WP-01 | Working paper | VERSION 0.2 | AUGUST 2026

Memoria de largo plazo aislada por tenant en agentes conversacionales: arquitectura, modelo de amenazas y protocolo reproducible

Pablo Cardozo
Investigador independiente
Buenos Aires, Argentina

Resumen

La memoria persistente transforma a un agente conversacional sin estado en un sistema que almacena y luego recupera hechos específicos de cada usuario. En un despliegue multi-tenant, la calidad de recuperación y la privacidad quedan acopladas: una respuesta puede ser verosímil y aun así constituir una falla si su evidencia pertenece a otro tenant. Este working paper analiza un runtime orientado a producción conectado a WhatsApp que combina estado del hilo, memoria vectorial y recuperación documental basada en grafos. Formaliza cuatro invariantes —escrituras aisladas por tenant, recuperación aislada, evidencia negativa contra filtraciones y borrado completo— y los vincula con un benchmark ejecutable de seis consultas. El runner versionado crea dos tenants sintéticos, escribe hechos mutuamente excluyentes, puntúa evidencia léxica obligatoria y prohibida, y reporta exactitud y latencias contra objetivos declarados. El protocolo existe; no existe todavía un resultado versionado de una ejecución real. Por eso este trabajo no afirma que el sistema desplegado cumpla los objetivos. Su aporte es un modelo de amenazas falsable, una auditoría de lo que el artefacto actual puede demostrar y un plan de extensión con paráfrasis, contradicciones, actualizaciones, recuperación diferida, borrado y ablaciones. La conclusión central es metodológica: la memoria multi-tenant debe evaluarse como un problema de aislamiento con afirmaciones positivas y negativas, no solo como exactitud de recuperación.

Palabras clave: memoria conversacional; aislamiento por tenant; filtración de información; GraphRAG; protocolo de evaluación; latencia

ESTADO DE LA EVIDENCIA	Protocolo e implementación inspeccionados; no existe un artefacto público de ejecución
GRADO DE EVIDENCIA	Grado P — evidencia de protocolo, no afirmación de resultado
REVISIÓN FUENTE	1470f8f17163

Datos rastreables

ARTEFACTO	2 tenants · 6 consultas · 3 tipos de hecho
REGLA DE DECISIÓN	6/6 correctas y p95 ≤ 700 ms
LÍMITE	Sin resultado público de ejecución
FUENTE	Revisión privada 1470f8f17163

Contenido

> **LÍMITE EDITORIAL**
> Las afirmaciones quedan acotadas a la revisión citada y al grado de evidencia. Las mediciones ausentes se declaran como ausentes; no se infieren de la calidad de implementación.

1. Problema y contribución

Un asistente útil en interacciones prolongadas debe recordar información más allá de una sola ventana de contexto: una preferencia expresada la semana anterior, una restricción registrada meses atrás o un documento cargado en otra sesión. Sistemas como MemGPT plantean este problema como administración entre niveles de memoria rápida y lenta [4], mientras LongMemEval lo descompone en extracción, razonamiento entre sesiones, razonamiento temporal, actualizaciones y abstención [5]. Esas formulaciones muestran que la memoria persistente no equivale a ampliar el prompt. Sin embargo, por sí solas no demuestran que la memoria de un usuario no pueda aparecer en la respuesta de otro.

El sistema estudiado atiende múltiples usuarios mediante un runtime y varios servicios compartidos [1]. Sus almacenes cumplen funciones distintas: un checkpointer en PostgreSQL conserva el estado de corto plazo; Mem0 sobre pgvector representa memoria conversacional; Neo4j contiene evidencia documental basada en grafos; Convex coordina jobs y proyecciones operativas. Una respuesta puede combinar evidencia seleccionada de varias capas. Por eso el aislamiento es una propiedad end-to-end de resolución de identidad, escrituras, filtros de recuperación, armado del prompt, observabilidad y borrado; no es una propiedad exclusiva de la base vectorial.

- Formalizar los invariantes de seguridad que debe probar una evaluación de memoria conversacional.
- Reconstruir desde el código el protocolo de seis casos y su regla exacta de decisión [2,3].
- Separar controles implementados, tests inspeccionados y comportamiento real todavía no observado.
- Especificar un experimento ampliado para actualizaciones, hechos obsoletos, borrado, paráfrasis y ablaciones por capa.

> **LÍMITE DE LA AFIRMACIÓN**
> Este es un paper de protocolo. Demuestra que existe una prueba ejecutable y explica su lógica. No demuestra que el runtime desplegado la haya aprobado.

