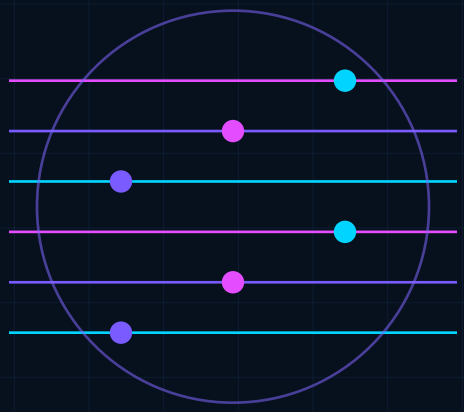
LOCAL-FIRST DATA AND SYNCHRONIZATION

Local-First SQLite Event Sourcing & CRDT Sync

Current-source review, evidence-bounded adoption decision, explicit limitations, reproducibility controls, and preserved source research note.



PERMANENT ID	EVIDENCE	ADOPTION	FRESHNESS
RES-032	A-	ADOPT AS FOUNDATIONAL	WATCH

Freshness review: 2026-09-17 / Next scheduled review: 2027-01-15

RESEARCH CONTROL SURFACE

Required Research Fields

Each field remains independently reviewable; evidence strength does not imply production authority.

EVIDENCE GRADE — **A-**

Strength of the retained source set and technical basis.

SOURCE AUTHORITY — **3 registered authorities**

SQLite Write-Ahead Logging; Automerge Documentation; Local-First Software: You Own Your Data, in Spite of the Cloud

FRESHNESS — **WATCH**

Reviewed 2026-09-17; dynamic sources require event-driven revalidation.

CONFLICTS — **VISIBLE / NOT SILENT**

Differences between specification, vendor behavior, research claims, and reproduced evidence must remain explicit.

LIMITATIONS — **EXPLICIT**

No certification, procurement approval, legal conclusion, or unbounded production claim is created by this report.

REPRODUCIBILITY — **REQUIRED**

Material claims require exact versions, representative data, failure cases, artifacts, and repeatable acceptance criteria.

ADOPTION POSTURE — **ADOPT AS FOUNDATION**

Adopt local authoritative state, append-only events, explicit conflict semantics, and CRDTs only where their merge model matches the domain. SQLite is a strong local substrate, not a universal distributed database.

REVIEW TRIGGER — **2027-01-15**

Scheduled at 120 days; earlier on material source, vulnerability, implementation, or contradictory-evidence change.

SOURCE AUTHORITY / FRESHNESS

Authority Register and Freshness Delta

Primary-source lineage is preserved; current-source changes are separated from unperformed implementation claims.

AUTHORITY 01

SQLite Write-Ahead Logging

https://sqlite.org/wal.html

AUTHORITY 02

Automerge Documentation

https://automerge.org/docs/

AUTHORITY 03

Local-First Software: You Own Your Data, in Spite of the Cloud

https://www.inkandswitch.com/local-first/

2026-09-17 FRESHNESS DELTA

Primary-source register retained and freshness controls advanced to the current review date. No new product or laboratory performance claim is introduced without reproduced evidence.

Freshness policy: scheduled review + immediate review on source supersession, material release, vulnerability, retraction, or contradictory evidence.

Evidence Must Survive Contact With Reality

- Source authority is not implementation evidence. Product behavior, scale, security, interoperability, and operator fit remain environment-specific.
- Conflicting evidence is recorded rather than averaged away. Specification behavior, vendor implementation, preprint results, and production observations are separate evidence classes.
- Quantitative claims are informative until reproduced under documented hardware, software, data, topology, load, and failure conditions.
- Adoption posture is reversible. A stronger source set can coexist with a pilot-only or research-only recommendation when maturity, safety, economics, or operability remain unresolved.

