Memoria Persistente Local-First en Antigravity (Capas L0–L3)

Cómo convertimos la práctica real creando mods y cursos 4K en agentes especialistas

Laboratorio Verificado: `laboratorio_memoria_persistente_antigravity.py` · **Visualizador:** `visualizador_memoria_persistente_antigravity.html`
Motor en Producción: `persistent-memory` (`memctl.py`, `memory_mcp_server.py`, `hooks.json`) · **Fuente:** *Memory for Personal AI* (`urn:orm:video:0642572475086`)

1. Cómo Trabajamos con Antigravity y el Choque Teoría vs Realidad Empírica

En nuestro trabajo diario con **Antigravity** (dirigiendo agentes basados en modelos de frontera como **Carbon**), una sola persona opera un estudio completo de desarrollo: construimos mods complejos para **Fallout 4** (*Crónicas del Refugio 13*, *Piper Wright Local Generative AI Companion*, *Project Astro-Cosmos*), mods reactivos de diálogos y voz para **Cyberpunk 2077** (*Ghost Echo*, *Chispa NC*) y una tubería automatizada de cursos técnicos en 4K a 60 fps.

El punto de partida del modelo impresiona por su dominio teórico sobre arquitectura de motores comerciales —grafos de escena, máquinas virtuales de scripting y gestión binaria, tal como expone **Jason Gregory** en *Game Engine Architecture, 3rd Edition*. Sin embargo, al bajar a la trinchera del ejecutable real en Windows, esa teoría choca constantemente contra peculiaridades empíricas no documentadas de cada compilador y parche: investigar `Papyrus.0.log`, `redscript_rCURRENT.log` o `cyber_engine_tweaks.log`, descartar intuiciones falsas y aislar la causa raíz exige tiempo y miles de tokens.

Si ese conocimiento práctico queda encerrado en el historial volátil de un chat, al día siguiente el agente sufre amnesia operativa y vuelve a caer en la misma trampa. Consolidarlo con rigor convierte cada obstáculo superado en especialización acumulativa.

> **Impacto Cuantificado en Nuestro Laboratorio (`laboratorio_memoria_persistente_antigravity.py`):**
>
> En frío, diagnosticar una trampa combinada en plugins `.esl` exige **5 turnos y 9.240 tokens**. Con el sistema `L0–L3`, el agente consulta el índice, recupera la solución por `FTS5 + BM25` y converge a la primera (**Pass@1 en 1 turno**) con **1.120 tokens**: un **87,9 % de ahorro en tokens de misión** y entre **87,0 % y 90,3 %** de ahorro en sobrecarga fija de arranque.

2. Límites Reales Descubiertos (`24.000 B` y `20.000 Tokens`) y Visión de Duncan Davidson

Al principio anotábamos las lecciones en un único `AGENTS.md` y un diario plano creciente. En producción chocamos contra dos límites arquitectónicos de Antigravity: el tope duro de **24.000 bytes** por archivo de regla (después de expandir inclusiones `@[label](ruta)`), donde `Mods_Fallout/AGENTS.md` alcanzó **23.585 B** (**98,3 %** antes del truncamiento silencioso), y el presupuesto conjunto **Always-On** de **20.000 tokens**, que obliga a limpiar la regla global `GEMINI.md` (`5.321 B -> 1.364 B`) y mover reglas de subdominio bajo demanda.

Cruzando la charla de **Duncan Davidson** (*Memory for Personal AI*, O'Reilly) con la literatura de **O'Reilly Learning** (Labaschin, Se, Johnson, Alake, Osmani, Koenigstein, Breunig, Jambor, Tekiner, Huyen y Albada), adoptamos cuatro pilares: **(1) Fin de silos propietarios**, **(2) Local-First y Escotilla de Escape (Escape Hatch)** en archivos Markdown bajo Git, **(3) Acceso por Ámbitos (Scoped Access)** contra el **Context Rot**, y **(4) Compuerta anti-envenenamiento con Trampa Semántica y Provenance**.

