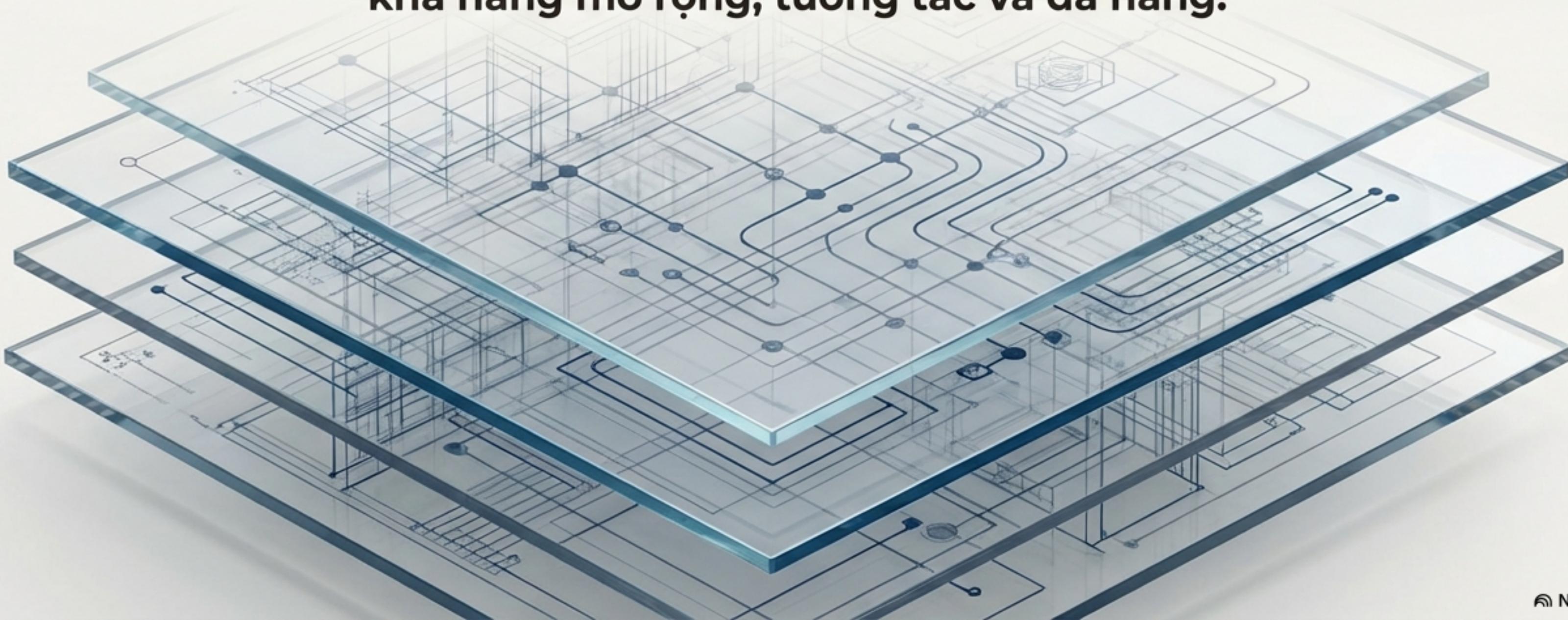


Xây Dựng AI Agent Hiện Đại: Một Kiến Trúc Phân Lớp

Từ giới hạn của LLM đến một hệ sinh thái agent có
khả năng mở rộng, tương tác và đa năng.



Nền Tảng: "Bộ Não" LLM và Những Thách Thức Căn Bản

Các hạn chế cốt lõi của LLM trong việc xây dựng AI Agent hoàn chỉnh

Mỗi AI Agent hiện đại đều bắt đầu với một LLM mạnh mẽ ở lõi. Tuy nhiên, bản thân LLM không phải là một agent hoàn chỉnh. Chúng đối mặt với ba thách thức lớn:

1. Trí nhớ ngắn hạn (Cửa sổ Ngữ cảnh Hạn chế):

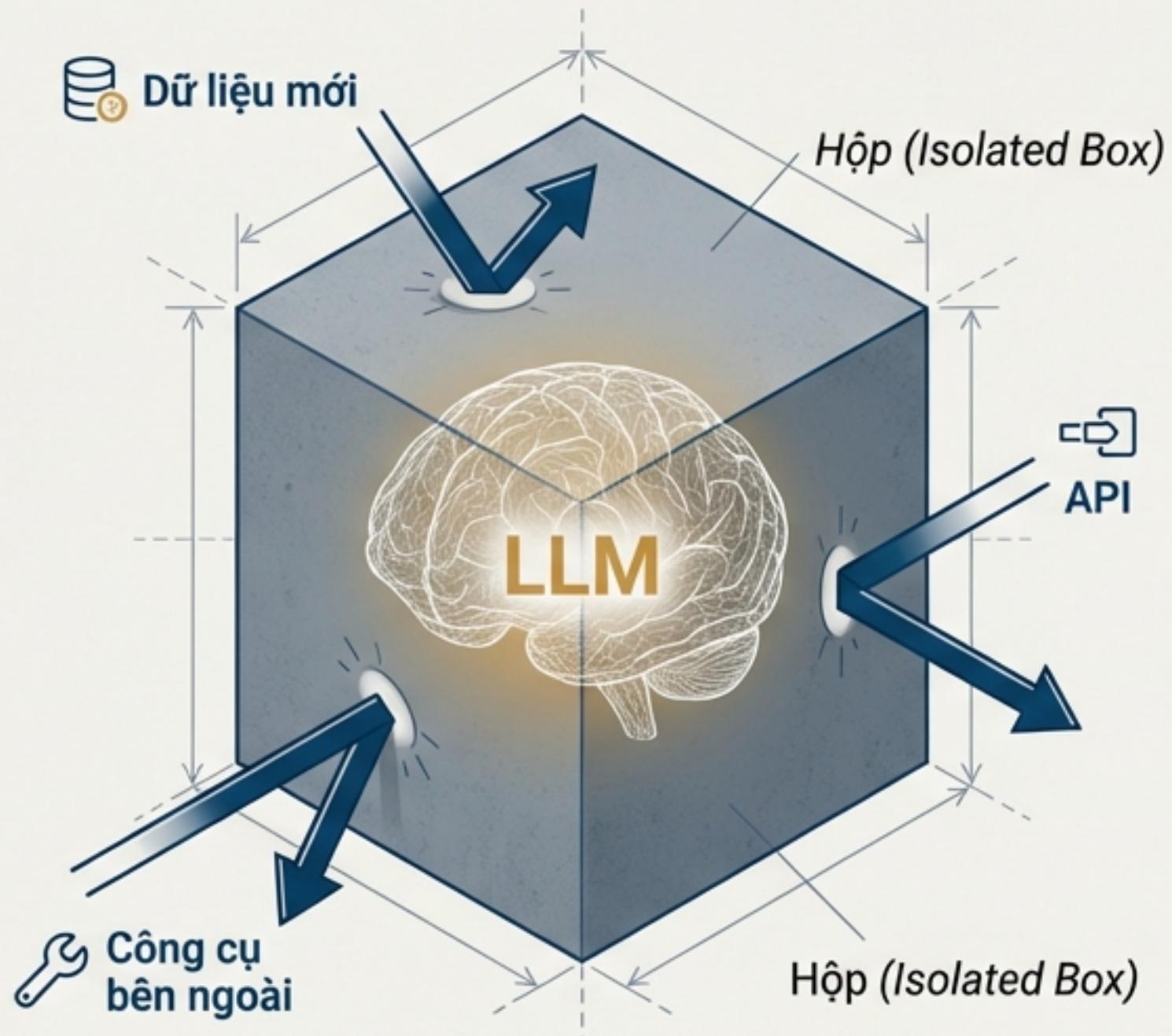
- 🕒 LLM chỉ có thể “nhớ” một lượng thông tin giới hạn trong một lần tương tác, giống như một người có trí nhớ ngắn hạn. Thông tin cũ sẽ nhanh chóng bị lãng quên.

