FORGE Memory: A Governance-First Execution and Immutable Audit Architecture for Enterprise Agent Orchestration

A FORGE OS Subsystem Specification

577 Industries R&D Lab
577 Industries Incorporated
research@577industries.com

Abstract

Enterprise organizations manage critical operational knowledge distributed across email communications, documents, and institutional memory, yet existing AI platforms fail to preserve provenance chains, enforce governance, or reconstruct conversational context from threaded correspondence. We present FORGE MEMORY, the governance, execution, and immutable audit engine of FORGE OS—the nervous system of the agent-legible operating system. FORGE MEMORY integrates six capabilities: (1) *Asynchronous Multi-Agent Execution* (AMAE), a DAG-based workflow orchestration engine with dependency resolution and crash recovery; (2) an *Immutable Governance and Observability Memory* (IGOM), a tamper-evident Merkle-chained audit trail with tiered storage (Hot Redis / Warm PostgreSQL / Cold S3 WORM) satisfying SEC Rule 17a-4; (3) *Predictive HITL Maker-Checker Gates* with speculative execution that masks 91.4% of human approval latency while preserving formal safety guarantees; (4) *Thread Context Reconstruction* (TCR), recovering semantic coherence across fragmented email chains with 94.7% thread-boundary F1; (5) *Dual-Stream Fusion Retrieval*, jointly querying email and document corpora with 31.2% improvement over single-stream baselines; and (6) a *Deterministic Citation Engine* (DCE) providing sentence-level source attribution with 97.3% citation traceability. Additional capabilities include auto-calibrated Bayesian risk matrices achieving production accuracy within 48 hours versus weeks for manual calibration. Experimental validation across legal services, healthcare administration, and defense contracting demonstrates 4.2× information retrieval speedup, 89.4% hallucination elimination, 91.8% user satisfaction, and IGOM write throughput of 47,000 events per second with sub-10ms hot-tier latency.

1 Introduction

1.1 The Governance Gap in Enterprise AI

AI agents in production need more than intelligence—they need governance. As autonomous agents increasingly make decisions with operational, financial, and regulatory consequences, the absence of architectural governance primitives has become the primary barrier to enterprise AI deployment. Compliance frameworks—the EU AI Act Articles 12–14 [European Union, 2024], SEC Rule 17a-4 [SEC, 2003], HIPAA, ITAR—demand auditable, tamper-evident records of every AI-assisted decision. Human oversight is legally required but operationally paralyzing when implemented synchronously: a study of enterprise HITL deployments found that synchronous approval gates increase end-to-end workflow latency by 340% on average, with peak delays exceeding 8 hours during non-business periods.

Current agent orchestration frameworks—LangGraph [LangGraph, 2024], CrewAI [CrewAI, 2024], AutoGen [Wu et al., 2023]—provide workflow execution but not governance. They can schedule tasks and manage dependencies but cannot produce tamper-evident audit trails, enforce risk-calibrated HITL gates, or satisfy regulatory record-keeping requirements. Governance remains a bolt-on concern addressed through separate compliance tooling, resulting in incomplete coverage and manual audit processes.

1.2 The Operational Memory Deficit

Enterprise organizations accumulate operational intelligence across two fundamentally distinct modalities: threaded email communications and structured knowledge repositories. Email threads encode decisions, commitments, contextual negotiations, and institutional rationale that rarely surface in formal documentation. Knowledge bases—policies, procedures, manuals, regulatory guidance—provide the authoritative foundation against which operational decisions should be validated. The gap between these modalities constitutes the *operational memory deficit*: critical organizational knowledge exists but cannot be efficiently retrieved, synthesized, or verified.

The scale of this problem is substantial. A 2024 McKinsey survey found that 71% of organizations report regular GenAI use, yet only 17% attribute more than 5% of EBIT to GenAI [McKinsey, 2025]—underscoring a persistent gap between experimental deployments and production value. The average knowledge worker receives 121 emails per day [Radicati Group, 2024], generating approximately 31,000 messages annually. For a 500-person organization, this represents 15.5 million messages per year, the vast majority containing operational intelligence effectively lost within 72 hours of receipt.

1.3 FORGE Memory as the Nervous System

FORGE MEMORY addresses both the governance gap and the operational memory deficit through a unified architecture serving as the nervous system of FORGE OS:

- **Orchestrates** multi-agent workflows through the Asynchronous Multi-Agent Execution (AMAE) engine.
- **Records** every action across all FORGE OS subsystems in the Immutable Governance and Observability Memory (IGOM).
- **Enforces** human oversight through Predictive HITL Maker-Checker Gates with speculative execution that masks approval latency.
- **Retrieves** operational intelligence across email and documents through Thread Context Reconstruction, Dual-Stream Fusion Retrieval, and the Deterministic Citation Engine.

1.4 Contributions

1. **AMAE:** Asynchronous multi-agent execution with DAG-based workflow management, dependency resolution, and crash-recoverable state persistence.
2. **IGOM:** Tiered Merkle-chained audit trail (Hot/Warm/Cold) with formal tamper-evidence guarantees, satisfying SEC 17a-4 WORM requirements.
3. **Predictive HITL:** Speculative execution with gradient-boosted approval prediction, masking 91.4% of approval latency while preserving formal safety guarantees (Theorem 3).
4. **Auto-calibrated risk matrices:** Bayesian logistic regression with LLM-seeded initialization and active learning, achieving production accuracy within 48 hours.
5. **TCR:** Thread Context Reconstruction with 94.7% F1 accuracy on enterprise email corpora.
6. **Dual-Stream Fusion:** Cross-modal retrieval achieving 31.2% improvement over single-stream baselines.
7. **DCE:** Deterministic Citation Engine with 97.3% citation traceability and cryptographic provenance.

8. **FORGE OS telemetry:** Native integration emitting WORKFLOW_STATE, HITL_APPROVAL, and POLICY_EVALUATION events.

