

Mortal Knowledge Protocol: A Governance Framework for AI Memory Systems

Richard “Ricky” Valentine
Independent Researcher, Houston, TX, USA
Ricky@RichardValentine.dev

Version 1.2 · March 2026

Abstract

We present the Mortal Knowledge Protocol (MKP), a governance framework for AI memory systems built on the thesis that knowledge must be mortal to remain valuable. MKP defines a six-state lifecycle (NOVEL, ACTIVE, CRYSTALLIZED, DECAYING, FROZEN, DEAD) governed by attention-weighted decay, where knowledge that is not retrieved gradually loses salience and migrates from active storage to cold archive. The protocol introduces three novel mechanisms:

1. *Two-Temperature Storage Model* — a finite Hot Mesh separated from an effectively unlimited Cold Archive, with cryptographically enforced lifecycle transitions.
2. *Organ Donor Permission Model* — sovereign knowledge transfer with graduated consent, explicit opt-in, and post-donation independence.
3. *Complement Discovery* — emergent relationship identification through co-retrieval pattern analysis and spectral clustering.

A Memory Wizard governance agent, implemented as a Model Context Protocol (MCP) server, orchestrates these mechanisms while preserving human authority over all write operations. MKP builds on Crystallization Theory (CT), extending the $W = (C \times R \times V)/A$ confidence formula with operational storage semantics and permission primitives. Version 1.2 adds protocol-level sequence flows, a cryptographic enforcement model, and an evaluation framework with falsifiable success criteria.

Keywords: knowledge governance, AI memory, lifecycle management, attention-weighted decay, federated knowledge, cognitive sovereignty, MCP, knowledge mortality

Related Patents: USPTO #63/953,509 (Crystallization Theory), #63/962,609 (KEOS)

1. Introduction

AI knowledge systems suffer four fundamental problems that current architectures fail to address as a unified challenge:

Cognitive Bloat. Knowledge bases grow without bound. Every fact, observation, and retrieval result is retained indefinitely, creating retrieval noise that degrades system performance. The “lost in the middle” phenomenon demonstrates that models struggle with mid-context information as context windows expand [Liu et al., 2024]. Model collapse from recursively generated data further compounds the quality degradation of stored knowledge [Shumailov et al., 2024].

Knowledge Obsolescence. Information decays in value over time, but current systems treat all stored knowledge as equally valid regardless of age or usage. A market analysis from 2019 occupies the same

retrieval priority as one from yesterday. No mechanism exists to identify, demote, or archive knowledge that has lost relevance.

Sovereignty Gaps. In multi-user and federated deployments, knowledge ownership is poorly defined. Users have limited control over how their contributed knowledge is used, shared, or retired. No existing framework provides end-of-life directives for knowledge assets.

Synthesis Failure. Current retrieval systems return documents, not insights. They cannot identify that two knowledge units, individually unremarkable, become valuable when considered together. Swanson’s seminal work on undiscovered public knowledge demonstrated this gap decades ago [Swanson, 1986]; it remains unaddressed in AI memory architectures.

Existing approaches such as MemGPT [Packer et al., 2023] address memory management through tiered storage but lack lifecycle governance, sovereignty semantics, and synthesis mechanisms. MKP addresses all four problems through a single architectural thesis: **knowledge must be mortal to remain valuable**. By introducing lifecycle semantics, attention-weighted decay, sovereign permissions, and co-retrieval synthesis into a unified protocol, MKP creates knowledge systems that stay lean, current, sovereign, and generative.

2. Knowledge Mortality

2.1 The Mortality Thesis

MKP is built on a single axiom: **knowledge that is not used should not persist in active memory**. This is not a storage optimization—it is an epistemic position. Knowledge derives value from its relationship to the observer’s current context and decision needs. Knowledge that no one retrieves has, by definition, zero operational value, regardless of its truth content.

This mirrors biological memory: the brain does not store all experiences with equal fidelity. Frequently accessed memories are strengthened (long-term potentiation); unused memories weaken (synaptic pruning). The lottery ticket hypothesis [Frankle and Carlin, 2019] demonstrates an analogous principle in neural networks: aggressive pruning can preserve or improve performance. MKP implements this principle at the knowledge-system level through attention-weighted decay.

2.2 Six-State Lifecycle

Every knowledge unit in MKP exists in exactly one of six states, defined by its crystallization weight W and operational status (Figure 1):

NOVEL ($\text{Age} < 48\text{h}$)

Newly ingested, awaiting first retrieval and validation. Not yet part of the active knowledge graph.

ACTIVE ($0.4 \leq W < 0.8$)

In regular use, subject to standard attention-weighted decay. The primary operational state.