2. Kiến thức lõi thời:

- 📅 Kiến thức của LLM bị đóng băng tại thời điểm huấn luyện. Chúng không thể truy cập thông tin mới hoặc dữ liệu thời gian thực.

3. Sự cô lập:

- 🔒 **Sự cô lập:** Một LLM đơn thuần là một hệ thống khép kín. Nó không thể tương tác với các công cụ bên ngoài, truy cập cơ sở dữ liệu, hoặc giao tiếp với các hệ thống khác.



Lớp 0: Nền Tảng - Giải Mã Cửa Sổ Ngữ Cảnh

Cửa Sổ Ngữ Cảnh là gì?

- **Định nghĩa:** Là lượng văn bản (đo bằng token) mà LLM có thể xem xét cùng một lúc. Nó bao gồm cả đầu vào (prompt, lịch sử) và đầu ra (phản hồi của mô hình).
- **Công thức:** `Tổng số Token (Đầu vào + Đầu ra) <= Độ dài Tối đa của Mô hình`.
- **Ảnh dụ:** "Trí nhớ làm việc" hay "trí nhớ ngắn hạn" của LLM.

Giới hạn không chỉ là kích thước.

- **Chi phí Tính toán:** Cửa sổ càng lớn, chi phí và độ trễ càng cao.
- **"Lost in the Middle":** Các mô hình có xu hướng quên thông tin ở giữa các văn bản dài.

So sánh Cửa sổ Ngữ cảnh của các LLM phổ biến.

Mô hình LLM	Cửa sổ Ngữ cảnh (Token)
GPT-3.5 Turbo	4,096 / 16,384
GPT-4	8,192 / 32,768
GPT-4 Turbo	128,000
Claude 2.1	200,000
Gemini 1.5 Pro	128,000 (có thể lên 1 triệu)
Mixtral 8x7B	32,768

Lớp 1: Lớp Quản Lý Trí Nhớ - Vượt Qua Giới Hạn Ngữ Cảnh

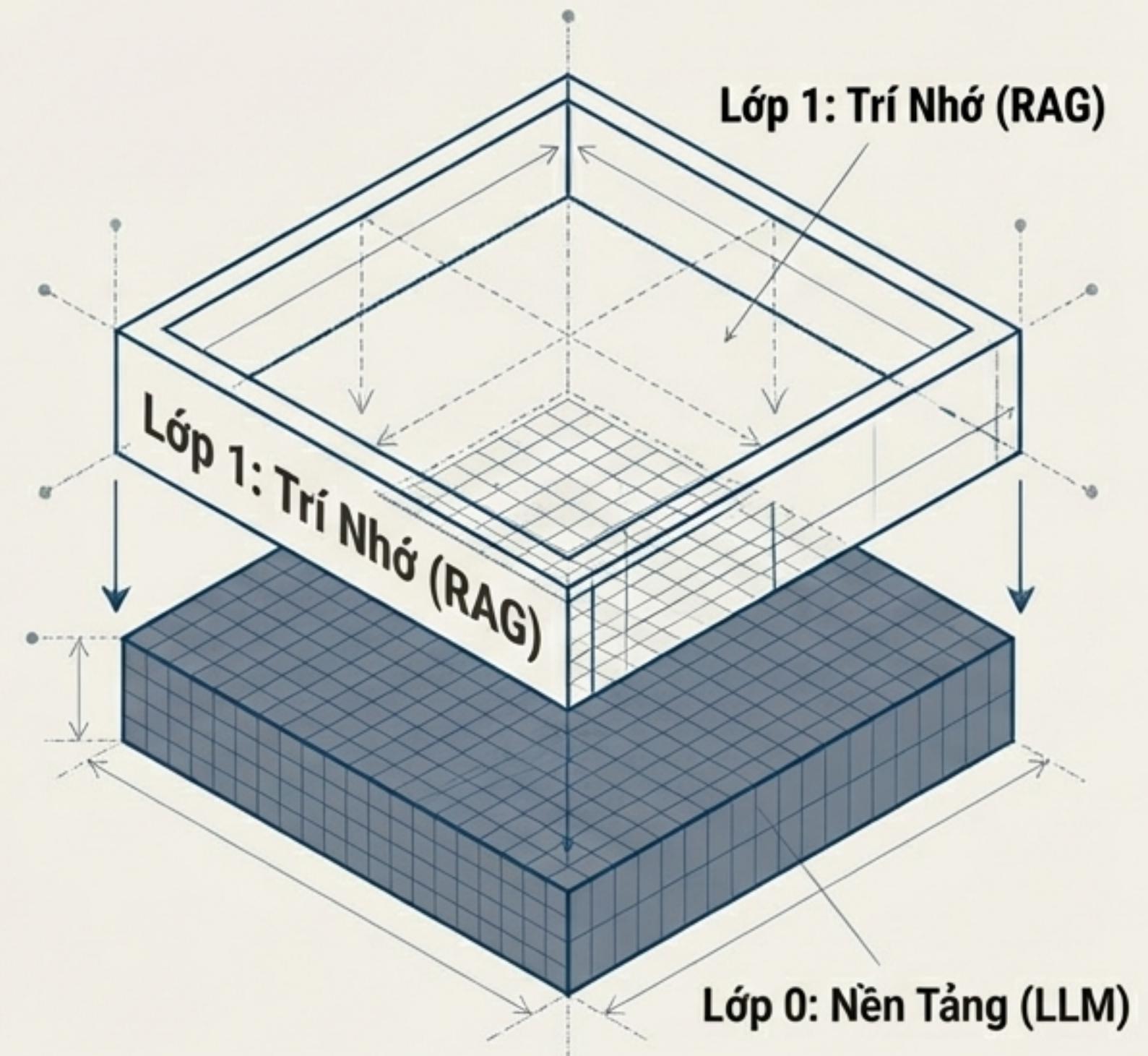
Xây dựng 'bộ não mở rộng' với kiến trúc RAG

Vấn đề: Làm thế nào để agent truy cập một lượng kiến thức khổng lồ, luôn cập nhật mà không cần nhồi nhét tất cả vào cửa sổ ngữ cảnh?

