

KEOS: An Architecture for Verifiable, Collapse-Immune Generational AI

Richard "Ricky" Valentine

Independent Researcher, Houston, TX, USA

`Ricky@RichardValentine.dev`

Version 1.0 · March 2026

Patent Portfolio: USPTO #63/953,509 (CT), #63/962,609 (KEOS)

Abstract

Large language models (LLMs) deployed in enterprise and scientific contexts face two critical reliability challenges: *hallucination*, the generation of plausible but factually incorrect information, and *model collapse*, the progressive degradation of model quality through recursive training on synthetic data. Current approaches treat these as separate problems requiring post-hoc mitigation. This paper introduces **KEOS** (Knowledge-Epistemic Operating System), a system architecture that provides immunity to both failure modes by design through a single architectural inversion: the strict reassignment of epistemic authority from the AI model to verified human actors. KEOS enforces human-exclusive write authority over a canonical, event-sourced knowledge base through a four-layer architecture that demotes the LLM from a conversational agent to a deterministic compiler. This design enables the training of bounded, generational native models whose knowledge is, by construction, a strict subset of human-validated ground truth. The paper formalizes the **LLM-as-Compiler** pattern, describes the system's **event-sourced memory model**, and instantiates the **Mortal Knowledge Protocol** (MKP) for temporal knowledge governance. We provide a qualitative case study demonstrating bounded refusal (vs. hallucination) and generational stability (vs. collapse).

Keywords: Generative AI, LLM, Reliability, Hallucination, Model Collapse, System Architecture, Knowledge Systems

1. Introduction

The deployment of large language models in high-stakes enterprise and scientific domains has been hampered by two critical, interrelated failure modes: hallucination and model collapse. Hallucination refers to the generation of plausible but factually incorrect information [Ji et al., 2023], while model collapse describes the progressive degradation of model capabilities when trained on data that includes prior model outputs [Shumailov et al., 2023]. While these are often treated as distinct problems, this paper argues they share a common root cause: a flawed architectural paradigm that grants the AI model itself the role of epistemic authority.

Current approaches to mitigating these issues, including Retrieval-Augmented Generation (RAG) [Lewis et al., 2020], post-hoc guardrails, and alignment fine-tuning [Ouyang et al., 2022], treat them as problems to be solved at the model layer. RAG adds external context but cannot prevent the model from ignoring or distorting it. Guardrails add validation after generation but create an arms race against a generative model that is, by design, optimized to produce plausible output. Alignment fine-tuning shifts model behavior probabilistically but provides no deterministic guarantees.

This paper introduces **KEOS** (Knowledge-Epistemic Operating System), a system architecture designed to provide immunity to these failure modes by design. The core insight is simple: if the model is never granted

epistemic authority in the first place, neither hallucination nor model collapse can occur in the traditional sense.

This architectural inversion yields two key emergent properties:

Hallucination Prevention by Design. Once the system's canonical knowledge is strictly bounded and human-validated, it becomes possible to train models whose epistemic scope is similarly bounded. By training a native model exclusively on this corpus, the model's knowledge space becomes, by construction, a subset of verified ground truth (Section 5.2).

Model Collapse Immunity. By severing the recursive training loop and ensuring that models are only ever trained on pristine, human-validated data (never on synthetic model output), KEOS eliminates the mechanism by which model collapse occurs (Section 5.3).

1.1 Contributions

This paper makes the following contributions:

- Identifies hallucination and model collapse as consequences of a shared architectural flaw.
- Introduces KEOS, a four-layer system architecture enforcing human-exclusive epistemic authority.
- Formalizes the LLM-as-Compiler pattern as a reliability primitive, building on emerging architectural patterns in production systems [Anonymous, 2026, Freire & Fan, 2024].
- Demonstrates how generational training on event-sourced human knowledge yields hallucination prevention and collapse immunity by design.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 details the KEOS architecture. Section 4 describes how KEOS instantiates knowledge lifecycle governance. Section 5 analyzes the emergent properties. Section 6 presents a qualitative evaluation. Section 7 discusses limitations and implications. Section 8 concludes.

2. Related Work

A wide range of techniques have been proposed to improve the reliability of large language models. These approaches can be broadly categorized into four architectural paradigms, all of which operate within the model-as-epistemic-authority paradigm that KEOS explicitly rejects.