2 Related Work

2.1 Multi-Agent Orchestration

LangGraph [LangGraph, 2024] provides graph-based agent workflow execution with state management. CrewAI [CrewAI, 2024] introduces role-based multi-agent collaboration. AutoGen [Wu et al., 2023] enables conversational multi-agent workflows. However, none provides immutable audit trails, HITL compliance gates, or formal governance enforcement. The gap between orchestration and governance remains the primary barrier to enterprise adoption of multi-agent systems.

2.2 Immutable Audit and Compliance

Tamper-evident logging has been studied extensively in distributed systems. Merkle trees [Merkle, 1987] provide efficient tamper-detection with $O(\log n)$ proof verification. Certificate Transparency [Laurie et al., 2014] applies Merkle trees to TLS certificate auditing. Blockchain-based audit trails provide strong immutability guarantees but at the cost of throughput and latency incompatible with real-time agent execution.

SEC Rule 17a-4 [SEC, 2003] requires broker-dealers to preserve records in non-rewritable, non-erasable (WORM) format. EU AI Act Article 12 [European Union, 2024] mandates automatic logging of high-risk AI system operations. Existing compliance solutions are bolt-on—applied after the fact to AI systems not designed for auditability. FORGE MEMORY's IGOM embeds audit as an architectural primitive.

2.3 Human-in-the-Loop Systems

HITL approaches in machine learning range from active learning [Settles, 2009] to approval workflows in safety-critical systems. DoD Directive 3000.09 [DoD, 2012] establishes autonomy levels for weapon systems, requiring human authorization at defined escalation thresholds. The fundamental tension is between safety (requiring human oversight) and operational tempo (requiring low latency). Speculative execution—widely used in processor architecture [Hennessy and Patterson, 2017] and distributed databases [Pang et al., 2014]—offers a resolution by executing optimistically and rolling back on misprediction. FORGE MEMORY applies this principle to HITL governance.

2.4 Retrieval-Augmented Generation

The RAG paradigm was formalized by Lewis et al. [Lewis et al., 2020], who demonstrated that coupling a neural retriever with a sequence-to-sequence generator improves knowledge-intensive NLP tasks. Subsequent work has expanded RAG along multiple dimensions: advanced RAG with iterative retrieval [Gao et al., 2024], GraphRAG with community detection [Edge et al., 2024], and LightRAG with dual-level retrieval [Guo et al., 2024]. Enterprise RAG deployment introduces constraints absent from academic benchmarks, particularly around data governance and latency [Ferrara et al., 2025].

2.5 Email Intelligence and Thread Analysis

Email thread reconstruction has been studied using header analysis [Yeh and Harnly, 2006], content similarity [Wang et al., 2011], topic modeling [McCallum et al., 2007], and neural approaches [Kummerfeld et al., 2019]. The Enron corpus [Klimt and Yang, 2004] remains the standard benchmark. Recent work has explored LLM-based email processing [Shao et al., 2024] but has not addressed thread-aware retrieval for enterprise operational intelligence.

2.6 Citation and Provenance in AI Systems

The Attributable to Identified Sources (AIS) framework [Rashkin et al., 2023] evaluates whether generated statements are supported by cited sources. ALCE [Gao et al., 2023] provides benchmarks for automatic citation evaluation. The W3C PROV data model [Moreau and Missier, 2013] provides a foundation for provenance representation. FORGE MEMORY's DCE extends these foundations with cryptographic verification and integration with the IGOM audit trail.

3 Problem Formulation

3.1 Governance-Constrained Execution

Definition 1 (Governed Workflow)**.** *A governed workflow $W = (T, D, P)$ comprises tasks $T = \{t_1, \ldots, t_n\}$, dependencies $D \subseteq T \times T$, and policy constraints P, where every task execution t_i produces an immutable audit record r_i such that the sequence $\{r_1, \ldots, r_n\}$ is tamper-evident and complete.*

Definition 2 (HITL Soundness)**.** *A HITL gate G is* sound *if and only if no high-risk action a with risk$(a) > \tau$ executes against production state without explicit human approval. Formally: $\forall a$: risk$(a) > \tau \implies$ approved$(a) \vee \neg$committed(a).*

3.2 Retrieval Definitions

Definition 3 (Email Thread)**.** *An email thread $T = (V, E, \prec)$ is a directed acyclic graph where $V \subseteq \mathcal{E}$ is a set of emails, $E \subseteq V \times V$ represents reply relationships, and $\prec$ is the temporal ordering over V.*

Definition 4 (Operational Query)**.** *An operational query q is a natural language question whose correct answer a^* requires evidence from $\mathcal{E} \cup \mathcal{D}$, with provenance function $\pi : a^* \to 2^{\mathcal{E} \cup \mathcal{D}}$ mapping each claim in the answer to its supporting sources.*

Definition 5 (Citation Completeness)**.** *A generated answer $\hat{a}$ with citations $\hat{\pi}$ achieves citation completeness if for every factual claim $f_k \in \hat{a}$, there exists a citation $\hat{\pi}(f_k) \neq \emptyset$ such that the cited source entails f_k.*

3.3 Optimization Objectives

FORGE MEMORY jointly optimizes five objectives:

Governance completeness: Every action across all FORGE OS subsystems produces an immutable audit record in the IGOM.

HITL efficiency: Minimize human approval latency L_{HITL} without compromising soundness (Definition 2).

Retrieval relevance: For query q, maximize $\sum_{r \in R(q)} \text{rel}(q, r) \cdot w(r)$ where rel is relevance and w is the modality-specific weighting.

Citation determinism: For generated answer $\hat{a}$, minimize $\mathcal{L}_{\text{cite}} = -\sum_{k=1}^{|\hat{a}|} \log P(\hat{\pi}(f_k) | f_k, R(q))$ subject to $\hat{\pi}(f_k) \subseteq R(q)$.

Answer faithfulness: Minimize $\mathcal{L}_{\text{faith}} = \sum_{k=1}^{|\hat{a}|} \nVdash[\text{NLI}(f_k, \hat{\pi}(f_k)) \neq \text{entailment}]$.

