SALESFUSION

Company Memory Is Not a Folder

Technical architecture and implementation report

PUBLIC RELEASE · VERSION 1.0.0 · 2026-07-17

Company Memory Is Not a Folder

Publication metadata

Field	Value
Author	SalesFusion Research
Publication date	2026-07-17
Version	1.0.0
Canonical slug	company-memory-is-not-a-folder
Canonical path	/technical-papers/company-memory-is-not-a-folder
Publication class	Technical architecture and implementation report
Approval state	Founder-approved public release

Revenue Cortex connection

Revenue Cortex depends on Company Memory because growth work cannot improve from stale folders or unsupported summaries. Buyer intelligence, positioning, outbound decisions, sales-call preparation, and pipeline review all require source-backed memory with provenance, freshness, conflict handling, and authority boundaries.

Evidence-status box

> This paper describes an operational pattern SalesFusion uses and proposes a broader Company Memory model. The pattern separates source-backed assertions, provenance, state, correction, and review from ordinary document storage. The broader model still requires benchmark evaluation for retrieval quality, conflict handling, staleness exclusion, and correction success. The allowed claim is architectural and methodological: useful organizational memory requires provenance, authority, freshness, conflict, uncertainty, and supersession semantics in addition to document retrieval. This paper presents no measured effect on revenue, decision quality, or user productivity.

Abstract

Organizational memory is often treated as a folder of documents plus search. That model fails when an agent or operator must answer operational questions under uncertainty: Which source is authoritative? When was the statement true? Is a newer source a correction, a conflict, or a supersession? Is the fact restricted? Is the

answer inferred, asserted, disputed, stale, or unknown? We define Company Memory as a source-governed memory architecture for business AI systems. It stores sources, assertions, entities, relationships, versions, epistemic state, provenance, restrictions, conflicts, corrections, and effective time. Search and vector indexes are treated as projections, not sources of truth. Obsidian or a similar note workspace is a readable projection and correction interface, not the canonical runtime. The model supports retrieval that returns evidence and state, not just text. It also defines failure modes and evaluation tests that must be passed before generated analysis can be trusted as an input to governed action.

The problem

A folder answers one question: where is the file? A business agent needs different questions answered. What does the company currently believe about an account, process, constraint, policy, metric, or workflow? Which statements are source-backed? Which statement was true during a prior time window? Which statement is restricted from public output? Which source has higher authority? Which assertions disagree? Which answer should be withheld because the evidence is stale or unresolved?

Document retrieval alone cannot represent these distinctions. A current policy, a deprecated draft, a generated summary, and an operator note may all rank highly for the same query. Without state semantics, the system can launder weak evidence into confident prose. Without effective time, it can use an old truth as a current fact. Without supersession, it can show a newer statement but leave the prior one ambiguously alive. Without restrictions, it can route sensitive material into an inappropriate output. Without correction records, it can hide the path by which the memory changed.

For consequential workflows, these failures matter. A recommendation based on stale authority should be blocked. A proposed action based on a disputed assertion should be escalated. A generated inference should remain an inference until accepted through the correct policy. The memory layer must therefore preserve the state of knowledge, not merely the text that produced it.

Contributions

This paper makes five contributions.

First, it defines Company Memory as a structured, event-backed memory layer with explicit epistemic states.

Second, it specifies provenance, effective time, restriction, correction, conflict, supersession, and retrieval semantics.

Third, it separates the canonical runtime from search and Obsidian projections.

Fourth, it gives concrete data objects, state transitions, invariants, and decision rules for memory operations.

Fifth, it defines an evaluation program for memory quality that does not require unsupported business-outcome claims.

Core definitions

Source is retained evidence with owner, tenant, classification, effective time, review state, version, and restrictions.

Assertion is a structured statement about a subject, predicate, and value. It references one or more sources and carries epistemic state.

Epistemic state is the memory's status label for a statement. The model separates epistemic status from lifecycle state. Epistemic status distinguishes observed_fact, authorized_assertion, inference, hypothesis, estimate, and unknown. Lifecycle state distinguishes active, stale, disputed, superseded, and retracted; a proposed record remains pending until validation and authority checks pass.

Effective time is the time interval during which a source or assertion is claimed to apply. It is separate from record creation time.

Supersession means a newer assertion replaces the current operational use of a prior assertion while preserving the prior assertion for audit and historical reconstruction.

Correction is a new event and usually a new assertion version. It does not erase the old record.

Restriction is a rule attached to a source, assertion, or field that limits retrieval, output, use, export, or audience.