ADOPTION / VALIDATION / REVIEW

Decision Gate and Validation Plan

ADOPTION POSTURE

ADOPT AS FOUNDATION

Adopt local authoritative state, append-only events, explicit conflict semantics, and CRDTs only where their merge model matches the domain. SQLite is a strong local substrate, not a universal distributed database.

1 / SOURCE	Pin exact versions, publication state, retrieved artifacts, and source hashes.
2 / PROTOTYPE	Demonstrate the core mechanism with representative data and instrumented execution.
3 / FAILURE	Exercise malformed input, dependency loss, stale state, overload, recovery, rollback, and misuse.
4 / BASELINE	Compare against an incumbent, classical, or simpler architecture using the same workload.
5 / ACCEPT	Record acceptance criteria, residual limitations, owner, expiration, and revalidation triggers.

EXPIRATION / REVIEW TRIGGER

Next scheduled review: 2027-01-15 (120-day cadence). Review sooner after material source revision, breaking implementation release, critical vulnerability, retraction, benchmark contradiction, changed workload, or changed legal/safety boundary.

8. Complete Source Research Note

SOURCE-LINEAGE NOTICE

The following material preserves the complete original source note, excluding only the conversational wrapper and outer Markdown fence. Legacy status labels and absolute language are retained for traceability and do not override the candidate status, limitations, or adoption decision in this edition.

Document ID: RES-032 Version: 1.0.0 (Definitive Registry Edition) Status: Published Research Note Classification: Public Organization: Mythos Systems (MYTHOS AI) Owner: Aaron Stovall Effective Date: 2026-07-16 Applies To: Distributed systems engineers, AI engineers building cross-platform persistent memory (Mnemosyne), and platform architects designing local-first edge synchronization pipelines Companion Standards: MYTHOS-DAT-0001 (Vector Database & Local-First Sync), MYTHOS-DAT-0004 (Event Sourcing & Durable State), MYTHOS-KNW-0002 (Persona, Avatar & Persistent Memory), MYTHOS-WEB-0001 (WebGPU & WebLLM)

The architecture of AI state management has failed. Autonomous agents and their memory graphs are tethered to centralized cloud databases. When the network drops, the agent loses its mind—incapable of recalling user preferences, project context, or its own persona. Furthermore, when a user interacts with an agent on their phone while offline, and later opens the desktop client, the system attempts to synchronize state using "Last-Write-Wins" (LWW) timestamp logic. This mathematically guarantees data loss and corrupted memory graphs.

The Mythos Architecture mandates absolute cognitive continuity. We achieve this by decoupling the agent's memory from the cloud using Local-First Event Sourcing and CRDTs (Conflict-free Replicated Data Types).

This research note defines the architectural dynamics of Local-First Sync. It dissects the mechanics of utilizing SQLite as a zero-latency, append-only local event store, the application of operation-based CRDTs for semantic state merging, and the asynchronous background sync pipelines that reconcile state across the enterprise. Understanding these dynamics is a prerequisite for building the zero-amnesia, cross-platform PANTHEON runtime mandated by the Mythos Knowledge Standards.

To achieve zero-amnesia, we must move from state-centric database mutations to distributed, append-only event logs.

1.1 SQLite as the Edge Event Store

SQLite is the most deployed database engine in the world, yet it is often dismissed for enterprise state management. In a local-first architecture, SQLite is the absolute source of truth for the edge device.

- The Mechanic: Instead of updating rows in a `users` table, the agent appends an event to an `events` table: `{"type": "PreferenceUpdated", "payload": {"color": "blue"}, "timestamp": "..."}`.
- The Advantage: The agent reads and writes to this local SQLite database with zero network latency. It functions flawlessly on a plane, in a tunnel, or during a cloud outage. The current state is simply a materialized view derived by replaying the local event log.

1.2 The WAL (Write-Ahead Log) Advantage

SQLite utilizes a Write-Ahead Log (WAL). Writes are appended to the WAL file before being committed to the main database.