3. Arquitectura Real en 4 Capas (`L0–L3`) e Integración Nativa en Antigravity

Capa	Artefactos en Disco	Mecánica de Carga	Límites y Guardianes
L0 · Constitución	`AGENTS.md` y `GEMINI.md` global	Inyección incondicional (**Always-On**) al abrir cada sesión.	Mantener `< 12.000 B`; alerta `>= 16.000 B`; tope `24.000 B` y techo `20.000 tok`.
L1 · Modular	`.agents/rules/*.md` y `SKILL.md`	Carga progresiva (`trigger: model_decision`, 30 tok en reposo).	Tope seguro `<= 18.000 B`. Destino del ascenso automático cuando $N \geq 2$.
L2 · Episódica	`memoria/MEMORIA_INDICE.md` y `diarios/*.md`	Única Fuente de Verdad legible en Markdown y auditada en Git.	Ahorro fijo de `87,0 %–90,3 %` frente a cargar diarios completos.
L3 · FTS5 + MCP	`memoria_fts.sqlite`, `memctl.py`, `hooks.json`	Búsqueda `BM25`, federación solo lectura entre proyectos y hooks.	Hooks `SessionStart` + `Stop` expanden `@includes` y evitan bucles.

4. Casos Reales Capturados en Modding y Producción Audiovisual 4K

Documentar únicamente qué cambió en el código deja al agente vulnerable a repetir su primera intuición errónea. Por eso cada entrada en nuestros diarios separa explícitamente la **Trampa Semántica** (qué parecía lógico pero falló en vivo) de la **Solución Determinista**:

ID	Proyecto / Motor	Trampa Semántica Descartada	Solución Determinista Verificada
FO4-EP-001	**Fallout 4** · Adytum	Quitar PlayerFaction con Aggression = 2.0 y lanzar StartCombat.	Aliar grupo en facción común (self=Ally) con Aggression = 1.0.
FO4-EP-005	**Fallout 4** · West Tek	Mover TERM persistente con SetPosition() en celdas ya en .fos.	GetPositionX() conserva coordenada vieja; crear instancia virgen.
FO4-EP-008	**Fallout 4** · .esl / .ba2	Máscara 24 bits (0x00FFFFFF) y .ba2 v8 en 1.10.163.	Máscara 12 bits (0x00000FFF), HEDR v1.0, .ba2 v1 y LatestCLR.
CP77-EP-002	**Cyberpunk** · Redscript	Opción única amarilla con fondo sin animar (LinearWipe_0).	Envolver DialogChoiceLogicController fijando LinearWipe_0 = 0.00.
CP77-EP-004	**Cyberpunk** · Audioware	Fiar de Session/Ready único y Duration() sin idioma ni dilación.	Resincronizar en ticks 1, 3, 7, pasar LocaleExt y affectedByTimeDilation = false.
DC-EP-001	**Camtasia 4K** · .tscproj	Pasar len(text) Unicode a rangeEnd en rótulos con tildes o rayas.	Calcular len(text.encode("utf-8")) porque C++ indexa en bytes UTF-8.
DC-EP-004/5	**ElevenLabs** · ES/EN	Deletrear siglas sueltas (hache) o subir Similarity al 95 % en inglés.	Aislar "...",, escribir ache, usar Multilingual v2 y Similarity 68–75 %.

5. Anatomía del Motor, Telemetría Real y Guardián Automático `Stop`

El motor `memctl.py` (Python 3 estándar, cero dependencias externas) y el servidor `memory_mcp_server.py` operan cinco garantías deterministas en producción:

1. **Tokenización Unicode segura en `SQLite FTS5 + BM25`:** `sanitize_fts_terms()` sanea puntos, guiones y dos puntos (`Papyrus.0.log`, `FO4-EP-001`, `TweakDB:SetFlat`) evitando errores de `MATCH`.
2. **Compuerta Anti-Envenenamiento (`PROVISIONAL -> VERIFIED`):** `record_memory()` rechaza (`REJECTED_MEMORY_POISONING`) cualquier entrada con `exit_code != 0`, errores en logs o sin **Trampa Semántica** ni **Provenance**; bloquea el ascenso prematuro hasta confirmar en vivo.
3. **Ascenso modular $N \geq 2$ hacia Capa L1:** Las incidencias verificadas repetidas se promueven a `.agents/rules/*.md` (`<= 18.000 B`) con `trigger: model_decision` sin inflar `AGENTS.md`.
4. **Expansión de inclusiones `@[label](ruta)` y presupuesto global:** Mide el tamaño efectivo expandido (hasta profundidad 3) y suma todas las reglas `always_on` del workspace y de `~/.gemini/` frente al techo de `20.000 tokens`.
5. **Guardián `SessionStart + Stop` sin alterar permisos:** Al iniciar sesión registra una línea base; al finalizar el turno, `hook-stop` exige refactorizar (`decision: "continue"`) si una regla creció por encima de `16 KB / 18 KB / 18.500 tok`, cortando cualquier bucle mediante `continuationActive`.