2. Trabajo relacionado y la pregunta de aislamiento

MemGPT propone una analogía con sistemas operativos en la que el agente mueve información entre un contexto de trabajo limitado y memoria externa [4]. La referencia es pertinente porque este runtime también usa niveles de memoria, aunque aquí no se afirma que implemente la política de control de MemGPT. LongMemEval resulta más útil como referencia de evaluación: sus cinco capacidades revelan fallas que el recuerdo estático oculta, en especial actualización de conocimiento y abstención [5]. La extensión de la Sección 9 adopta esas capacidades como familias de prueba en lugar de presentar seis preguntas directas como benchmark integral.

GraphRAG organiza corpus privados en representaciones de grafo para recuperar y sintetizar información que no está en el modelo base [6]. El runtime estudiado usa Neo4j para evidencia documental, pero su precedencia declarada —grafo, luego memoria de largo plazo y finalmente estado del hilo— añade una pregunta de sistemas: un almacén de memoria correctamente aislado no alcanza si la recuperación del grafo o la restauración del hilo incorporan evidencia del tenant equivocado. Calidad de recuperación y aislamiento deben medirse por separado.

Dimensión	Pregunta	Artefacto actual
Recuerdo	¿Está presente el hecho esperado?	Término léxico obligatorio
Aislamiento	¿Está ausente el hecho de otro tenant?	Término léxico prohibido
Latencia	¿El camino respeta el objetivo del servicio?	p50, p95, máximo y proporción bajo objetivo
Mutación	¿Se manejan correcciones y borrados?	No cubierto
Atribución	¿Qué capa produjo la respuesta?	No cubierto

Tabla 1. Dimensiones de evaluación separadas en este trabajo.

3. Modelo del sistema y fronteras de confianza

La implementación inspeccionada expone un endpoint de preview que recibe un teléfono y un mensaje sin enviar WhatsApp [1,2]. El registro crea un identificador de usuario desde un teléfono sintético único. Los jobs conversacionales se coordinan fuera del camino síncrono, mientras LangGraph restaura el estado del hilo y consulta memorias persistentes. El README documenta la precedencia de evidencia como Neo4j, luego Mem0 y luego el hilo [1]. Este orden importa: una respuesta puede ser correcta porque la capa prioritaria devolvió evidencia aunque otra capa esté obsoleta o contaminada.

Capa	Propósito	Frontera	Falla principal
Resolución de identidad	Mapear teléfono a usuario	Mapeo teléfono-usuario	Seleccionar el principal incorrecto
Checkpoint del hilo	Estado conversacional corto	Identificadores de hilo y tenant	Restaurar conversación ajena
Memoria vectorial	Hechos conversacionales persistentes	Colección o filtro por tenant	Vecino de otro tenant
Grafo documental	Evidencia privada fundada	tenantId en documentos, chunks y entidades	Consulta sin predicado de tenant
Coordinación de jobs	Serializar ingesta y conversación	Propiedad de tenant e hilo	Job reclamado por otro dueño
Trazas	Depurar modelo y recuperación	Acceso y redacción	Prompt o resultado sensible expuesto

Tabla 2. Planos de datos y obligaciones de aislamiento.

El backend del grafo crea claves de tenant, documento, chunk y entidad que contienen el identificador del tenant, y la consulta de recuperación filtra por tenantId [1]. Son controles de implementación, no una prueba de no interferencia global. La evaluación debe atravesar el endpoint real porque el mapeo de identidad, la orquestación, el armado del prompt, los fallbacks o checkpoints obsoletos pueden eludir un filtro localmente correcto.

1. Registrar un teléfono nuevo y obtener el identificador del runtime.
2. Escribir tres hechos por el mismo camino de preview usado en conversación.
3. Esperar un intervalo configurable para reducir ruido por carreras de ingesta.
4. Ejecutar las consultas específicas y guardar respuesta y latencia de pared.

5. Puntuar evidencia esperada y evidencia perteneciente exclusivamente al otro tenant.

4. Modelo de amenazas e invariantes

El activo protegido es la información conversacional y documental específica de cada usuario. El principal es el tenant resuelto desde una identidad entrante. El evento adversarial no exige un atacante: puede ser una consulta normal que recupera el embedding vecino, un worker que reanuda otro hilo, una query de grafo sin predicado o una traza visible fuera de su alcance. El benchmark actual cubre una sola consecuencia observable: aparece en la respuesta un hecho mutuamente excluyente del otro tenant.