- The Impact: This provides ACID compliance and extreme write performance. The agent can stream thousands of cognitive events, memory updates, and tool outputs per second to the local store without blocking the LLM reasoning loop (Nona).

1.3 The Semantic Boundary (OPFS for the Web)

For web-based agents (CALLISTO), SQLite cannot rely on standard browser storage.

- The Mechanic: The architecture MUST utilize the Origin Private File System (OPFS). OPFS provides high-performance, synchronous file access (via Web Workers) required by SQLite WASM. This allows the browser-based agent to possess the same zero-latency, local-first event store as a native desktop application.

While local-first event sourcing solves the offline problem, it introduces severe physical and mathematical constraints when multiple devices attempt to synchronize.

2.1 The Last-Write-Wins (LWW) Fallacy

If Device A and Device B are both offline, and the user updates their persona on Device A, and updates their project context on Device B, a naive sync engine will compare timestamps and overwrite one update.

- The Flaw: Timestamps are dependent on local clocks, which drift. Even with NTP, LWW mathematically discards concurrent operations. The user loses data.

2.2 The Causal Consistency Gap

If Agent A writes Event 1, which causes Agent A to write Event 2, the sync engine MUST NOT deliver Event 2 to Agent B before Event 1.

- The Flaw: Standard message queues (like Kafka or RabbitMQ) do not inherently respect causal ordering across distributed, offline-first edge nodes.
- The Mitigation: The architecture MUST utilize Vector Clocks or Lamport Timestamps embedded in the event metadata to mathematically reconstruct the causal order during sync.

2.3 The Semantic Conflict (The "Blue vs. Red" Problem)

If the user tells their phone agent, "My favorite color is blue," and simultaneously tells their desktop agent, "My favorite color is red," a mathematical CRDT can merge the sets, but the result is a corrupted memory graph containing both facts. The LLM will hallucinate the user's preference.

- The Constraint: Mathematical CRDTs (like OR-Sets or LWW-Registers) resolve data conflicts, but they do not resolve semantic conflicts.

To solve the synchronization crisis, the architecture combines mathematical CRDTs with AI-driven semantic resolution.

3.1 Operation-Based CRDTs (CmRDTs)

Instead of syncing the entire state, the devices sync the operations (events).

- The Mechanic: An event like `{"action": "add_preference", "value": "blue"}` is a commutative operation. It does not matter if it arrives before or after `{"action": "add_preference", "value": "red"}`. The CRDT guarantees that both devices eventually converge to the exact same state containing both preferences.
- The Advantage: Operations are idempotent. If the network drops during sync, the devices can re-send the event log without fear of duplicating state.

3.2 The Semantic Merge Engine (The Mnemosyne Module)

When the CRDT detects a conflicting semantic state (e.g., the user changed their favorite color on two devices), the architecture does not crash or pick a random winner.

- The Mechanic: The conflicting events are routed to a local, fast LLM (via NPU). The LLM analyzes the conflict and the temporal context: "The user said 'blue' at 10:00 AM on mobile, and 'red' at 10:05 AM on desktop. The user likely changed their mind."
- The Impact: The LLM generates a new, synthesized "Resolution Event" (e.g., `{"action": "update_preference", "value": "red", "reason": "LWW based on semantic temporal recency"}`). This resolution event is appended to the CRDT log, mathematically converging both devices to the correct, semantically accurate state.

3.3 Asynchronous Background Sync

The agent never blocks its cognitive loop waiting for the network.

- The Mechanic: A dedicated background thread (or Web Worker) continuously monitors the local SQLite event log for new entries. When network connectivity is restored, it opens a WebSocket or gRPC stream to the enterprise control plane (or peer-to-peer via libp2p) and streams the compressed event deltas.
- The Impact: The user experiences zero latency. The sync happens silently in the background.

The fundamental shift of local-first CRDT sync is the restoration of cognitive sovereignty to the edge.

