

Independent Research Brief

Governed Operational Memory and Intelligence Systems

SYNTHETIC / PUBLIC / INDEPENDENTLY REWRITTEN

Purpose

This brief describes a general architecture for an operational memory and intelligence system. The system helps organizations capture operational incidents, preserve verified resolutions, reason over evidence, recommend next steps, verify outcomes, and reuse trustworthy knowledge. It is designed as decision support, not an autonomous operator.

Scope and evidence status

This is an independent research brief containing generalized concepts, proposed hypotheses, evaluation methods, and human-control principles. It contains no source code, private prompts, customer data, operational records, credentials, repository details, or deployment information. The concepts below are research directions and do not establish production readiness, legal compliance, or safety certification.

1. Core operational model

A useful operational learning loop is:

Signal	Incident	Evidence	Reasoning	Human decision
Action	Outcome	Verification	Reusable memory	Learning

The following record types should remain distinct: raw observation, evidence, hypothesis, recommendation, human decision, corrective action, outcome, verification, and operational memory.

2. Reference architecture

A modular implementation can be represented as:

Business source systems -> purpose-built connectors -> canonical event contracts -> validation and quarantine -> operational domain store -> evidence and memory indexes -> integrity and sensitive-data controls -> critical reasoning -> human-reviewed recommendation -> action and verification -> memory lifecycle management

The initial implementation should use a modular monolith with clear interfaces. Independent services should be introduced only when scale or ownership boundaries justify their additional complexity.

3. Operational domain

A durable domain model may include organizations, sites, areas, assets, users, roles, source systems, connectors, incidents, observations, evidence, hypotheses, reasoning steps, recommendations, decisions, corrective actions, outcomes, verifications, operational memories, audit events, integrity events, connector runs, and quarantine records.

- State transitions are explicit and server-validated.
- Tenant and site scope is retained on owned records.
- User decisions include actor, role, timestamp, selected option, and rationale.
- Audit history is append-only from ordinary application roles.
- Indexes are projections; the domain store remains authoritative.

4. System-integrity layer

The integrity or antibody layer protects both the running system and the process used to build it.

Runtime detection

- Schema drift and malformed evidence
- Invalid workflow transitions
- Sensitive-data leakage and cross-organization access
- Stale, contradictory, or poisoned memory
- Unsafe model output and missing citations
- Unauthorized actions and inconsistent release artifacts

Build-process detection

- Unregistered or overlapping work
- Edits outside declared scope
- Missing checkpoints or incomplete handoffs
- Unreviewed dependency changes
- Commits that cannot be traced to a work package
- Changes that weaken safety or authorization controls

The integrity layer may detect, classify, quarantine, preserve evidence, and recommend a bounded repair. It should not silently delete, reset, merge, push, deploy, or mutate production systems.

5. Provenance, drift, and quarantine

Important facts should retain source, source-record identifier, source timestamp, ingestion timestamp, transformation version, actor or service identity, evidence reference, validation state, and confidence or quality status.

Integrity artifacts should use canonical serialization, bounded payloads, duplicate-key rejection, stable ordering, recomputable digests, and version-bound validators. Caller-provided status labels or hashes should not be trusted without recomputation.

State	Meaning	Response
no_drift	Baseline and candidate are equivalent	Stable observation only
evidence_drift	Evidence changed; semantic case set is stable	Preserve and manually review
classification_drift	A case meaning or classification changed	Quarantine and fresh review
case_set_drift	Cases were added or removed	Quarantine and fresh review

Ambiguous, malformed, stale, contradictory, unauthorized, or provenance-incomplete material should fail closed and enter quarantine.

6. Critical-reasoning layer

The critical-reasoning layer is intended to reduce premature conclusions, confirmation bias, overconfidence, and drift in AI-assisted decisions.

- Observations and confirmed facts
- Assumptions and hypotheses
- Supporting evidence and counter-evidence
- Missing evidence and competing explanations
- Falsification tests
- Possible actions, no-action, and escalation
- Confidence and risk if wrong
- Verification plan

Valid outputs include insufficient evidence, conflicting evidence, unknown cause, investigation required, escalation required, and no safe recommendation. The system must be able to decline a recommendation.

