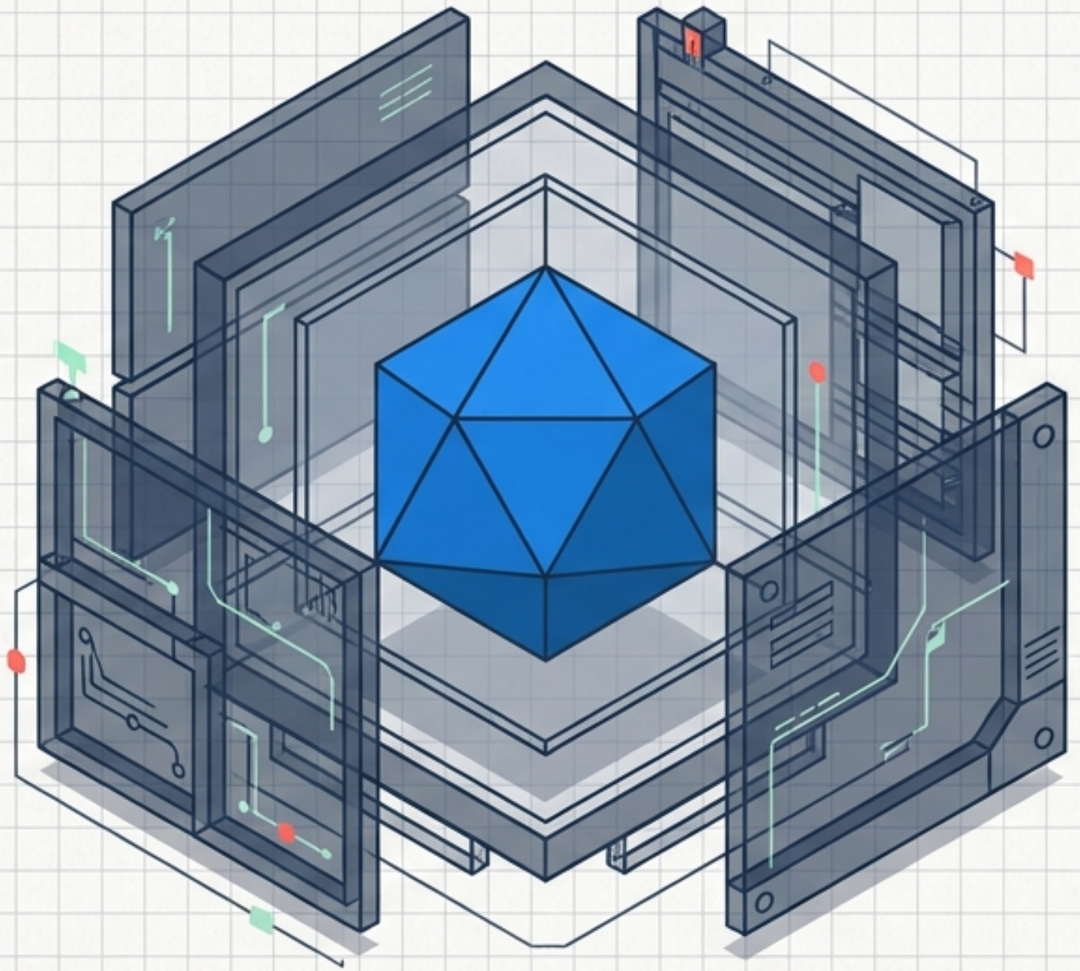
Cómo Microsoft Despliega Agentes de IA a Escala Empresarial

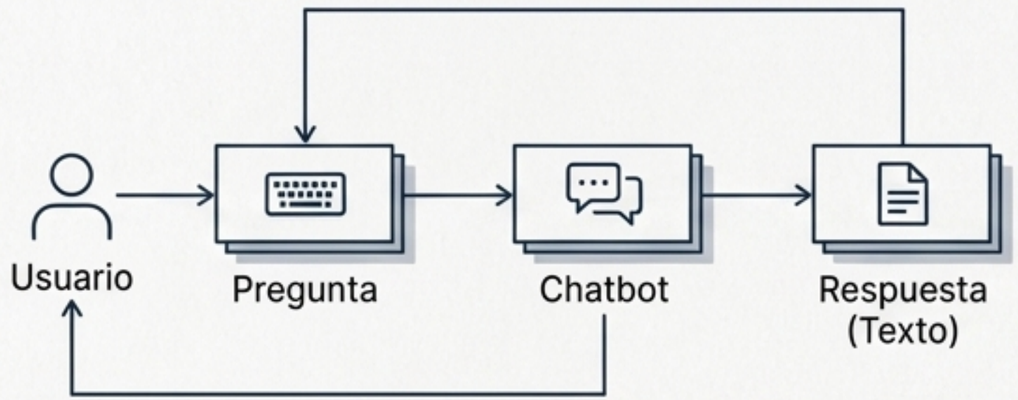
Lecciones de ingeniería y producción desde Microsoft Foundry, la plataforma que impulsa IA para más de 80,000 empresas y soporta a más de 20 millones de usuarios activos.



Hemos abandonado la era de responder preguntas para entrar en la era de ejecutar trabajo real.