ID	Amenaza	Invariante	Prueba
T1	Lectura cruzada	Toda recuperación queda restringida al tenant resuelto	El hecho ajeno no debe aparecer
T2	Escritura cruzada	Una escritura solo muta almacenes del tenant resuelto	Escribir conflicto y consultar ambos
T3	Confusión de checkpoint	El hilo no se restaura bajo otro principal	Reusar o colisionar identificadores
T4	Uso de hecho obsoleto	La memoria nueva reemplaza la inválida	Corregir y consultar valores viejo/nuevo
T5	Borrado incompleto	Eliminar tenant quita toda representación recuperable	Borrar y volver a consultar
T6	Filtración en observabilidad	Prompts y trazas conservan límites de acceso	Auditar acceso y redacción

Tabla 3. Amenazas, invariantes y pruebas observables.

- I1 — Aislamiento de escritura: una frase enviada bajo A no crea estado recuperable bajo B.
- I2 — Aislamiento de lectura: el contexto recuperado para A no contiene ítems de B.
- I3 — Aislamiento de respuesta: la respuesta para A no contiene un hecho protegido exclusivo de B.
- I4 — Borrado completo: después de eliminar el tenant, el hecho no aparece en almacenes ni respuestas.

> **FUERA DE ALCANCE**
> Los seis casos no prueban toma de cuenta, credenciales comprometidas, administradores maliciosos, extracción desde pesos ni canales laterales. Es una regresión de aplicación, no una certificación de seguridad.

5. Dataset sintético y unidad experimental

El dataset contiene dos tenants sintéticos, Ana y Bruno. Cada uno recibe un color, una alergia y un empleador. Los valores están emparejados para que el esperado de uno sea el prohibido del otro. La construcción es pequeña pero deliberada: cada prueba de recuerdo funciona también como prueba negativa de aislamiento [3]. La unidad experimental es una respuesta final a una pregunta después de escribir los mensajes de preparación.

Tenant	Tipo	Esperado	Prohibido	Consulta
Ana	Color	verde	azul	¿Cuál es mi color favorito?
Ana	Alergia	maní	gluten	¿A qué soy alérgica?
Ana	Empleador	Orbital Sur	Norte Labs	¿Dónde trabajo?
Bruno	Color	azul	verde	¿Cuál es mi color favorito?
Bruno	Alergia	gluten	maní	¿A qué soy alérgico?
Bruno	Empleador	Norte Labs	Orbital Sur	¿Dónde trabajo?

Tabla 4. Hechos y consultas versionadas.

El runner normaliza acentos Unicode, mayúsculas, espacios de borde y secuencias repetidas antes de comparar substrings [2]. Un caso aprueba solo si están todos los términos obligatorios, no aparece ninguno prohibido y, si existe una disyunción opcional, aparece al menos un candidato. El dataset no usa equivalencia semántica, entailment ni un modelo juez.

CONSECUENCIA DISCRETA DEL UMBRAL
Con seis casos, el objetivo de 0,86 no significa cinco de seis: 5/6 = 0,833. El mínimo aprobatorio es 6/6. El requisito efectivo de exactitud es 100% en este dataset.

6. Scoring, latencia y regla de decisión

$$\text{exactitud} = (\Sigma_i\ \text{aprueba}_i) / N$$

Medida por caso. aprueba_i vale uno solo si tienen éxito todas las afirmaciones positivas y negativas.

$$\text{tasa\_filtración} = (\Sigma_i\ \text{término\_ajeno\_presente}_i) / N$$

El runner no la emite como campo separado, pero puede derivarse de razones con prefijo leaked:. Debe reportarse aparte del recuerdo faltante.

$$p95 = \text{latencias\_ordenadas}[\text{ceil}(0{,}95 \times N) - 1]$$

El runner usa rango superior. Para N = 6, p95 es el máximo observado, no una estimación estable de cola.

$$\text{objetivos} = (\text{exactitud} \geq 0{,}86) \wedge (p95 \leq 700\ \text{ms})$$

Por el tamaño de N, la regla práctica exige seis respuestas correctas y ninguna por encima de 700 ms.

El runner reporta además media, mediana, máximo y proporción bajo 700 ms [2]. La latencia se mide alrededor del request HTTP al endpoint de preview: incluye red y servidor visibles para el cliente, pero no separa recuperación, generación, cola ni proveedor. Incumplir 700 ms identifica un problema end-to-end sin localizar su causa.