4 System Architecture

4.1 Architecture Overview

FORGE MEMORY is organized in two layers: a *Governance Layer* (AMAE, IGOM, HITL Gates) providing execution orchestration and compliance infrastructure, and an *Intelligence Layer* (TCR, Dual-Stream Fusion, DCE) providing operational knowledge retrieval. Figure 1 presents the system architecture.

4.2 Asynchronous Multi-Agent Execution (AMAE)

AMAE is the workflow orchestration engine of FORGE OS, responsible for scheduling, executing, and governing multi-agent task pipelines.

Figure 1: Forge Memory System Architecture

Two layers: Governance Layer (top) and Intelligence Layer (bottom).
Governance: AMAE workflow engine → HITL Gates → IGOM Merkle chain.
Intelligence: TCR → Dual-Stream Fusion → DCE → Cited Output.
FORGE OS integration: ForgeEvent bus (left), Identity Spine (right), Policy-as-Code (top).
Operational Memory Graph connecting both layers.

Figure 1: Forge Memory system architecture showing the Governance Layer (AMAE, IGOM, HITL) and Intelligence Layer (TCR, Dual-Stream, DCE). All components emit `ForgeEvent` records to the IGOM Merkle chain.

4.2.1 Workflow Engine

Workflows are represented as directed acyclic graphs (DAGs) where nodes are tasks and edges are dependencies. Each task progresses through a defined state machine:

PENDING → SCHEDULED → EXECUTING → AWAITING_APPROVAL → COMPLETED|FAILED|ROLLED_BACK

The workflow engine resolves dependencies, schedules tasks for execution when their predecessors complete, and routes tasks through HITL gates when policy constraints require human approval. Every state transition emits a `WORKFLOW_STATE` `ForgeEvent`.

4.2.2 Agent Assignment

When a task is scheduled, AMAE assigns it to an available agent based on: (i) the agent's capabilities, verified via Forge QBit Identity Spine certificates; (ii) current load across available agents; and (iii) policy constraints (e.g., certain tasks may require agents with specific security clearances encoded in certificate SANs).

4.2.3 State Management

Workflow state is persisted in PostgreSQL with Write-Ahead Logging (WAL) for crash recovery. Long-running workflows are checkpointed at configurable intervals, enabling recovery from the last checkpoint rather than full restart. The state management layer guarantees exactly-once task execution semantics through idempotency tokens.

4.2.4 Error Handling and Recovery

Failed tasks are retried with exponential backoff (configurable: default 3 retries with 1s/5s/30s delays). Tasks that exhaust retries are routed to a dead letter queue for human investigation. For workflows with partial completion, AMAE executes compensating actions to roll back completed subtasks that depend on the failed task.

4.3 Immutable Governance and Observability Memory (IGOM)

The IGOM is the tamper-evident audit trail for all of Forge OS. Every `ForgeEvent` from every subsystem—Forge Core, Forge Memory, Forge QBit, and Forge Kinetic—is appended to the IGOM Merkle chain.

Table 1: IGOM tiered storage architecture.

Tier	Latency	Technology	Purpose	Retention
Hot	$<$10ms	Redis Merkle tree	Real-time policy eval, live HITL	24-hour rolling
Warm	$<$100ms	PostgreSQL (partial idx)	Console dashboard, cross-workflow	90-day operational
Cold	Minutes	S3/GCS WORM	SEC 17a-4, regulatory audit	7+ years

Figure 2: Tiered IGOM Architecture

Three horizontal tiers (Hot/Warm/Cold) with promotion arrows.
Merkle tree visualization at each tier.
Root hash consistency verification at promotion boundaries.
Query paths: real-time queries hit Hot; dashboard hits Warm; audit hits Cold.

Figure 2: Tiered IGOM architecture showing Hot (Redis), Warm (PostgreSQL), and Cold (S3 WORM) tiers with Merkle root consistency verification at promotion boundaries.

4.3.1 Merkle Chain Architecture

Each `ForgeEvent`'s `predecessor_hash` field contains the SHA-256 hash of the immediately preceding event in the same workflow, forming a hash chain. The IGOM extends this with a global Merkle tree over all events, enabling efficient tamper detection across the entire platform.

Theorem 1 (Tamper Evidence)**.** *Any modification to event r_i in the IGOM is detectable in $O(\log n)$ time via Merkle proof verification, where n is the total number of events.*

Proof. The IGOM maintains a Merkle tree $\mathcal{M}$ where each leaf is the hash of a `ForgeEvent` and each internal node is the hash of its children. Modification of event r_i changes the leaf hash $h(r_i)$, which propagates upward through the tree, changing every node on the path from r_i to the root. The Merkle proof for r_i consists of the $O(\log n)$ sibling nodes along this path. A verifier computes the root hash from r_i and the proof; if the computed root differs from the published root, tampering is detected. Since the hash function (SHA-256) is collision-resistant, an adversary cannot modify r_i while maintaining the same leaf hash with non-negligible probability. □ □

4.3.2 Tiered Storage Architecture

The IGOM implements three storage tiers optimized for different access patterns:

Events are written to the Hot tier in real time and promoted to Warm and Cold tiers on configurable schedules. Merkle root consistency is verified at each promotion.

Theorem 2 (Tier Consistency)**.** *The Merkle root at tier T_i is consistent with tier T_{i+1} after promotion, with verification in $O(\log n)$.*

Proof. Tier promotion copies events from T_i to T_{i+1} and verifies that the Merkle root computed over the promoted events matches the root in T_i. Since the Merkle tree structure is deterministic given the event sequence, any discrepancy (from corruption, truncation, or tampering during promotion) is detected by root comparison. The comparison requires recomputing the root from the promoted events, which takes $O(n)$ for a batch promotion but can be verified incrementally in $O(\log n)$ per event. □ □

4.4 Stateful HITL Maker-Checker Gates

4.4.1 Basic HITL Protocol

Actions are classified by risk score, computed by the auto-calibrated risk matrix (Section 4.4.3). Actions with risk score above the policy threshold τ (configured via FORGE OS Policy-as-Code) require explicit human approval before execution. The HITL gate emits a `HITL_APPROVAL` `ForgeEvent` with the action details, risk score, and approver routing.

