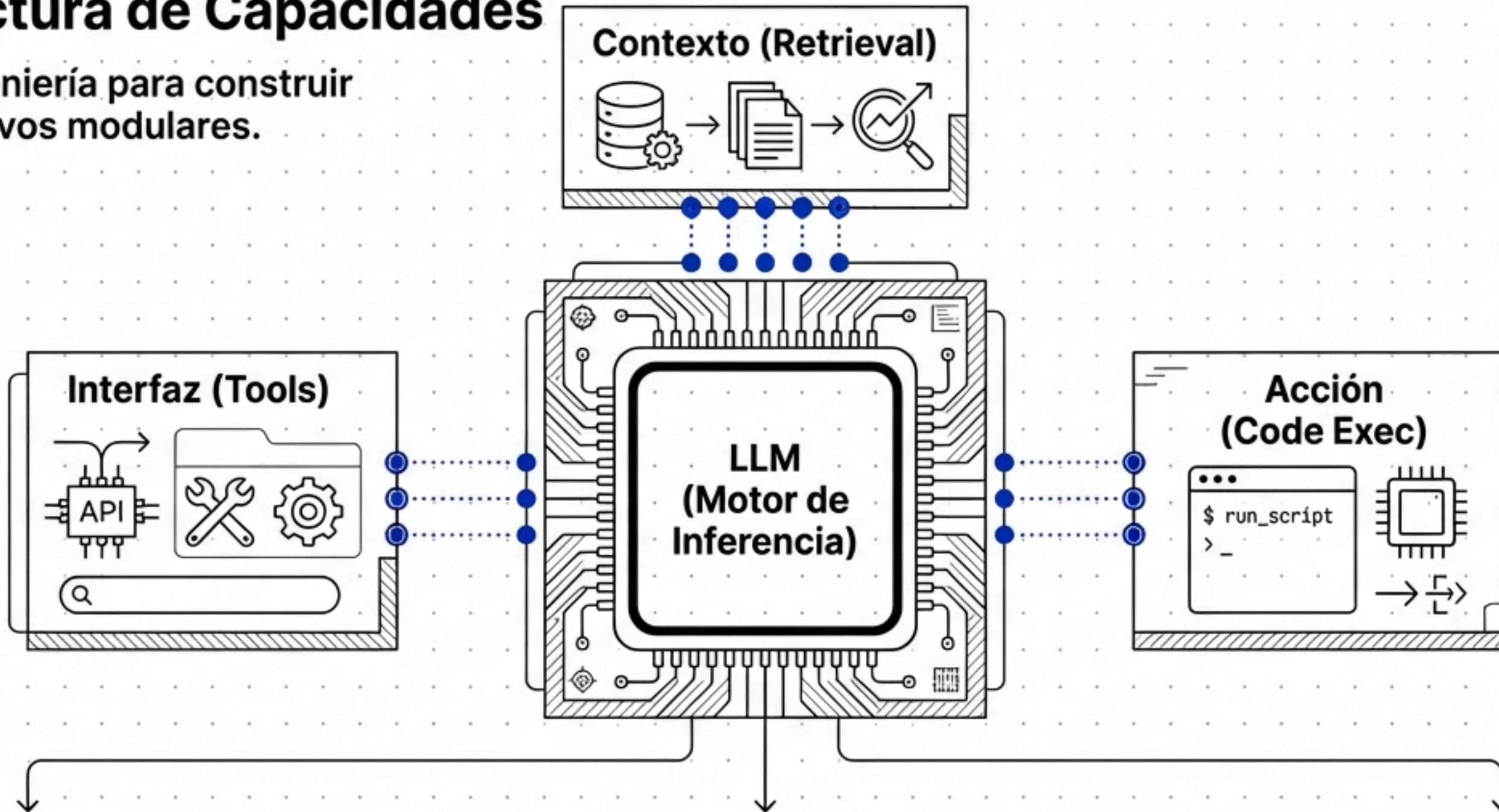


De Chatbot Pasivo a Agente Activo: La Arquitectura de Capacidades

Una guía de ingeniería para construir
sistemas cognitivos modulares.

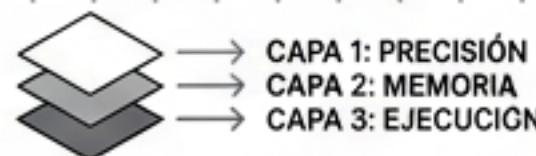


Un modelo de lenguaje sin herramientas
está aislado y limitado a su entrenamiento
estático.

Para construir un agente real, debemos integrar
capas arquitectónicas que resuelvan
problemas específicos: precisión, memoria y
ejecución lógica.

Propósito: Desglosar la implementación
técnica de las tres capacidades
fundamentales que transforman texto en
acción.