CRYSTALLIZED ($W \geq 0.8$)

High-confidence knowledge validated through repeated co-firing. Decay is paused. Reached after sustained active use with high endorsement and minimal contradictions.

DECAYING ($0.1 \leq W < 0.4$)

Below survival threshold with declining usage. Scheduled for evaluation. If usage does not recover, transitions to FROZEN.

FROZEN ($W < 0.1$)

Moved to Cold Archive. Can be thawed if relevance returns. No longer in Hot Mesh; retrieval requires explicit thaw authorization.

DEAD

Explicitly deleted by human actor. Irrecoverable. Marked in event log for audit. FROZEN units may also transition to DEAD when omission mandates require permanent destruction.

All state transitions are triggered either by human-approved events or by deterministic MKP rules executing on decay thresholds. No state transition occurs without a logged event.

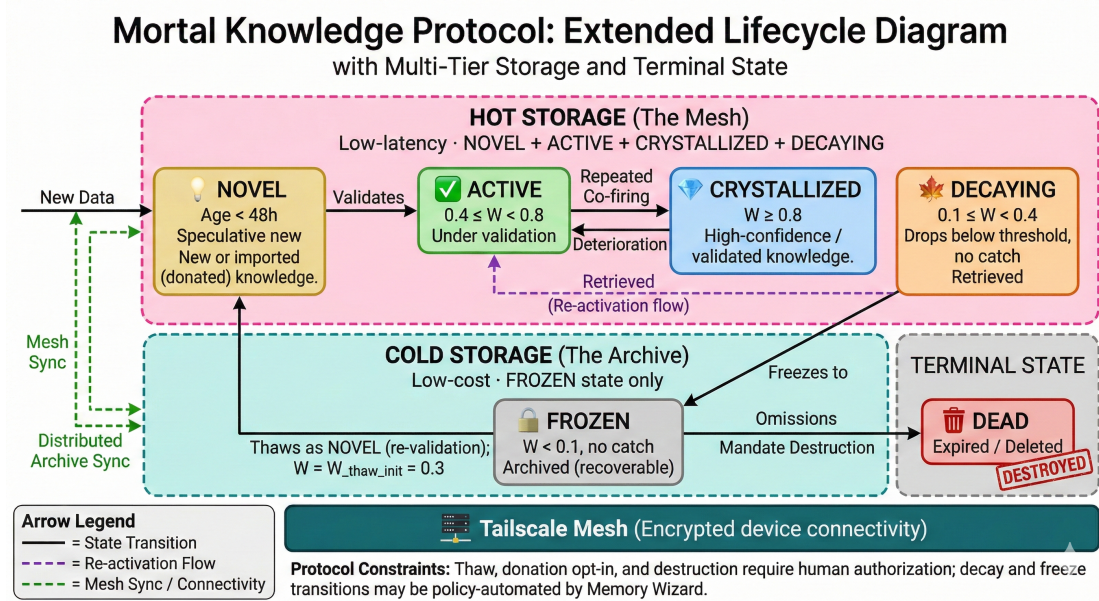


Figure 1: MKP Extended Lifecycle Diagram. State transitions are governed by crystallization weight W and human authorization rules. Hot Storage (The Mesh) retains all active states including DECAYING; Cold Storage (The Archive) holds FROZEN units pending potential thaw. Thawed units re-enter as NOVEL at $W = W_{\text{thaw_init}} = 0.3$, requiring fresh validation before advancing.

2.3 Attention-Weighted Decay

Knowledge decay follows a Hebbian principle: knowledge that is frequently retrieved stays alive; unused knowledge fades. The survival score is computed as:

$$\text{Survival Score} = W \cdot e^{-\lambda t} \quad (1)$$

where W is the crystallization weight (usage + endorsement), λ is the decay constant (configurable; default 0.01 per day), and t is days since last retrieval. When a knowledge unit is retrieved, the decay timer resets. CRYSTALLIZED knowledge has decay paused entirely. ACTIVE knowledge decays slowly unless used. DECAYING knowledge faces accelerated decay.

When the survival score drops below the freeze threshold (default 0.1), the knowledge unit transitions to FROZEN and is moved to Cold Archive.

2.4 Thaw Semantics

When a FROZEN knowledge unit is thawed, its weight **resets to a defined initial thaw value** ($W_{\text{thaw_init}}$), not its pre-freeze weight. The unit must re-earn its weight through fresh usage. This prevents immortality through freeze/thaw cycling, which would defeat the purpose of mortality.