Giải pháp: Xây dựng một "bộ não mở rộng" cho agent. Thay vì cố gắng **nhớ** mọi thứ, agent sẽ học cách **truy xuất** thông tin khi cần.

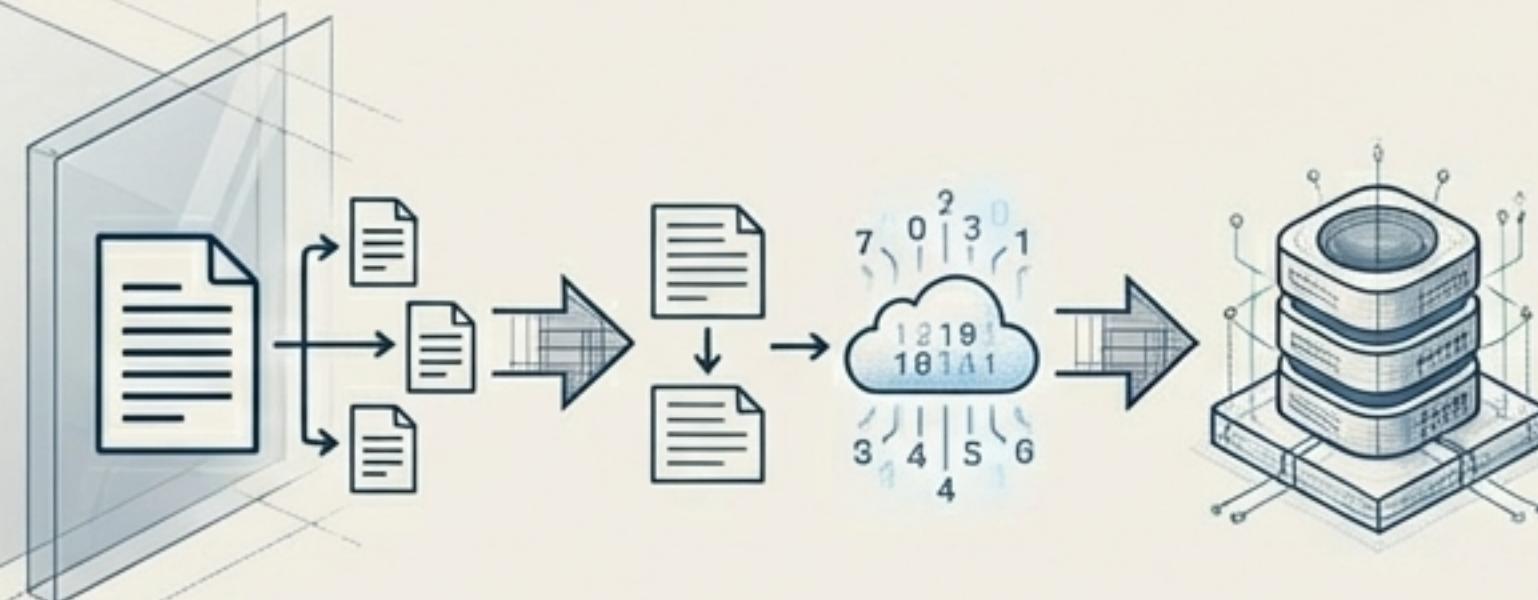
Kiến trúc thống trị: Retrieval-Augmented Generation (RAG)

- **Định nghĩa:** Một kiến trúc kết hợp LLM (bộ sinh) với một hệ thống truy xuất thông tin bên ngoài (bộ truy xuất).
- **Ánh dụ:** Cho phép LLM "thi trong kỳ thi mở" (open-book exam), nơi nó có thể tra cứu tài liệu tham khảo trước khi trả lời.
- **Lợi ích cốt lõi:**
 - Giảm thiểu "ảo giác" (hallucination)
 - Kiến thức luôn cập nhật
 - Khả năng trích dẫn nguồn



Quy trình Hoạt động của RAG

Giai đoạn Indexing (Offline): Xây dựng Cơ sở Tri thức



Tài liệu (văn bản, PDF, v.v.) được chia thành các đoạn nhỏ (chunks).

Mỗi chunk được chuyển đổi thành một vector số học bằng một mô hình embedding.

Các vector được lưu trữ và lập chỉ mục trong một Vector Database.

Giai đoạn Retrieval & Generation (Real-time): Trả lời Truy vấn



Người dùng đặt câu hỏi.

Truy vấn được chuyển thành vector. Hệ thống tìm kiếm các vector chunk tương đồng nhất (tìm kiếm ngữ nghĩa).

Các chunk liên quan nhất được lấy ra và chèn vào prompt của LLM.

LLM nhận prompt đã được "tăng cường" và tạo ra câu trả lời.

"**Tìm kiếm ngữ nghĩa** (Semantic Search) cho phép tìm thấy các đoạn văn liên quan về mặt ý nghĩa, ngay cả khi chúng không chứa các từ khóa chính xác trong câu hỏi."

Lựa Chọn Chiến Lược: RAG so với Fine-tuning

RAG và Fine-tuning giải quyết các vấn đề khác nhau. Lựa chọn đúng đắn phụ thuộc vào mục tiêu của bạn.

Khía cạnh	Fine-tuning	Retrieval-Augmented Generation (RAG)
Mục tiêu chính	Dạy cho mô hình một phong cách , giọng điệu hoặc định dạng mới.	Cung cấp cho mô hình kiến thức cập nhật, theo lĩnh vực cụ thể.
Cập nhật kiến thức	Rất khó khăn và tốn kém. Phải huấn luyện lại toàn bộ mô hình.	Dễ dàng và nhanh chóng. Chỉ cần cập nhật dữ liệu trong vector store.
Chi phí	Cao (yêu cầu GPU, thời gian và dữ liệu huấn luyện lớn).	Thấp hơn đáng kể (chi phí chủ yếu là lưu trữ và embedding).
Nguy cơ 'ảo giác'	Vẫn có thể xảy ra, mô hình có thể quên kiến thức đã học.	Thấp hơn nhiều, vì câu trả lời được neo vào dữ liệu thực tế.
Khả năng giải thích	Thấp (mô hình là 'hộp đen').	Cao (có thể trích dẫn chính xác nguồn thông tin đã sử dụng).

