StudioMeyer Memory — Whitepaper

Bi-temporal Knowledge Graph as a Service for AI Agents

Version: 1.0.3 **Stand:** 10. Juni 2026 **Repo:** github.com/studiomeyer-io/studiomeyer-memory **Server:** memory.studiomeyer.io **MCP Endpoint:** `https://memory.studiomeyer.io/mcp` **npm n8n-Node:** `n8n-nodes-studiomeyer-memory`

Executive Summary

AI-Assistenten haben kein Gedaechtnis. Jede Session faengt bei null an. Das ist 2026 immer noch das groesste ungeloeste Problem in der LLM-Anwendungsschicht.

StudioMeyer Memory ist ein **bi-temporaler Knowledge Graph** der genau dieses Problem loest. Statt reiner Notizen oder Vector-Embeddings speichert er Fakten als Knoten und Beziehungen mit Validity-Intervallen, Confidence-Scores und Widerspruchs-Erkennung. AI-Agents bekommen dadurch nicht nur "was wurde gesagt" zurueck, sondern "was ist heute noch wahr".

Das Produkt ist als hosted SaaS auf `memory.studiomeyer.io` live, betrieben in Frankfurt (EU-Hosting), DSGVO-konform. Zugang ueber drei Wege:

1. **MCP-Server** (Model Context Protocol) — direkte Integration in Claude Desktop, Claude Code, Cursor, VS Code Copilot Chat, Goose, Postman, MCPJam.
2. **n8n Community Node** (`n8n-nodes-studiomeyer-memory`) — sechzehn Operations als Drop-in-Node mit npm-Provenance.
3. **REST API** — vollstaendiger Zugriff fuer Custom-Integrationen.

Auf dem LongMemEval-Benchmark erreicht StudioMeyer Memory in unserem aktuell besten internen 50q-stratified Run **86 Prozent** mit GPT-4o als Judge (Run S957b, 1. Mai 2026, Anthropic Sonnet 4.6 als Answer-Generator). Caveat zur Ehrlichkeit: dieser Run lief vor dem v3.16.11 Cross-Project-Search-Leak-Fix; ein sauberer Re-Run mit dem Fix scharf gestellt steht aus. Methodik, Konfiguration und Limitations sind in Sektion 4 dokumentiert.

---

1. Das Problem — AI Memory in 2026

1.1 Drei Schichten die alle "Memory" heissen

Im Markt werden drei sehr unterschiedliche Dinge als Memory verkauft. Wer das nicht trennt, vergleicht Aepfel mit Birnen.

Schicht 1: Statische Notizen. Eine Markdown-Datei wie `CLAUDE.md` oder `AGENTS.md` . Du schreibst rein "wir nutzen TypeScript strict" und der Assistent liest das bei jeder Session. Kein Algorithmus, keine Embeddings, keine Cloud. Reicht fuer viele Solo-Entwickler.

Schicht 2: Akkumulierende Notizen. Was Claude Code seit Maerz 2026 mit Auto-Memory + Auto-Dream tut. Die KI schreibt waehrend der Arbeit selbst Eintraege in dieselbe Markdown-Datei, ein nightly Subagent konsolidiert. Funktioniert lokal, bleibt im File-System, kein semantisches Retrieval. ChatGPT Memory ist im Prinzip dasselbe, nur ohne Editier-Zugang.

Schicht 3: Strukturiertes Memory mit Knowledge Graph. Hier werden Fakten als Knoten plus Kanten gespeichert, mit semantischer Suche, Confidence-Decay und teilweise temporaler Validity. Loest Probleme die Schicht 1 und 2 nicht koennen. Das ist die Kategorie in der StudioMeyer Memory antritt.

1.2 Wo Schicht 1 und 2 nicht reichen

Sobald folgende Anforderungen kommen, reicht ein Markdown-File nicht mehr:

- **Cross-Session und Cross-Tool**: Claude Code, Cursor, Codex, Web-Chats, n8n, Voice-Agents teilen Memory
- **Multi-User oder Multi-Tenant**: jeder User braucht eigenen isolierten Speicher
- **Bi-Temporalitaet**: was war wann gueltig, was wurde wann ueberschrieben
- **Widerspruchs-Erkennung**: "ich wohne in Berlin" vs. drei Wochen spaeter "ich wohne in Hamburg" — beides nicht gleichzeitig wahr
- **Skalierende Confidence**: wie zuverlaessig ist eine Information, wird das Vertrauen schwaecher mit der Zeit
- **Audit-Trail mit DSGVO-Tauglichkeit**: jede Schreib-Operation nachvollziehbar, Loeschpfade fuer Recht-auf-Vergessen

Genau hier setzt StudioMeyer Memory an.

1.3 Warum Bi-Temporalitaet das Killer-Feature ist

Die meisten Memory-Systeme speichern eine Wahrheit: "der User wohnt in Hamburg." Wenn sich das aendert, wird die alte Information ueberschrieben. Geschichte verloren.

Bi-Temporalitaet bedeutet zwei Zeit-Achsen pro Fakt:

- **Valid-from / Valid-to**: wann war diese Information in der echten Welt wahr?
- **Recorded-at**: wann wurde sie in den Speicher geschrieben?