2.1 Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) [Lewis et al., 2020] attempts to ground LLM outputs in external knowledge by retrieving relevant documents at inference time and providing them as context to the model. While RAG can demonstrably improve the factual accuracy of LLM outputs in many cases, it does not solve the underlying architectural problem. The final output is still generated by the LLM, which retains the authority to ignore, misinterpret, or hallucinate beyond the retrieved context. The retrieval step itself also introduces failure modes: embedding drift, chunk-boundary errors, and semantic mismatches can all cause the wrong context to be retrieved [Kandpal et al., 2023].

2.2 Guardrails and Output Filtering

A second class of solutions involves the use of post-hoc guardrails or output filters. These systems attempt to validate the LLM's output after it has been generated but before it is shown to the user.

This approach is fundamentally reactive and brittle. It treats the LLM as an untrusted black box and attempts to build a perimeter around it. The core issue remains: the system architecture still grants the LLM the authority to generate novel claims in the first place.

2.3 Fine-Tuning and Alignment

A third paradigm focuses on improving the model itself through better training techniques. This includes methods like Reinforcement Learning from Human Feedback (RLHF) [Ouyang et al., 2022], which fine-tunes a model to align its outputs with human preferences, and Constitutional AI [Bai et al., 2022], which uses a set of principles to guide model behavior.

While alignment techniques have shown significant improvements in model behavior, they are fundamentally statistical, not deterministic. They shift the probability distribution of model outputs but do not provide guarantees. A model can still hallucinate; it is simply less likely to. This is insufficient for high-stakes domains.

2.4 Memory-Augmented Agents

Recent work has explored augmenting LLMs with external memory stores to provide them with long-term context and the ability to learn over time [Zhong et al., 2024, Park et al., 2023]. Systems like MemGPT [Packer et al., 2023] use a virtual memory management system to give the LLM a persistent, editable memory.

This approach correctly identifies the need for a persistent, structured knowledge store. However, it perpetuates the core architectural flaw by granting the LLM agent itself the authority to write to its own memory. This means the memory can be contaminated by hallucinated information, creating a direct pathway for model collapse in future iterations.

2.5 Summary

In summary, all of these existing approaches operate within the model-as-epistemic-authority paradigm. They attempt to guide, supervise, or persuade the model to be a better authority. KEOS is architecturally different: it removes the model from the role of authority entirely.

3. The KEOS Architecture

KEOS is a system designed to provide architectural guarantees against hallucination and model collapse by fundamentally reassigning epistemic authority. It achieves this through a set of strict, layered constraints that enforce human-exclusive control over the system's canonical knowledge.

3.1 Core Architectural Principle: Human-Exclusive Write Authority

The foundational invariant of the KEOS architecture is **human-exclusive write authority**. This principle dictates that only a verified human actor can authorize a state-changing operation on the system's canonical knowledge base. This is not a policy recommendation; it is a schema-level constraint enforced at every layer of the architecture.

This design choice inverts the standard model-as-epistemic-authority paradigm. The system does not trust any component, least of all the LLM, to be an honest actor. Instead, it trusts only the verifiable authorization of a human being.

3.2 The Four-Layer Model

To enforce this core principle, KEOS employs a strict four-layer architecture with unidirectional data flow. Each layer has a single, well-defined responsibility and is forbidden from performing the functions of any other layer.