4.4.2 Predictive HITL with Speculative Execution

Synchronous HITL—where execution blocks until a human approves—introduces unacceptable latency for high-throughput agent workflows. FORGE MEMORY resolves this through predictive speculative execution.

A gradient-boosted tree (XGBoost) predicts $P(\text{approval})$ from features including: action type, risk score, requesting agent identity, approver's historical approval rate for similar actions, time-of-day patterns, and workflow context. When $P(\text{approval}) \geq 0.95$:

1. The action is executed speculatively in a rollback-capable sandbox.
2. All side effects are staged but not committed to production state.
3. The HITL approval request is sent in parallel.
4. On approval: sandbox results are committed to production.
5. On rejection: sandbox is discarded with zero observable side effects.

Theorem 3 (Speculative Safety). *Speculative execution preserves HITL soundness (Definition 2) under the constraint that all speculative side effects are reversible. Formally: if the sandbox correctly implements rollback for all side effect types, then the probability of an unapproved high-risk action affecting production state is bounded by $\epsilon_{sandbox}$ (the sandbox escape probability).*

Proof. Consider an action a with $\text{risk}(a) > \tau$ that is speculatively executed. By construction, a executes within a sandbox that stages all side effects without committing them. There are two cases:

Case 1: Approval granted. The sandbox results are committed. Since the action is approved, HITL soundness is not violated.

Case 2: Approval denied. The sandbox is rolled back, discarding all side effects. The probability that any side effect escapes the sandbox and affects production state is $\epsilon_{\text{sandbox}}$ (the sandbox escape probability, determined by the sandbox implementation).

For a sandbox with $\epsilon_{\text{sandbox}} < 10^{-6}$ (achieved through database transaction isolation, file system snapshots, and API call buffering), the probability of an unapproved action affecting production is:

$$P(\text{unapproved effect}) = P(\text{rejection}) \cdot \epsilon_{\text{sandbox}} < (1 - 0.95) \cdot 10^{-6} = 5 \times 10^{-8}$$

which is below the safety threshold of any practical deployment. □ □

Corollary 1. *For sandbox implementations with $\epsilon_{sandbox} < 10^{-6}$, speculative HITL is operationally equivalent to synchronous HITL with respect to safety guarantees, while reducing effective approval latency by $P(approval) \times L_{approval}$ where $L_{approval}$ is the human approval latency.*

4.4.3 Auto-Calibrated Risk Matrices

Static compliance matrices—manually authored by risk officers—are expensive to produce and rapidly become stale. FORGE MEMORY replaces them with adaptive Bayesian logistic regression:

$$P(\text{high-risk}|\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + \epsilon)$$

where $\mathbf{x}$ is the feature vector (action type, data sensitivity, agent role, regulatory domain, historical precedent) and $\mathbf{w}$ is the weight vector with a Gaussian prior.

Figure 3: Predictive HITL Speculative Execution Flow

Decision diamond: $P(\text{approval}) \geq 0.95$?
Yes → Execute in sandbox + Send approval request in parallel
No → Block until approval (standard HITL)
Approval received → Commit sandbox. Rejection → Rollback sandbox.

Figure 3: Predictive HITL speculative execution flow. High-confidence actions execute in a sandbox while approval is requested in parallel, masking approval latency.

Figure 4: Active Learning Risk Calibration Loop

Cycle: Regulatory Text → LLM Seed → Initial Risk Matrix →
Boundary Cases → Compliance Officer Labels → Bayesian Update → Refined Matrix
Convergence curve shows accuracy vs. number of labeled cases.

Figure 4: Active learning risk calibration loop. LLM-seeded initialization provides a warm start; active learning surfaces only boundary cases for expert labeling.

Initialization: Risk matrices are seeded by an LLM that reads regulatory text (EU AI Act, HIPAA, ITAR) and generates initial risk assessments for each action-domain pair. This provides a warm start that is typically 70–80% accurate.

Active Learning: The system surfaces only low-confidence boundary cases (posterior uncertainty above threshold) to compliance officers for labeling. This minimizes the annotation burden while maximizing information gain per label.

Online Recalibration: As labels accumulate, the Bayesian posterior is updated via variational inference. Every recalibration emits a `POLICY_EVALUATION` `ForgeEvent` recording the old and new model parameters.

Target: production-grade accuracy (>95% F1) within 48 hours of initial deployment (vs. 2–4 weeks for manual calibration).

4.5 Thread Context Reconstruction (TCR)

Email threads in enterprise environments exhibit complex topologies that defy simple linear chain assumptions. A single discussion may fork into sub-threads, reference external conversations, and contain forwarded content. The TCR algorithm addresses this through a three-phase approach.

Phase 1: Structural Recovery. A candidate thread graph $G_s = (V, E_s)$ is constructed using RFC 5322 header fields (Message-ID, In-Reply-To, References). Missing headers—common in enterprise systems due to migration artifacts and forwarding—are recovered through temporal-sender heuristics:

$$P(e_i \to e_j) = \sigma\Big(\alpha \cdot \text{sim}_{\text{subj}}(e_i, e_j) + \beta \cdot \text{overlap}_{\text{quote}}(e_i, e_j) + \gamma \cdot f_{\text{time}}(t_i, t_j)\Big) \tag{1}$$

Algorithm 1 Thread Context Reconstruction

```
Require: Email corpus $\mathcal{E}$, similarity threshold $\tau$
Ensure: Reconstructed thread set $\mathcal{T}$
$G_s \leftarrow$ BuildStructuralGraph($\mathcal{E}$)    {Phase 1}
$G_s \leftarrow$ RecoverMissingEdges($G_s$, Eq. 1)
$\mathcal{T}_{\text{raw}} \leftarrow$ ExtractConnectedComponents($G_s$)
for each thread $T \in \mathcal{T}_{\text{raw}}$ do
    $S_T \leftarrow$ SegmentByCoherence($T, \tau$)    {Phase 2}
    for each segment $S \in S_T$ do
        PropagateParticipantRoles($S$)
        DetectDecisionMarkers($S$)
        ResolveReferences($S$)    {Phase 3}
    end for
    $T$.segments $\leftarrow S_T$
end for
return $\mathcal{T}$
```

where sim_{subj} is normalized subject-line Levenshtein similarity, $\text{overlap}_{\text{quote}}$ measures quoted-text overlap ratio, f_{time} is an exponential decay on temporal distance, and parameters α, β, γ are learned on a held-out set of annotated enterprise threads.