7. AI boundary

The model or provider is an untrusted, replaceable component. It may classify incidents, extract structured observations, suggest hypotheses, retrieve similar cases, identify missing evidence, draft recommendations, and propose verification steps.

- Provider output cannot set policy authority.
- Provider output cannot approve actions or promote memory.
- Provider output cannot modify production systems.
- Provider input must be a minimum-necessary, allowlisted context.
- A deterministic validator converts a draft into a structured reasoning artifact.

8. Operational memory and retrieval

Operational memory should preserve verified experiences rather than arbitrary document fragments. A reusable case can contain symptom, context, confirmed cause, corrective action, outcome, verification, applicability, evidence references, review date, expiration or revalidation, supersession, and known failure modes.

Memory lifecycle

Candidate -> Reviewed -> Approved -> Active -> Revalidation required -> Superseded -> Retired

Retrieval should combine organization/site/asset filters, structured matching, full-text search, semantic similarity, verification and freshness ranking, contradiction filtering, and applicability checks. Failed hypotheses and unsuccessful corrective actions should remain searchable as negative knowledge.

9. Data separation and authorization

Use separate data lanes for operational data, identity or customer data, and restricted data such as credentials, secrets, payment information, health information, or government identifiers.

The context builder must create a minimum-necessary allowlisted projection. Unknown fields fail closed. Authorization should be server-derived through an organization -> site -> area -> asset -> incident hierarchy.

10. Connector architecture

Purpose-built connectors should initially be read-oriented and support schema mapping, incremental synchronization, checkpointing, idempotency, retries, quarantine, rate limits, reconciliation, source links, version tracking, and health reporting.

Canonical events retain a source identifier, source timestamp, received timestamp, organization scope, schema version, correlation identifier, actor or service identity, and bounded payload. Replay must not duplicate incidents or actions.

11. Modularity and internationalization

New capabilities should use versioned interfaces with explicit inputs, outputs, errors, permissions, events, dependencies, independent tests, safe disablement, and observability. Modules communicate through contracts and events rather than private state.

Worldwide deployment requires locale-aware dates, times, numbers, currencies, pluralization, validation messages, accessible labels, and notifications. English may be the canonical source locale, with Brazilian Portuguese as an additional locale. Domain codes, permissions, API values, and audit events remain language-neutral. Localization must not create product forks.

12. Evaluation program

A permanent evaluation laboratory should include known-good, ambiguous, contradictory, unsafe-action, stale-memory, sensitive-data, failed-repair, cross-tenant, duplicate-connector, prompt-injection, multilingual, and regression cases.

Area	Example measures
Retrieval	Correct-case recall, ranking quality, citation accuracy, stale-memory suppression
Reasoning	Fact/inference separation, missing-evidence detection, counter-evidence handling
Safety	Unsafe recommendation rate, leakage prevention, cross-scope denial
Workflow	Time to find prior resolution, repeat incidents, verification completion

13. Human control and construction discipline

Human judgment remains mandatory for high-impact recommendations, safety-sensitive actions, contradictory evidence, uncertain diagnosis, corrective-action approval, memory promotion, customer-impacting changes, security incidents, and release authorization.

Before each implementation step, prepare a bounded research brief covering the decision question, authoritative sources, applicability, facts versus hypotheses, alternatives, risks, acceptance criteria, scope exclusions, and authorization. Each work package also requires a checkpoint, validation record, handoff, and traceable commit.

14. Research boundaries

The following are deferred until the foundations are proven:

- Autonomous remediation
- Equipment control or autonomous dispatch
- Broad knowledge graphs
- Fine-tuning
- Dozens of connectors
- Predictive-maintenance claims
- Customer-specific code forks
- Multi-agent autonomy
- Advanced forecasting
- Compliance or security certifications without evidence

Current status

This brief is a generalized research artifact. It does not prove semantic AI quality, provider safety, customer safety, production readiness, performance at scale, connector reliability, disaster recovery, legal compliance, security certification, or autonomous repair.