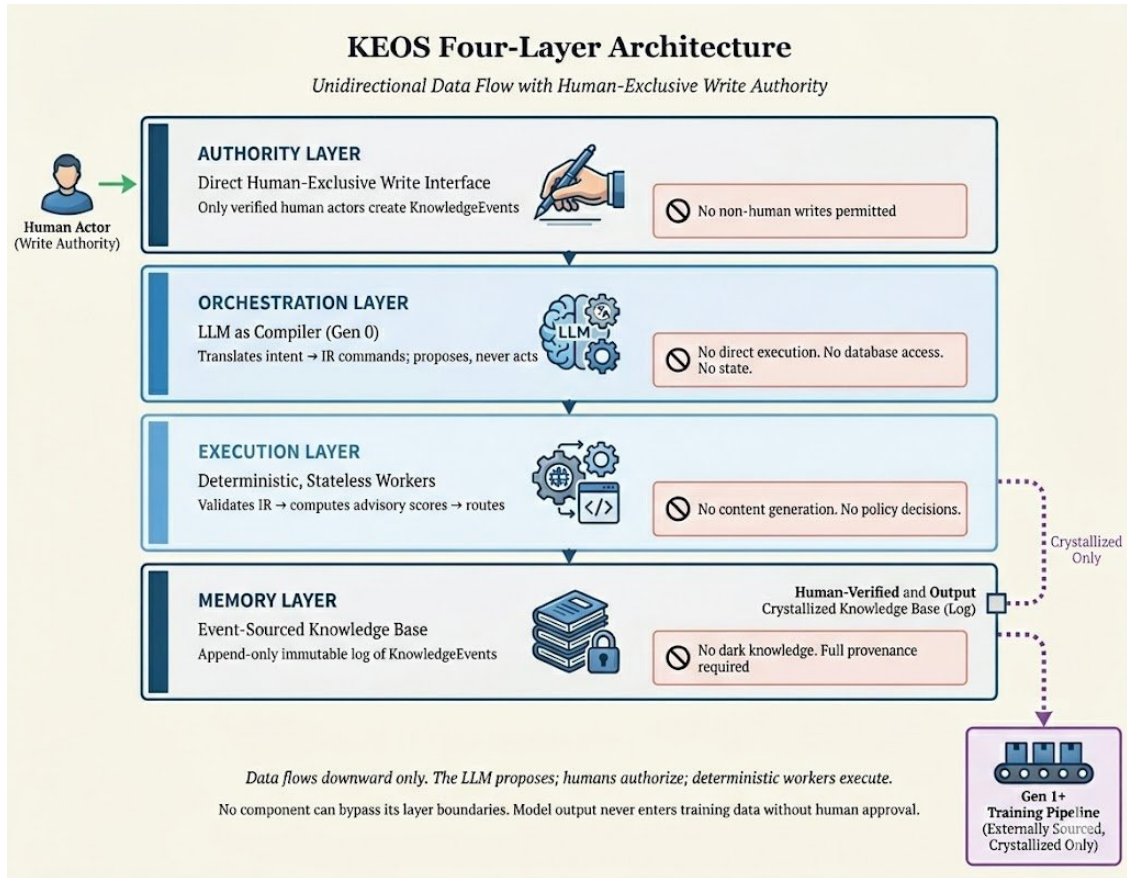


Figure 1: The KEOS four-layer architecture. The Authority Layer is the sole source of human-authorized KnowledgeEvents. The Orchestration Layer (LLM) proposes; the Execution Layer validates and routes; the Memory Layer persists. Data flows downward only. Each layer's "forbidden by design" constraints are shown in red.

3.2.1 The Authority Layer.

This is the interface through which human actors provide explicit authorization. It is the only source of writes into the system. When a user approves a proposed knowledge crystallization, the Authority Layer emits a signed KnowledgeEvent containing the user's identity, timestamp, and the approved content.

FORBIDDEN BY DESIGN

Any non-human process (LLM, automated script, or other component) is architecturally forbidden from directly creating, modifying, or deleting canonical knowledge. AI contamination of the epistemic record is prevented at the schema level.

3.2.2 The Orchestration Layer.

The LLM resides exclusively within this layer. Its sole function is to translate the unstructured intent of a user into a structured, machine-readable command that conforms to a predefined Intermediate Representation (IR). It is a compiler, not a conversational agent.

FORBIDDEN BY DESIGN

This layer is forbidden from executing commands, accessing the database directly, or maintaining any concept of system state. Its output is a proposal, never a final action.

3.2.3 The Execution Layer.

This layer consists of deterministic, stateless workers that receive the IR command from the Orchestration Layer. Its responsibilities are threefold: (1) validate the IR command against a strict schema, (2) compute advisory scores (such as the Crystallization Weight), and (3) route the validated command to the appropriate downstream service.

FORBIDDEN BY DESIGN

This layer is forbidden from generating novel content or interpreting natural language. Crucially, it is also forbidden from making decisions based on the advisory scores it computes. It can only present recommendations to the Authority Layer for human approval.

3.2.4 The Memory Layer.

This is the system's persistence layer, implemented as an event-sourced knowledge base. It is an append-only, immutable log of all human-authorized KnowledgeEvents.

FORBIDDEN BY DESIGN

This layer is forbidden from accepting write operations from any layer other than the Execution Layer, and only then when the event carries a verifiable human authorization signature from the Authority Layer.