The initial thaw value is a protocol constant (e.g., $W_{\text{thaw_init}} = 0.3$) or a bounded function of pre-freeze state, but is **never equal** to the unit's pre-freeze W . This ensures that thawed knowledge re-enters the Hot Mesh as NOVEL (re-validation) and must prove renewed relevance through fresh local retrieval patterns, consistent with MKP's epistemic position that value is usage-relative.

W Reset Convention. On thaw, W resets to $W_{\text{thaw_init}} = 0.3$. On donation ingestion, receiving nodes initialize $W = 0.3$. This value is the canonical “unproven but plausible” starting point—above the DECAYING threshold, below the ACTIVE threshold, requiring demonstrated local retrieval to advance.

3. The Cognitive Stack

MKP operates within a layered architecture called the **Cognitive Stack** (Figure 2). Each layer has a defined responsibility and communicates through explicit handoff protocols. The stack is **local-first**: all components run on the user’s device or mesh, with no cloud dependency required.

Figure 1: The MKP Cognitive Stack (Local-First Architecture)

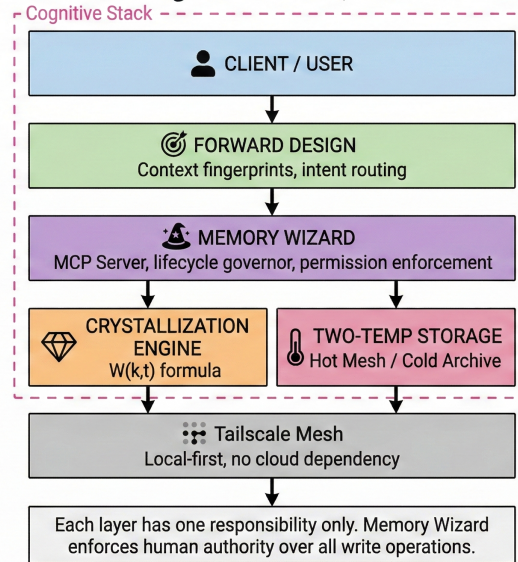


Figure 1: The MKP Cognitive Stack (Local-First Architecture)

Figure 2: The MKP Cognitive Stack: Local-First Architecture. Five layers—Client/User, Forward Design, Memory Wizard, Crystallization Engine with Two-Temperature Storage, and Tailscale Mesh—communicate through explicit handoff protocols with unidirectional data flow. The Memory Wizard is the sole governance agent, enforcing human authority over all write operations. The stack is local-first—no cloud dependency required.

3.1 Forward Design

Forward Design is the context-awareness layer. It maintains awareness of the user’s current intent, active tools, and conversational state. Forward Design orchestrates tool calls and determines which knowledge to surface based on **context fingerprints**—structured representations of the user’s current task, domain, and decision needs.

Forward Design does not store knowledge. It routes queries to the Memory Wizard, receives retrieval results, and presents them to the user with appropriate confidence indicators and complement suggestions.

3.2 Memory Wizard

The Memory Wizard is the knowledge governance agent, implemented as a **Model Context Protocol (MCP) server** [Anthropic, 2024]. It is the sole arbiter of knowledge lifecycle transitions and permission enforcement. The Memory Wizard:

- Receives retrieval requests from Forward Design
- Queries the Hot Mesh and, when necessary, the Cold Archive
- Enforces permission checks using cryptographic verification
- Manages freeze, thaw, and donation operations
- Runs complement discovery (query-time suggestions and background refinement)
- Monitors decay patterns and proposes lifecycle transitions

The Memory Wizard operates on behalf of the human owner but does not possess write authority over knowledge content. It governs the lifecycle and permissions of knowledge, not its substance.

3.3 Crystallization Engine

The Crystallization Engine computes confidence scores using the W formula from Crystallization Theory [Valentine, 2025]. For each knowledge unit k at time t :

$$W(k, t) = \frac{C(k) \times R(k, t) \times V(k)}{A(k)} \quad (2)$$

Here $C(k)$ is the confirmation count from independent contributors, $R(k, t)$ is the recency function reflecting retrieval frequency, $V(k)$ is the utility score based on retrieval-to-action conversion, and $A(k)$ is the age since (re)introduction into the Hot Mesh (creation, donation ingestion, or thaw), i.e., the current validation epoch. The Crystallization Engine updates W on every retrieval event and resets the associated decay timer.

3.4 Two-Temperature Storage

Knowledge is stored across two tiers with fundamentally different performance and capacity characteristics:

Hot Mesh (Active Knowledge). In-memory or fast distributed cache containing NOVEL, ACTIVE, CRYSTALLIZED, and DECAYING knowledge units. Capacity-limited by organizational cognitive bandwidth (typically 10,000–100,000 units for enterprise). Sub-millisecond retrieval. Synchronized across mesh nodes via encrypted network.