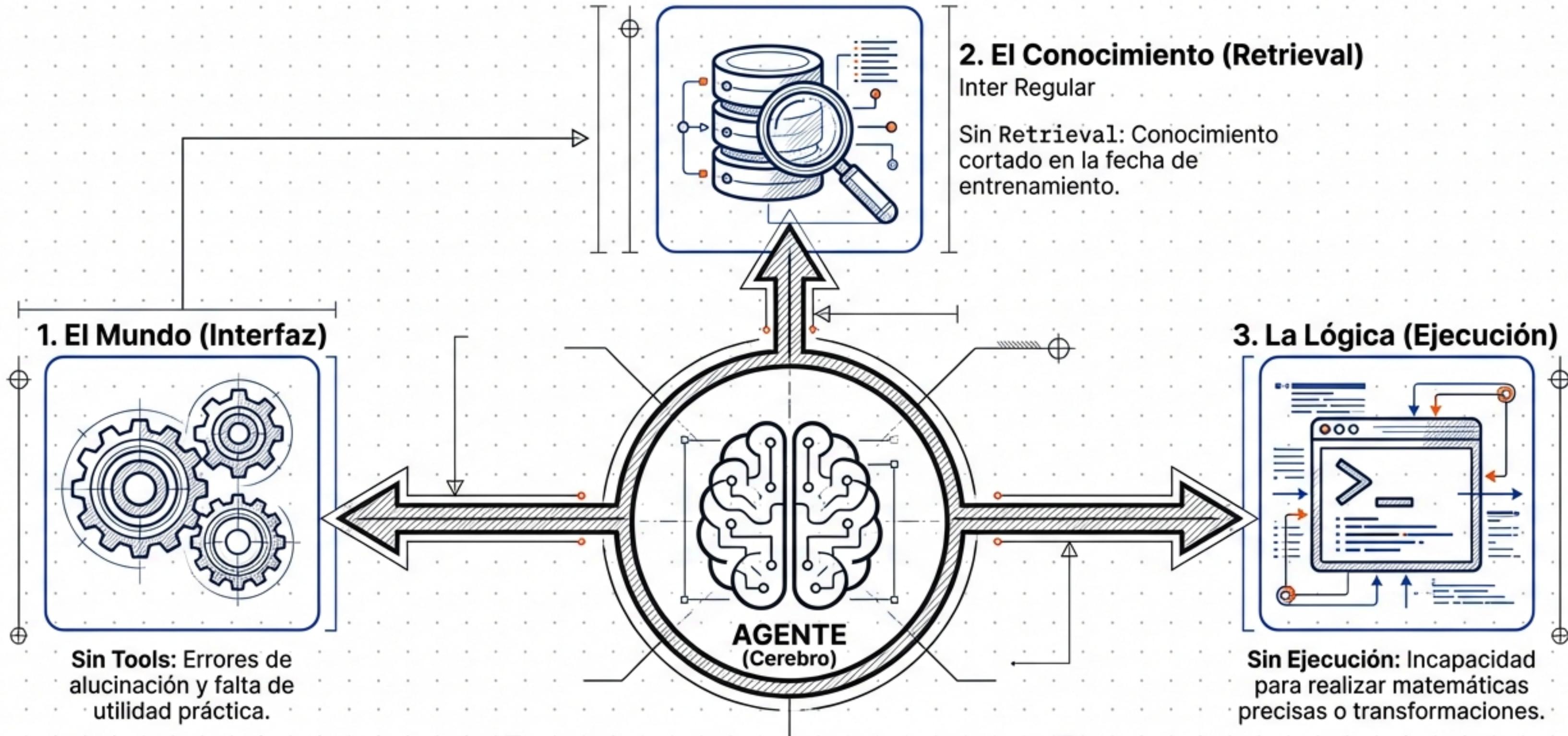
ADVERTENCIA: CAPACIDAD LIMITADA.
ESTADO: AISLADO.
SIN INTERFAZ EXTERNA.



→ CAPA 1: PRECISIÓN
→ CAPA 2: MEMORIA
→ CAPA 3: EJECUCIÓN

OBJETIVO: IMPLEMENTACIÓN MODULAR.
ESTADO: EN PROGRESO.
JSON CONFIG: { "modules": ["tools", "retrieval", "code_exec"] }.

Anatomía de un Agente: Tres Módulos Críticos



El Diseño de la Interfaz Determina la Inteligencia

El modelo no 'adivina' qué herramienta usar; deduce su utilidad basándose en la definición que usted escribe.



Responsabilidad Única

Atomicidad absoluta. Una tool hace una sola cosa bien.

Naming Convention

Patrón estricto `Verbo + Sustantivo` (ej: `buscar_usuario`, no `gestor_datos`).

Contexto de Uso

La descripción debe explicar explícitamente **cuándo** usar la herramienta.

Tipado Fuerte

Parámetros con restricciones claras y ejemplos.