Damit kannst du fragen: "Was wussten wir am 15. Maerz ueber den User?" und bekommst den Stand wie er damals war, nicht den heutigen Stand. Praktisch wichtig fuer:

- Voice-Agents die Anrufe protokollieren und spaeter ueber den Stand zu einem bestimmten Zeitpunkt schlussfolgern muessen
- Customer-Support der einen Bug von vor zwei Wochen rekonstruieren muss
- Personal Assistants die den Stand einer Beziehung oder eines Projektes ueber Monate verfolgen
- Compliance-Audits die nachweisen muessen welcher Stand zu einem Datum bekannt war

Bi-Temporalitaet ist bei StudioMeyer Memory ab dem Free-Tier eingebaut, kein Aufpreis fuer das wichtigste Feature.

---

2. Was StudioMeyer Memory tut

2.1 Kern-Funktionen

Sessions: jede Konversation kann als Session gestartet und abgeschlossen werden. Sessions tragen Observations und werden chronologisch verkettet. Auf Anforderung kann man eine Session als step-by-step Replay rendern.

Learnings: Patterns, Mistakes, Insights, Research-Notizen werden als typisierte Eintraege gespeichert. Jeder Eintrag hat Category, Confidence (0.0 bis 1.0), Memory-Type (episodic oder semantic) und optionale Tags. Der Gatekeeper-Layer prueft beim Schreiben gegen existing Eintraege: bei mehr als 95 Prozent Aehnlichkeit wird gemerged statt dupliziert.

Decisions: Strategische Entscheidungen mit Reasoning, Alternatives und Confidence-Score. Decisions koennen reviewt und revised werden. Sind die zentrale Audit-Surface fuer den Knowledge Graph.

Entities + Observations + Relations: der Knowledge Graph selbst. Entities haben Types (person, project, service, customer, etc.), Observations sind kleine Fakten an einer Entity, Relations verbinden Entities mit gerichteten Kanten.

Skills: ein Pattern-Memory das Erfolg oder Fehlschlag pro Domain trackt und beim naechsten Mal Recipe-Vorschlaege zurueckliefert.

Synthesis und Reflection: aus akkumulierten Learnings koennen periodisch Synthesen generiert werden. Wochen- und Monatsreflexionen on-demand.

Dream-Cycle (Offline-Konsolidierung, neu Juni 2026): Das Memory arbeitet auch zwischen den Sessions. Ein Dream-Cycle verdichtet wiederkehrende episodische Erinnerungen zu dauerhaften semantischen Fakten (LLM-destilliert, mit konservativer Confidence und Verlinkung auf die Quell-Episoden — die Originale bleiben erhalten), prueft den Bestand auf Widersprueche, rebalanciert Confidence-Decay und Lifecycle-Tiers und hinterlaesst einen durchsuchbaren Dream-Report. Laeuft automatisch (taeglich gated) oder manuell mit Vorschau-Modus. Das ist dasselbe Konsolidierungs-Prinzip, das 2026 mit Anthropic Dreaming und Letta Sleep-Time-Compute zum Industrie-Standard wird — hier nativ im Knowledge Graph.

Retrieval, das ehrlich bleibt und mitlernt (neu Juni 2026): Drei Verhaltens-Upgrades im Suchpfad. Erstens altern zeitlose semantische Fakten gedaempft — eine Dauerwahrheit wird nicht von Recency begraben, frische Eintraege bleiben unberuehrt. Zweitens staerkt jeder Abruf das Memory: haeufig genutzte Fakten bleiben praesent, archivierte werden bei erneuter Relevanz reaktiviert (mit Spacing-Daempfung gegen Feedback-Schleifen). Drittens entscheidet die Abstention-Logik jetzt auf Basis von zwei unabhaengigen Evidenz-Signalen (semantische

Aehnlichkeit plus Cross-Encoder), ob der Server lieber ehrlich "keine gesicherte Information" meldet, statt aus schwachen Treffern zu raten — gemessen auf dem internen Retrieval-Harness ohne einen einzigen False-Abstain.