Cold Archive (Dormant Knowledge). Append-only blob store containing FROZEN units. Effectively unlimited capacity. Retrieval latency in seconds (requires thaw authorization). Encrypted at rest with per-unit data encryption keys.

This separation ensures the active working knowledge base remains performant regardless of historical archive size. As organizational knowledge accumulates over years, the Hot Mesh stays lean through continuous decay-driven pruning.

4. Permission and Sovereignty

4.1 Cognitive Sovereignty

MKP establishes that each human user has **absolute sovereignty** over their contributed knowledge assets. Sovereignty means:

- The owner decides what knowledge enters the system (ingestion control)
- The owner decides who can retrieve their knowledge (access control)
- The owner decides whether knowledge can be donated on freeze (donation control)
- The owner decides when knowledge is permanently destroyed (destruction control)

No system component—including the Memory Wizard—can override these sovereignty primitives. They are cryptographically enforced, not policy-enforced.

4.2 Organ Donor Permission Model

When a knowledge unit freezes, its owner may have previously marked it as **donatable**—an explicit, signed declaration that this knowledge may be offered to other mesh nodes. The Organ Donor Permission Model draws on the semantics of biological organ donation:

Explicit opt-in. The donatable flag must be set by the owner before freeze. The Memory Wizard cannot set this flag autonomously.

Graduated consent. Owners can specify which knowledge categories are donatable, to which recipients, and under what conditions.

Post-donation independence. Donated knowledge arrives at the receiving node as NOVEL ($W = W_{\text{thaw_init}}$). It must earn its place through local retrieval. The donor has no ongoing control over the recipient’s copy.

End-of-life directives. Owners can specify what happens to their knowledge on account deactivation: donate all, destroy all, or selective disposition.

4.3 Permission Enforcement

Permissions in MKP are **cryptographically enforced**, not policy-enforced. This means a compromised Memory Wizard cannot bypass access controls because access requires possession of the correct cryptographic key, not merely a policy flag evaluation.

4.4 Cryptographic Enforcement Model

This section specifies the minimum viable cryptographic architecture required to fulfill the enforcement claim in Section 4.

Design Principle. Every knowledge unit is encrypted at rest. Access requires possession of the correct key, not merely a policy check that could be bypassed by a compromised component.

Table 1: Cryptographic enforcement model. Each knowledge asset is encrypted independently; access requires key possession, not policy evaluation. KEK = Key Encryption Key (owner-held); DEK = Data Encryption Key (per-unit, wrapped).

Asset	Cipher	Key Type	Key Holder
Content	AES-256-GCM	Symmetric DEK	Memory Wizard / Cold Archive
Metadata	AES-256-GCM	Separate DEK	Memory Wizard
Event Log	HMAC chain	HMAC key	Node operator
Perm. Manifest	Ed25519	Asymmetric	Owner (private key)
Donatable Flag	Ed25519	Asymmetric	Owner (private key)

Asset	On Freeze	On Thaw	On Donate
Content	DEK wrapped with owner KEK	Owner unwraps DEK	Re-encrypted under new DEK; original rotated
Metadata	Wrapped independently	Unwrapped with KEK	Optionally shared per scope
Event Log	Freeze event appended	Thaw event appended	Donation event on both chains
Perm. Manifest	Frozen with unit	Verified before restore	New manifest signed by donor (owner) granting recipient access
Donatable Flag	Frozen with unit	N/A	Verified before broadcast

4.4.1 Key Hierarchy

The system uses a **two-tier envelope encryption model**:

Key Encryption Keys (KEKs). Each human owner holds a KEK, derived from identity credentials or hardware tokens. KEKs never leave the owner’s device. Cross-node transfer uses recipient public keys (or device-bound secure channels) to deliver wrapped DEKs end-to-end over authenticated mesh tunnels.

Data Encryption Keys (DEKs). Each knowledge unit has a unique DEK generated at creation time. DEKs are wrapped by the owner’s KEK for storage. The Memory Wizard operates on unwrapped DEKs in memory during active use and re-wraps them on any state transition.

4.4.2 Ownership Transfer and Revocation

Because donation re-encrypts content under a recipient-specific DEK, recipients never possess the donor’s DEK. This enables **revocation of the donor’s access to recipient-held copies** and supports explicit ownership transfer when the donor deletes their local ciphertext and destroys their wrap state.

Note: The donor retains access to their own archived ciphertext unless they explicitly destroy it—revocation is of cross-node copies, not self-held originals.