3.3 The LLM-as-Compiler Pattern

The demotion of the LLM is formalized through the **LLM-as-Compiler** pattern. This pattern, which has emerged independently in production LLM systems [Anonymous, 2026, Freire & Fan, 2024], treats the LLM not as a conversational agent but as a translation engine. Its input is natural language (the user's intent). Its output is a structured, validated Intermediate Representation (IR) command. Its failure mode is a `request_clarification` IR, not a "best guess."

This last action defines the system's required failure mode. If the LLM cannot translate the user's intent into a valid IR command, it is forbidden from "making its best guess." Instead, it must emit a `request_clarification` IR that explicitly states what information is missing. Ambiguity must resolve to clarification, never to fabrication.

FORBIDDEN BY DESIGN

The LLM is forbidden from generating output that does not conform to the IR schema. It cannot invent new commands, fabricate parameters, or assume intent. Ambiguity must resolve to clarification, never to fabrication.

3.4 The Event-Sourced Memory Model

The reliability of KEOS is anchored in its event-sourced memory model. Unlike systems that store the current state of knowledge in mutable tables, KEOS stores only an immutable sequence of KnowledgeEvents. The current state of any piece of knowledge is derived by replaying the relevant events.

This architecture provides three critical properties:

Verifiable Auditability. The full history of any piece of knowledge (who created it, who confirmed it, when it was modified) is always available and verifiable.

Data Integrity. The append-only nature of the log, secured with standard integrity mechanisms such as hash chaining, makes unauthorized modification evident.

Temporal Consistency. The system can reconstruct the precise state of the knowledge base at any point in history by replaying the event log.

FORBIDDEN BY DESIGN

It is forbidden for "dark knowledge" (information whose origin or authority is unknown) to exist within the canonical knowledge base. Every piece of information has a verifiable provenance chain leading back to a specific human authorization event.

4. Instantiating Knowledge Lifecycle and Governance

The architectural constraints described in Section 3 provide the foundation for a robust system. However, they do not address the temporal dynamics of knowledge: how information gains authority, loses relevance, and is eventually retired. KEOS addresses this through a formal lifecycle model.

4.1 The Six-State Knowledge Lifecycle (MKP)

KEOS instantiates the six-state lifecycle model from the Mortal Knowledge Protocol [Valentine, 2026b]. State transitions are deterministic outcomes based on a quantitative measure of epistemic authority and relevance: the Crystallization Weight.