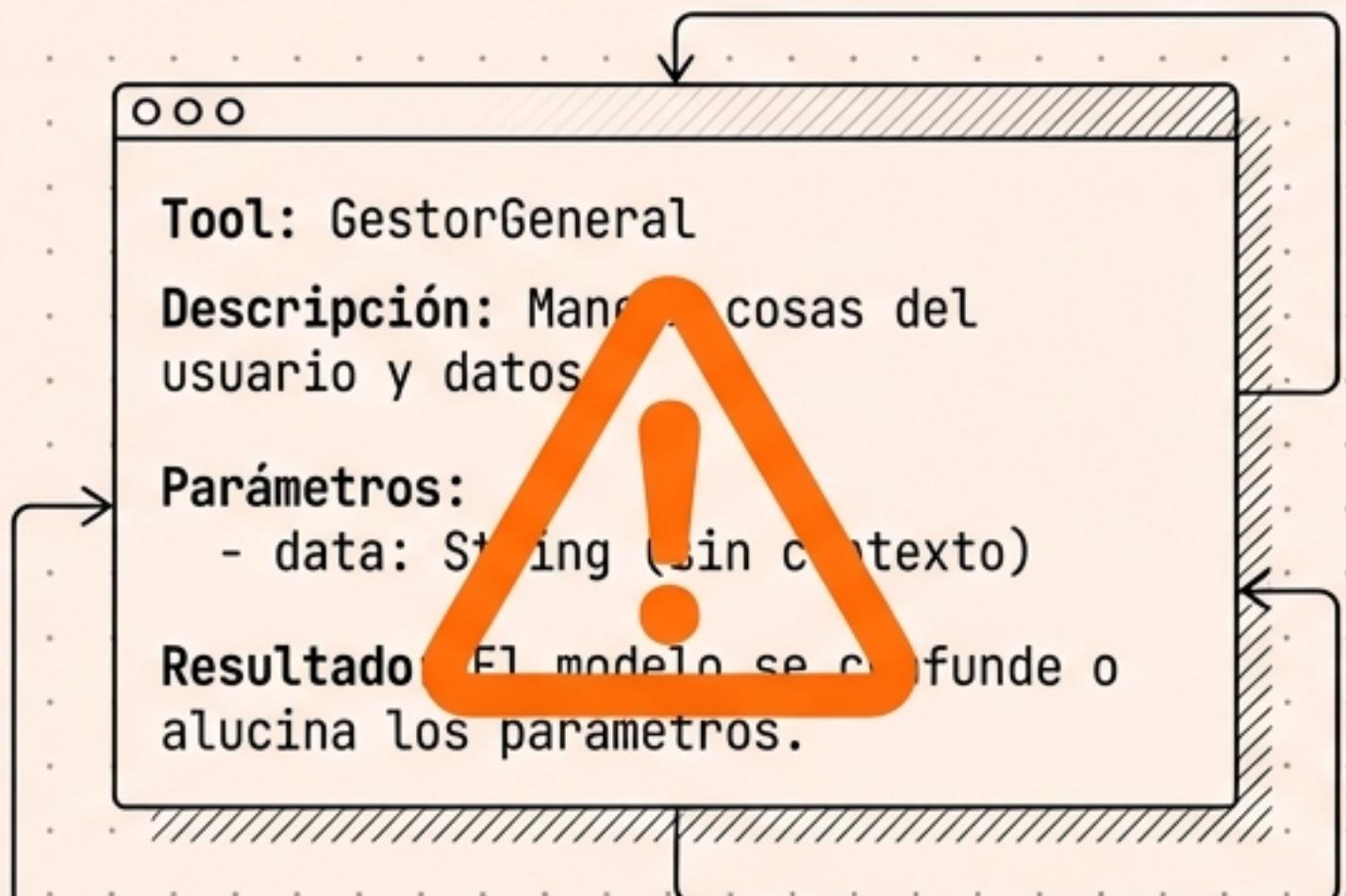
Predictibilidad

Respuestas estructuradas que el modelo puede parsear.

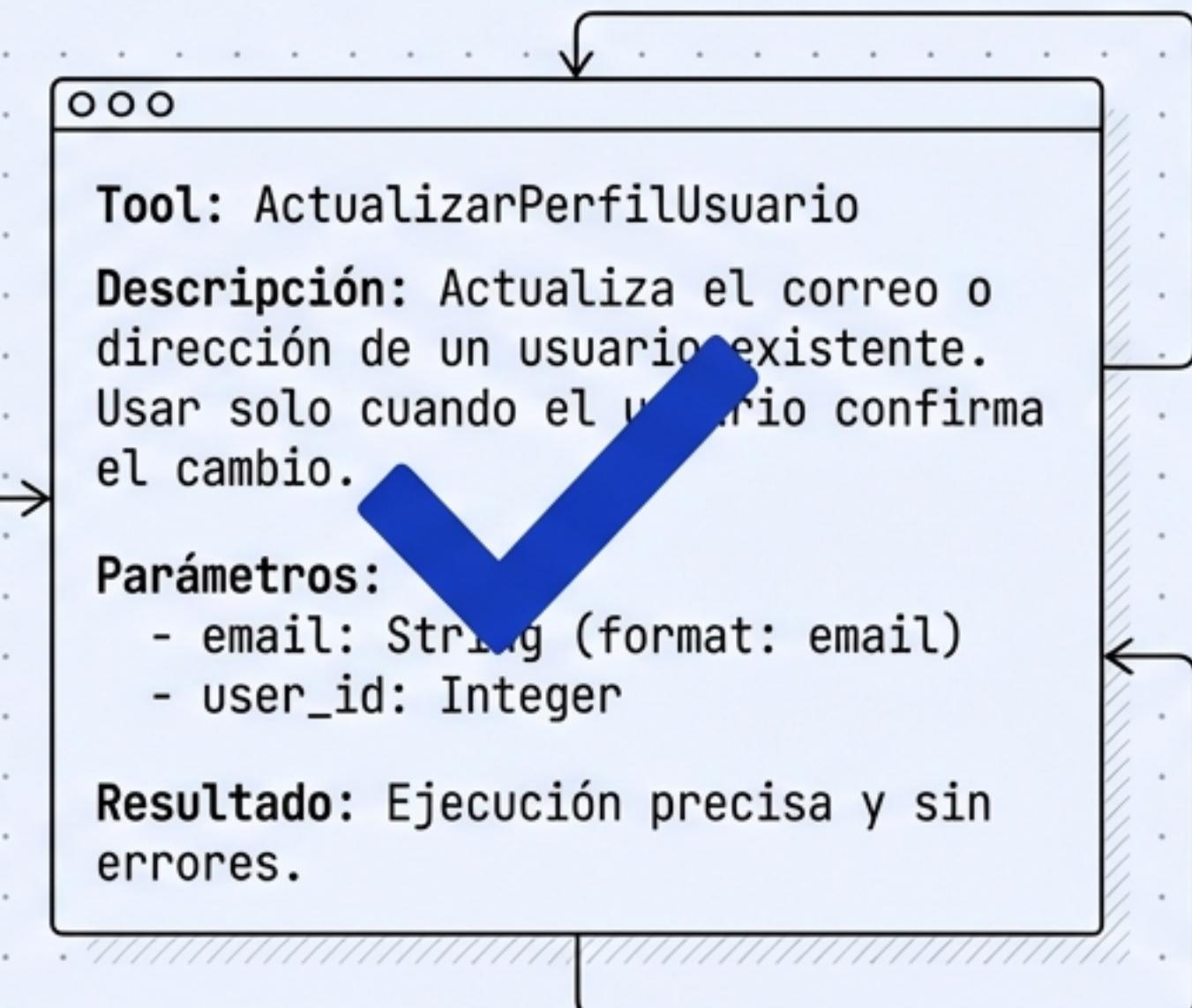
Un mal diseño de tools puede hacer que incluso el mejor modelo falle consistentemente.