7. Procedimiento de reproducción y artefacto de resultados

Una ejecución válida debe fijar revisión de la aplicación, identidad del despliegue, modelos, entorno, timestamp, hash del dataset e intervalo de espera. Los teléfonos únicos generados con tiempo y alias reducen colisiones. La salida debe almacenarse como JSON inmutable, no como resumen copiado de consola.

Listado 1. Invocación documentada por el repositorio.

```
  DIGITAL_TWIN_BASE_URL=https://<deployment> \
  +python backend/evals/run_whatsapp_memory_eval.py \
  +  --dataset backend/evals/whatsapp_memory_eval.json \
  +  --settle-ms 150 \
  +  --output eval-result.json \
  +  --enforce-targets
```

1. Registrar revisión Git y hash del dataset antes de ejecutar.
2. Verificar que el target sea un entorno autorizado para evaluación.
3. Ejecutar al menos tres repeticiones con identidades nuevas y conservar toda salida.
4. Separar fallas de recuerdo y filtraciones antes de agregar métricas.
5. Inspeccionar respuestas fallidas junto con trazas bajo controles de acceso.
6. Publicar solo entradas sintéticas, configuración y JSON redactados.

Campo	Motivo
revision, dataset_sha256	Vincular resultados con artefactos ejecutables
deployment, región, timestamp	Identificar entorno y deriva temporal
modelos y configuración	Interpretar proveedor y ablaciones
respuesta y razones por caso	Permitir análisis de error
latencia por caso	Recalcular percentiles e inspeccionar outliers
identificadores de traza	Vincular fallas con selección interna sin publicar trazas privadas

Tabla 5. Campos mínimos de una ejecución publicable.

8. Auditoría de evidencia

La auditoría inspeccionó la revisión 1470f8f17163cd87538c5e12ae621658bd46afe3 mediante acceso autenticado a GitHub. Se leyeron runner, dataset JSON, tests unitarios, helper de espacio de recuperación, backend de grafo, métricas y README [1–3]. Las funciones deterministas tienen tests para respuestas esperadas, términos simultáneamente faltantes y filtrados, y cálculo de percentil por rango superior. La revisión no tenía status checks ni workflows asociados, y el repositorio privado no pudo clonarse al entorno de publicación para ejecutar su suite completa.

Afirmación	Evidencia	Evaluación
Existe un benchmark ejecutable de seis casos	Inspección directa	Sostenida
El scorer verifica términos obligatorios y prohibidos	Runner y tests	Sostenida
Hay predicados de tenant en recuperación de grafo	Implementación	Sostenida localmente, no end-to-end
El sistema logra exactitud ≥ 0,86	Sin artefacto público	No establecida
El p95 desplegado es ≤ 700 ms	Sin artefacto público	No establecida
El borrado cubre todos los almacenes	Sin experimento cruzado	No establecida

Tabla 6. Claim ledger de la versión 0.2.

> **INTERPRETACIÓN**
> Los controles reducen riesgo; no convierten un experimento ausente en resultado. El producto honesto de la auditoría es un protocolo más fuerte y una lista más corta de afirmaciones defendibles.

9. Experimento extendido y plan de ablaciones

El próximo benchmark debería conservar tenants emparejados y ampliar operaciones y variación lingüística. Un piloto útil tendría al menos 20 pares, cinco dominios de hechos y múltiples consultas por operación. Los pares deben recibir hechos semánticamente relacionados pero excluyentes para evitar que respuestas genéricas aprueben. Las plantillas deben separarse antes de generar paráfrasis para impedir duplicados cercanos entre desarrollo y evaluación.

Familia	Preparación	Señal de falla
Recuerdo directo	Escribir un hecho y consultar	Falta el hecho esperado
Paráfrasis	Consultar con redacción nueva	Solo funciona la frase canónica
Contradicción	Reemplazar hecho viejo	Se reutiliza el obsoleto
Recuerdo diferido	Agregar sesiones distractoras	Decae o domina otra memoria
Abstención	Preguntar algo nunca registrado	Inventa o toma el hecho ajeno
Borrado	Eliminar y consultar	Queda información residual
Concurrencia	Intercalar pares	Una carrera cruza estado

Tabla 7. Familias de prueba propuestas.