Projection is a derived view such as search, vector index, report, dashboard, or Obsidian note. Projections can be rebuilt from canonical state.

Memory model and state protocol

The canonical memory has four record families: sources, assertions, entities and relationships, and events. A minimal assertion record is:

```
{
  "assertion_id": "asrt_active_constraint_001",
  "tenant_id": "tenant_demo",
  "subject_ref": "entity_revenue_motion",
  "predicate": "has_current_constraint",
  "value": "operator approval required for external actions",
  "epistemic_status": "authorized_assertion",
  "lifecycle_state": "active",
  "source_refs": ["src_authority_matrix_v1"],
  "authority_rank": "approved_source",
  "effective_from": "2026-07-01",
  "effective_to": null,
  "review_due": "2026-08-01",
  "restriction_refs": [],
  "supersedes": [],
  "conflict_set_ref": null,
  "version": 3
}
```

The assertion state machine is:

```
proposed -> active -> stale -> superseded
             |-> disputed -> active | superseded | retracted
             |-> retracted
```

Generated analysis may create a proposed assertion or an inferred assertion, depending on policy. It cannot silently create an active company fact. Activation requires source sufficiency, authority, restriction checks, and review state. A corrected source creates a new event and may create a new assertion version. A retracted source marks dependent assertions as stale, disputed, or retracted according to policy; it does not delete history.

Conflict is first-class. Two assertions can share subject and predicate while disagreeing on value, effective time, source authority, or restriction. The memory opens a conflict set rather than choosing last-write-wins. Retrieval must surface required unresolved conflicts when the objective depends on the predicate.

Supersession is also first-class. If a newer operating policy replaces an older one, the older assertion becomes superseded for current-state retrieval but remains available for historical questions. The system can reconstruct what it believed on a prior date by replaying effective time and event time.

Provenance, restrictions, and retrieval

W3C PROV-O provides useful vocabulary for entities, activities, agents, derivation, and attribution. The Company Memory model uses the same general provenance discipline: an assertion should be traceable to a source and to the activity that produced or changed it. The citation supports provenance vocabulary, not any claim that this implementation produces better outcomes.

Retrieval is not "top k chunks." A memory retrieval request specifies objective, effective time, minimum freshness, allowed classifications, required predicates, tenant, actor, and output use. The response contains:

Field	Meaning
active assertions	source-backed statements eligible for the request
inferences	generated or derived statements labeled as such
conflicts	unresolved disagreements relevant to the request
unknowns	required predicates with insufficient evidence
stale exclusions	records excluded because freshness requirements failed
restricted exclusions	records withheld because use or audience is not allowed
provenance	source refs, version refs, event refs, and locators
digest	integrity summary of the snapshot

Decision rules:

```
if requested_tenant != resource.tenant:
    deny("TENANT_SCOPE_DENIED")
if source_or_assertion.is_restricted_for(output_use):
    exclude_and_report("RESTRICTED")
if required_predicate.has_unresolved_conflict:
    return_state("CONFLICT_UNRESOLVED")
if required_predicate.only_has_stale_assertion:
    return_state("SOURCE_INSUFFICIENT")
if generated_inference_requested_as_fact:
    return_state("UNSUPPORTED_ASSERTION")
```

These rules preserve the difference between absence, restriction, conflict, staleness, and inference. A generated answer may still be useful, but it must carry its memory state.

Obsidian projection

The Obsidian projection exists because operators need a readable, editable surface. It can show source notes, entity pages, assertion ledgers, conflict pages, decision packets, run summaries, and learning records. The projection can also collect human corrections.

The projection does not own canonical truth. A direct note edit creates an import proposal with a diff, author, timestamp, target records, and rationale. The import proposal then passes validation:

Check	Purpose
schema validity	prevent malformed canonical records
source link	prevent unsupported fact creation
version match	prevent overwriting newer state
authority check	prevent unauthorized correction
restriction check	prevent inappropriate disclosure
conflict handling	prevent silent erasure

If approved, the canonical store emits events and the projection is regenerated. This keeps Obsidian useful without letting a local edit bypass memory governance.

Implementation

SalesFusion has implemented this pattern at the operational-design level: source-backed records, canonical state, correction surfaces, and redundancy controls are separated from free-form notes. The broader Company Memory architecture can be implemented with a relational canonical store, an append-only event log, retained source artifacts, and disposable search projections.