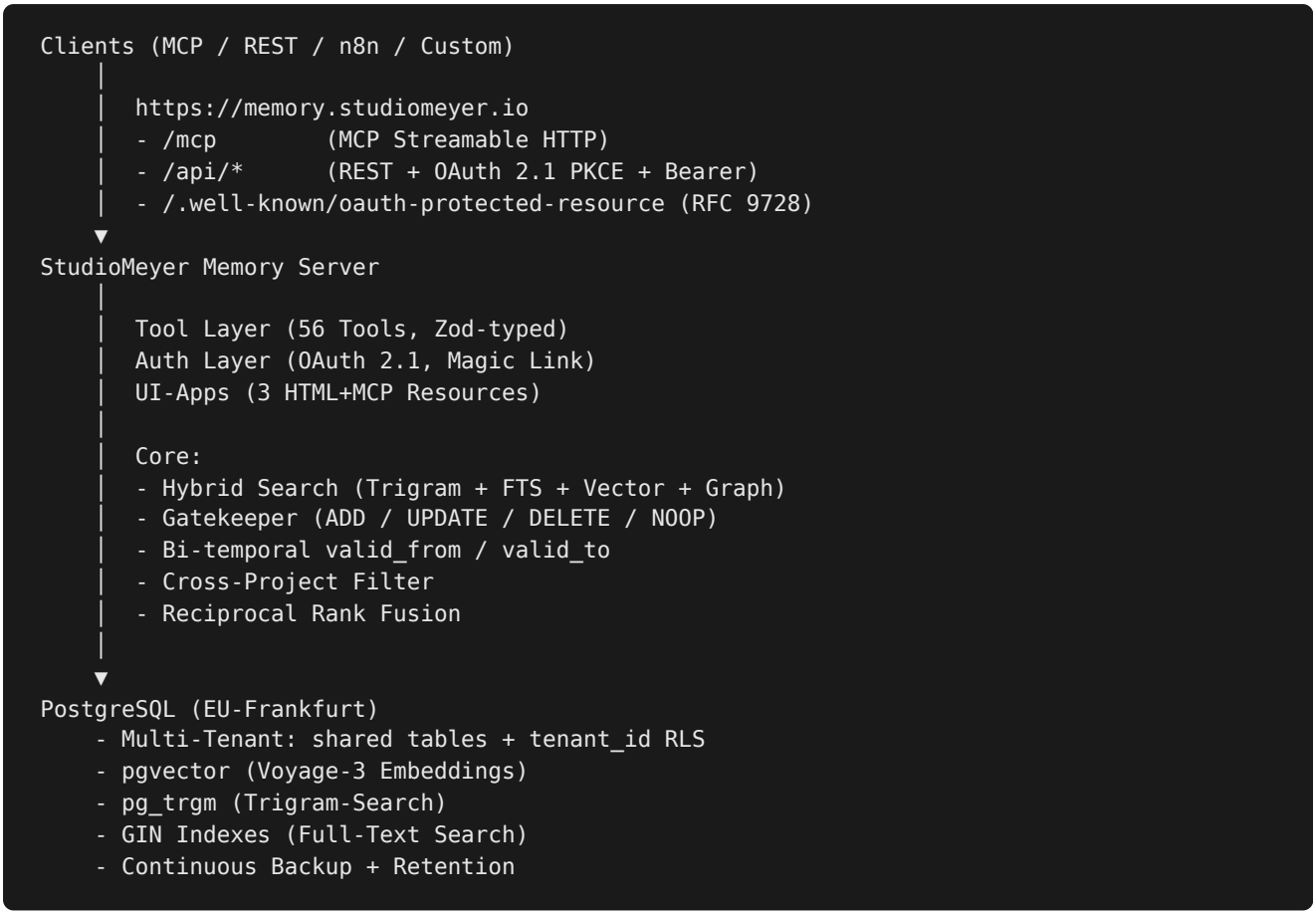
2.2 Tool-Inventar

56 MCP-Tools in sieben funktionalen Bloecken:

Block	Tool-Anzahl	Beispiele
Session Lifecycle	5	Session start/end, summarize, reflect, replay
Learnings + Decisions	12	learn, decide, update, link, archive
Knowledge Graph	14	entity create/observe/relate/search/open/graph
Search + Retrieval	8	hybrid search, recall, recall-timeline, rerank
Synthesis + Insights	7	synthesize, insights, proactive, chronicle
Maintenance + Health	7	health, dedupe, decay, consolidate, export
UI Resources (MCP Apps)	3	graph_view, recall_timeline, session_replay

Die UI-Resources sind nach der MCP Apps Spec vom 26. Januar 2026 gebaut. Verifiziert in Claude.ai (Web + Desktop) — dort rendern interaktiver 3D-Knowledge-Graph, vertikale Timeline und Step-by-Step Session-Replay direkt im Chat als sandboxed iframes. Offiziell von der MCP Apps Spec angekuendigte UI-Hosts: Claude (Web/Desktop), Goose, VS Code Insiders, ChatGPT. Andere MCP-Clients (Postman MCP Client, MCPJam, weitere) bekommen die strukturierte JSON-Antwort als Fallback — voll backward-kompatibel.

2.3 Architektur



Hosting: EU-Region Frankfurt. Active-Standby-HA mit Auto-Failover via Watchdog. Drill-getestet.

SSL: Let's Encrypt Wildcard, autonomous Renewal.

2.4 Search-Stack

Die Suche ist hybrid und mehrstufig. Eine Query laeuft durch:

1. **Trigram-Match** fuer fuzzy keyword + typo-tolerant
2. **Full-Text-Search** (PostgreSQL GIN-Index, ts_vector) fuer exakte Phrasen
3. **Semantic-Vector** (pgvector, Voyage-3 Embeddings) fuer paraphrasiertes Material
4. **Knowledge-Graph-Walk** (entity-based Traversal mit Hop-Distance-Filter)
5. **RRF-Merging** (Reciprocal Rank Fusion) der vier Streams
6. **Temporal-Decay** (juengere Eintraege gewinnen, konfigurierbar)
7. **Optional Cross-Encoder Rerank** (opt-in)
8. **Noise-Filter** (entfernt Low-Quality-Treffer wenn Score-Verteilung keine Discrimination zeigt)

Dieses 8-Schicht-Pattern findet "das Ding mit SSL" auch wenn es als "certbot renewal" gespeichert war. Jedes Search-Ergebnis kommt mit `relevance`, `recency`, `rank`, `matchMethod` und optional einer `explain`-Section die zeigt warum der Treffer gefunden wurde.

2.5 Gatekeeper — Anti-Duplication

Beim Schreiben eines Learning oder einer Observation prueft der Gatekeeper-Layer:

- Hash-Identitaet (auf normalized content) → SKIP wenn identisch
- Embedding-Cosine ueber 0.95 → UPDATE existing statt ADD new (mit "previously: ..."-Trail)
- Embedding-Cosine zwischen 0.85 und 0.95 → wird als verwandt verlinkt
- Unterhalb 0.85 → ADD als neuer Eintrag

Plus optional Fast-Mode (`fast: true` per call) der den Embedding-Round-trip skipped fuer high-volume Ingest-Workloads. Default ist Full-Gatekeeper, Fast-Mode ist explizit pro Operation aktivierbar.