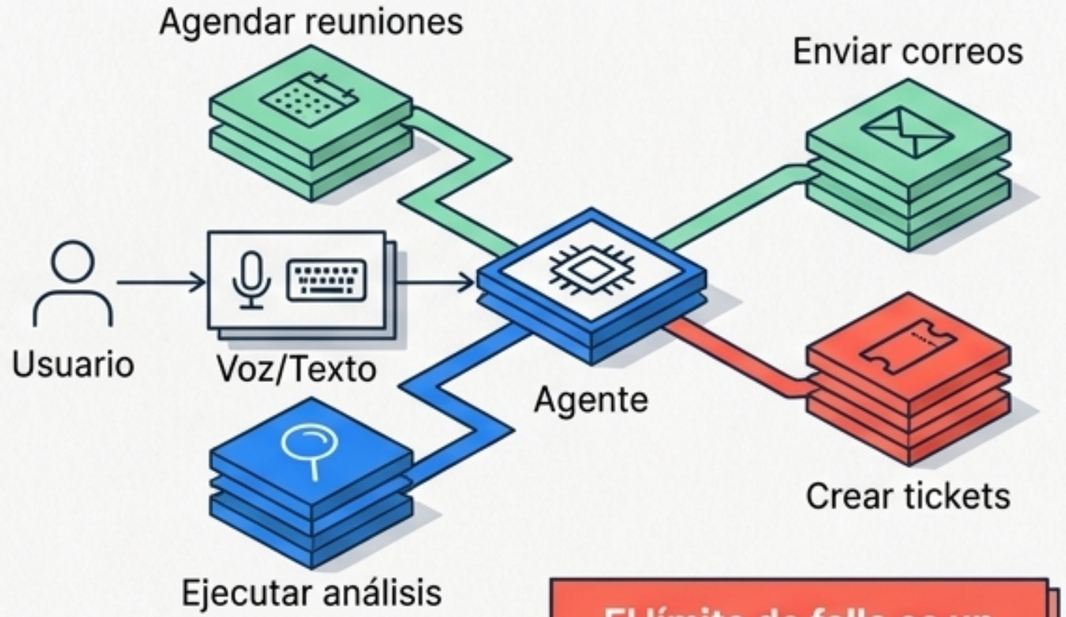
En 2026, el uso de voz como interfaz principal expone una nueva realidad. Los agentes ejecutan acciones significativas en nombre del usuario, elevando drásticamente el **estándar de fiabilidad requerido** para llegar a **producción**.

El Pasado: Chatbots



El límite de falla es una mala experiencia.

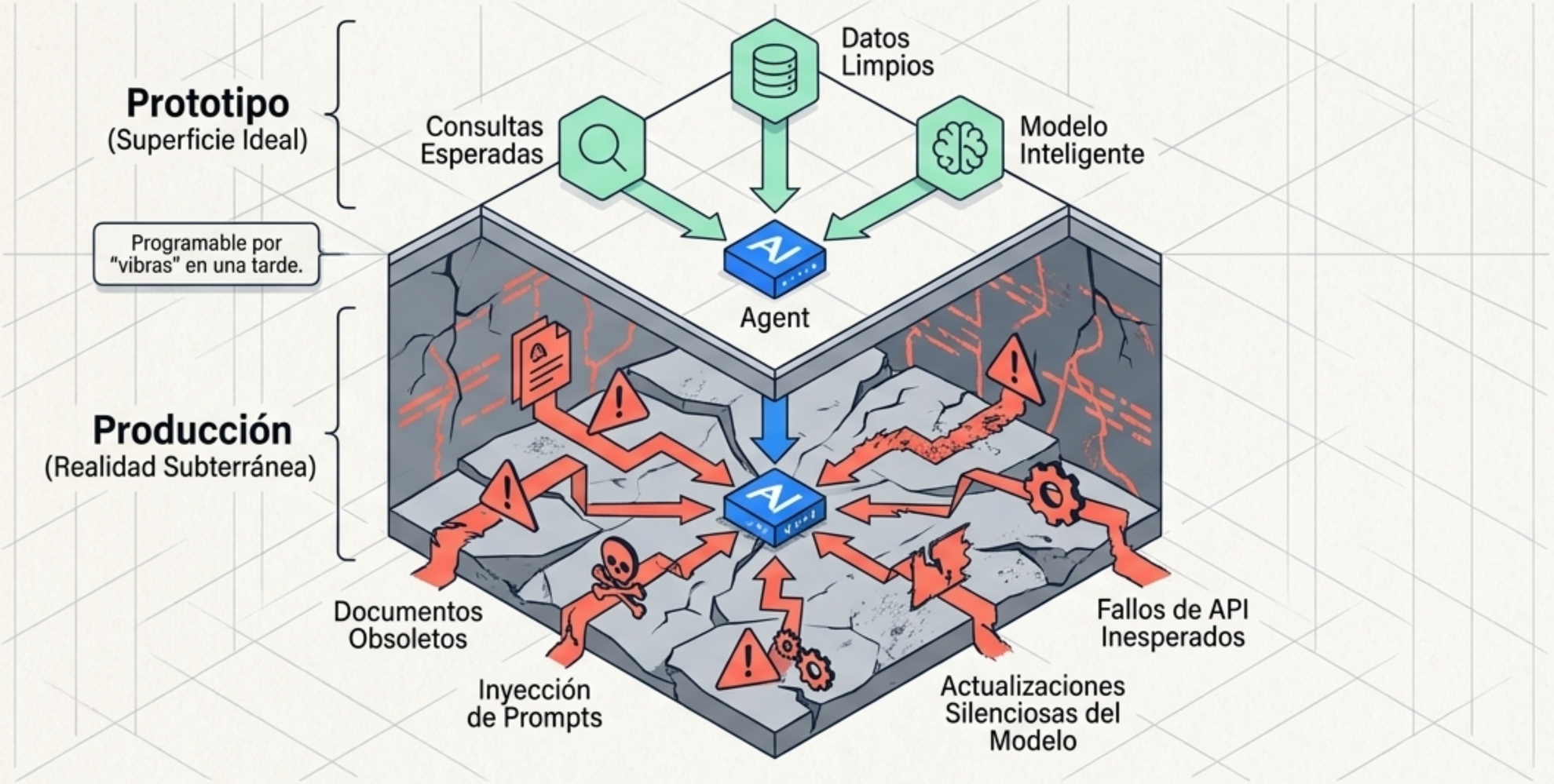
El Presente: Agentes



El límite de falla es un incidente de negocio.