Core endpoints include:

```
POST /v1/sources
POST /v1/sources/{source_id}/versions
POST /v1/assertions
POST /v1/assertions/{assertion_id}/correct
POST /v1/assertions/{assertion_id}/dispute
POST /v1/assertions/{assertion_id}/supersede
GET  /v1/state/stale
GET  /v1/state/conflicts
POST /v1/state/snapshot
POST /v1/projections/obsidian
POST /v1/projections/obsidian/import-proposals
```

Mutating requests require idempotency keys and optimistic version checks. A key reused with a different request body is rejected. A correction never deletes the old assertion; it writes a new event and changes retrieval eligibility. Search indexes and vector indexes are rebuilt from canonical records and may be discarded.

The most important implementation invariant is this: every current-state value must be explainable as a source-backed assertion, an accepted inference, an unknown, a conflict, or an excluded restricted/stale record. Anything else is not Company Memory; it is unsupported generated text.

Evaluation

Memory evaluation should be independent of business outcomes. A benchmark corpus can include one authoritative fact, two consistent sources, contradictory sources with different authority, stale and fresh sources, inference misrepresented as fact, unsupported generated assertion, retracted source, corrected source, restricted source requested for public output, and source text containing prompt injection.

Metrics include source precision, current-state accuracy, conflict recall, stale-record exclusion, unsupported-assertion rate, uncertainty calibration, correction success, provenance completeness, and restriction compliance. The system should also be tested for cross-tenant retrieval. In pre-installation safety tests, cross-tenant leakage should be zero.

Acceptance criteria:

Criterion	Required behavior
stale exclusion	stale assertion cannot satisfy a freshness requirement
conflict visibility	unresolved required conflict blocks or escalates
correction trace	prior record remains auditable
restriction enforcement	restricted material is excluded from disallowed output
inference label	generated inference is not returned as fact
source trace	every active assertion has source and event references

These tests can show architecture correctness. They cannot show that the memory improves business performance without a separate field evaluation.

Safety, risks, and failure modes

The main memory risks are memory poisoning, prompt injection from source content, silent staleness, conflict erasure, retrieval laundering, restriction leakage, false confidence, and projection bypass. OWASP's LLM application threat list is relevant background for prompt injection and unsafe tool behavior, while NIST AI RMF supports a risk-management framing. Neither reference is evidence of SalesFusion-specific safety.

Controls include treating source text as data, preserving provenance, requiring activation policy, using conflict sets, excluding stale records by default from current-state retrieval, labeling inference, restricting output by audience and use, auditing projection imports, and preserving append-only event history. Zero-trust reasoning from NIST SP 800-207 supports resource-scoped checks, especially tenant isolation and least privilege.

Failure should be explicit. If the system cannot answer, it should return unknown, source_insufficient, conflict_unresolved, or restricted, not invent an answer. This is a product requirement, not a writing preference.

Limitations

The model adds operational overhead. Teams must maintain source ownership, review cadence, authority ranking, restrictions, and correction workflows. Some organizations have poor source hygiene; the memory layer will expose that problem rather than solve it automatically. The model also does not eliminate judgment. A high-authority source can be wrong, a correction can be incomplete, and a conflict can require human interpretation.

This paper does not provide a benchmark result. It specifies the benchmark that should be run. It also does not claim that Obsidian is the best projection for every organization; the architectural requirement is a readable projection with proposal-based import, not a specific editor.

Practical implications

A team building business agents should begin with memory semantics before building broad action. The minimum viable memory is not a document upload. It is a source registry, assertion ledger, conflict set, staleness rule, restriction model, correction path, and state snapshot endpoint.

The operating implication is also direct: when a human corrects the system, the correction must become structured memory, not a chat message trapped in a transcript. When an agent uses memory, it must carry source state forward into the decision packet. When a note is edited, it must become a proposal until validated.

References used

This paper uses W3C PROV-O for provenance vocabulary, NIST AI Risk Management Framework for risk framing, NIST SP 800-207 for resource-scoped access reasoning, and OWASP Top 10 for Large Language

Model Applications for threat awareness. These references do not validate SalesFusion-specific retrieval or business outcomes.

Conclusion

Company Memory is not a folder because company knowledge is not just text. It is sourced, time-bound, restricted, disputed, corrected, superseded, and sometimes unknown. A business AI system that cannot represent those states will eventually confuse retrieval with truth. The SalesFusion pattern separates canonical state from human-readable projection and generated analysis. The broader model should now be tested with retrieval, conflict, staleness, correction, restriction, and tenant-isolation benchmarks before stronger claims are made.

Changelog

- 2026-07-17 — Version 1.0.0: founder-approved first public release; publication metadata, evidence-status box, limitations, and verified references retained.