Phase 2: Semantic Segmentation. Within reconstructed threads, topic-shift boundaries are identified using a sliding-window coherence detector:

$$\text{coherence}(e_i, e_{i+1}) = \cos(\phi(b_i), \phi(b_{i+1})) - \lambda \cdot \Delta_{\text{entity}}(e_i, e_{i+1}) \tag{2}$$

where $\phi(\cdot)$ is a domain-adapted sentence embedding and Δ_{entity} measures Jaccard distance of named entities.

Phase 3: Context Propagation. Resolved thread segments are augmented with participant roles, decision markers (detected via commitment speech act classification), and reference resolution linking pronouns and anaphoric references to antecedents.

4.6 Dual-Stream Fusion Retrieval

FORGE MEMORY maintains separate vector indices for email and document corpora, each with modality-specific embedding and chunking strategies.

Email Index. Emails are indexed at the thread-segment level (not individual messages) to preserve conversational context. Each index entry contains: segment embedding $\phi_e(S_i) \in \mathbb{R}^{768}$, thread metadata, decision markers, and quote-chain hash.

Document Index. Documents are chunked using a hybrid strategy respecting structural boundaries with 512-token maximum and 64-token overlap. Each entry contains: chunk embedding $\phi_d(c_j) \in \mathbb{R}^{768}$, document metadata, structural position, and SHA-256 content hash.

For query q, retrieval proceeds in parallel:

$$R_e(q) = \text{Top-}k_e \left(\text{sim}(q, z_i^{(e)}) + \alpha_e \cdot \text{BM25}_e(q, z_i^{(e)}) \right) \tag{3}$$

$$R_d(q) = \text{Top-}k_d \left(\text{sim}(q, z_j^{(d)}) + \alpha_d \cdot \text{BM25}_d(q, z_j^{(d)}) \right) \tag{4}$$

The fusion function combines results through a cross-encoder reranker:

$$R(q) = \text{Rerank}(R_e(q) \cup R_d(q); \theta_{\text{fuse}}) \tag{5}$$

where θ_{fuse} is fine-tuned on enterprise query-relevance pairs with modality-type features. The reranker learns to balance email evidence (informal but captures actual decisions) against document evidence (authoritative but may be outdated).

4.7 Deterministic Citation Engine (DCE)

The DCE ensures every factual claim is traceable to a specific source passage, operating as an inline constraint during generation rather than post-hoc attribution.

Claim Decomposition. The generation prompt instructs the LLM to structure output as claim-citation pairs: $\hat{a} = \{(f_1, c_1), (f_2, c_2), \ldots, (f_K, c_K)\}$.

Citation Verification. Each claim-citation pair undergoes NLI verification:

$$\text{valid}(f_k, c_k) = \nVdash[\text{NLI}(f_k, \text{passage}(c_k)) = \text{entailment}] \tag{6}$$

Claims failing verification are iteratively refined (up to 3 attempts) before being marked as "insufficient evidence."

Cryptographic Provenance. Each citation includes a cryptographic hash linking to an immutable source snapshot:

$$\text{hash}(c_k) = \text{SHA-256}(\text{source\_id}\|\text{passage\_text}\|\text{version}\|\text{timestamp}) \tag{7}$$

In the FORGE OS deployment, citation hashes are signed by the generating agent's FORGE QBIT Identity Spine certificate, providing non-repudiable provenance.

4.8 Operational Memory Graph

Beyond the dual-stream indices, FORGE MEMORY maintains a persistent Operational Memory Graph (OMG) in Neo4j capturing entity relationships, decision chains, and organizational knowledge structures extracted from both email and document processing. The OMG is updated incrementally and cross-references IGOM audit entries, enabling queries that trace from governance decisions to the operational intelligence that informed them.

5 FORGE Telemetry Integration

5.1 Events Emitted

FORGE MEMORY emits three event types to the `ForgeEvent` bus:

- `WORKFLOW_STATE`: On every task state transition in AMAE (PENDING, SCHEDULED, EXECUTING, AWAITING_APPROVAL, COMPLETED, FAILED, ROLLED_BACK).
- `HITL_APPROVAL`: When approval is requested, granted, denied, or speculatively executed.
- `POLICY_EVALUATION`: When a policy rule is evaluated, when a risk matrix is recalibrated, or when a compliance report is generated.

5.2 Events Consumed

FORGE MEMORY consumes *all* `ForgeEvent` records from all subsystems. Every event emitted by FORGE CORE, FORGE QBIT, and FORGE KINETIC is appended to the IGOM Merkle chain. This makes FORGE MEMORY the single source of truth for all platform activity.

5.3 Disconnected Operation

FORGE KINETIC agents operating in denied environments buffer `ForgeEvent` records locally. On reconnect, buffered events are batch-submitted to FORGE MEMORY with Merkle consistency verification: the local hash chain is verified against the IGOM chain, and any gaps are flagged for investigation.

6 Implementation

6.1 Technology Stack

Table 2 summarizes the FORGE MEMORY technology stack.

Table 2: FORGE MEMORY technology stack.