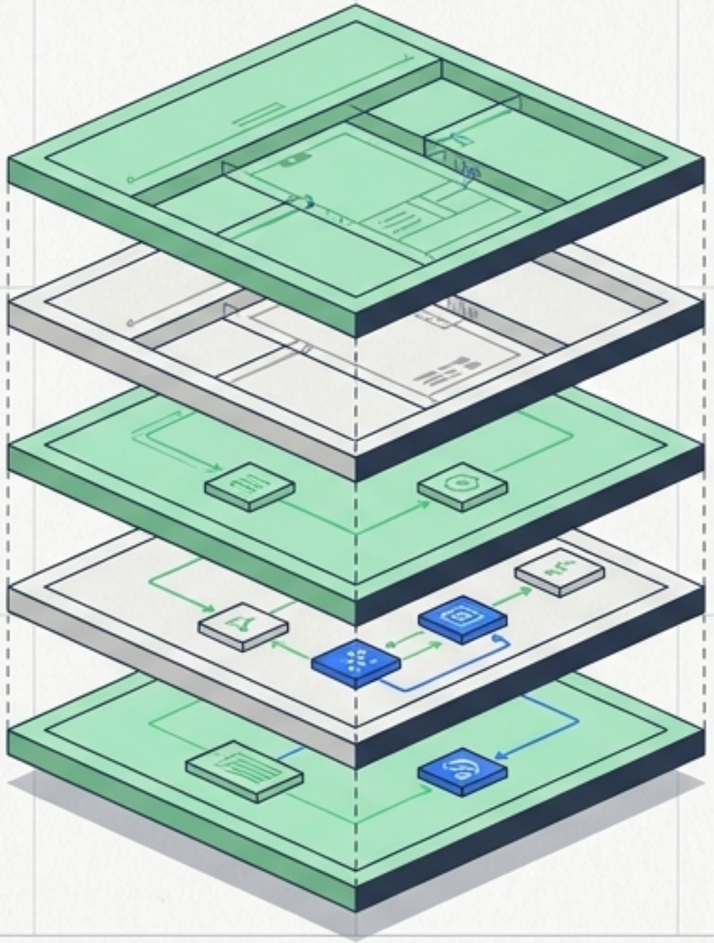
Por qué un prototipo de IA rara vez sobrevive en un entorno de producción.

El modelo base casi nunca es el problema. Las fallas en producción surgen de todo el ecosistema circundante: usuarios reales, documentos desactualizados y casos límite que jamás aparecieron en el conjunto de evaluación inicial.



El ‘arnés’ de ingeniería es tan crítico como el modelo de lenguaje subyacente.

Los modelos de IA no son bases de datos estables. El "Arnés" absorbe esta volatilidad, actuando como la maquinaria defensiva que permite al modelo operar con seguridad.