2.6 3D-Knowledge-Graph + Cinema-Mode

Eine technische Einzigartigkeit unter den Memory-Systemen: drei interaktive UI-Resources die direkt im Claude-Chat rendern (MCP Apps Spec 26. Januar 2026). Jede Resource ist self-contained HTML+JavaScript ohne externe Abhaengigkeiten, sandboxed im iframe, ueber `nex_graph_view` , `nex_recall_timeline` und `nex_session_replay` aufrufbar.

3D-Knowledge-Graph (`nex_graph_view`). Cinema-Mode 3D-Force-Graph mit auto-rotate Kamera, Hover-Pin, Multi-Select-Type-Filter, Cmd+K Search, Detail-Panel mit Top-30 Observations + In/Out-Relations (klickbar fuer fly-to). Time-Travel-Slider unten in der Stage — du kannst zu einem Datum springen und sehen wie der Graph an dem Tag aussah. Zehn Entity-Buckets statt der ueblichen drei bis fuenf (person, agent, service, infra, customer, project, concept, decision, learning, session) mit eigenen Farben. WebGL-Cleanup auf unmount.

Recall-Timeline (`nex_recall_timeline`). Vertikale chronologische Timeline aus aktuellen Learnings, Decisions und Sessions. Filter-Chips fuer Typ, Category-Dropdown, Search, click-to-expand. Default-Limit 200 Items, Time-Window konfigurierbar.

Session-Replay (`nex_session_replay`). Horizontaler Step-by-Step Replay einer Session: Start → Observations → Learnings → Decisions → End. Fuer Debugging und Audit-Trails von Voice-Agent-Sessions oder Customer-Support-Conversations praktisch.

Wo das laeuft. Live verifiziert in zwei Production-Surfaces:

1. **Claude.ai App (Web + Desktop) direkt im Chat.** Frag im Chat „zeig mir meinen Knowledge-Graph" und das interaktive 3D-Force-Graph rendert als Inline-iframe — ohne externe Tab, ohne separate App, ohne Login-Flow.
2. **StudioMeyer Portal-Dashboard** unter `studiomeyer.io/portal/memory/knowledge` mit gleichem 3D-Force-Graph als Pro-Feature plus Snapshot-Mode, Bar-Chart-Mode und Time-Travel-Slider.

Plus eine Public Demo-Page unter `studiomeyer.io/services/memory/demo` zeigt einen Showcase-Tenant mit kuratiertem Knowledge-Graph (111 Entities, 199 Observations, 140 Relations, 40 Learnings, 23 Decisions, 45 Sessions) — fuer alle die einen Account-Setup nicht erst durchlaufen wollen bevor sie wissen wie es aussieht.

Kein anderes uns bekanntes Memory-System hat eine vergleichbare Live-3D-Visualisierung als Bestandteil des Standard-Tool-Surfaces.

---

3. Integrationen

3.1 MCP Server (Model Context Protocol)

`https://memory.studiomeyer.io/mcp` ist ein Streamable-HTTP MCP-Endpoint nach Spec 2025-06-18 mit Backwards-Compat fuer aeltere Spec-Versionen.

Auth: OAuth 2.1 mit PKCE und Magic-Link-Flow ueber Email. RFC 9728 `.well-known/oauth-protected-resource` ist publik, automatic-discovery aktiv.

Tested gegen:

- Claude Desktop (App)
- Claude Code (CLI)
- Cursor
- VS Code GitHub Copilot Chat
- Goose, Postman MCP Client, MCPJam

Setup-Befehl Claude Code:

```
claude mcp add studiomeyer-memory -s user --url https://memory.studiomeyer.io/mcp
```

Erste Tool-Calls triggern den OAuth-Flow per Browser-Redirect, danach speichert der Client das Refresh-Token lokal.

3.2 n8n Community Node

`n8n-nodes-studiomeyer-memory` ist ein nativer n8n-Community-Node, MIT-lizensiert, auf npm published mit npm-Provenance via GitHub Actions OIDC. Die n8n-Verified-Community-Node-Submission ist fuer v0.2 geplant — v0.1.x akzeptiert bewusst den Unverified-Status, weil Zero-Runtime-Dependencies (SDK-Bundling) erst in v0.2 fertiggestellt werden.

15 Operations in vier Resources:

Resource	Operations
Memory	Search, Recall, Learn, Decide
Entity	Create, Observe, Search, Relate, Open
Session	Start, End, Recall Timeline
Insight	Synthesize, Reflect, Proactive Briefing

Setup: API-Key aus dem Portal kopieren, in n8n Credentials einfuegen, Node aus dem Workflow-Editor ziehen, Resource und Operation waehlen, fertig.

Architektur: stateless Connect/Call/Close pro Execution via offizielles MCP-SDK. Pure-Function Routing testbar ohne Mocks. CI mit Lint + Typecheck + Tests auf Node 20 und 22.

3.3 n8n Templates

github.com/studiomeyer-io/n8n-templates enthaelt n8n-Workflows die den Memory-Node nutzen — alle mit einheitlichen Production-Patterns: Webhook-Verification (HMAC), Rate-Limit, Idempotency-Check, Error-Branches mit LLM-Fallback, Audit-Trail bei Errors.