Kết luận then chốt: Hãy sử dụng **Fine-tuning** để thay đổi cách agent nói chuyện. Hãy sử dụng **RAG** để thay đổi *những gì* agent biết. Hai kỹ thuật này có thể bổ trợ cho nhau.

Lớp 2: Lớp Giao Thức - Phá Vỡ Sự Cô Lập

Vượt qua giới hạn của sự cô lập: Từ vấn đề $M \times N$ đến giải pháp $M+N$ có thể mở rộng.

Vấn đề mới:

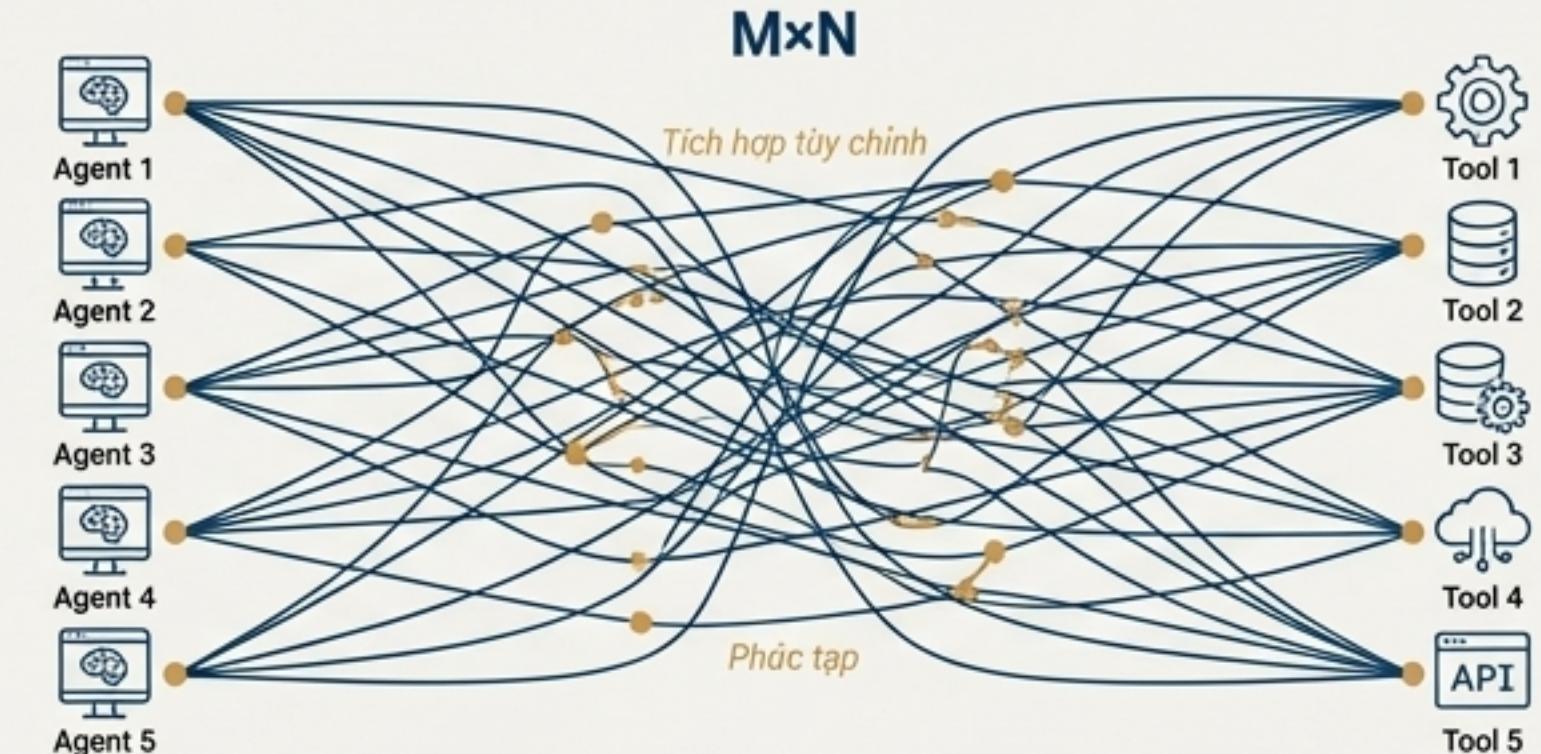
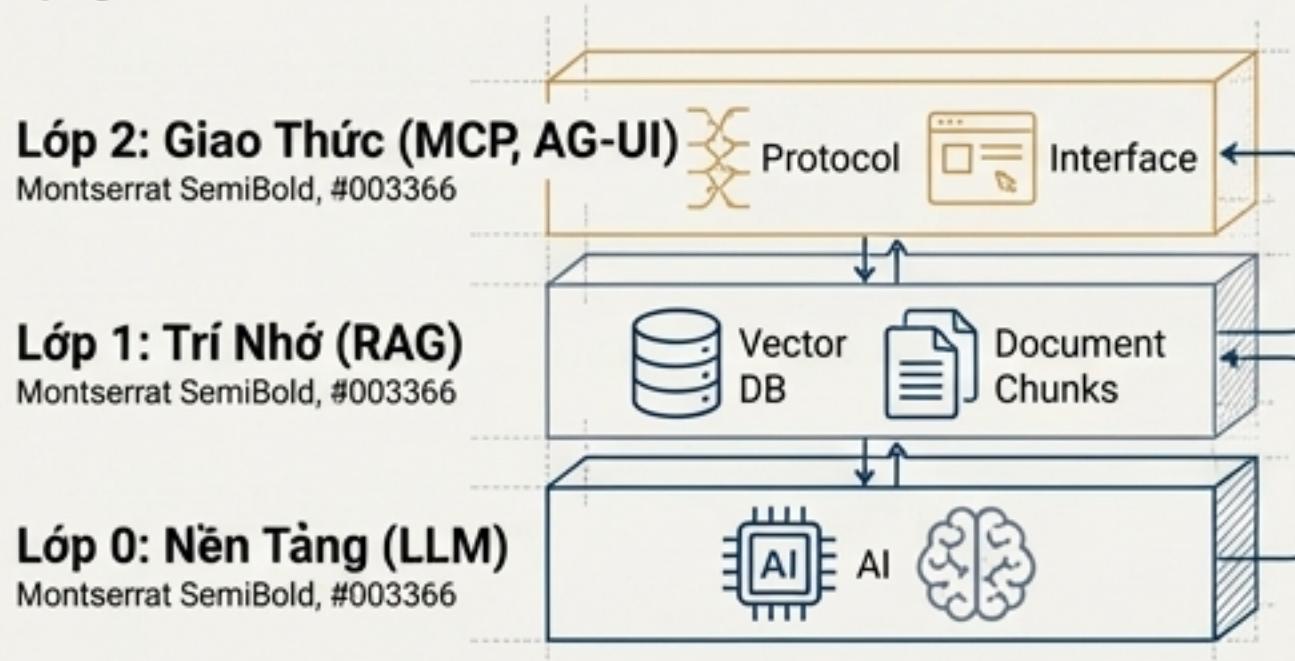
Agent của chúng ta giờ đã thông minh và có trí nhớ, nhưng nó vẫn bị “mắc kẹt trong hộp”. Làm thế nào để nó tương tác một cách an toàn và có thể mở rộng với vô số công cụ, API, nguồn dữ liệu và người dùng?