4.4.3 Key Rotation

DEKs are rotated on donation. KEKs are rotated on credential change. The HMAC chain provides tamper evidence (not confidentiality). Nodes may additionally sign periodic log checkpoints (hash anchors) for third-party audit.

Scope Note: This section specifies the cryptographic model, not the implementation. Cipher suite selection, key derivation functions, and HSM integration are deferred to reference implementations.

5. Protocol Operations

5.1 Freeze and Thaw

Freeze is the transition from Hot Mesh to Cold Archive. When a knowledge unit’s survival score drops below the freeze threshold, the Memory Wizard initiates the freeze sequence: pause decay, wrap the DEK with the owner’s KEK, serialize content and metadata, store in Cold Archive, and log the event.

Thaw is the reverse: a query matches a FROZEN unit, thaw authorization is obtained (from owner or admin only), the DEK is unwrapped, content is restored to Hot Mesh, and the unit re-enters the Hot Mesh as NOVEL (re-validation) at $W_{\text{thaw_init}}$ with a fresh decay timer.

5.2 Complement Discovery

Complement Discovery identifies emergent relationships between knowledge units by analyzing co-retrieval patterns. It operates in two phases:

Phase A (Query-Time). On every retrieval, the Memory Wizard performs a constant-time lookup on the cached complement graph (maintained by Phase B). It scores and ranks complement suggestions based on link strength, recency weight, and context similarity, returning the top $k=5$ above a threshold of 0.3.

Phase B (Background). A periodic job (e.g., daily) scans the full retrieval history, builds a co-firing matrix of session-based co-retrievals, performs spectral cluster analysis, and updates complement links. New high-co-firing pairs without existing links are proposed; existing links with declining co-firing are marked as weakening.

Complement links have an independent lifecycle (PROPOSED → CONFIRMED → WEAKENING → REMOVED) distinct from the knowledge unit lifecycle, allowing relationships to adapt even when units remain stable.

All complement suggestions are advisory. The client decides whether to retrieve the complement. No automatic retrieval of unrequested knowledge occurs.

5.3 MCP Tool Interface

The Memory Wizard exposes its operations as MCP tools, enabling integration with any MCP-compatible client:

- `retrieve(query, context_fingerprint) → content + W + complements`
- `freeze(unit_id) → confirmation + archive location`
- `thaw(unit_id, authorization) → restored content + new W`
- `donate(unit_id, recipient_node) → donation status`
- `set_donatable(unit_id, flag, scope) → confirmation`
- `get_complements(unit_id, k) → ranked complement list`
- `get_lifecycle(unit_id) → full state history`

6. Wizard Operations

6.1 Decay Management

The Memory Wizard monitors decay patterns across the Hot Mesh, identifying three categories:

Steady decline

Usage naturally tapering—proceed to freeze.

Oscillating

Periodic usage spikes—hold in DECAYING, do not freeze.

Cliff drop

Sudden cessation—flag for review, may indicate context shift rather than irrelevance.

6.2 Co-Retrieval Analysis

Beyond complement discovery, the Memory Wizard tracks co-retrieval for operational intelligence: which knowledge units are always retrieved together (candidates for merge), which are retrieved in alternation (candidates for disambiguation), and which are retrieved by different users in similar contexts (candidates for cross-pollination).

6.3 Evaluation Framework

While empirical validation remains future work, this section defines the metrics against which any MKP implementation should be measured. These targets establish falsifiability criteria for the protocol's core claims.

Table 2: Evaluation metrics and falsifiable success criteria. These targets establish measurable benchmarks for any MKP implementation; complement metrics require human-labeled evaluation sets.

Metric	Target	Measurement	Rationale
Hot mesh stability	$\pm 10\%$ / 90d	Weekly count of ACTIVE + CRYSTALLIZED units	Validates that decay and freeze maintain equilibrium
Retrieval latency (hot)	<50 ms at p95	Instrumented Memory Wizard query pipeline	Confirms hot mesh supports real-time interaction
Retrieval latency (thaw)	<5 s end-to-end	Verification + key unwrap + restore (excludes human approval dwell time)	Validates cold archive remains practically accessible
Freeze/thaw fidelity	100% integrity	Hash comparison pre- and post-transition	Lifecycle transitions must be lossless and permission-preserving
Complement precision	>0.70 at $k=5$	Human judgment labels on suggested pairs	Surfaced complements are useful, not noise
Complement recall	>0.50	Expert-labeled known-good pairs	System finds obvious cross-unit relationships
False-freeze rate	$<5\%$ / quarter	Units thawed within 7 d of freeze	Calibrates decay constants against premature archival
Donation acceptance	$>30\%$ in 30 d	Acceptances / offers received by eligible (complement-matched) recipients	Donated knowledge produces useful transfers
Decay calibration	W within ± 0.1	Predicted vs. actual retrieval probability	Validates W formula’s predictive power

Auto-tuning rule. If the false-freeze rate exceeds its target, increase the decay half-life (or raise the freeze threshold) until the metric returns to band. This creates a self-calibrating feedback loop: observed thaw-within-7-days events directly inform decay parameter adjustment.