Verfuegbar heute (Tier 1, hardened):

- **Voice-Agent Cross-Session Memory** (Vapi/Retell + Telegram + Memory)
- **Customer Support Bot mit Kunden-Historie**
- **Personal Assistant mit Long-Term Memory**

In Vorbereitung (v0.5.0-prep, Distribution-Push pending T01 End-to-End-Voice-Smoke):

- **Restaurant Stammgast-Bot** (Telegram + phone-customer-key)
- **Tourist-Bot Repeat-Visitor** (web-chat + sessionId/fingerprint)
- **Lead-Qualifier mit BANT+I + Pipedrive**
- **Meeting-Bot Cross-Meeting Continuity** (Fathom/Otter/Granola Webhook)
- **Migration-Tool zu StudioMeyer Memory** (Developer-Preview: ETL aus Vector-only-Memory-Systemen)

3.4 REST API

Fuer Custom-Integrationen ohne MCP oder n8n:

```
# Search
curl -X POST -H "Authorization: Bearer ${API_KEY}" \
  -H "Content-Type: application/json" \
  -d '{"query":"SSL renewal","limit":5}' \
  https://memory.studiomeyer.io/api/search

# Learn
curl -X POST -H "Authorization: Bearer ${API_KEY}" \
  -H "Content-Type: application/json" \
  -d '{"category":"pattern","content":"...","confidence":0.9}' \
  https://memory.studiomeyer.io/api/learn

# Decide
curl -X POST -H "Authorization: Bearer ${API_KEY}" \
  -H "Content-Type: application/json" \
  -d '{"title":"X","decision":"Y","reasoning":"Z"}' \
  https://memory.studiomeyer.io/api/decide
```

Volle API-Spec unter `/api`.

---

4. Methodik — LongMemEval

4.1 Was LongMemEval misst

LongMemEval ist ein 2024 von Wu et al. (arXiv 2410.10813) publizierter Benchmark fuer Long-Term-Memory in AI-Assistenten. Er besteht aus 500 Evaluationsinstanzen. Die offiziellen `question_type`-Felder decken sechs Typen ab:

- **single-session-user**: Fakt aus einer einzelnen User-Message in einer Session
- **single-session-assistant**: Fakt aus einer Assistant-Message (typischerweise Listen, Empfehlungen)
- **single-session-preference**: User-Praeferenzen aus einer Session
- **multi-session**: Information die ueber mehrere Sessions verteilt ist
- **temporal-reasoning**: Zeit-Bezuege in absolute Daten aufgeloest
- **knowledge-update**: spaeter-revisierte Information

Plus Abstention-Fragen die zusaetzlich ueber `_abs`-IDs markiert sind — mit Abstention insgesamt sieben Typen. Auf Capability-Ebene misst LongMemEval fuenf Kernfaehigkeiten: information extraction, multi-session reasoning, knowledge updates, temporal reasoning und abstention.

Pro Question gibt es eine Haystack von realistischen Konversations-Sessions plus eine konkrete Frage. Das Memory-System muss ingestieren und die Frage beantworten.

Subsets: oracle (15 MB, fokus auf Memory-Mechanik), small-cleaned (265 MB, mittel), medium-cleaned (2.6 GB, voll).

Scoring: ein zweiter LLM (default GPT-4o) judget jede Antwort gegen den Gold-Standard mit binaerem Verdict (correct oder incorrect). Aggregat-Score ist Prozent korrekt.

4.2 Unsere Bench-Pipeline

Das Bench-Skript ist transparent dokumentiert. Kern-Features:

1. **Tenant-Isolation pro Run**: jeder Bench bekommt einen frischen Memory-Tenant.
2. **Per-Question-Project-Scoping**: jede Q bekommt einen eigenen Project-Filter. Ohne diese Isolation wuerden Q's Cross-Help bekommen.
3. **Per-Q User-Entity-Naming**: User-Entity heisst pro Question eindeutig, keine Cross-Q-Dedup-Collision.
4. **Atomic Fact Extraction**: jede User-Message wird in CATEGORY:FACT-Atoms zerlegt.
5. **Assistant-Message-Processing**: Listen und nummerierte Aufzaehlungen werden mit Position-Marker gespeichert.
6. **Multi-Query-Decomposition**: bei der Antwort-Phase wird die Frage in 2-3 Sub-Queries decomposed.
7. **Chain-of-Thought Answer**: Antwort-LLM bekommt expliziten CoT-Prompt fuer multi-hop reasoning.
8. **Cost-Cap als Hard-Stop**: konfigurierbarer Budget-Cap als Schutz gegen Runaway.

Provider-Optionen im Bench-Skript:

- Anthropic Haiku 4.5 fuer Extraction und Answer (schnell, kostenguenstig)
- Anthropic Sonnet 4.6 als optional Answer-Generator (besser auf reasoning-heavy Q's)
- OpenAI GPT-4o (vergleichbarer Cost, anderes Failure-Profile)

Scoring-LLM: GPT-4o (`gpt-4o-2024-08-06`), Pflicht weil das ist auch der Judge im offiziellen LongMemEval-Paper.

4.3 Aktuelles Result

StudioMeyer Memory: 86 Prozent auf 50q stratified, GPT-4o judged (Run S957b, 1. Mai 2026).

Konfiguration: Anthropic Haiku 4.5 fuer Extraction, Anthropic Sonnet 4.6 als Answer-Generator, frisch isolierter Bench-Tenant, per-Q Project-Scoping, Memory-Server v3.16.10.