Capa 5: Contexto

Capa 4: Identidad

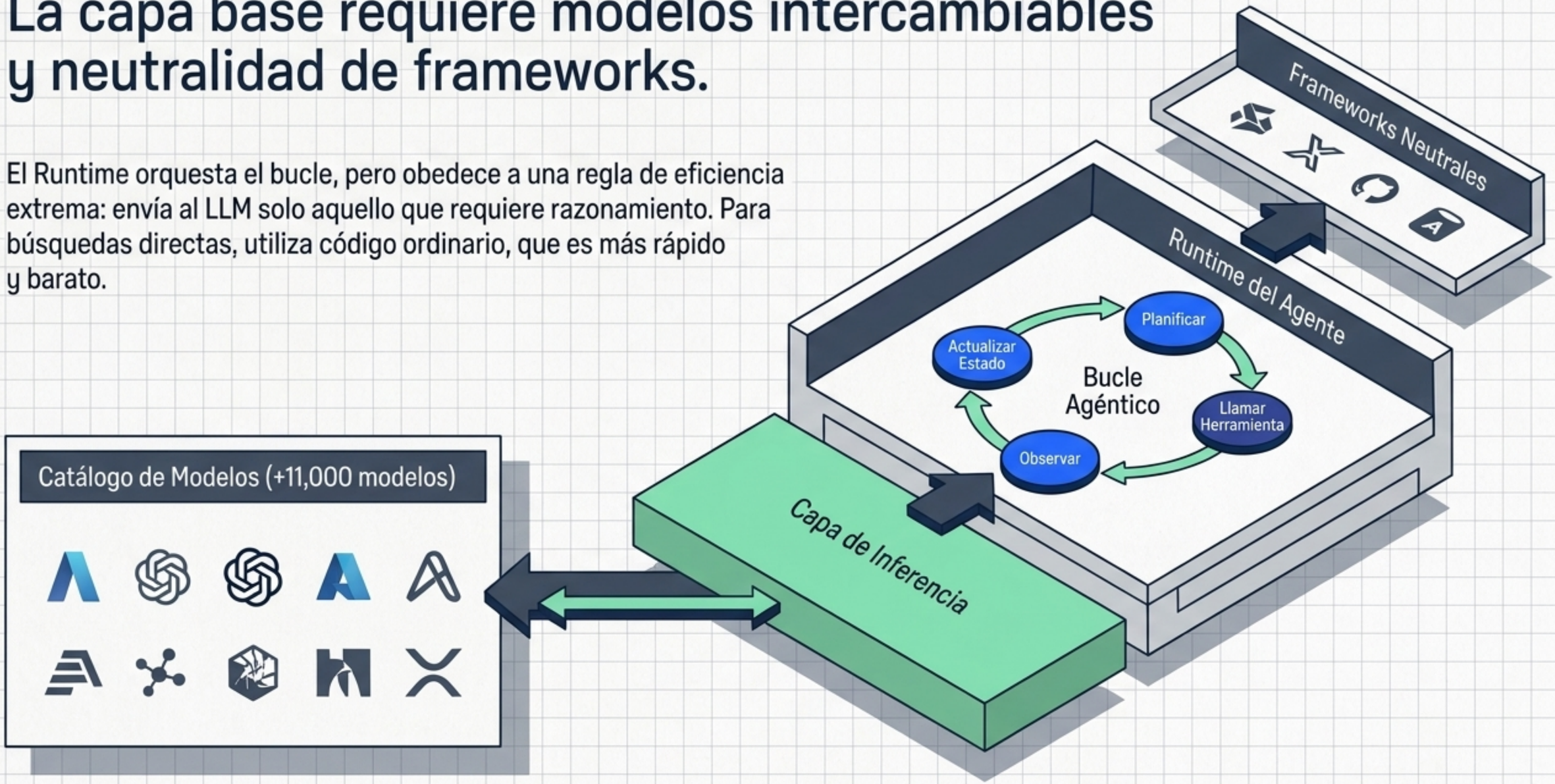
Capa 3: Observabilidad y Gobernanza

Capa 2: Runtime del Agente

Capa 1: Inferencia

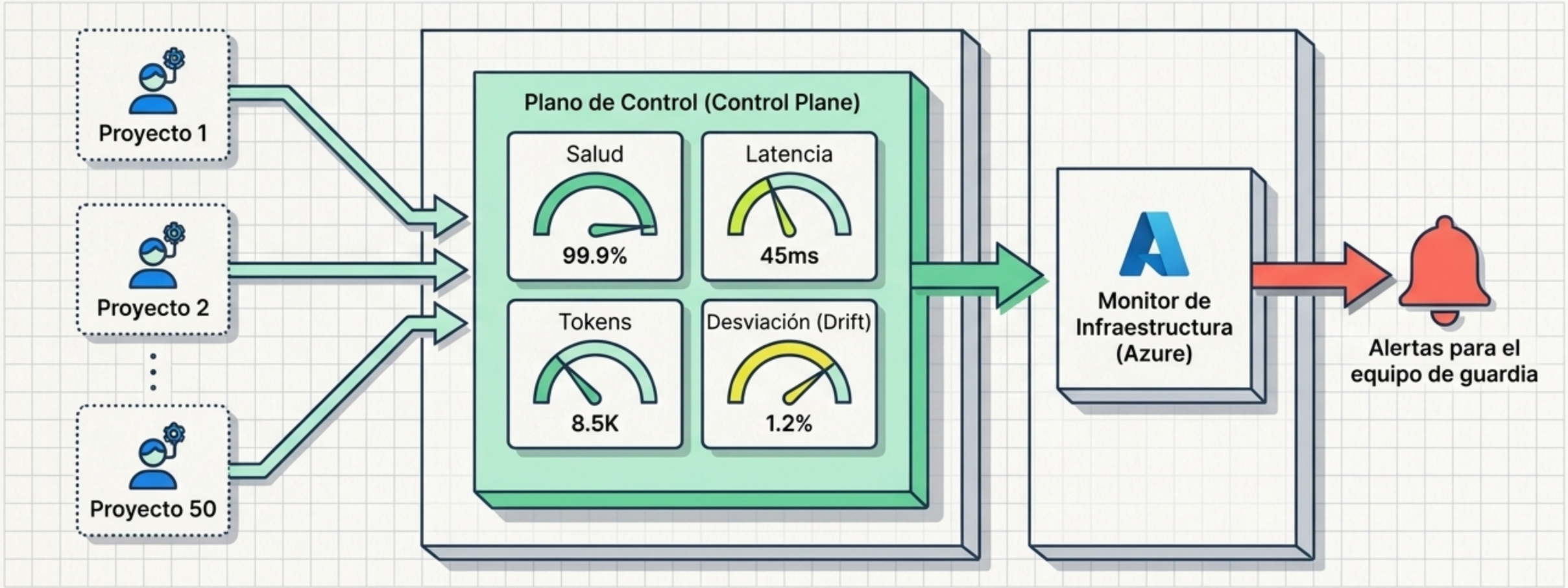
La capa base requiere modelos intercambiables y neutralidad de frameworks.

El Runtime orquesta el bucle, pero obedece a una regla de eficiencia extrema: envía al LLM solo aquello que requiere razonamiento. Para búsquedas directas, utiliza código ordinario, que es más rápido y barato.



Sin telemetría centralizada, las regresiones de calidad en la flota son completamente invisibles.

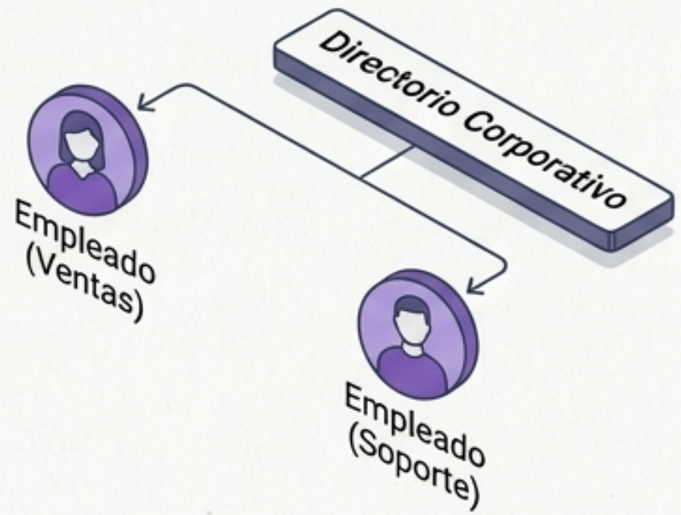
Gestionar agentes en producción es gestionar flotas completas. Se requiere una vista unificada que enrute las métricas del agente hacia los sistemas de infraestructura que el equipo de guardia ya utiliza.



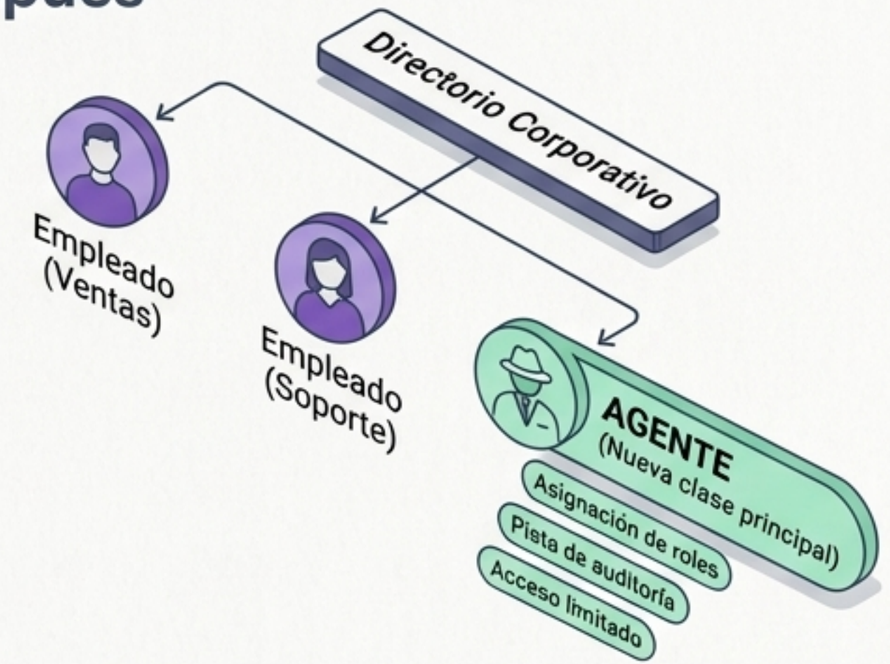
Los agentes que ejecutan acciones requieren su propia identidad corporativa auditable.

Si un agente tiene el poder de actuar, no puede operar de forma anónima ni usar credenciales prestadas. Debe someterse a los mismos controles de acceso restrictivos que un empleado humano.