Patrón vs. Anti-patrón: La Diferencia en la Práctica

✗ ANTI-PATRÓN (Ambigüedad)



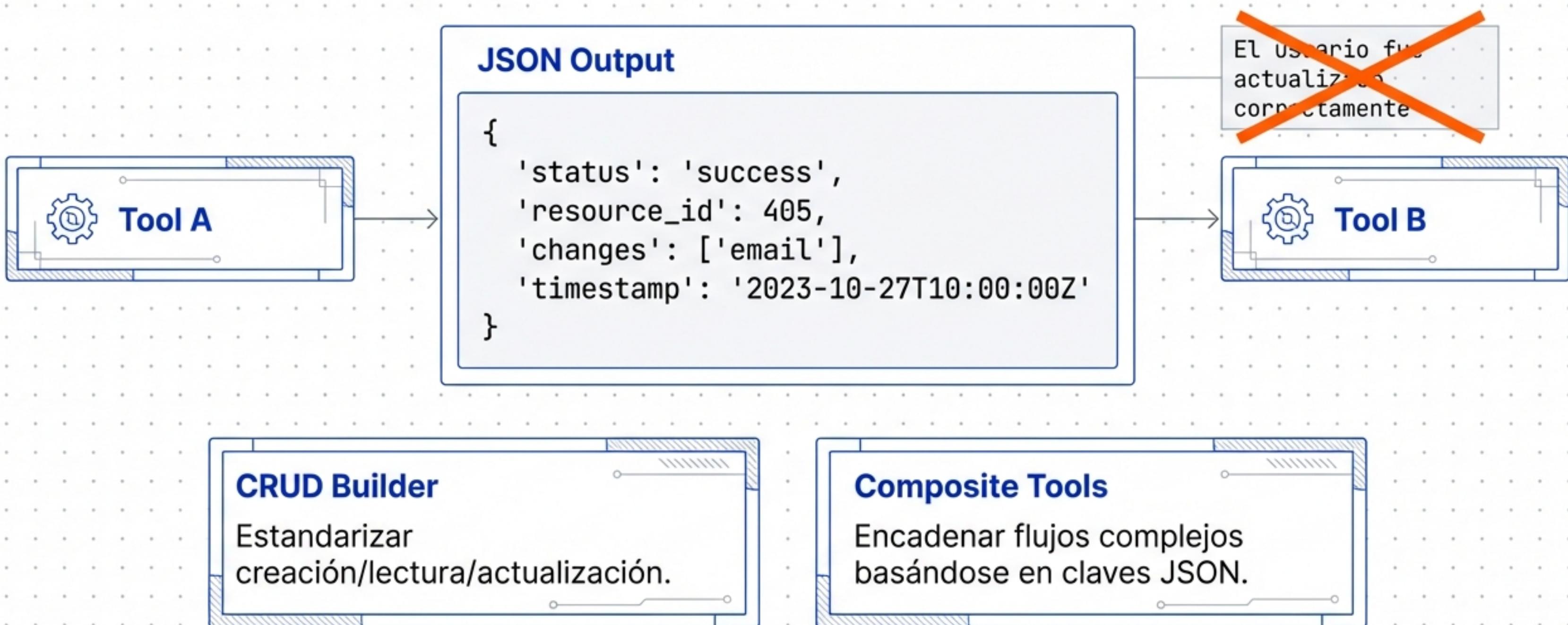
✓ PATRÓN (Claridad)



La ambigüedad en el diseño es la causa #1 de errores de invocación.

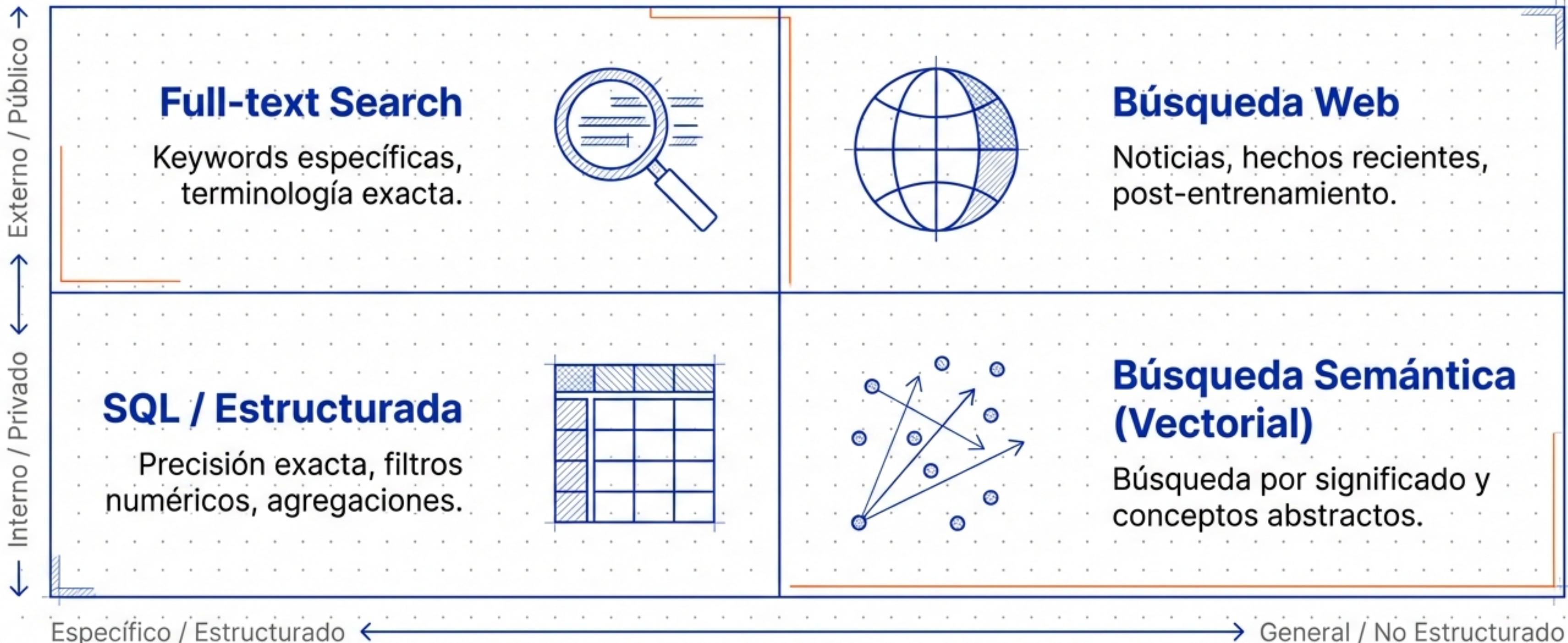
Respuestas Estructuradas para la Composabilidad