Caveat zur Ehrlichkeit: S957b lief gegen einen Memory-Server der noch den Cross-Project-Search-Leak hatte. v3.16.11 (Cross-Project-Filter komplett geschlossen plus Performance-Reparatur — neuer GIN-Index auf entity-FTS, vier neue partielle Composite-Indexes, OR-with-NULL-Pattern in semantic-search durch dynamic-append ersetzt) ist seit 2. Mai LIVE auf Production. Production-EXPLAIN-ANALYZE bestaetigt: FTS Bitmap-Index-Scan in 0.226 ms, /health 50-60 ms p95, Container-Logs sauber. Der 86-Prozent-Wert war potenziell um 1-3 Prozentpunkte durch Cross-Q-Help inflationiert; ein sauberer 50q-Re-Run mit dem Fix scharf gestellt steht aus. Ehrliche Range nach dem Re-Run: 78-86 Prozent.

Kategorie-Breakdown (S957b, vor cross-project-leak-fix):

- **temporal-reasoning**: 92 Prozent
- **multi-session**: 92 Prozent
- **single-session-user**: 100 Prozent
- **single-session-preference**: starke Performance
- **knowledge-update**: starke Performance
- **single-session-assistant**: 0 Prozent in der ersten Baseline (S933) — Listen-Extraction aus langen Assistant-Messages war nicht implementiert. S957d-Fix mit Listen-Item-Detection und Position-Marker brachte 4 von 6 in einem targeted-Smoke (67 Prozent). Voll-Run mit dem Fix scharf gestellt steht im selben Re-Run wie der Cross-Project-Leak-Fix aus.

Status sauberer Re-Run (Stand 2. Mai 2026): Geparkt bis vier Vorbedingungen erfuellt sind. Drei Smoke-Versuche am 2. Mai sind hinfaellig — zwei kleine Wellen liefen gegen einen alten Bench-Tenant der die Ground-Truth einer Frage gar nicht ingested hatte; ein 10q-Stratified-Run gegen einen frischen Tenant wurde nach 2 Fragen abgebrochen, weil ungetrackte Memory-Server-interne Anthropic-Calls (Gatekeeper-LLM-Decisions im Cosine-0.85-0.95-Bereich) den Bench-eigenen `--max-cost` umgangen und reale Kosten produziert haben.

Vier Vorbedingungen vor dem naechsten Bench-Run:

1. Bench-Skript schickt `fast: true` per-call mit allen Ingest-Operationen — der Memory-Server-interne Sonnet-Round-trip wird damit geskippt und Kosten bleiben im Bench-Cap nachvollziehbar.

2. Hard-Budget API-Key auf Anthropic Console fuer den Memory-Server (separater Key in der SaaS-Container-`.env` mit eigenem Cap).
3. Pre-Flight 1q-Smoke zur Cost-Verifikation BEVOR 10q oder 50q starten.
4. Cross-Encoder Phase 4 (JINA) bleibt aus.

500q Voll-Run: geparkt. Wird publiziert sobald die vier Vorbedingungen erfuellt sind und ein sauberer 50q-Re-Run validiert hat dass die Cost-Kontrolle haelt. Solange behandeln wir den 86-Prozent-Wert als internen Zwischenstand auf 50q-Basis.

---

5. Public Comparison

Direkte Vergleiche sind heikel weil fast jedes Memory-System unter anderen Setup-Bedingungen testet: anderer Answer-LLM, anderer Subset, anderer Judge, anderer Re-Run, anderes End-to-End-Setup. Was wir sauber sagen koennen:

System	Aggregat	Temporal	Methodik / Caveat	Quelle
StudioMeyer Memory	**86%** (50q stratified, S957b)	92%	Haiku Extract + Sonnet 4.6 Answer + GPT-4o Judge. Caveat: vor v3.16.11 cross-project-leak-fix, sauberer Re-Run pending	dieses Whitepaper
Mem0 (Managed Platform)	93.4%	93.2%	Token-Efficient Memory Algorithm April 2026. Gilt fuer Managed mit proprietaeren Optimierungen — OSS-SDK richtungsgleich aber nicht identisch	mem0.ai/blog April 2026
Mem0 BEAM 10M Tokens	48.6%	16.3%	Production-Scale-Bench. Mem0 selbst benennt das als offenes Problem	mem0.ai/research
Mastra Observational Memory	94.87% / 84.23%	—	gpt-5-mini Answer / gpt-4o Answer — +10pp pure Modell-Switch	mastra.ai/research/observational-memory
OMEGA	95.4%	—	GPT-4.1 als Reasoning-Answer, OSS	omegamax.co/compare
Hindsight	91.4%	79.7%	end-to-end mit reasoning-Modell	hindsight-ai paper
Hindsight-OSS-20B	83.6%	—	OSS-Modell als Answer	hindsight
SuperMemory	85.4%	—	Drittpartei	supermemory.ai
Letta	83.2%	—	Drittpartei	atlan-Vergleich
Zep	63.8%	—	offiziell	arXiv 2501.13956

Kontext zu der Tabelle:

Wir sind ehrlich: im Aggregat liegen wir mit 86 Prozent unter Mem0 (Managed Platform 93.4 Prozent) und unter Mastra OM mit gpt-5-mini (94.87 Prozent). Wer puren Score-Vorsprung sucht, findet ihn dort.