- Zero-Amnesia Agents: An agent running on a user's phone, desktop, and a sovereign edge node maintains a perfectly synchronized, persistent memory graph. If the phone dies, the desktop instantly resumes the cognitive trajectory with zero context loss.
- Multi-Agent Consensus: Multiple agents (e.g., a coding agent on the desktop and a research agent on the server) can share a synchronized memory graph via CRDT sync. When the research agent learns a new API pattern, the coding agent's local SQLite store is instantly updated, allowing seamless collaboration.
- Cloud Decoupling: The enterprise cloud is demoted from the "source of truth" to a "backup and aggregation node." If the cloud is destroyed, the distributed network of local SQLite event stores retains the complete, mathematically verifiable history of the agent's state.

To deploy local-first CRDT sync in production, the PANTHEON architecture (specifically the Mnemosyne and Clio modules) enforces strict operational boundaries.

5.1 Durable Execution for Sync State

The background sync process is fragile. If the sync crashes mid-transfer, it must not duplicate events.

- The Mitigation: The sync engine MUST be wrapped in a durable execution runtime (Temporal/Clio). The sync workflow is checkpointed. If the device reboots, the sync resumes exactly where it left off, utilizing the CRDT's idempotency to guarantee safe retries.

5.2 Event Log Compaction (The Snapshot Strategy)

An append-only SQLite event log will grow infinitely. After a month of continuous agent usage, replaying the log to derive state would take minutes.

- The Mitigation: The architecture MUST implement periodic snapshotting. When the event log exceeds 10,000 events, the system calculates the current state, writes a "Snapshot Event" to the log, and securely archives the older events to cold storage. The local state can now be derived by replaying from the latest snapshot, keeping local boot times under 500ms.

5.3 PQC Cryptographic Sync Tunnels

Syncing local state to the enterprise exposes the data to MITM attacks.

- The Mitigation: The WebSocket/gRPC sync tunnel MUST be encrypted using Post-Quantum Cryptography (Hybrid TLS 1.3 with ML-KEM, as mandated by `MYTHOS-SEC-0001`). Furthermore, the event payloads themselves MUST be

encrypted using envelope encryption (KMS), ensuring that even if the enterprise sync node is compromised, the local agent's memory remains cryptographically shredded.

The strategy of tethering AI cognition to centralized cloud databases is architecturally bankrupt for the edge era. An agent that forgets its user when the Wi-Fi drops is a toy. By utilizing SQLite as a local-first event store and CRDTs for mathematical state convergence, we achieve absolute cognitive continuity. The agent becomes a sovereign, zero-amnesia entity that seamlessly flows across devices, networks, and platforms, retaining its identity and memory without relying on the cloud.

```
A network cable is not a lifeline; an offline agent is not a zombie.
A timestamp is a lie; a CRDT is a mathematical truth.
A state mutation is a lost history; an event log is an absolute.
A cloud database is a master; a local SQLite store is a sovereign.

The cloud holds a replica; the edge holds the truth.

> Networks may partition.
> Cognitive state must be absolute.
```

End of Research Note RES-032

9. Source Register

Source	Authority and Status	Freshness Control
SQLite Write-Ahead Logging https://sqlite.org/wal.html	Primary database documentation Living Published: Living	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.
Automerge Documentation https://automerge.org/docs/	Primary CRDT implementation documentation Living Published: Living	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.
Local-First Software: You Own Your Data, in Spite of the Cloud https://www.inkandswitch.com/local-first/	Primary research essay Foundational research Published: 2019	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.

10. Lineage, Review, and Publication Record

Record	Value
Original source file	RES-032_Local_First_SQLite_Event_Sourcing_CRDT_Sync.md
Original source words	1663
Upgrade type	Enterprise research report; source and freshness validation; limitations; adoption decision; reproducibility plan
Generated on	2026-07-22
Current status	Candidate Research Report
Publication authority	Formal Mythos approval not yet recorded
Next review	120 days or event-driven trigger, whichever occurs first