Thách thức của Tích hợp: Vấn đề $M \times N$

Không có một tiêu chuẩn chung, mỗi agent (M) cần một tích hợp tùy chỉnh cho mỗi công cụ (N). Điều này dẫn đến sự bùng nổ về độ phức tạp.

Giải pháp: Một Lớp Giao Thức Chung

Giới thiệu một lớp trừu tượng, một “ngôn ngữ chung” để chuẩn hóa giao tiếp, biến vấn đề $M \times N$ thành một giải pháp $M+N$ có khả năng mở rộng.



MCP: Chuẩn "USB" cho Giao Tiếp giữa Agent và Công Cụ

Giống như USB đã chuẩn hóa cách các thiết bị ngoại vi kết nối với máy tính, MCP đang chuẩn hóa cách các AI Agent kết nối với dữ liệu và công cụ.

Model Context Protocol (MCP) là gì?

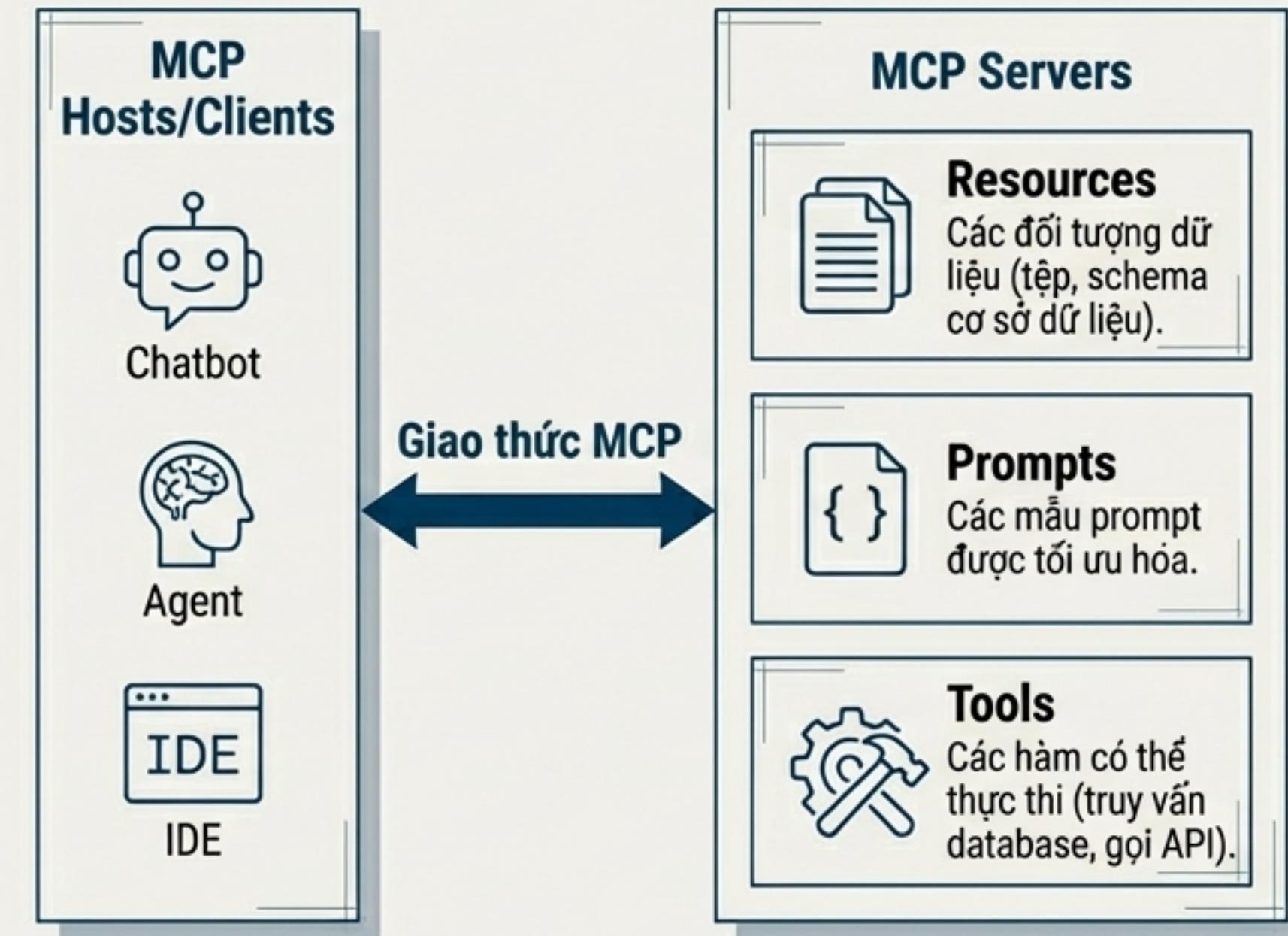
Một giao thức mở, được giới thiệu bởi Anthropic, nhằm chuẩn hóa việc tích hợp giữa các ứng dụng LLM và các nguồn dữ liệu/công cụ bên ngoài.

Kiến trúc Client-Server:

- **MCP Hosts/Clients:** Các ứng dụng AI (agent, chatbot, IDE) muốn truy cập dữ liệu.
- **MCP Servers:** Các chương trình "phơi bày" khả năng của một công cụ hoặc nguồn dữ liệu (ví dụ: database, GitHub, Slack) thông qua giao thức MCP.

Lợi ích:

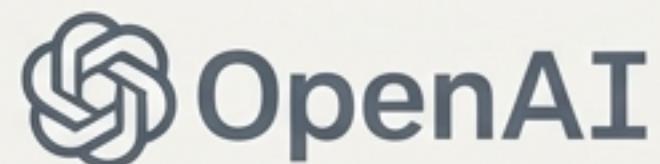
Giảm thời gian phát triển, tăng cường khả năng tương tác, thúc đẩy một hệ sinh thái module hóa.