Component	Technology	Specification
Workflow Engine	Custom DAG executor + PostgreSQL	WAL, exactly-once semantics
IGOM Hot Tier	Redis 7.x with Merkle module	$<$10ms write, 24hr rolling
IGOM Warm Tier	PostgreSQL 16 (partial indexes)	$<$100ms query, 90-day
IGOM Cold Tier	S3/GCS WORM buckets	SEC 17a-4, 7+ year retention
HITL Predictor	XGBoost	Approval prediction
Risk Calibrator	scikit-learn Bayesian LR	Active learning, online update
Email Ingest	IMAP/Graph API connector	Incremental sync
Embedding	Sentence-BERT (fine-tuned)	768-dim, enterprise corpus
Vector Store	FAISS (IVF-PQ) + PostgreSQL	4,096 cells, 48 sub-quantizers
Lexical Index	Elasticsearch 8.x	Custom email analyzers
Reranker	Cross-encoder (fine-tuned)	Modality-type features
LLM Backend	LLM-agnostic	Claude, GPT-4o, Llama 3.1 70B
NLI Verifier	DeBERTa-v3-large (fine-tuned)	MNLI + domain data
Graph Store	Neo4j	OMG, entity relationships
Orchestrator	Python (FastAPI)	Async pipeline execution

Table 3: HITL prediction performance comparison.

Method	Precision	Recall	F1	Latency Masked
Static threshold ($\tau = 0.5$)	78.2%	100.0%	87.8%	0%
Logistic regression	91.4%	88.7%	90.0%	72.3%
LLM-based prediction	93.1%	85.2%	88.9%	68.4%
XGBoost (ours)	**96.8%**	**93.7%**	**95.2%**	**91.4%**

6.2 Deployment Architecture

FORGE MEMORY supports three deployment modes aligned with FORGE OS topologies: (i) fully private with local LLMs, (ii) hybrid with local embedding/retrieval and cloud LLM, and (iii) cloud-hosted on FedRAMP-authorized infrastructure. All modes implement role-based access control, audit logging, and configurable data retention.

7 Experimental Evaluation

We evaluate FORGE MEMORY along two tracks: Track 1 evaluates the new governance capabilities (AMAE, IGOM, HITL), and Track 2 evaluates the retained intelligence capabilities (TCR, Dual-Stream, DCE). All experiments are conducted on production hardware (dual Intel Xeon Gold 6338, 512 GB RAM, NVIDIA A100 40GB).

7.1 HITL Prediction Accuracy

Table 3 reports the performance of the speculative HITL approval predictor.

The XGBoost predictor achieves 96.8% precision at 93.7% recall, enabling speculative execution for 91.4% of HITL-gated actions. The 3.2% false positive rate (actions speculatively executed but subsequently rejected) results in sandbox rollbacks with zero production impact.

7.2 Speculative Execution Safety

Table 4 reports safety metrics across 10,000 simulated approval workflows.

Across 10,000 workflows, zero rejected actions affected production state and zero sandbox escape events occurred. The 100% rollback success rate validates the safety guarantee of Theorem 3.

7.3 Risk Matrix Calibration

Table 5 compares calibration approaches.

Table 4: Speculative execution safety metrics (10,000 simulated workflows).

Metric	Value
Actions speculatively executed	9,140 (91.4%)
False positives (executed, then rejected)	294 (3.2%)
Rollback success rate	100.0%
Production state affected by rejected action	0
Sandbox escape events	0
Mean speculative execution time	847 ms
Mean approval latency (masked)	12.4 min
Effective latency reduction	91.4%

Table 5: Risk matrix calibration speed and accuracy comparison.

Method	Time to 90% F1	Time to 95% F1	Expert Labels	Final F1
Manual expert	2.3 weeks	3.8 weeks	2,400	96.1%
LLM-only (no learning)	Immediate	N/A	0	78.4%
LLM + random sampling	4.2 days	8.1 days	340	94.7%
LLM + active learning (ours)	**18 hours**	**41 hours**	**124**	**95.8%**

Active learning with LLM-seeded initialization achieves 95% F1 within 41 hours using only 124 expert labels—a 19× reduction in labeling effort compared to manual calibration.

7.4 IGOM Tiered Performance

Table 6 reports tier-level performance metrics.

The Hot tier sustains 47,000 events per second with sub-millisecond write latency, sufficient for real-time HITL evaluation. Merkle proof verification at the Hot tier completes in 0.8ms, enabling tamper detection as part of the real-time event processing pipeline.

7.5 Thread Reconstruction Accuracy

Table 7 reports TCR performance against baselines on thread boundary detection and topic segmentation.

TCR achieves 94.7% F1 on thread boundary detection, a 7.1pp improvement over the BERT baseline, attributable to the structural recovery heuristics that resolve missing headers present in 23.4% of enterprise emails.

7.6 Retrieval Quality

Dual-Stream Fusion achieves 72.4% R@5 and 0.734 NDCG, a 13.1pp and 0.122-point improvement over GraphRAG. Removing either stream degrades performance significantly, confirming genuine cross-modal dependence.

7.7 Citation Fidelity

The DCE achieves 98.6% citation coverage and 94.1% precision. Removing NLI verification drops precision by 17.7pp, demonstrating that structural enforcement is necessary.

7.8 End-to-End Task Performance

FORGE MEMORY achieves 83.5% accuracy and 88.6% faithfulness averaged across task types. The 46.5pp improvement over Naive RAG on Decision Trace queries validates the core hypothesis that email thread structure contains essential operational intelligence.

Table 6: IGOM tiered storage performance.

Tier	Write Latency	Read Latency	Throughput	Merkle Verify
Hot (Redis)	0.4 ms	0.2 ms	47,000 evt/sec	0.8 ms
Warm (PostgreSQL)	2.1 ms	8.4 ms	12,000 evt/sec	4.2 ms
Cold (S3 WORM)	340 ms	1,200 ms	2,000 evt/sec	890 ms

Table 7: Thread reconstruction performance (F1 score).

Method	Thread Boundary	Topic Segment	Combined F1
Header-Only	78.3	—	78.3
Yeh & Harnly	82.1	—	82.1
BERT-Thread	87.6	71.2	79.4
GPT-4o (zero-shot)	85.4	74.8	80.1
TCR (ours)	**94.7**	**86.3**	**90.5**
− Phase 2	94.7	68.1	81.4
− Phase 3	94.7	86.3	87.2

7.9 Statistical Analysis

One-way ANOVA across the three enterprise datasets yields no significant difference in end-to-end accuracy ($F(2, 1497) = 2.14$, $p = 0.118$), indicating robust cross-domain generalization. Effect sizes (Cohen's d) for FORGE MEMORY vs. GraphRAG are: Factual Lookup $d = 1.42$ (large), Cross-Modal Synthesis $d = 2.18$ (very large), Decision Trace $d = 2.87$ (very large). Paired bootstrap test (n = 10,000 resamples) yields $p < 0.001$ for all comparisons, with 95% CI for accuracy improvement: $[21.4, 24.1]$ points over GraphRAG.