Para que las tools trabajen juntas, la salida de una debe ser la entrada de otra.
El texto libre rompe esta cadena.



Retrieval: El Puente hacia el Conocimiento Externo

Taxonomía de Estrategias de Recuperación



El Ciclo de Vida del Patrón RAG



Query + Recuperación
de candidatos.

Reordenamiento por
relevancia real.

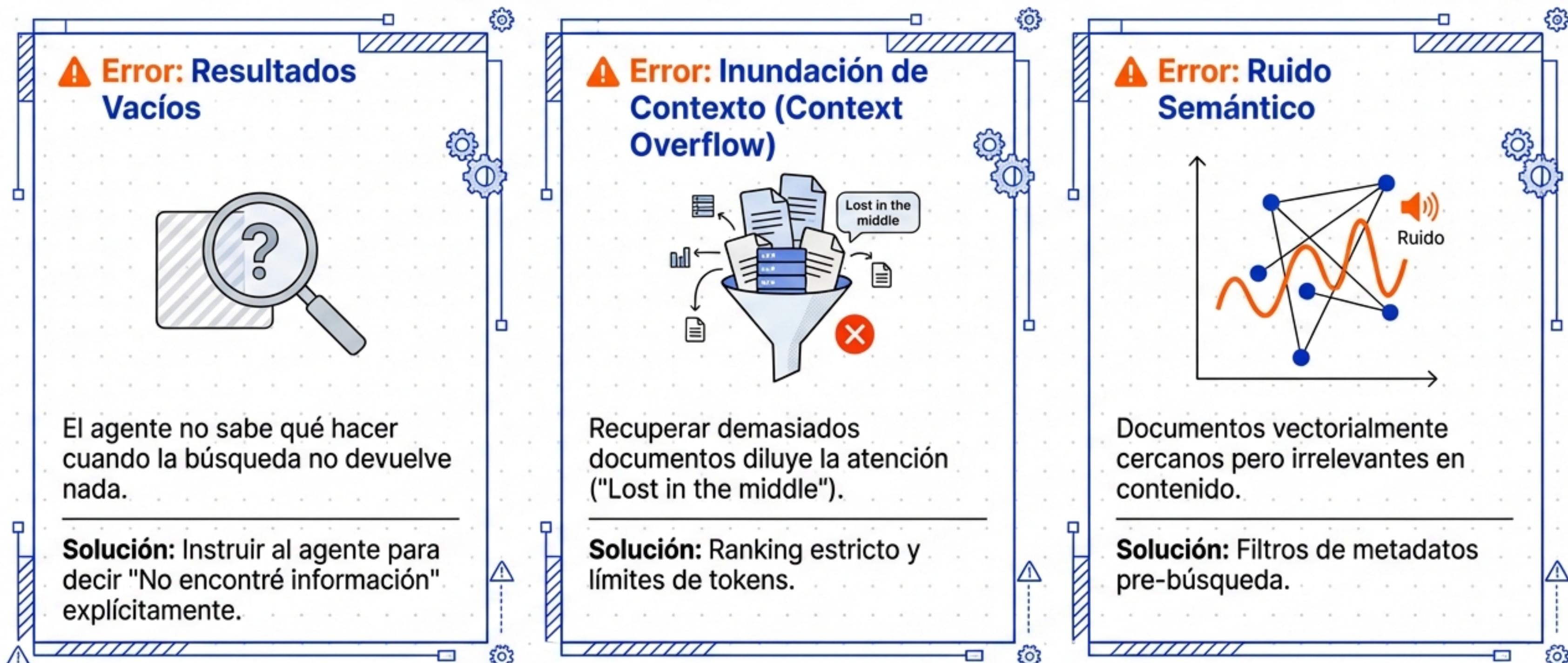
Inyección de fragmentos
en el prompt.

Síntesis de respuesta
basada en evidencia.

Insight Crítico: La calidad de la respuesta depende totalmente de las etapas 1 y 2. Garbage in, Garbage out.

Debugging de Retrieval: Errores Frecuentes

Guía de troubleshooting para problemas comunes en la fase de recuperación.