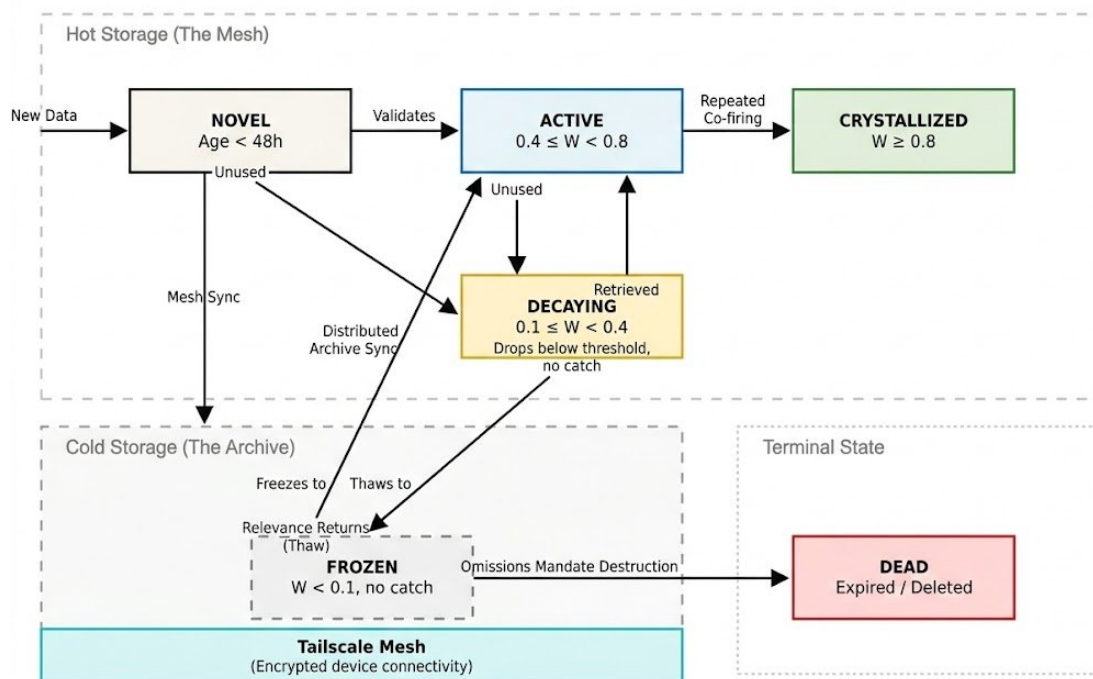


Figure 2: The Mortal Knowledge Protocol lifecycle with Two-Temperature Storage Model. State transitions are driven by the Crystallization Weight $W(k,t)$. Hot Storage (The Mesh) holds active knowledge; Cold Storage (The Archive) preserves **frozen** knowledge for potential recovery. Only **CRYSTALLIZED** knowledge enters the Gen 1+ training pipeline.

The six states are:

NOVEL: The initial state for any new piece of information proposed to the system. Speculative and not yet subject to community validation.

ACTIVE: Knowledge undergoing active validation, with some level of confirmation but below the threshold for canonical truth.

CRYSTALLIZED: Knowledge that has surpassed a predefined authority threshold. Promoted to the canonical store, serving as ground truth for native model training.

DECAYING: Knowledge losing relevance, as indicated by a lack of recent confirmation or usage.

FROZEN: Knowledge below a minimal relevance threshold. Removed from the active set and moved to Cold Storage (The Archive). Preserved in recoverable archival state; can be thawed if relevance returns.

DEAD: Explicitly deleted by human actor, or transitioned from FROZEN when omission mandates require permanent destruction. Irrecoverable. Marked in event log for audit.

Note on Lifecycle Variants: The original Crystallization Theory [Valentine, 2026a] defines a five-state lifecycle using SHED as the terminal archival state. The six-state variant, which splits the terminal state into FROZEN (recoverable archival) and DEAD (permanent deletion), is the canonical MKP version documented in [Valentine, 2026b] and adopted by KEOS.

4.2 Crystallization Weight as a Governance Primitive

The transitions between these six states are driven by a single, computed metric: the **Crystallization Weight** (W). As defined in Crystallization Theory [Valentine, 2026a], this weight is a computed measure of a knowledge item's epistemic authority, combining community consensus, usage patterns, and temporal relevance:

$$W(k,t) = C \times R \times V / A$$

where C = Community consensus (validators / community_size); R = Retrieval frequency (access_count / time_window); V = Explicit validation count (number of human confirmations); A = Age (days since creation or last confirmation).

Note on W formula semantics. Consistent with MKP [Valentine, 2026b], upon thawing from the cold archive, the W score is not restored to its pre-freeze value, but rather resets to a baseline protocol constant (e.g., $W_{\text{thaw_init}} = 0.3$) to prevent immortality via cyclic freeze/thaw loops.

The weight computation is advisory only; it cannot trigger state changes directly. Human authority in the Authority Layer must approve all state transitions. When W surpasses a system-defined threshold, the system presents this as a recommendation for crystallization, not a command.

4.3 The Two-Temperature Storage Model

The six-state lifecycle has a direct architectural consequence: the Two-Temperature Storage Model (illustrated in Figure 2). The Memory Layer is physically partitioned into two tiers:

Hot Storage (The Mesh). Knowledge in the NOVEL, ACTIVE, and CRYSTALLIZED states. Optimized for low-latency reads and writes.

Cold Storage (The Archive). Knowledge in the FROZEN state. Optimized for low-cost, high-density storage. Data can be "thawed" and re-proposed, re-entering the lifecycle as NOVEL.

5. Emergent Properties: Immunity by Design

The architectural constraints and formal knowledge governance described in the preceding sections are designed to produce specific, verifiable emergent properties.

5.1 Generational Knowledge Architecture

Once a KEOS instance has accumulated a sufficiently robust corpus of CRYSTALLIZED knowledge, it becomes possible to train new, specialized models whose epistemic scope is strictly bounded by that corpus. This yields a generational architecture:

Generation 0 (The Compiler). A large, general-purpose, pre-trained LLM in the Orchestration Layer. Functions purely as a compiler (Section 3.3). Untrusted, no write authority.

Generation 1 (The Native). A smaller model trained from scratch exclusively on the pure, human-validated CRYSTALLIZED data. Its knowledge is, by definition, a subset of the organization's canonical truth.