These metrics are measurable with standard instrumentation (event logs, timestamps, hash functions). Complement discovery precision and recall require human-labeled evaluation sets, consistent with MKP’s epistemic position that knowledge value is usage-relative—there is no ground truth for “correctness,” but there is ground truth for “useful co-retrieval.”

7. Relationship to Crystallization Theory

MKP builds directly on Crystallization Theory [Valentine, 2025]. The $W = (C \times R \times V) / A$ formula remains the core confidence calculation. CT’s five-state lifecycle provides the governance foundation, which MKP extends to six states by distinguishing FROZEN (recoverable cold storage) from DEAD (permanent deletion). Variance-as-signal remains the mechanism for extracting meta-knowledge from disagreement.

7.1 Terminology Evolution

The W formula variables have evolved between CT and MKP to better reflect their operational roles. CT’s original C (Confirmations) and R (Recency) map to MKP’s C (Consensus, reflecting multi-source agreement) and R (Retrieval, reflecting usage-based recency rather than time-of-creation recency). This relabeling does not change the mathematical structure—the formula operates identically—but clarifies the semantic intent within MKP’s governance context.

The key extensions MKP adds beyond CT are: operational storage semantics (two-temperature model), permission primitives (Organ Donor model, cognitive sovereignty), governance agency (Memory Wizard as MCP server), and complement discovery (co-retrieval synthesis). **CT provides the validation physics; MKP provides the governance framework.**

8. Scope and Limitations

This paper presents an architectural framework, not empirical results. The following limitations are acknowledged:

No benchmarks. The evaluation framework (Section 6.3) defines targets but does not report measurements. Empirical validation requires a reference implementation.

Decay constant calibration. The default $\lambda = 0.01/\text{day}$ is illustrative. Optimal decay rates will vary by domain, organizational size, and knowledge velocity.

Complement discovery cold-start. Phase A (query-time) depends on Phase B having built the complement graph. New deployments will have no complement suggestions until sufficient retrieval history accumulates.

Cryptographic overhead. Per-unit encryption adds latency to every lifecycle transition. The performance impact on high-throughput systems has not been measured.

Single-owner assumption. The current permission model assumes each knowledge unit has one owner. Multi-author knowledge (collaborative documents, team-generated insights) requires extension to joint ownership semantics.

Scale validation. Hot mesh size stability and complement discovery accuracy may behave differently at enterprise scale (100K+ units, 1000+ users) versus the single-node case described here.

9. Conclusion

The Mortal Knowledge Protocol introduces a governance framework for AI memory systems built on the thesis that knowledge must be mortal to remain valuable. By combining attention-weighted decay, two-temperature storage, cryptographic permission enforcement, sovereign donation semantics, and co-retrieval-based complement discovery, MKP addresses the four fundamental problems of cognitive bloat, knowledge obsolescence, sovereignty gaps, and synthesis failure as a unified architectural challenge.

Version 1.2 adds three concrete mechanisms that advance MKP from architectural description toward implementable specification: protocol-level sequence flows defining message-level interactions for retrieval, freeze/donation, and complement discovery; a cryptographic enforcement model specifying the key hierarchy and encryption-per-operation guarantees; and an evaluation framework with nine falsifiable metrics and an auto-tuning feedback loop.

MKP is designed as the governance companion to Crystallization Theory’s validation framework. Together, they provide the complete lifecycle: **CT determines what knowledge is worth keeping; MKP determines how that knowledge lives, migrates, and dies.**

Acknowledgments. This work was developed independently. The author thanks the open-source AI research community for tools and frameworks that enabled rapid prototyping and specification development.

Conflicts of Interest. The author has filed patent applications related to the Crystallization Theory framework (USPTO #63/953,509), the KEOS architecture (USPTO #63/962,609), and therapeutic peptide delivery systems (USPTO #63/960,158).

Code and Data Availability. A reference implementation of the Memory Wizard MCP server and evaluation datasets will be released under an open-source license upon completion of initial testing. Repository link will be added to the Zenodo record metadata at that time.