Sự Hội Tụ của Ngành: Hệ Sinh Thái Đang Chấp Nhận MCP

MCP không chỉ là một ý tưởng. Nó đang nhanh chóng trở thành một tiêu chuẩn được các tên tuổi lớn nhất trong ngành AI và công nghệ áp dụng để đảm bảo khả năng tương tác.

Những người tiên phong về Mô hình (Roboto || Regular) Montserrat SemiBold



Các Nền tảng Đám mây & Doanh nghiệp



Microsoft



Công cụ dành cho Nhà phát triển



"MCP đang nhanh chóng trở thành một tiêu chuẩn mở cho kỷ nguyên tác tử AI."

- Demis Hassabis, CEO của Google DeepMind

AG-UI: Chuẩn Hóa Cuộc Trò Chuyện giữa Agent và Con Người

Vấn đề:

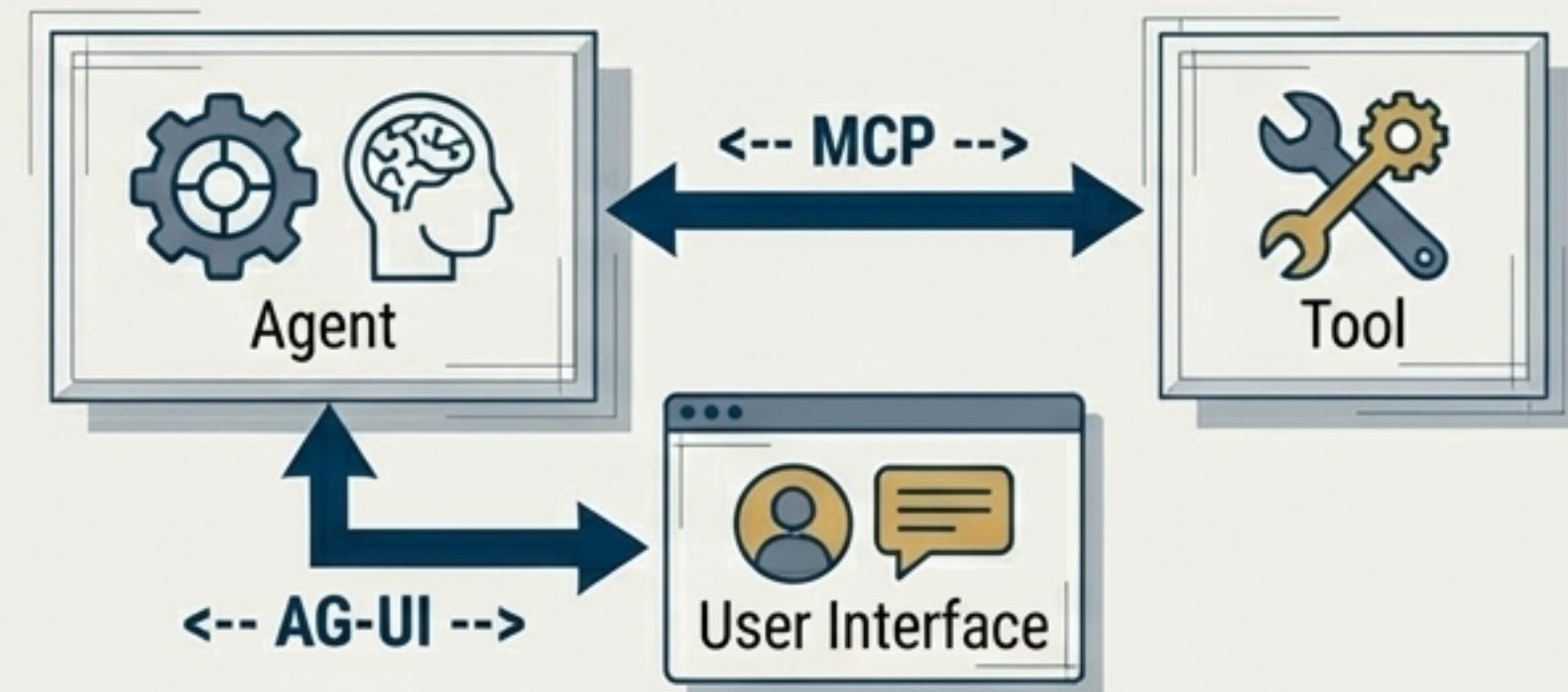
Làm thế nào để giao diện người dùng (UI) biết agent đang làm gì? ("đang suy nghĩ", "đang gọi công cụ", "gặp lỗi", "cần bạn phê duyệt"). Nếu không có tiêu chuẩn, mỗi ứng dụng phải tự xây dựng cơ chế này.

AG-UI (Agent-User Interaction Protocol) là gì?

Một giao thức mở, nhẹ, dựa trên sự kiện, được thiết kế để chuẩn hóa luồng thông tin giữa backend của agent và giao diện người dùng.

Các tính năng chính:

- Trò chuyện và phát trực tuyến (streaming) theo thời gian thực.
- Đồng bộ hóa trạng thái hai chiều.
- Hỗ trợ các quy trình cần sự tham gia của con người (Human-in-the-loop).



Hệ sinh thái Framework hỗ trợ:

Framework	Trạng thái
LangGraph	Hỗ trợ
Mastra	Hỗ trợ
CrewAI	Hỗ trợ
AG2	Hỗ trợ
OpenAI Agent SDK	Mở cho đóng góp