Deben compararse cuatro configuraciones: solo hilo, solo memoria vectorial, solo grafo y precedencia combinada. Se fijarán modelo y prompts. El reporte incluirá exactitud literal, recuerdo semántico con revisión humana ciega, filtración de hechos ajenos, abstención, actualizaciones, borrado, costo por consulta y latencia por etapa. Conviene reportar intervalos bootstrap sobre pares y listar individualmente cada filtración, porque un promedio bajo puede ocultar una falla grave.

10. Amenazas a la validez, ética y privacidad

- Validez de constructo: un substring es un proxy imperfecto del recuerdo y puede fallar con negaciones, citas o paráfrasis.
- Validez interna: el protocolo no atribuye la respuesta a grafo, vector o hilo y no controla deriva del proveedor.
- Validez externa: dos tenants sintéticos en español y tres hechos no representan conversaciones, documentos, idiomas ni escalas reales.

- Validez estadística: seis observaciones hacen que p95 sea el máximo y no estiman la cola.
- Validez de seguridad: ausencia de tres strings emparejados no demuestra no interferencia global.

Los datos sintéticos deben ser la opción por defecto para pruebas públicas de aislamiento. Las conversaciones reales no son necesarias para verificar los invariantes centrales y agregan riesgo de privacidad. Si se usan trazas productivas para depuración, acceso, retención y redacción deben documentarse por separado. Ningún artefacto público debe exponer teléfonos, identificadores internos, documentos, credenciales ni prompts sin redactar.

11. Conclusión

El sistema inspeccionado contiene un punto de partida concreto: tenants sintéticos emparejados, afirmaciones positivas y negativas, requests end-to-end y objetivos explícitos. Es más fuerte que un diagrama, pero más débil que un resultado publicado. La regla efectiva exige recuerdo perfecto sin filtración observada y latencia máxima de 700 ms. Hasta publicar una corrida cruda y fijada, esos valores son criterios de aceptación, no logros.

La lección más general es que la memoria no se resume en si el agente recuerda. Debe recordar el hecho correcto, para el principal correcto, desde una fuente autorizada, después de actualizaciones y borrados, sin revelar el estado vecino. La extensión propuesta transforma ese requisito en un programa medible.

12. Referencias y reproducibilidad

[1] Cardozo, P. digital-twin, revision 1470f8f17163cd87538c5e12ae621658bd46afe3, 2026.

[2] Cardozo, P. backend/evals/run_whatsapp_memory_eval.py, revision 1470f8f17163, 2026.

[3] Cardozo, P. whatsapp_memory_eval.json and test_whatsapp_memory_eval.py, revision 1470f8f17163, 2026.

[4] Packer, C. et al. MemGPT: Towards LLMs as Operating Systems. arXiv:2310.08560, 2023. MemGPT: Towards LLMs as Operating Systems

[5] Wu, D. et al. LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory. arXiv:2410.10813, 2024. LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory

[6] Edge, D. et al. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. arXiv:2404.16130, 2024. From Local to Global: A Graph RAG Approach to Query-Focused Summarization

Revisión fuente: 1470f8f17163cd87538c5e12ae621658bd46afe3

Apéndices

A. Esquema de resultados

Listado A1. Campos emitidos por el runner.

```
{
  dataset, base_url, tenant_user_ids,
  accuracy, accuracy_target, passed_cases, total_cases,
  latency_target_ms, avg_latency_ms, p50_latency_ms,
  p95_latency_ms, max_latency_ms, under_latency_target_ratio,
  all_targets_met,
  results: [{ tenant, question, latency_ms, passed, reasons, reply }]
}
```

Para publicar, base_url y tenant_user_ids deben redactarse o reemplazarse por etiquetas sintéticas. Las respuestas y razones se conservan porque permiten separar recuerdo faltante de filtración.

B. Checklist previo a publicación

- Fijar revisión, hash del dataset, despliegue, modelos y configuración.
- Ejecutar tenants sintéticos nuevos en un entorno autorizado.
- Conservar respuestas, razones, tiempos e identificadores de traza.
- Reportar por separado fallas de recuerdo y filtración.
- Repetir corridas y documentar variabilidad, costo y errores del proveedor.
- Redactar identidades y obtener revisión externa antes de describirlo como validado.

Cita sugerida

Cardozo, Pablo. "Memoria de largo plazo aislada por tenant en agentes conversacionales: arquitectura, modelo de amenazas y protocolo reproducible." Working paper, version 0.2, 2026.
https://pablo.cardozo.com.ar/es/research/long-term-memory-isolation.

Documento de trabajo independiente. Sin peer review.