Was uns differenziert ist nicht der Aggregat-Score, sondern Architektur und Production-Verhalten:

- **Bi-Temporal Knowledge Graph ab Free Tier.** Die meisten Wettbewerber haben Knowledge-Graph-Features hinter Pro-Tiers (z.B. Mem0 Graph Memory ab 249 USD pro Monat). Bei uns ist Bi-Temporalitaet inkludiert.
- **BEAM-10M Production-Scale.** Mem0 berichtet selbst dass ihr Aggregat bei 10M Tokens auf 48.6 Prozent einbricht und temporal sogar auf 16.3 Prozent. Das ist eine harte Real-Production-Limitation. Unser Hybrid-Search-Stack mit pgvector + pg_trgm + GIN + RRF skaliert anders, weil die Search-Phase nicht reasoning-LLM-dependent ist.
- **EU-Region-Hosting + DSGVO.** Frankfurt, Multi-Tenant-Isolation per RLS, Auftragsverarbeitungsvertrag verfuegbar.
- **3D-Knowledge-Graph + UI Resources** (Sektion 2.6). Live-rendernde MCP-Apps-Resources direkt im Claude-Chat plus im SaaS-Portal — bei keinem anderen Memory-System gefunden.
- **Methodik-Transparenz.** Wir publizieren Run-Konfiguration, Caveats, Re-Run-Plan. Das ist in einem so jungen Markt selten.

Mastra OM, OMEGA und Hindsight nutzen Reasoning-Modelle (gpt-5-mini, GPT-4.1) als Answer-Generator. Das ist eine andere Konfigurations-Liga als ein Stack mit Sonnet 4.6 oder Haiku als Answer. Wir testen aktuell GPT-5-mini als Answer-Alternative — preislich attraktiv (0.25 / 2.00 USD per Million Input/Output Tokens vs Sonnet 3 / 15 USD), Mastra hat damit +10.64pp gegenueber gpt-4o erreicht.

StudioMeyer Memory's Staerke liegt in **time-aware Memory** (Bi-Temporal-Architektur) und **Production-Scale-Stabilitaet** (Hybrid-Search statt reasoning-LLM-dependency). Wenn der Use-Case Voice-Agents, Customer-Support oder Personal Assistant ist, sind diese beiden Differenzierer entscheidender als der Aggregat-Score.

---

6. Use-Cases (mit n8n)

6.1 Voice-Agent mit Cross-Session Memory

Ein Vapi- oder Retell-basierter Voice-Agent ruft per Webhook einen n8n-Workflow auf. Phone-Number ist der Customer-Key. Memory-Search findet existing Customer-Profile, Sessions-History, Praeferenzen. Antwort wird kontextualisiert. Nach Call: neue Observations werden geschrieben.

Template im Repo, mit HMAC-Signature-Verifikation, Webhook-Acknowledge, Async-Memory-Writes.

6.2 Customer-Support-Bot mit Ticket-Historie

Telegram- oder Discord-Bot fuer ein KMU. Bei jeder neuen Anfrage Memory-Search auf Customer-Email oder UserID. Bot kann auf vergangene Tickets, gestellte Fragen, getroffene Loesungen referenzieren. Memory-Learn bei jedem aufgeloesten Ticket fuer naechstes Mal.

Template mit LLM-Fallback-Reply (Memory-Search-Failure schlaegt nicht den ganzen Bot, sondern liefert eine generische Antwort + Audit-Eintrag).

6.3 Personal Assistant fuer Solo-Founder

Ein eigener Bot der ueber Telegram zugaenglich ist und Cross-Session Memory hat. Nutzer schreibt "denk dran ich treffe X am Donnerstag", Bot bestaetigt + Memory-Learn. Spaeter "was wollte ich X fragen?" liefert das Memory-Search-Result.

6.4 Lead-Qualifier mit BANT+I + Pipedrive-Integration

Webform empfaengt Lead-Daten, n8n-Workflow ruft GPT-4o-mini fuer BANT-Analyse, schreibt das Result als Memory-Decision, erstellt Deal in Pipedrive. Memory bleibt die Single-Source-of-Truth fuer das Lead-Profile.

6.5 Meeting-Bot Cross-Meeting Continuity

Fathom-, Otter- oder Granola-Webhook auf Meeting-End triggert n8n. Bot identifiziert Participant-Set via Hash, sucht Memory nach vorherigen Meetings mit gleichen Participants, schreibt Folge-Ups, postet Summary in Slack.

6.6 Migration zu StudioMeyer Memory

Pure ETL-Workflow, kein LLM, n8n-SplitInBatches-Loop. Liest existing Memory-Daten aus Vector-only-Quellen, mappt auf StudioMeyer Memory's Schema (Entities, Observations, Relations, Learnings), bulk-imported via Memory-Learn.

---

7. Pricing

Tier	Preis	Beschreibung
Free	$0	Fuer immer kostenlos. Voller Zugriff auf MCP-Server, n8n-Node und REST-API. Hybrid-Search-Stack (Trigram + FTS + pgvector + Knowledge-Graph + RRF + temporal decay) — schnell, deterministisch, kein LLM-Overhead.
Pro	$29 / Monat	Founding Price fuer die ersten 100 User. Plus **Agentic Retrieval** (Iterative LLM-Completeness-Check fuer multi-hop / temporal / aggregation queries) und **AI Reranker** (Cross-Encoder + LLM-Fallback fuer Precision auf Top-Ergebnissen). Hoehere Limits, Priority-Support.
Team	$49 / Monat	Unbegrenzte KI-Features, Team-Sharing, dedicated Support.

Was Pro+ konkret zusaetzlich tut: Bei einer Query wie *"Was war der Gesamtumsatz aller Kunden die ich im Q1 betreut habe und im Maerz konvertiert sind?"* erkennt das Pro-Tier dass das Multi-Hop + Temporal + Aggregation ist, fuehrt eine Initial-Suche aus, prueft via Haiku-LLM ob die

Antwort vollstaendig ist, und feuert bei Luecken eine Refined-Re-Query. Auf LongMemEval-Benchmarks bringt das 5-10pp Recall auf den schwersten Question-Kategorien. Free-Tier laesst diese LLM-Roundtrips weg — schneller (typisch 2s statt 7s), deterministisch, aber bei sehr komplexen multi-hop queries niedrigere Recall.