Document Version: 1.2 | Publication Date: March 2026 | License: CC BY 4.0 (text), MIT (reference code)

A. Protocol Mechanics

The following sequence flows specify the message-level interactions between MKP components for the three core operations. These flows formalize the architectural interactions described in Sections 3–6 at the protocol level.

Actors: Client (requesting system/user), Forward Design (FD, context layer), Memory Wizard (MW, governance/MCP server), Crystallization Engine (CE, W formula), Hot Mesh (HM, active store), Cold Archive (CA, dormant store), Owner/Admin (human authority).

A.1 Retrieval with Thaw

Covers the complete retrieval path, including cold-path thaw authorization.

```

Client → FD:      Query(context, intent)
FD → MW:          RetrievalRequest(query, context_fingerprint)
MW → HM:          Lookup(query, context_fingerprint)

--- Hot Path (unit found in hot mesh) ---
HM → MW:          Hit(unit_id, content, W_score)
MW → CE:          UpdateDecay(unit_id, retrieval_event)
CE → HM:          WriteBack(unit_id, new_W, decay_timer_reset)
MW → FD:          RetrievalResponse(content, W_score, complements[])
FD → Client:      Result(content, confidence, suggested_complements)

--- Cold Path (unit not in hot mesh) ---
HM → MW:          Miss()
MW → CA:          ArchiveQuery(query, context_fingerprint)
CA → MW:          FrozenHit(unit_id, ciphertext, wrapped_DEK,
                        permission_manifest)
MW → MW:          VerifyPermission(manifest, requester_identity)

[If requester lacks thaw authority]
MW → FD:          ThawRequired(unit_id, summary, owner_contact)
FD → Client:      NeedsAuthorization(unit_id, summary)
Client → Owner:   ThawRequest(unit_id, reason)
Owner → MW:       ThawApproval(unit_id, owner_KEK_unwrap)

[If requester IS owner/admin]
MW proceeds directly

MW → CA:          Thaw(unit_id, unwrapped_DEK)
CA → HM:          Restore(unit_id, content, metadata)
MW → CE:          InitializeDecay(unit_id, W=W_thaw_init)
CE → HM:          WriteBack(unit_id, W_thaw_init, new_decay_timer)
MW → FD:          RetrievalResponse(content, W_thaw_init, complements[])
FD → Client:      Result(content, confidence, source=thawed)

```

Key Properties:

- Hot path never touches Cold Archive—latency stays under 50ms
- Thaw requires cryptographic proof of authority (KEK unwrap), not just a policy flag

- Thawed unit's W resets to $W_{\text{thaw_init}}$ (not restored from pre-freeze); unit must re-earn weight through fresh usage, preventing immortality via cycling
- $W_{\text{thaw_init}}$ is a protocol constant (e.g., 0.3) or bounded function, but never equal to pre-freeze W
- Decay timer restarts from the thaw event

A.2 Freeze with Donation

Covers the complete freeze path, including organ donor broadcast when the unit is marked donatable.

```

CE → MW:      DecayAlert(unit_id, W_score=0.09, below_threshold)
MW → MW:      CheckLifecycleRules(unit_id, current_state=DECAYING)
MW → HM:      GetMetadata(unit_id)
HM → MW:      Metadata(unit_id, donatable_flag, owner_id,
                  complement_links)

--- Freeze Execution ---
MW → CE:      PauseDecay(unit_id)
MW → HM:      WrapAndRemove(unit_id, owner_KEK)
HM → MW:      WrappedUnit(unit_id, ciphertext, wrapped_DEK)
MW → CA:      Store(unit_id, ciphertext, wrapped_DEK,
                  permission_manifest)
CA → MW:      Stored(unit_id, archive_location)
MW → MW:      LogEvent(FREEZE, unit_id, timestamp, W_at_freeze)

--- Donation Broadcast (only if donatable_flag = true) ---
MW → MW:      CheckDonatableFlag(unit_id) // verify Ed25519 sig
MW → Mesh:    DonationOffer(unit_id, metadata_summary,
                  complement_profile)

[For each interested mesh node]
RemoteMW → MW: DonationInterest(unit_id, node_id,
                  complement_score)
MW → MW:      VerifyDonatableFlag(unit_id, owner_signature)
MW → RemoteMW: DonationGrant(unit_id,
                  ciphertext_reencrypted_under_new_DEK,
                  wrapped_new_DEK_for_recipient,
                  permission_manifest)
RemoteMW → RemoteMW: UnwrapDEK(wrapped_new_DEK_for_recipient,
                  recipient_key)
RemoteMW → RemoteMW: Decrypt(ciphertext_reencrypted_under_new_DEK,
                  new_DEK)
RemoteMW → RemoteHM: Ingest(unit_id, content, state=NOVEL)
RemoteMW → RemoteCE: InitializeDecay(unit_id, W=W_thaw_init)
RemoteMW → MW:      DonationAcknowledged(unit_id, node_id)

MW → MW:      LogEvent(DONATION_COMPLETE, unit_id,
                  recipients[], timestamp)