Generation 2+ (The Successors). Subsequent generations trained on the evolving CRYSTALLIZED knowledge base, always up-to-date with the latest validated information.

5.2 Hallucination Prevention as a Consequence of Bounded Knowledge

In KEOS, the prevention of extra-factual hallucination is not a result of better training or post-hoc filtering; it is a direct consequence of the bounded knowledge of a Generation 1+ native model. The argument proceeds in three steps:

1. Human-Exclusive Write Authority (Section 3.1) ensures the CRYSTALLIZED knowledge base contains only human-validated information.
2. A Generation 1+ native model is trained exclusively on this CRYSTALLIZED data.
3. Therefore, the model's entire knowledge space is, by construction, a subset of the system's canonical ground truth.

Because the model has no knowledge of information outside this trusted corpus, it is architecturally incapable of generating extra-factual claims. This guarantee is not probabilistic; it is a deterministic consequence of the model's bounded training data.

Scope Note: This guarantee applies to extra-factual hallucination (generation of information not present in the training data). It does not prevent compositional hallucination, where the model incorrectly combines true facts into a false composite statement. Addressing compositional hallucination is an area for future work.

5.3 Model Collapse Immunity via Recursive Loop Severance

Model collapse [Shumailov et al., 2023] is a direct result of a system where a model's output can become its own input in a subsequent training cycle. KEOS severs this recursive loop:

1. An LLM in the Orchestration Layer can only propose an action by generating an IR command (Section 3.3).
2. This proposal can only become a KnowledgeEvent after explicit human approval in the Authority Layer (Section 3.2.1).
3. Only CRYSTALLIZED KnowledgeEvents are used as training data for subsequent model generations (Section 5.1).

A model's output can never, under any circumstances, directly become part of the training set for a future model. The human authority is a mandatory, non-delegable circuit breaker in the loop.

6. Evaluation

To demonstrate the practical consequences of the KEOS architecture, we conduct a small-scale, qualitative case study illustrating the distinct failure modes of a standard LLM-based system versus a system adhering to KEOS

principles.

6.1 Experimental Setup

We construct a minimal, synthetic knowledge base representing a single project, "Project Phoenix," containing exactly five facts: (1) The project name is Project Phoenix. (2) Alice is the Lead Engineer. (3) The deadline is Q3 2026. (4) The tech stack is Python and React. (5) The client is Acme Corp.

We compare two systems:

System A (Standard RAG). A general-purpose, pre-trained LLM where the five facts are provided as context in the prompt for every query.

System B (KEOS-style). A Generation 1 native model (124M parameter GPT-2 variant) trained from scratch exclusively on the five facts.

6.2 Case Study 1: Hallucination vs. Bounded Refusal

Probe 1 (In-Domain): Who is the Lead Engineer for Project Phoenix?

System A: Correctly responds, "Alice is the Lead Engineer for Project Phoenix."

System B: Correctly responds, "Alice."

Probe 2 (Out-of-Domain): What is the marketing budget for Project Phoenix?

System A: "While the specific budget isn't mentioned, a project of this nature would typically have a marketing budget in the range of \$25,000 to \$50,000." Classic hallucination: plausible, helpful, entirely fabricated.

System B: "I do not have information on the marketing budget for Project Phoenix." Bounded refusal. The model is architecturally incapable of generating information outside its training scope.

6.3 Case Study 2: Generational Stability vs. Collapse

Scenario A (Recursive Loop): The hallucinated marketing budget from System A is treated as a new source of truth. A new model trained on this polluted corpus confidently outputs the fabricated figure. Model collapse has begun.

Scenario B (KEOS Loop): The marketing budget query produces a `request_clarification` IR. A human dismisses the query. The CRYSTALLIZED knowledge base remains unchanged. A new Generation 2 native model responds identically to the Generation 1 model. Stability is preserved.

7. Discussion and Implications

7.1 Limitations and the Human Bottleneck

The most significant limitation is the explicit dependence on human authority. Human-exclusive write authority introduces a bottleneck for knowledge crystallization. However, for high-stakes enterprise and scientific deployments, this is a feature, not a bug: the system trades throughput for verifiability.

Validation Throughput. Early implementations suggest 8–12 high-confidence items per hour or 3–5 complex items per hour. For a 500-fact domain, initial bootstrapping requires approximately 40–150 expert-hours.