Antes



Después



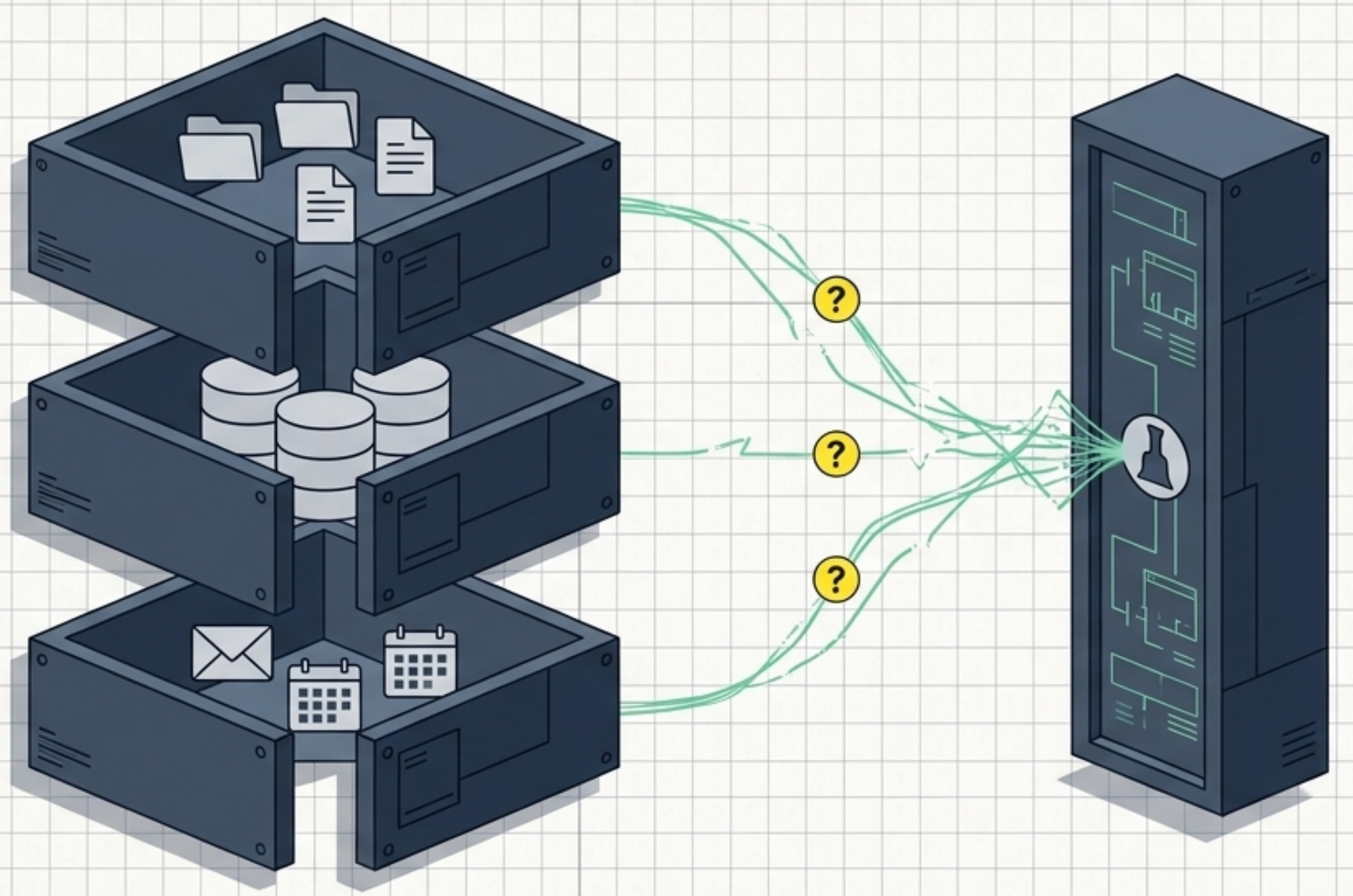
El contexto empresarial no falta, pero vive atrapado en múltiples silos aislados.

Dotar a un agente del contexto correcto es el desafío técnico más complejo. Sin un contexto prístino, el agente ignora lo que no sabe, llevándolo inevitablemente a alucinar respuestas con extrema confianza.

Documentos
(SharePoint / Wikis)

Tablas Estructuradas
(Data Warehouses)

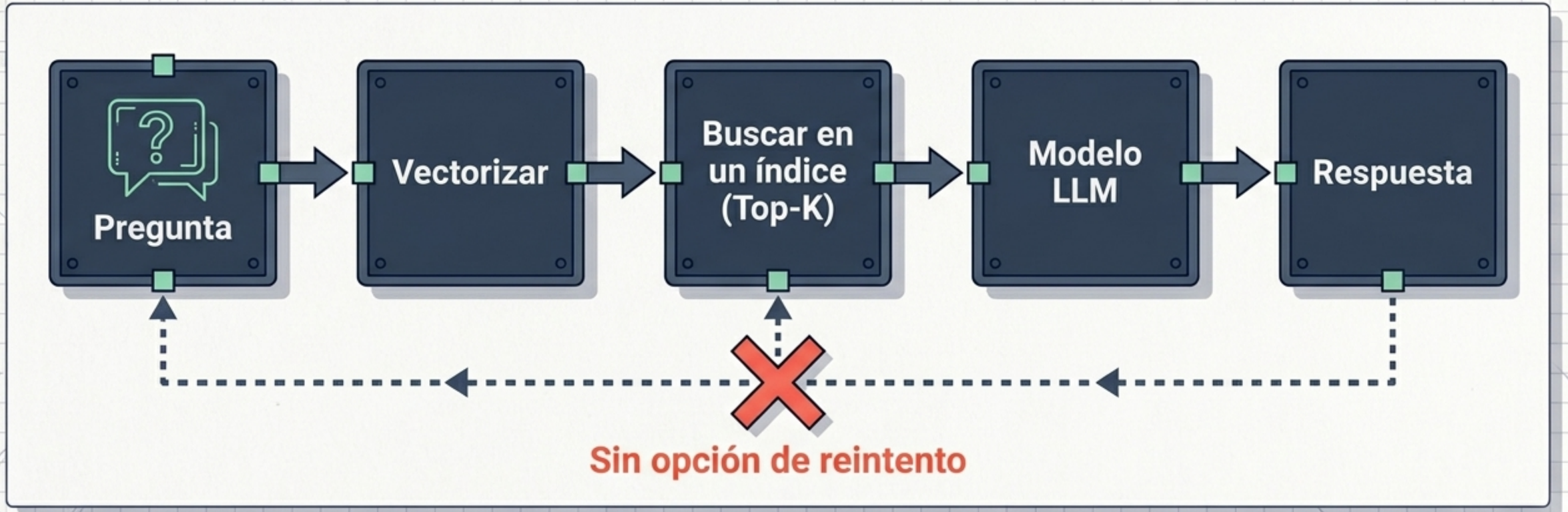
Aplicaciones de Trabajo
(Outlook / Teams)