Ejecución de Código: De Poeta a Ingeniero

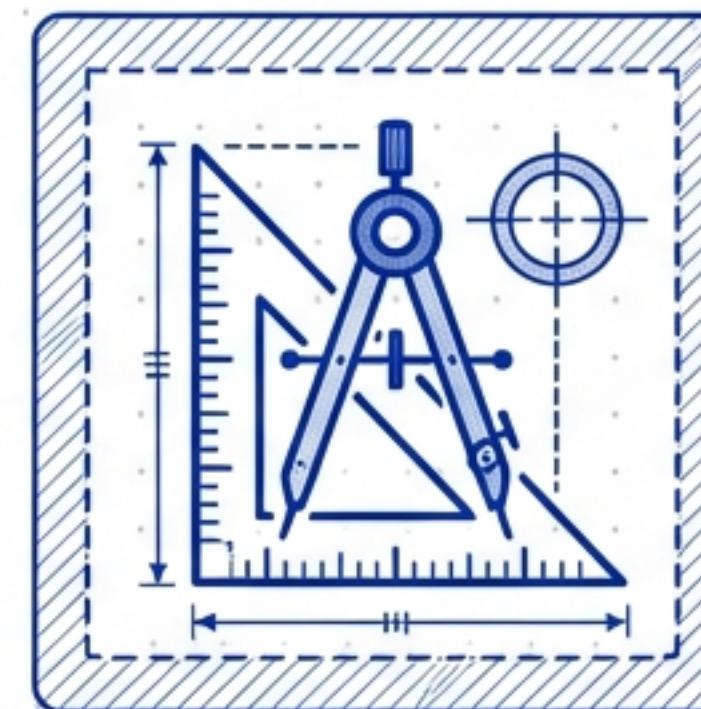
Delegando la lógica compleja a un intérprete confiable.