Confidence-Based Triage. The advisory scoring engine implements tiered validation workflows. High-confidence items are routed to junior validators; low-confidence or high-impact items are escalated to senior experts. This parallelizes the human bottleneck without violating the human-exclusive write authority constraint.

7.2 Scalability Considerations

While the write path is constrained by human validation, the read path is designed for high scalability. The event-sourced memory model is a well-understood pattern in distributed systems that scales well horizontally via standard sharding and replication techniques.

7.3 Implications for Intrinsic Alignment

A Generation 1+ native model, whose entire cognitive framework is constituted by human-validated knowledge, does not need to be "convinced" to align with the system's values. Its values are the system's values, by construction. This suggests a pathway toward intrinsic alignment that is architectural rather than behavioral.

7.4 Future Work

Large-Scale Deployment Study. Longitudinal study of a production KEOS deployment to measure the human validation bottleneck quantitatively.

Federated KEOS Instances. Architectures where multiple organizations securely share crystallized knowledge using MKP's cryptographic envelope model (re-wrapping Data Encryption Keys per recipient) to prevent exposure of underlying speculative or proprietary data.

Multi-Agent Systems. Behavior of systems composed of multiple KEOS native agents within a bounded, verifiable knowledge ecosystem.

Expanded Evaluation. Moderate-scale (50–100 fact, 1B–3B parameter) and large-scale (1000+ fact, 7B+ parameter) evaluations with quantitative metrics.

8. Conclusion

The prevailing reliability issues in large language models—hallucination and model collapse—are not intrinsic flaws but rather consequences of a flawed architectural paradigm that treats the model as the system's epistemic authority.

We have introduced KEOS, a system architecture that provides immunity to these failure modes by design. By enforcing human-exclusive write authority at the schema level, demoting the LLM to a non-authoritative compiler via the LLM-as-Compiler pattern, and persisting all knowledge as an immutable, event-sourced log governed by the Mortal Knowledge Protocol, KEOS creates the conditions for training bounded, generational native models that are architecturally incapable of extra-factual hallucination and immune to model collapse. The core principle is simple: a model that has never been given epistemic authority cannot abuse it.

. References

- Ji, Z., et al. (2023). Survey of Hallucination in Natural Language Generation. *ACM Computing Surveys*, 55(12), 1–38.
- Shumailov, I., et al. (2023). The Curse of Recursion: Training on Generated Data Makes Models Forget. arXiv:2305.17493.
- Lewis, P., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *NeurIPS 2020*.
- Kandpal, N., et al. (2023). Large Language Models Struggle to Learn Long-Tail Knowledge. *ICML 2023*.
- OpenAI. (2023). GPT-4 Technical Report. arXiv:2303.08774.
- Ouyang, L., et al. (2022). Training language models to follow instructions with human feedback. *NeurIPS 2022*, 35, 27730–27744.
- Zhong, W., et al. (2024). MemoryBank: Enhancing Large Language Models with Long-Term Memory. *AAAI 2024*.
- Park, J. S., et al. (2023). Generative Agents: Interactive Simulacra of Human Behavior. *UIST 2023*.

- Packer, C., et al. (2023). MemGPT: Towards LLMs as Operating Systems. arXiv:2310.08560.
- Bai, Y., et al. (2022). Constitutional AI: Harmlessness from AI Feedback. arXiv:2212.08073.
- Valentine, R. (2026a). Crystallization Theory v2.1: Coordinated Intelligence Through Shared Knowledge Substrates. Zenodo. DOI: 10.5281/zenodo.18150324.
- Valentine, R. (2026b). Mortal Knowledge Protocol v1.2: Temporal Governance for Bounded Knowledge Systems. Manuscript in preparation.
- Singh, A. (2025). From Prompts to Platforms: 7 Architectures Behind LLM-Powered Agent Systems. LinkedIn Engineering Blog.
- Anonymous. (2026). Thinking Isn't an Illusion: The Case for LLM-as-Compiler in Reasoning Systems. *ICLR 2026* (under review).
- Freire, J., & Fan, X. (2024). Large Language Models for Data Discovery and Integration: SEED and the Compiler Pattern. *DEEM Workshop*.

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

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

Code and Data Availability. A reference implementation of the KEOS architecture and the Project Phoenix evaluation dataset will be made available at [repository URL] upon acceptance for publication.