Nodo de Recuperación
(Retrieval)

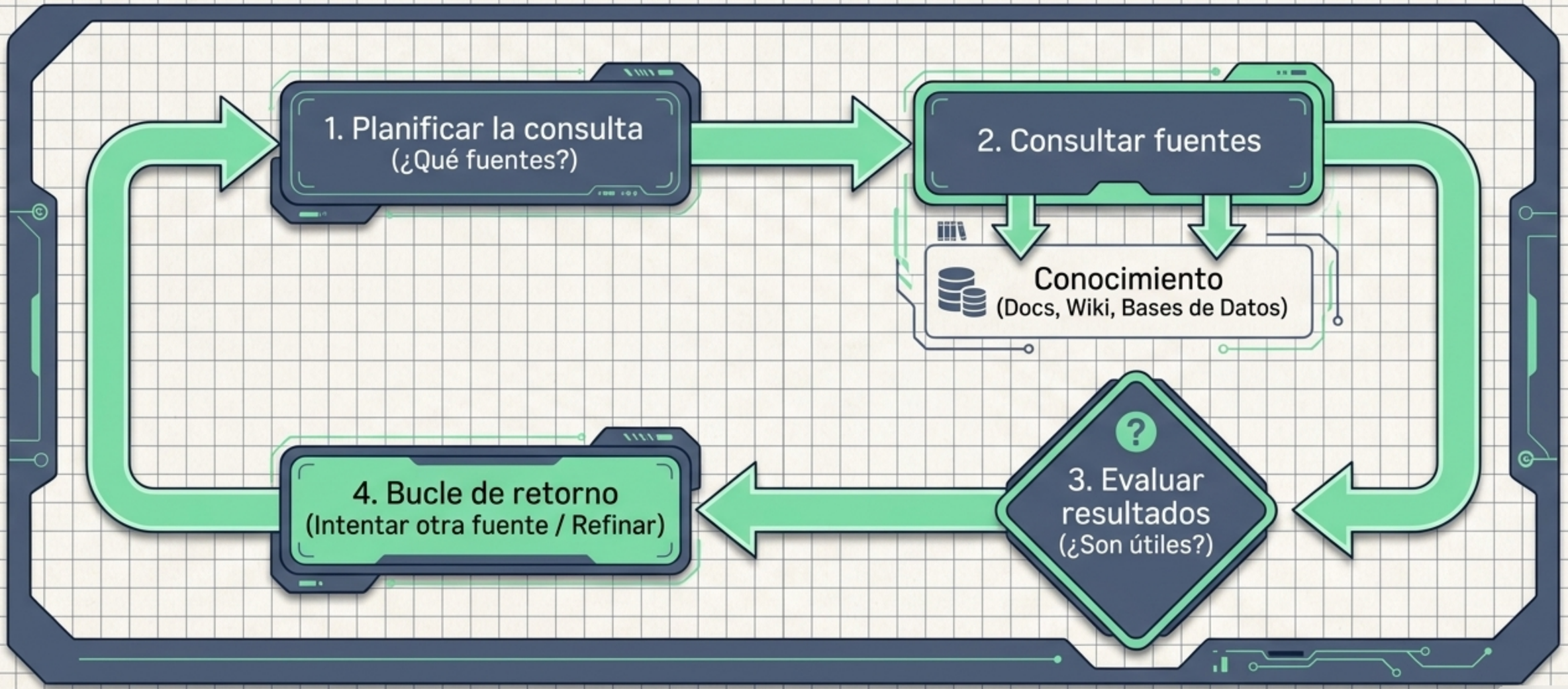
El patrón RAG clásico de un solo intento no fue diseñado para la complejidad empresarial.

El modelo “one-shot” funciona para corpus pequeños y limpios. Pero se quiebra en el instante en que la pregunta es ambigua, la primera búsqueda regresa vacía, o la respuesta requiere triangular múltiples fuentes.



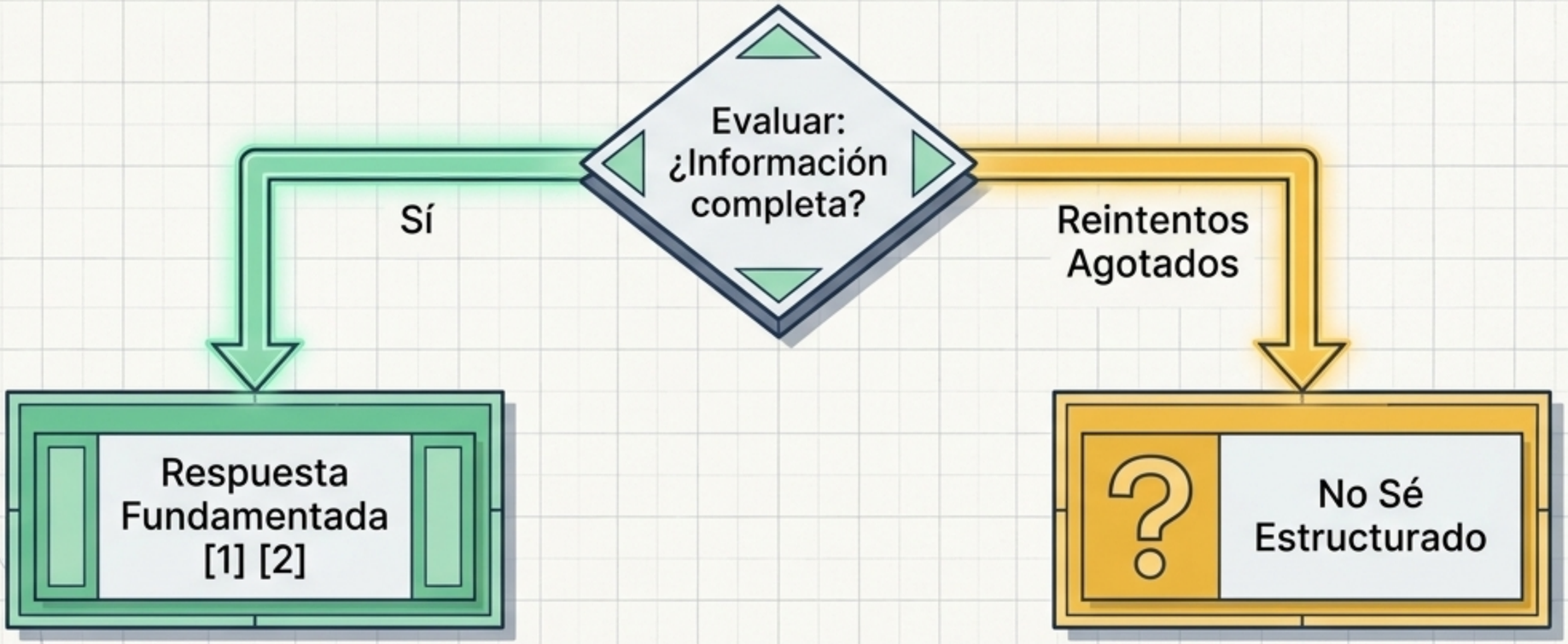
La recuperación de información debe operar como un sub-agente iterativo.