```

Key Properties:

- Donatable flag is a signed assertion by the owner—cannot be set by Memory Wizard autonomously
- Donated units arrive at receiving nodes as NOVEL ($W = W_{\text{thaw_init}}$), not ACTIVE—they must earn their place through local retrieval
- The original DEK is never shared; a fresh DEK is generated and wrapped for each recipient
- Donation and freeze are logged as separate events in the HMAC chain
- Non-donatable units simply freeze without the broadcast phase

A.3 Complement Discovery (Dual-Phase)

Covers both real-time query-time suggestion and background refinement phases.

```

=== Phase A: Query-Time Suggestion ===

[Triggered on every retrieval that returns a result]

MW → MW:      OnRetrieval(unit_id, context_fingerprint)
MW → HM:      GetComplementEdges(unit_id)
                // Cached adjacency list from Phase B
                // Constant-time lookup; no raw log scan
HM → MW:      ComplementEdges[(co_unit_id, link_strength,
                link_state)]
MW → MW:      ScoreComplements(edges, context_fingerprint)
                // Score = link_strength x recency x context_sim
MW → MW:      RankAndFilter(complements, k=5, threshold=0.3)
MW → FD:      ComplementSuggestions(ranked_complements[])
FD → Client:   Result(primary_content,
                suggested_complements[])

=== Phase B: Background Refinement ===

[Runs periodically, e.g., daily or on configurable schedule]

Scheduler → MW: RunComplementAnalysis()
MW → HM:      GetRetrievalLog(window=90d)
HM → MW:      FullRetrievalHistory[(unit_id, ts, context,
                session)]

MW → MW:      BuildCoFiringMatrix(history)
                // Matrix[i][j] = sessions where both i and j
                // were retrieved in same context window

MW → MW:      ClusterAnalysis(matrix, method=spectral)
MW → MW:      DetectNewComplements(clusters, existing_links)
MW → MW:      DetectStaleComplements(clusters, existing_links)

    [For each new complement candidate]
    MW → MW:      ProposedComplement(unit_a, unit_b, confidence,
    evidence)
    MW → HM:      WriteComplementLink(unit_a, unit_b,
    status=PROPOSED)

    [For each stale complement]
    MW → MW:      StaleComplement(unit_a, unit_b, decay_evidence)
    MW → HM:      UpdateComplementLink(unit_a, unit_b,
    status=WEAKENING)

MW → MW:      LogEvent(COMPLEMENT_ANALYSIS, new_count,
    stale_count, timestamp)

```

Key Properties:

- Phase A is latency-sensitive—performs constant-time edge lookup on the cached complement graph, not raw log scanning
- Phase B is compute-intensive but runs asynchronously—no impact on retrieval latency

- Complement links have their own lifecycle (PROPOSED → CONFIRMED → WEAKENING → REMOVED) independent of knowledge unit lifecycle
- The co-firing matrix uses session-based co-retrieval (units retrieved in the same task context), not mere temporal proximity
- Background refinement can discover cross-domain complements that query-time suggestion misses
- All complement suggestions are advisory; the client/user decides whether to retrieve the complement

References

- Anthropic. Model context protocol specification. <https://modelcontextprotocol.io>, 2024. Accessed March 2026.
- Jonathan Frankle and Michael Carlin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.
- Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638.
- Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. 2023.
- Iliia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI models collapse when trained on recursively generated data. *Nature*, 631:755–759, 2024. doi: 10.1038/s41586-024-07566-y.
- Don R. Swanson. Fish oil, Raynaud’s syndrome, and undiscovered public knowledge. *Perspectives in Biology and Medicine*, 30(1):7–18, 1986. doi: 10.1353/pbm.1986.0087.
- Richard Valentine. Crystallization theory: A framework for multi-contributor knowledge validation in AI memory systems. 2025. USPTO #63/953,509.
- Richard Valentine. Extracellular vesicle compositions comprising self-amplifying RNA encoding therapeutic peptides, 2026a. USPTO #63/960,158.
- Richard Valentine. KEOS: A knowledge-epistemic operating system for hallucination-free, collapse-immune organizational AI, 2026b. USPTO #63/962,609.