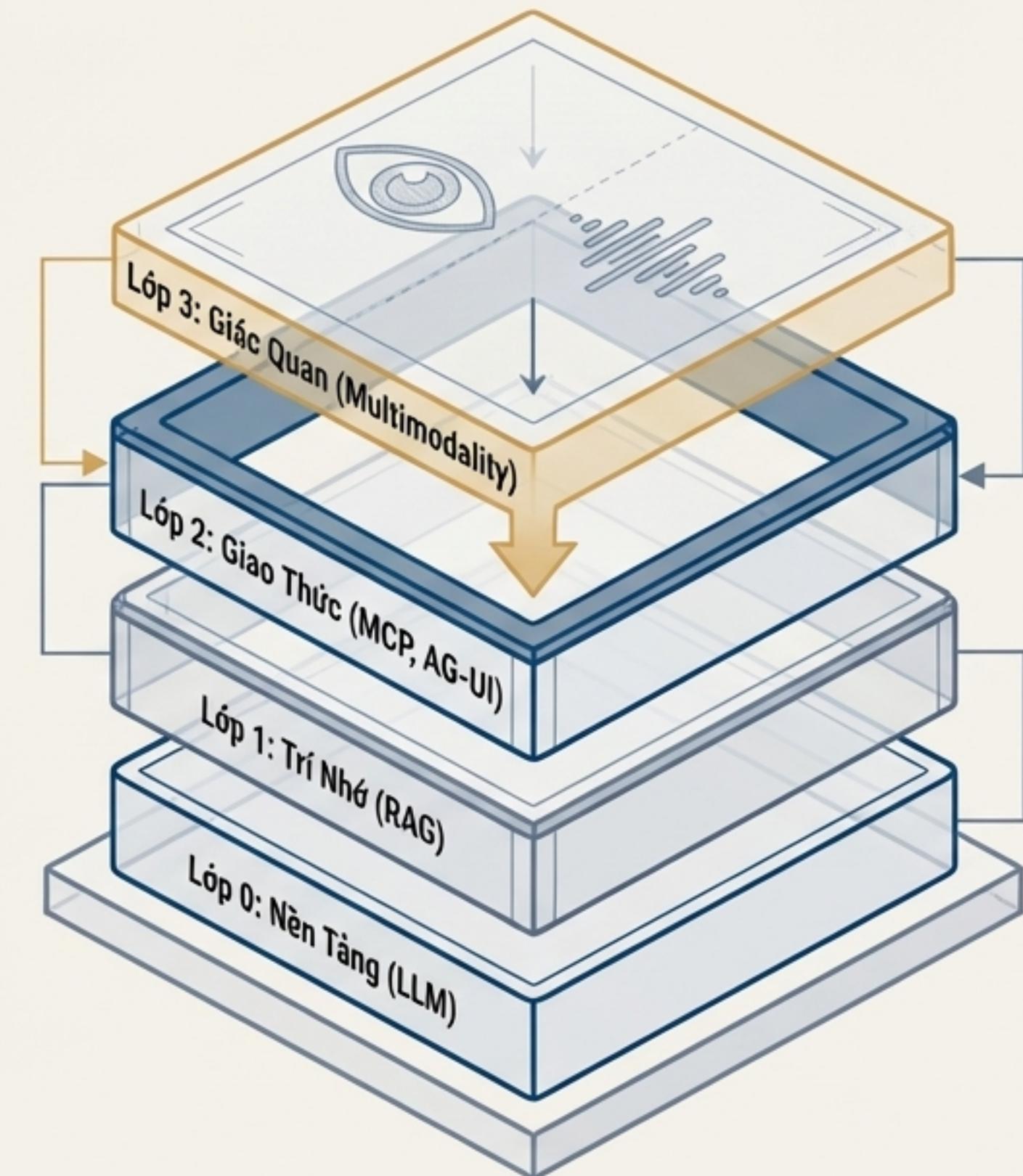
Lớp 3: Lớp Mở Rộng Giác Quan - AI Đa Phương Thức

Vấn đề mới:

Agent của chúng ta có thể suy luận, ghi nhớ và giao tiếp bằng văn bản. Nhưng thế giới thực không chỉ có văn bản. Làm thế nào để agent có thể "nhìn", "nghe" và hiểu được sự kết hợp phức tạp của nhiều loại thông tin?

Giải pháp: AI Đa phương thức (Multimodal AI)

- **Định nghĩa:** Một dạng AI có khả năng xử lý và hiểu đồng thời nhiều dạng dữ liệu khác nhau (phương thức), chẳng hạn như văn bản, hình ảnh, âm thanh và video.
- **So sánh:**
 - **Unimodal AI (Đơn phương thức):** GPT-3 (chỉ văn bản), ResNet (chỉ hình ảnh).
 - **Multimodal AI (Đa phương thức):** DALL-E (văn bản & hình ảnh), Gemini (văn bản, hình ảnh, video, âm thanh).



Kiến Trúc của một Hệ thống AI Đa Phương Thức

Mô-đun Đầu vào (Input Module)



Chức năng: Tiếp nhận và xử lý nhiều luồng dữ liệu thô.

Ví dụ: Sử dụng mô hình Computer Vision (như ViT) để trích xuất đặc trưng từ hình ảnh, sử dụng mô hình NLP để tạo embedding cho văn bản.

Mô-đun Hợp nhất (Fusion Module)



Chức năng: Kết hợp và tìm ra mối liên hệ giữa các đặc trưng từ các phương thức khác nhau.

Các phương pháp phổ biến:
Hợp nhất sớm (Early Fusion),
Hợp nhất muộn (Late Fusion).

Mô-đun Đầu ra (Output Module)



Chức năng: Tạo ra kết quả đầu ra mong muốn từ dữ liệu đã được hợp nhất.

Ví dụ: Tạo ra một đoạn mô tả văn bản cho một hình ảnh, trả lời câu hỏi bằng giọng nói về một video.

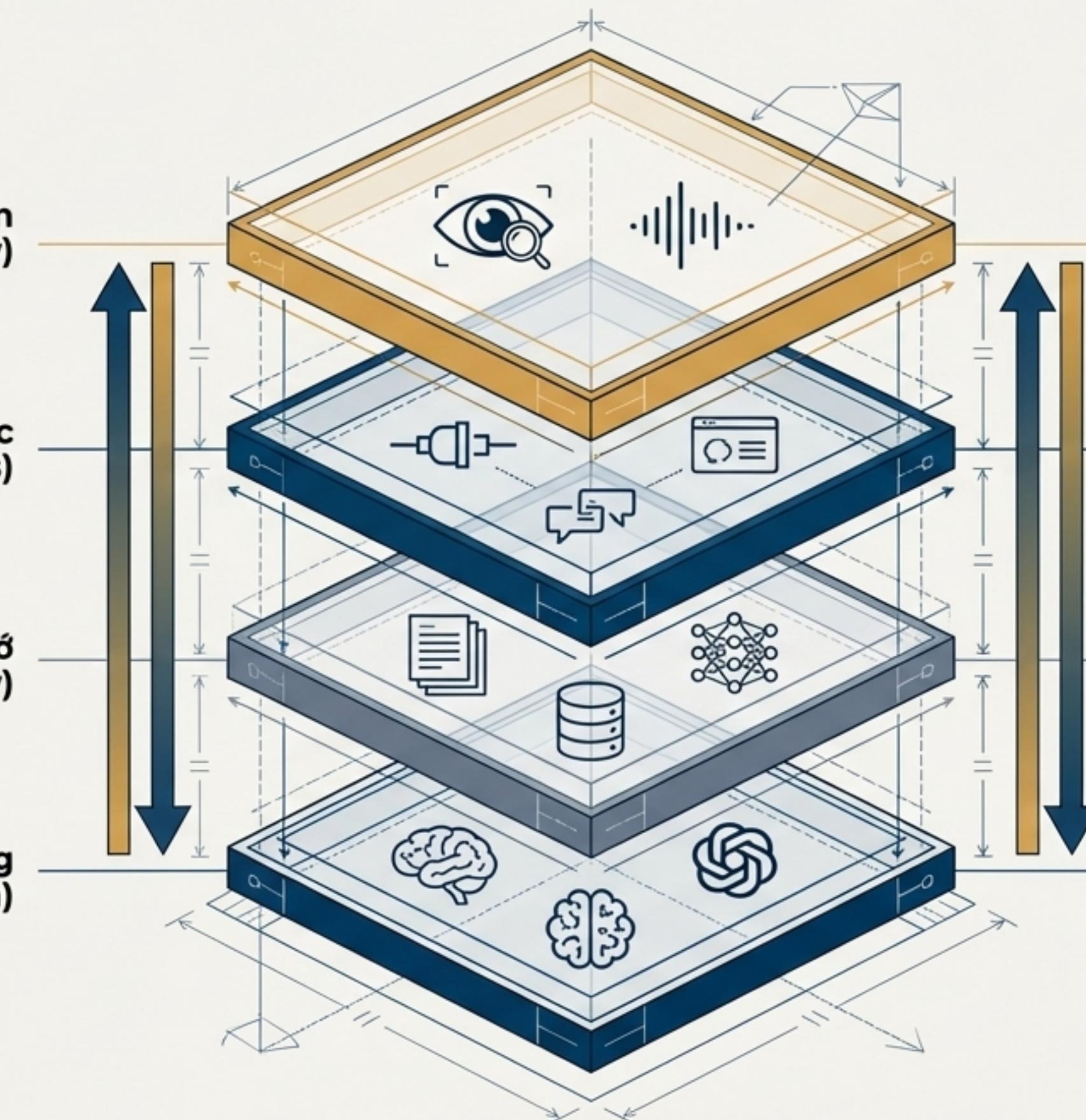
Sơ Đồ Kiến Trúc Hoàn Chỉnh của AI Agent Hiện Đại