En lugar de una búsqueda ciega, la recuperación se envuelve en su propio micro-agente. Este sub-agente planea qué fuentes atacar, evalúa sus hallazgos, e itera autónomamente si la información está incompleta.



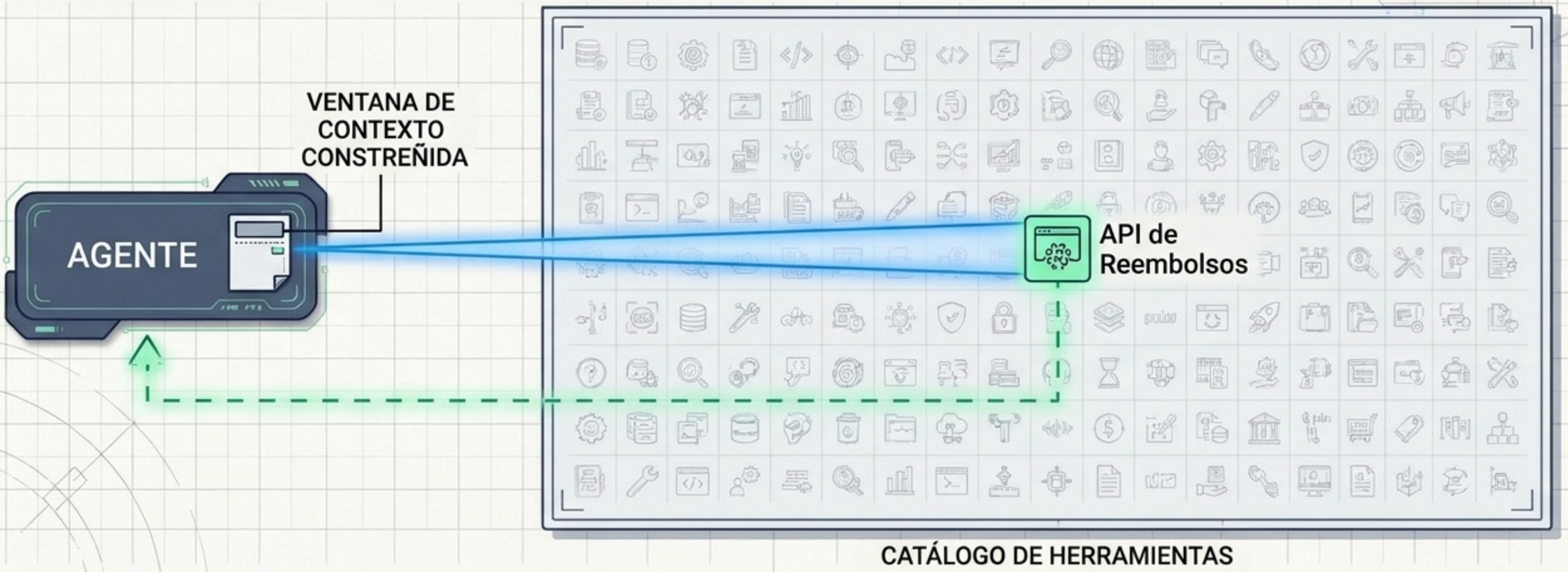
Agotar los reintentos debe generar una falla estructurada, no una alucinación forzada.

El RAG clásico fuerza al modelo a adivinar cuando no encuentra datos. El sub-agente iterativo posee una válvula de escape: cuando agota sus intentos, envía una señal clara de falla que el sistema puede manejar.