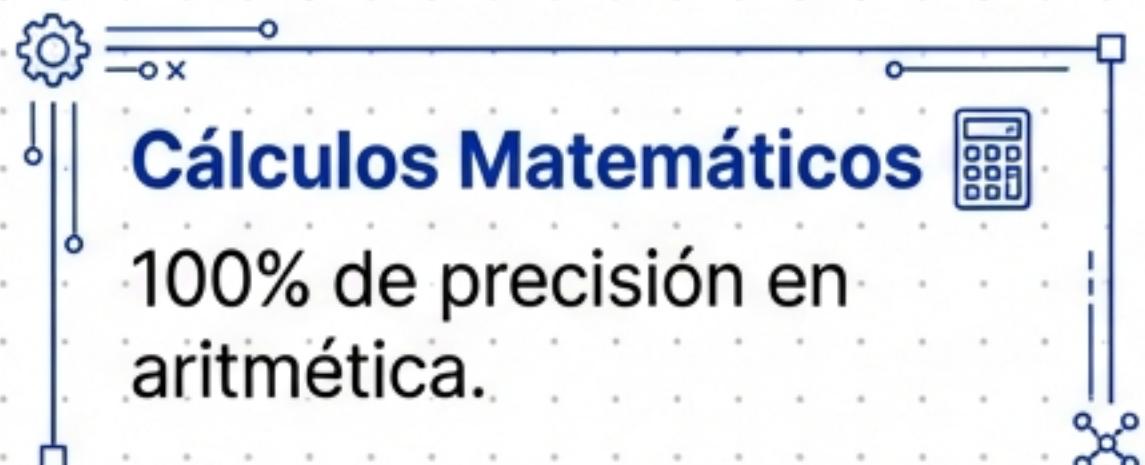
Poeta/LLM



Python Exec



Ingeniero



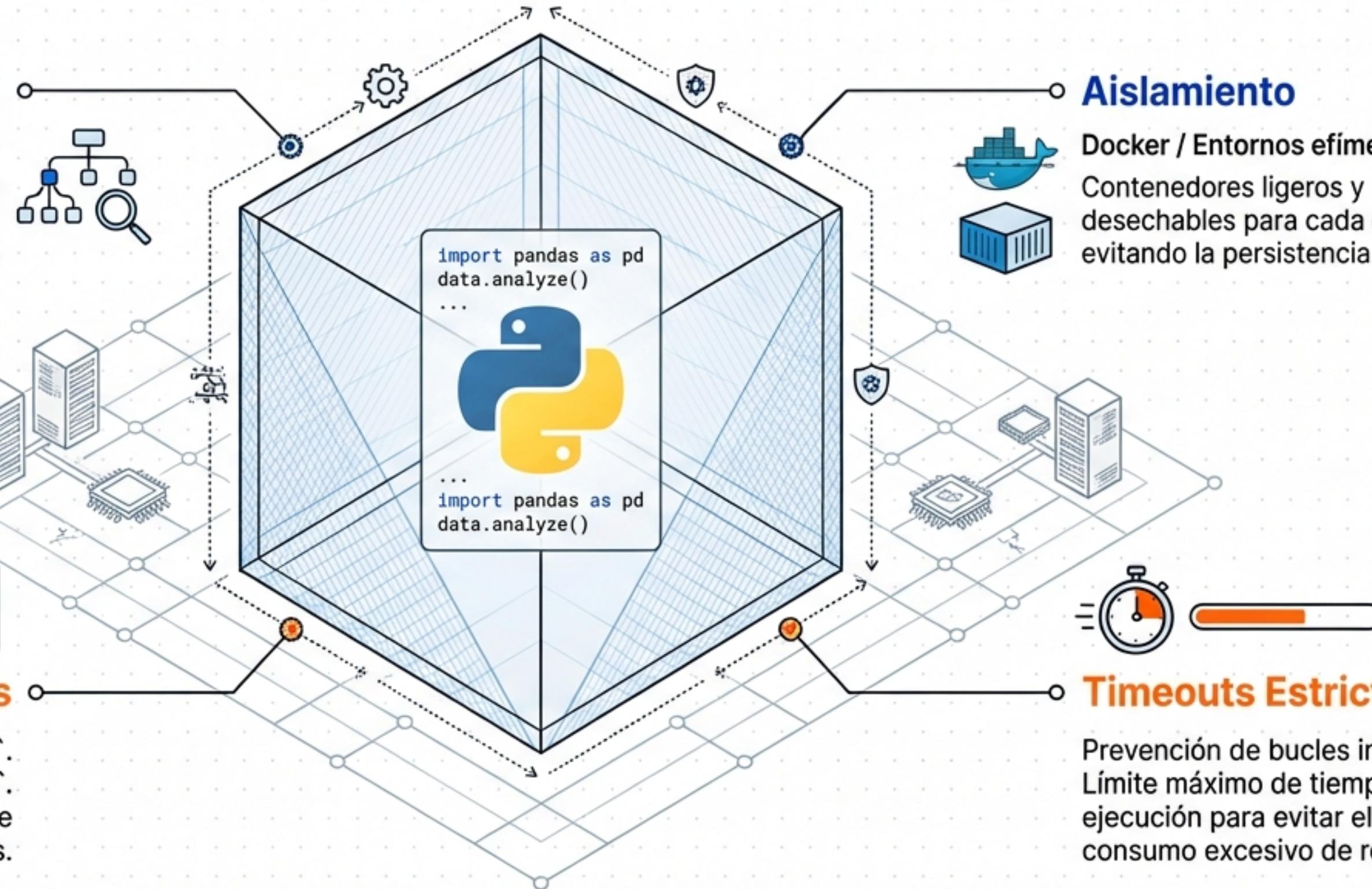
El Sandbox: Seguridad en la Ejecución

Mecanismos de defensa para la ejecución controlada de código.

Validación de AST

Análisis de sintaxis pre-ejecución.

Identificación de estructuras de código peligrosas antes de que se ejecuten.



Whitelist de Imports

Permitir: `'pandas'`, `'numpy'`.

Bloquear: `'os'`, `'sys'`, `'net'`.

Restricción estricta de bibliotecas accesibles.

Aislamiento

Docker / Entornos efímeros.

Contenedores ligeros y desechables para cada ejecución, evitando la persistencia.

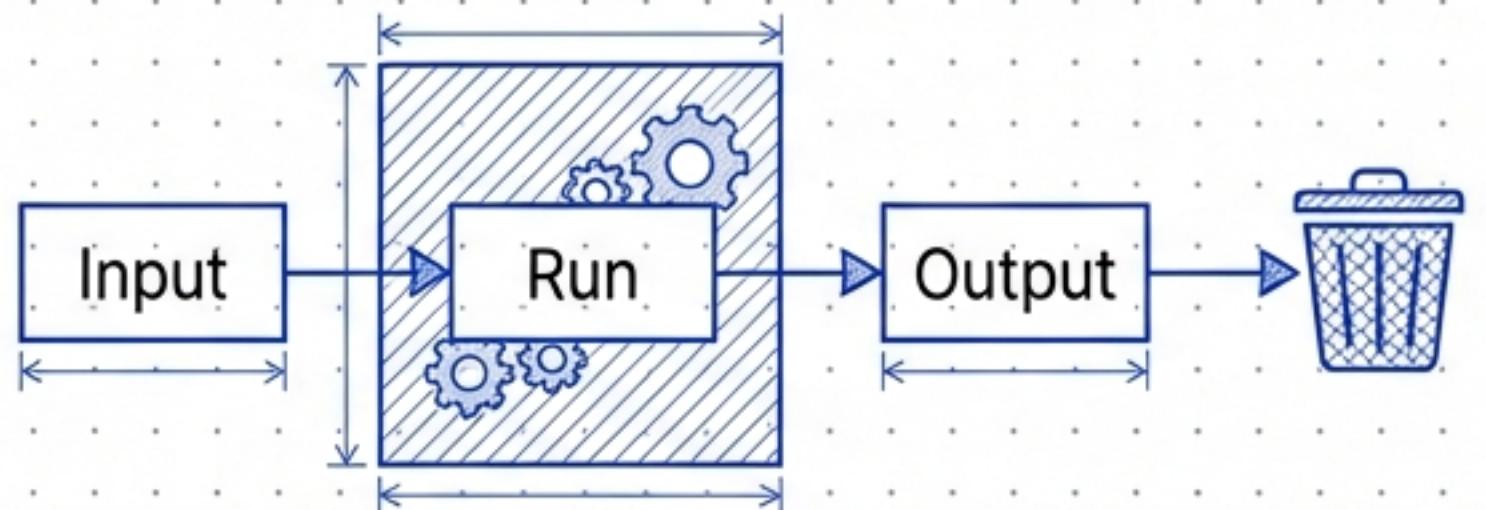
Timeouts Estrictos

Prevención de bucles infinitos.

Límite máximo de tiempo de ejecución para evitar el consumo excesivo de recursos.