Lớp 3: Lớp Mở Rộng Giác Quan
(Multimodality)

Lớp 2: Lớp Giao Thức
(Protocols)

Lớp 1: Lớp Quản Lý Trí Nhớ
(Memory)

Lớp 0: Lớp Nền Tảng
(Foundation)



Chức năng: Cung cấp khả năng "nhìn" và "nghe", xử lý đa dạng loại dữ liệu.

Công nghệ: Computer Vision, Audio Processing, Fusion Models.

Chức năng: Cho phép agent tương tác với thế giới bên ngoài một cách chuẩn hóa.

Công nghệ: MCP (để kết nối với Công cụ & Dữ liệu), AG-UI (để giao tiếp với Người dùng).

Chức năng: Cung cấp "trí nhớ dài hạn" và kiến thức bền ngoài, có thể cập nhật.

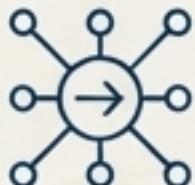
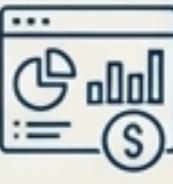
Công nghệ: RAG (Retrieval-Augmented Generation), Vector Databases, Embedding Models.

Chức năng: "Bộ não" suy luận ngôn ngữ cốt lõi.

Công nghệ: Large Language Models (LLM), Cửa sổ Ngữ cảnh.

Triển khai trong Thực tế: Tích hợp Nguồn Dữ liệu Doanh nghiệp qua MCP

Lớp Giao thức cho phép một agent duy nhất truy cập và tổng hợp thông tin từ nhiều hệ thống khác nhau, vốn thường bị cô lập (siloed).

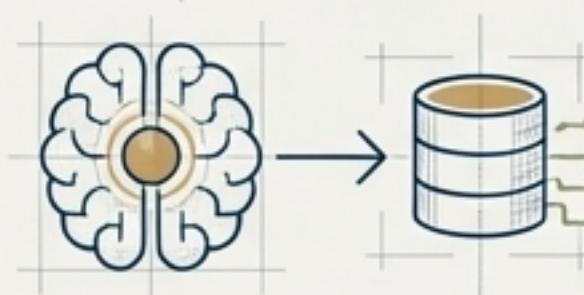
	Nguồn Dữ liệu	Chức năng trong Doanh nghiệp	Cách MCP Tích hợp	Lợi ích cho Agent
	Time-Series Databases	Dữ liệu hiệu suất và giám sát hệ thống theo thời gian.	Adapter MCP bảo tồn mối quan hệ thời gian, nén dữ liệu để giảm token.	Hiểu được xu hướng, phát hiện bất thường dựa trên lịch sử.
	Log Management Systems	Thông tin chi tiết về thực thi, lỗi và sự kiện hệ thống.	Trích xuất ngữ nghĩa từ log dài, chuyển đổi thành thông tin có cấu trúc.	Chẩn đoán nguyên nhân gốc rễ của sự cố từ thông tin chi tiết.
	Service Mesh Telemetry	Dữ liệu về giao tiếp và sự phụ thuộc giữa các microservice.	Biểu diễn các đồ thị phụ thuộc và luồng giao tiếp một cách hiệu quả.	Phân tích các lỗi phức tạp trong hệ thống phân tán.
	Business KPI Systems	Dữ liệu về tác động kinh doanh, doanh thu, khách hàng.	Ánh xạ các chỉ số kỹ thuật với các kết quả kinh doanh.	Đưa ra các khuyến nghị ưu tiên dựa trên tác động kinh doanh thực tế.

Các Nguyên Tắc Kiến Trúc Cho Tương Lai

Xây dựng AI Agent không chỉ là chọn một LLM. Đó là việc thiết kế một hệ thống nhiều lớp, có khả năng mở rộng.



1. **Bắt đầu từ Lõi, nhưng Đừng Dừng lại ở đó:** LLM là nền tảng, nhưng sức mạnh thực sự đến từ các lớp kiến trúc xung quanh nó.



2. **Bên ngoài hóa Tri thức và Trí nhớ:** Đừng cố gắng "nhồi nhét" mọi thứ vào LLM. Sử dụng RAG để tạo ra một bộ nhớ ngoài linh hoạt và luôn cập nhật.



3. **Chuẩn hóa Giao tiếp, Tối đa hóa Tương tác:** Áp dụng các giao thức như MCP và AG-UI để xây dựng một hệ sinh thái mở, thay vì các tích hợp tùy chỉnh khép kín.



4. **Mở rộng Giác quan để Hiểu Sâu hơn:** Hướng tới đa phương thức để cho phép agent tương tác với thế giới một cách tự nhiên và toàn diện hơn.

Tương lai của AI Agent là một hệ sinh thái được xây dựng trên các thành phần module hóa, có khả năng tương tác và được kết nối bởi các tiêu chuẩn mở.