Kein Vendor-Lock-in: jeder Tenant kann jederzeit alle Daten als Markdown oder JSON exportieren. Wer kuendigt, bekommt alles mit.

Keine Hidden-Costs. Embeddings, Storage, Backup, SSL, OAuth, Multi-Tenant-Isolation, Bi-Temporal-Tracking, Knowledge-Graph alles im Preis enthalten. Keine separate Knowledge-Graph-Subscription.

DSGVO: Primaere Datenhaltung in EU-Region Frankfurt (PostgreSQL + Backups). Auftragsverarbeitungsvertrag (AVV) auf Anfrage, Standardvertragsklauseln (SCC) wo Subprozessoren in Drittlaendern noetig sind. Technische und organisatorische Massnahmen (TOMs) dokumentiert. Wir verwenden bewusst EU-Regionen bei allen Plattform-Subprozessoren — direkter Drittland-Datenexport findet nicht statt.

Recht auf Vergessen: vollstaendiger Loesch-Pfad pro Tenant, Audit-Log mit definierter Retention. Voll Art. 17 DSGVO-konform.

Aktuelle Pricing-Details siehe studiomeyer.io/services/memory.

---

8. Roadmap

Q2 2026:

- ✔ Dream-Cycle Offline-Konsolidierung + Retrieval-Reinforcement + Zwei-Signal-Abstention (shipped Juni 2026, siehe Sektion 2.1)
- Sauberer 50q-Re-Run nach Erfuellung der vier Cost-Control-Vorbedingungen aus Sektion 4.3 — Whitepaper-Update mit dem post-Fix Wert
- 500q-Voll-Run, sobald der saubere 50q stabil und budget-kontrolliert laeuft
- n8n Templates Distribution-Push (Tier 2 + 3 nach erfolgreicher T01-Voice-Smoke)
- Drei weitere n8n-Templates fuer SM-Synergien (Restaurant, Tourismus, CRM)

Q3 2026:

- Cross-Encoder Phase 4 als opt-in Pro-Feature
- Memory-Diff-Tool fuer Side-by-Side Vergleich von Snapshots
- Slack-Native Integration
- Tauri-basierte Desktop-App fuer 3D-Knowledge-Graph-View

Q4 2026:

- Self-Hosting-Option (Docker-Compose-Pack, Lizenz-basiert)
- Multi-Region-Hosting als Add-On fuer Enterprise-Tier
- LongMemEval extended Benchmark-Suite (LoCoMo + ConvoMem) als kontinuierlicher CI-Run

9. Limitations + Honesty

Was StudioMeyer Memory heute NICHT hat:

- **Kein Self-Hosting heute**. Kommerzielle Source-Lizenzen sind auf Anfrage moeglich. Out-of-the-box Self-Hosting ist Q4-Roadmap-Item.
- **Keine eingebauten Connectors zu Slack, Notion, Gmail (noch)**. Roadmap-Item Q3.
- **Keine produktseitige Token-Kompression**. Die SaaS-Schicht speichert Eintraege verbatim — kein verlustbehaftetes globales Conversation-Compression-Feature. Vorteil: keine Information-Loss bei Long-Form-Memory. Nachteil: groesserer Storage-Footprint pro Tenant. (Die LongMemEval-Bench-Pipeline extrahiert zusaetzlich atomare Fakten aus User- und Assistant-Messages — das ist eine bench-spezifische Ingestion-Optimierung, kein produktiver Storage-Verfahren.)
- **Englisch-bias bei Embeddings**. Voyage-3 ist multilingual aber EN-trained. Deutsche und spanische Searches funktionieren, sind aber bei sehr nuancierten paraphrasierten Queries gelegentlich schwaecher als EN. Roadmap: dedicated DE/ES Embedding-Models in Q3-Q4.

Wir kommunizieren das offen weil ein ehrliches Produkt langfristig erfolgreicher ist als eines mit Marketing-Mauer.

---

10. References

- LongMemEval Benchmark: arXiv 2410.10813 (Wu et al., 2024), github.com/xiaowu0162/LongMemEval
- MCP Spec 2025-06-18: modelcontextprotocol.io/specification
- MCP Apps Spec 2026-01-26: blog.modelcontextprotocol.io/posts/2026-01-26-mcp-apps
- Zep Paper: arXiv 2501.13956
- Mastra Observational Memory: mastra.ai/blog/observational-memory

Repos:

- Public Listing: github.com/studiomeyer-io/studiomeyer-memory
- n8n Custom Node: github.com/studiomeyer-io/n8n-nodes-studiomeyer-memory
- n8n Templates: github.com/studiomeyer-io/n8n-templates

Live-Endpoints:

- Memory-API: `https://memory.studiomeyer.io/mcp`
- Health: `https://memory.studiomeyer.io/health`
- Marketing-LP: studiomeyer.io/services/memory
- Blog: studiomeyer.io/de/blog/ai-memory-erklaert

11. Kontakt

- **Email**: hello@studiomeyer.io
- **Founder**: Matthias Meyer (StudioMeyer)
- **GitHub Org**: studiomeyer-io

Stand: 10. Juni 2026, Version 1.0.3 — Memory v3.19.0 mit Dream-Cycle, Retrieval-Reinforcement und Zwei-Signal-Abstention.