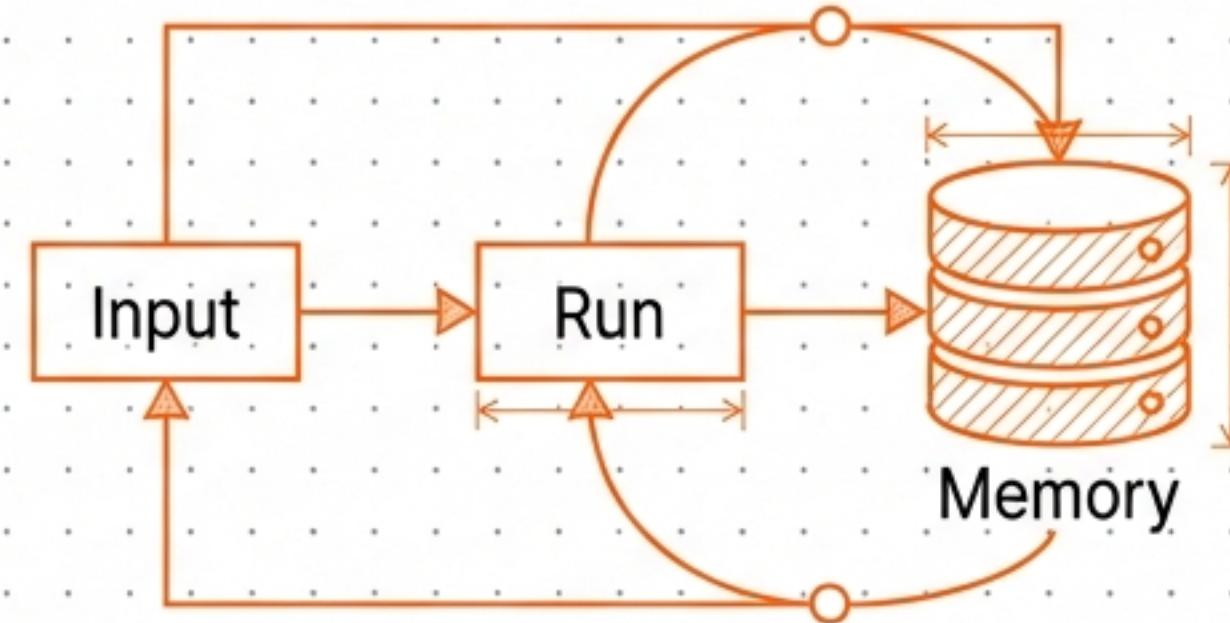
Patrones de Implementación de Ejecución

Patrón A: Stateless (Contenedor Efímero)



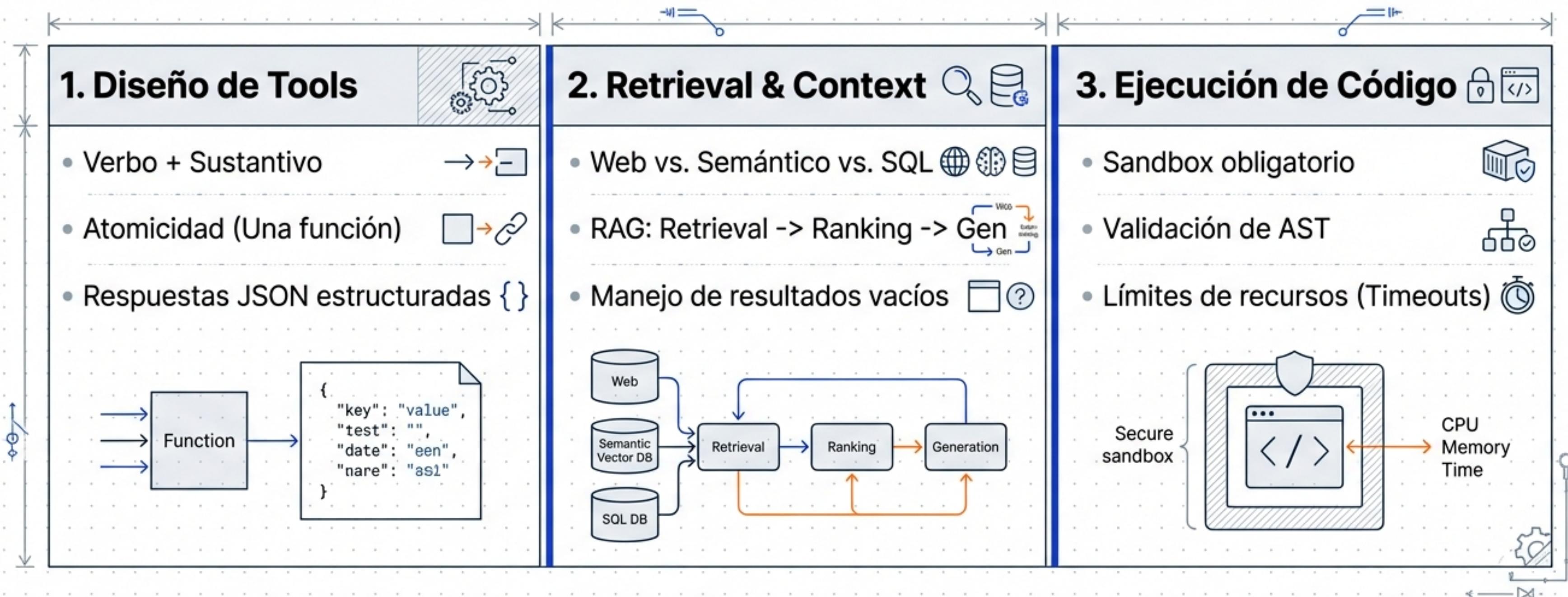
- **Uso:** Tareas de un solo paso.
- **Pros:** Máxima seguridad, limpieza automática.
- **Contras:** No hay memoria entre ejecuciones.

Patrón B: REPL con Estado (Interactive Session)



- **Uso:** Análisis exploratorio, depuración.
- **Descripción:** Mantiene variables como Jupyter.
- **Requisito:** Gestión de ciclo de vida de sesión.

La Arquitectura de la Agencia: Hoja de Ruta



La verdadera inteligencia del agente no reside en el modelo, sino en la arquitectura de sus herramientas.