Proyecto Registrado	`AGENTS.md` Actual	Reglas L1	`Always-On` (20.000)	Ahorro Índice · Entradas
DesarrolloCursos (DC)	3.860 B (16,1 %)	3 (model_decision)	2.082 tok (10,4 %)	87,0 % · 7 VERIFIED
Mods_Fallout (FO4)	5.071 B (21,1 %, antes 23.585 B)	4 (model_decision)	3.557 tok (17,8 %)	88,3 % · 8 VERIFIED
Mods_cyberpunk2077 (CP77)	6.525 B (27,2 %)	2 (model_decision)	2.749 tok (13,7 %)	90,3 % · 14 VERIFIED

6. Cápsula de Transferencia Universal: Del Modding a Cualquier Software

Este mismo diseño escala sin cambios a **Unreal Engine 5**, **Unity**, **Godot** o cualquier pila de ingeniería de software: basta conectar la compuerta de verificación al compilador o suite de pruebas del proyecto para que todo incidente depurado hoy se convierta en aciertos **Pass@1** mañana.

Pilar `L0–L3`	En Modding y Producción 4K	En Ingeniería de Software General
Capa L0 (`< 12 KB`)	Reglas nucleares `.esp/.ba2`, `@wrapMethod`, audio 4K.	Contratos de API, seguridad y arquitectura base del repo.
Capa L1 (`model_decision`)	Subdominios `Nexus`, `Audioware`, `Camtasia`, `ElevenLabs`.	Reglas por pila: `Kubernetes`, `PostgreSQL`, `Rust`, `WebGPU`.
Capa L2 (`diarios/` Git)	Bitácoras con **Trampa Semántica** y log real del juego.	**Post-mortems** y decisiones `ADR/` versionadas con el código.
Capa L3 (`FTS5 + Stop`)	Valida `PapyrusCompiler`, `redscript-cli` o `typst`.	Exige `cargo test`, `tsc --noEmit` o `pytest` antes de grabar.

7. Citas de Autoridad y Bibliografía Oficial O'Reilly (`MCP`)

> *«Cada herramienta de IA recuerda hoy una versión distinta de nosotros; la única salida sostenible es un hogar local-first de contexto propiedad del usuario, portable en Markdown y expuesto por ámbitos bajo permisos explícitos.»* — **Duncan Davidson** (`urn:orm:video:0642572475086`)
>
> *«Sin mecanismos estrictos de validación y olvido selectivo durante la consolidación, la memoria persistente se convierte en un amplificador de errores conocido como envenenamiento de memoria.»* — **Ksenia Se & Pete Johnson** (`urn:orm:video:0642572428136`)

#	Título y Autor(es)	Identificador (`ourn`)	Enlace O'Reilly
1	**Memory for Personal AI with Duncan Davidson** — B. Lorica & D. Davidson	`urn:orm:video:0642572475086`	Abrir en O'Reilly
2	**Agent Memory** — Benjamin Labaschin (*Early Release*)	`urn:orm:book:0642572370473`	Abrir en O'Reilly
3	**Managing Memory for AI Agents** — B. Labaschin et al. (2025)	`urn:orm:book:9798341661257`	Abrir en O'Reilly
4	**Expert Showcase: Agent Memory** — K. Se, L. Liu & P. Johnson (2026)	`urn:orm:video:0642572428136`	Abrir en O'Reilly
5	**Build Self-Improving Agents with Memory, Traces, and Skills** — R. Alake	`urn:orm:live-event-series:0642572457082`	Abrir en O'Reilly
6	**Agentic Engineering** — Addy Osmani (*Early Release*)	`urn:orm:book:0642572392291`	Abrir en O'Reilly
7	**Harness Engineering** — Nicole Koenigstein (*Early Release*)	`urn:orm:book:0642572422783`	Abrir en O'Reilly
8	**The Context Engineering Handbook** — Drew Breunig (*Early Release*)	`urn:orm:book:0642572260705`	Abrir en O'Reilly
9	**Agentic Reliability Engineering** — David Jambor (2026)	`urn:orm:book:0642572294809`	Abrir en O'Reilly
10	**Architecting Data Systems for AI Agents** — Firat Tekiner (*Early Release*)	`urn:orm:book:0642572354664`	Abrir en O'Reilly
11	**Game Engine Architecture, 3rd Edition** — Jason Gregory (2018)	`urn:orm:book:9781351974271`	Abrir en O'Reilly
12	**AI Engineering** — Chip Huyen (2024)	`urn:orm:book:9781098166298`	Abrir en O'Reilly
13	**Building Applications with AI Agents** — Michael Albada (2025)	`urn:orm:book:9781098176495`	Abrir en O'Reilly