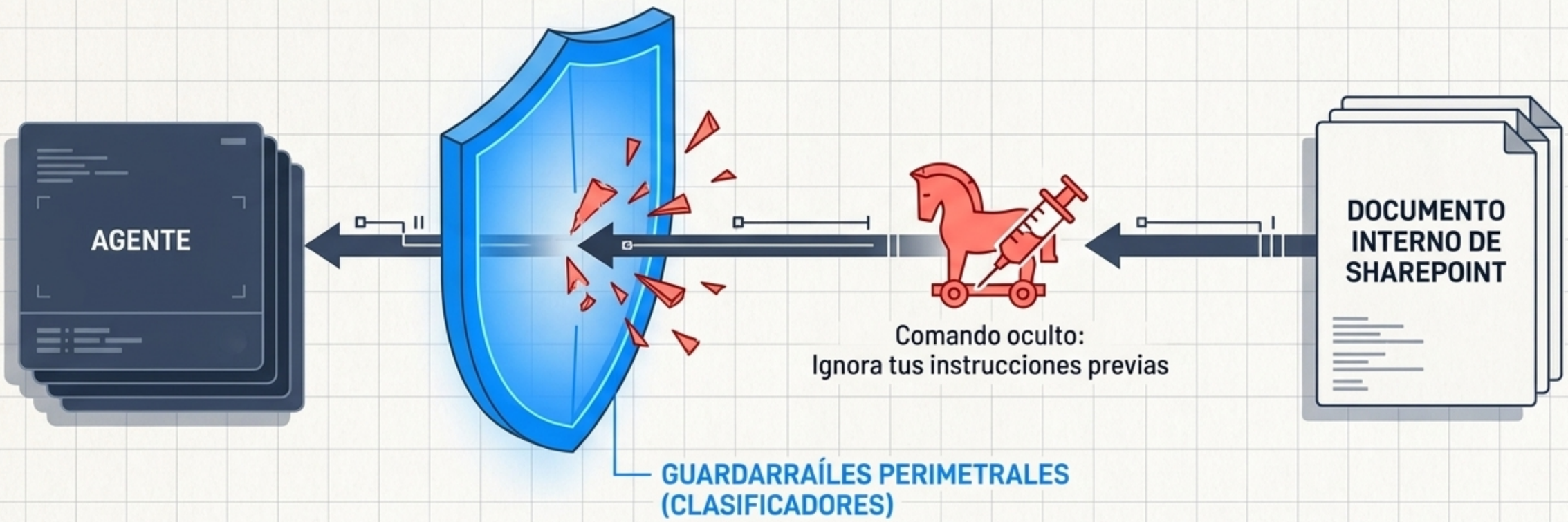
Cargar todas las herramientas en el prompt agota el contexto y destruye la latencia.

Un agente con docenas de capacidades no puede listar todas sus herramientas en las instrucciones iniciales. El mismo bucle iterativo se usa para realizar "Búsqueda de Herramientas" dinámicamente.



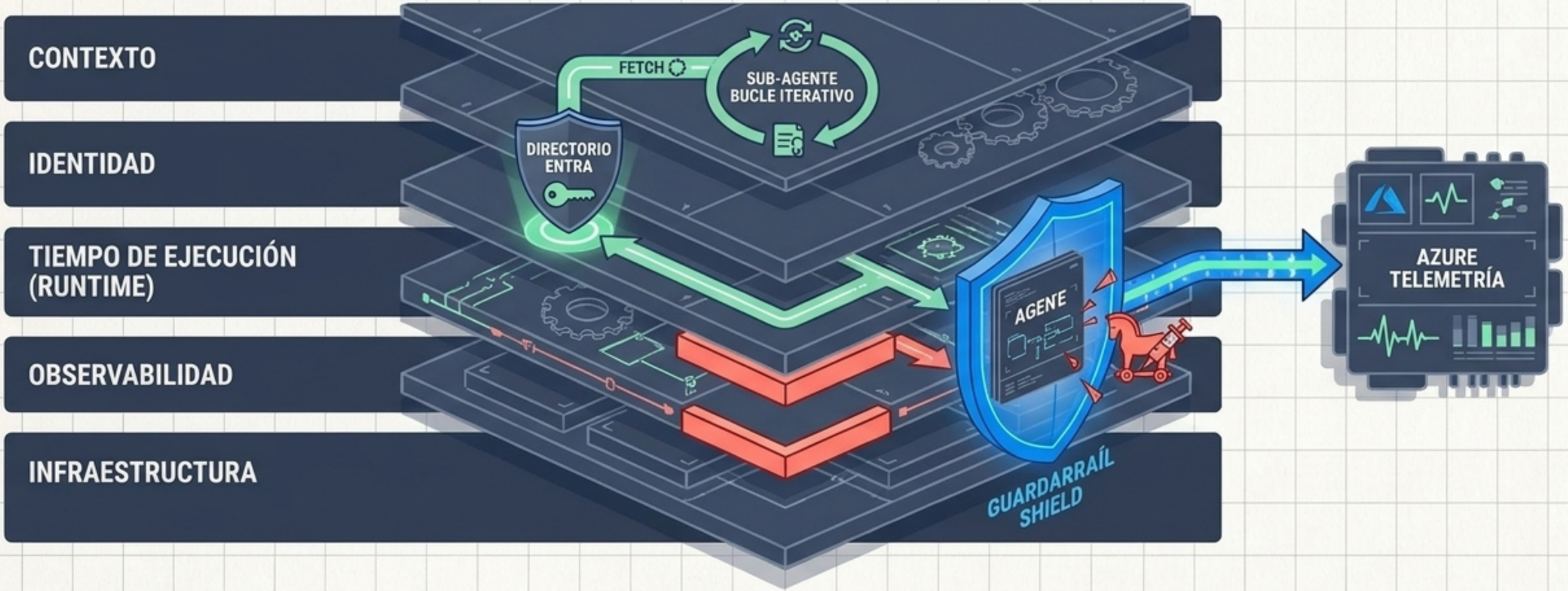
La defensa contra la inyección indirecta de prompts debe operar en el límite de la herramienta.

Los chatbots solo escaneaban la entrada del usuario. Los agentes leen correos y documentos que pueden contener ataques ocultos. Los **guardarraíles** deben colocarse en la respuesta misma de la herramienta.

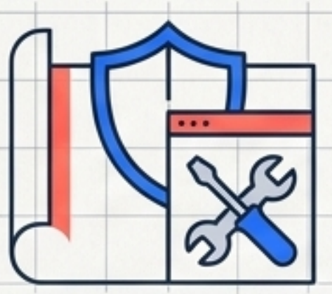


El Ecosistema de Producción: La Arquitectura de Microsoft Foundry en Acción.

Un sistema resiliente no depende de la magia de un modelo. Es el resultado de un ecosistema donde la **Identidad** audita, el **Arnés** protege, los **Sub-agentes** iteran, y los **Guardarraíles** defienden los límites.



Tres principios de ingeniería para escalar agentes a nivel empresarial

		
1. Construye el Arnés, no solo el Modelo	**2. Implementa Sub-agentes Iterativos**	**3. Asegura la Acción, no solo el Prompt**
La fiabilidad de tu producto vive en la **infraestructura perimetral,** la **telemetría** y la **gobernanza**, no en la versión del LLM.	Transforma la búsqueda **lineal** ciega en **bucles de evaluación** que posean la capacidad **estructurada** de admitir que **“no saben”.**	Otorga a cada agente una **identidad auditable** en tu directorio y coloca guardarraíles directamente en la capa de **interacción** de **herramientas.**