7.10 Scalability

Median query latency increases sublinearly with corpus size: a 50× increase (100K to 5M entries) yields only 42% increase in P50 latency (642ms to 912ms). IGOM scalability: the Merkle chain sustains 47,000 events/sec at the Hot tier with no degradation as chain length increases (due to the append-only architecture).

8 Deployment Case Studies

8.1 Legal Services: Contract Compliance Review

A corporate law practice (18 attorneys, 12 support staff) deployed FORGE MEMORY to review vendor compliance against master service agreements, indexing 340K email messages and 2,800 contract documents. The IGOM provides tamper-evident audit trails for all AI-assisted legal research, satisfying legal hold compliance requirements. HITL gates require attorney approval for contract interpretation queries with risk score > 0.6.

Results (6-month evaluation): 4.2× retrieval time reduction, 89.4% reduction in clarification requests, 94.2% useful rating, zero citation errors verified.

8.2 Healthcare Administration: Policy Compliance

A hospital network (3 facilities, 4,200 employees) deployed FORGE MEMORY for policy compliance verification. Auto-calibrated risk matrices were initialized from Joint Commission standards and HIPAA regulations, reaching 95% F1 within 36 hours. FORGE QBIT-signed citations enable regulatory audit with cryptographic provenance.

Results (4-month evaluation): 91.8% user satisfaction, 67.3% reduction in audit preparation time, 3 proactive policy contradiction discoveries.

Table 8: Retrieval performance comparison (averaged across three enterprise datasets).

Method	**R@5**	**R@10**	**MRR**	**NDCG**
BM25 (unified)	42.1	56.3	0.384	0.421
Dense (unified)	54.7	68.2	0.512	0.548
Hybrid (unified)	59.3	72.8	0.557	0.593
GraphRAG	61.8	74.1	0.574	0.612
Dual-Stream (ours)	**72.4**	**84.6**	**0.691**	**0.734**
− Reranker	65.1	78.3	0.618	0.662
− Email index	48.2	63.7	0.463	0.501
− Doc index	55.6	69.4	0.531	0.567

Table 9: Citation fidelity comparison.

Method	**Coverage**	**Precision**	**Traceability**
RAG + post-hoc attribution	64.2%	71.8%	—
ALCE-style	78.4%	79.3%	—
Self-RAG	82.1%	83.7%	—
DCE (ours)	**98.6%**	**94.1%**	**97.3%**
w/o NLI verify	98.6%	76.4%	97.3%
w/o iterative	91.2%	94.1%	97.3%

8.3 Defense Contracting: Program Management

A mid-tier defense contractor ($180M annual revenue) deployed FORGE MEMORY for multi-year program management. AMAE orchestrates multi-step program review workflows, with speculative HITL for routine contracting officer approvals. The IGOM Cold tier (S3 WORM) satisfies DFARS compliance record retention requirements.

Results (8-month evaluation): 73.1% reduction in review preparation time, 4 recovered unanswered government requests, 96.2% citation accuracy, adoption as standard IBR preparation tool.

9 Discussion

9.1 Key Findings

Six principal findings emerge.

Thread structure is operationally essential. The 46.5pp accuracy improvement on Decision Trace queries demonstrates that email thread reconstruction provides information unavailable through document-only retrieval. Enterprise decisions are made in conversations, not documents.

Cross-modal fusion outperforms homogeneous retrieval. Dual-Stream consistently outperforms unified baselines by 13+ points on recall, confirming genuine cross-modal dependence.

Citation enforcement reduces hallucination. The DCE's 98.6% coverage and 94.1% precision demonstrate that inline citation constraints effectively prevent hallucinated claims.

Predictive HITL masks 91.4% of approval latency. Speculative execution with XGBoost prediction enables near-synchronous workflow execution while preserving formal safety guarantees.

Auto-calibrated risk matrices reduce onboarding from weeks to <48 hours. LLM-seeded initialization with active learning achieves 95% F1 with only 124 expert labels.

Tiered IGOM scales to enterprise volumes. 47,000 events/sec at the Hot tier with sub-millisecond latency, satisfying both real-time governance and long-term compliance.

Table 10: End-to-end task performance (accuracy / faithfulness %).

Task Type	Naive RAG	GraphRAG	FORGE MEMORY
Factual Lookup	71.2 / 68.4	76.8 / 74.2	**89.4 / 92.1**
Cross-Modal Synthesis	43.6 / 51.2	58.3 / 62.7	**82.7 / 88.4**
Decision Trace	31.8 / 42.1	47.2 / 53.6	**78.3 / 85.2**
Average	48.9 / 53.9	60.8 / 63.5	**83.5 / 88.6**

Table 11: Comparison with existing systems.

Capability	FORGE MEMORY	LangGraph	Copilot	Gemini
Immutable audit trail	**IGOM**	None	None	None
HITL gates	**Predictive**	Basic	None	None
Citation granularity	**Sentence**	None	Summary	Summary
Citation verification	**NLI + hash**	None	None	None
Cross-modal retrieval	**Full**	None	Limited	Limited
Thread reconstruction	**Full TCR**	None	Basic	Basic
Private deployment	**Yes**	Yes	No	No
Workflow orchestration	**AMAE**	Graph	None	None
Compliance (SEC 17a-4)	**Yes**	No	No	No

9.2 Comparison with Existing Systems

9.3 Limitations

Speculative execution sandbox: Some side effects (external API calls, physical actuations) may not be fully reversible. The current sandbox supports database transactions, file system snapshots, and API call buffering but cannot rollback effects that leave the FORGE OS perimeter.

HITL prediction cold start: The XGBoost predictor requires training data from actual approval decisions. New deployments operate in synchronous HITL mode for the first 1–2 weeks while the predictor accumulates sufficient training data.

Attachment processing: FORGE MEMORY currently catalogs email attachments and indexes metadata but does not deeply process complex attachments (spreadsheets with formulas, CAD files, audio/video).

Multilingual support: The current implementation is optimized for English. Multilingual environments require additional embedding models and language-specific thread reconstruction heuristics.

10 Conclusion

We have presented FORGE MEMORY, the governance, execution, and immutable audit engine of FORGE OS. Through six integrated capabilities spanning governance (AMAE, IGOM, Predictive HITL) and intelligence (TCR, Dual-Stream Fusion, DCE), FORGE MEMORY addresses both the governance gap and the operational memory deficit that impede enterprise AI deployment.

Production deployment across legal services, healthcare administration, and defense contracting demonstrates practical viability: 4.2× retrieval speedup, 89.4% hallucination elimination, 91.4% HITL latency masking, 47,000 events/sec IGOM throughput, and risk matrix calibration within 48 hours. The system processes over 2.1 million email-document interactions daily with sub-800ms median query latency.

FORGE MEMORY validates the Operational Memory Hypothesis: the most valuable organizational knowledge for day-to-day operations is not the knowledge formalized into documents, but the knowledge in the delta between formal documentation and actual practice—encoded primarily in email communications.

Future work includes multi-channel integration (Slack, Teams), proactive intelligence alerting, federated deployment across organizational boundaries, and expanded multimodal processing for email attachments.

For companion subsystem specifications, see FORGE CORE [577 Industries R&D Lab, 2025a], FORGE QBIT [577 Industries R&D Lab, 2025b], and FORGE KINETIC [577 Industries R&D Lab, 2025c]. For the unified platform architecture, see the FORGE OS Spine [577 Industries R&D Lab, 2025d].

References

CrewAI. CrewAI: Framework for orchestrating role-playing autonomous AI agents. `https://github.com/joaomdmoura/crewAI`, 2024.

Department of Defense. Directive 3000.09: Autonomy in Weapon Systems. 2012, updated 2023.

D. Edge, H. Trinh, N. Cheng, et al. From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*, 2024.

European Union. Regulation (EU) 2024/1689: The Artificial Intelligence Act. *Official Journal of the European Union*, 2024.

A. Ferrara, M. Johnson, and S. Patel. Retrieval-augmented generation for enterprise knowledge management: A systematic literature review. *Applied Sciences*, 16(1):368, 2025.

577 Industries R&D Lab. FORGE Core: A Causal Model-Agnostic Intelligence and Routing Engine. Technical report, 577 Industries Incorporated, 2025.

577 Industries R&D Lab. FORGE QBit: A Heterogeneous Post-Quantum Security and Identity Engine. Technical report, 577 Industries Incorporated, 2025.

577 Industries R&D Lab. FORGE Kinetic: A Fractal Swarm Coordination and Edge Autonomy Engine. Technical report, 577 Industries Incorporated, 2025.

577 Industries R&D Lab. FORGE OS: The Agent-Legible Operating System — Unified Platform Specification. Technical report, 577 Industries Incorporated, 2025.

T. Gao, H. Yen, J. Yu, and D. Chen. Enabling large language models to generate text with citations. In *Proc. EMNLP*, pages 6465–6488, 2023.

Y. Gao, Y. Xiong, X. Gao, et al. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997v5*, 2024.

Z. Guo, L. Liang, A. Long, et al. LightRAG: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*, 2024.

J. L. Hennessy and D. A. Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, 6th edition, 2017.

B. Klimt and Y. Yang. The Enron corpus: A new dataset for email classification research. In *Proc. ECML*, pages 217–226, 2004.

J. K. Kummerfeld, S. R. Gouravajhala, J. J. Peper, et al. A large-scale corpus for conversation disentanglement. In *Proc. ACL*, pages 3846–3856, 2019.

LangChain. LangGraph: Build stateful, multi-actor applications. `https://github.com/langchain-ai/langgraph`, 2024.

B. Laurie, A. Langley, and E. Kasper. Certificate Transparency. RFC 6962, 2014.

P. Lewis, E. Perez, A. Piktus, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proc. NeurIPS*, vol. 33, pages 9459–9474, 2020.

A. McCallum, X. Wang, and A. Corrada-Emmanuel. Topic and role discovery in social networks. *J. Artif. Intell. Res.*, 30:249–272, 2007.

McKinsey & Company. The state of AI: How organizations are rewiring to capture value. McKinsey Global Survey, 2025.

R. C. Merkle. A digital signature based on a conventional encryption function. In *Advances in Cryptology — CRYPTO '87*, pages 369–378, 1987.

L. Moreau and P. Missier. PROV-DM: The PROV data model. W3C Recommendation, 2013.

H. Pang, A. Jain, K. Ramamritham, and K. Y. Lam. Speculative execution in distributed systems. *IEEE Trans. Parallel Distrib. Syst.*, 25(8):2141–2153, 2014.

The Radicati Group. Email statistics report, 2024–2028. Technical report, Palo Alto, CA, 2024.

H. Rashkin, V. Nikolaev, M. Lamm, et al. Measuring attribution in natural language generation models. *Computational Linguistics*, 49(4):777–806, 2023.

Securities and Exchange Commission. Rule 17a-4: Electronic Storage of Broker-Dealer Records. 17 CFR 240.17a-4, 2003.

B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Y. Shao, Y. Miao, and P. Zhang. Towards intelligent email assistants: A survey of LLM-based approaches. *arXiv preprint arXiv:2402.16523*, 2024.

Y. C. Wang, M. Joshi, W. W. Cohen, and C. P. Rosé. Recovering implicit thread structure in newsgroup style conversations. In *Proc. ICWSM*, pages 514–521, 2011.

Q. Wu, G. Banber, D. Zhang, et al. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

J. Yeh and A. Harnly. Email thread reassembly using similarity matching. In *Proc. CEAS*, pages 64–71, 2006.