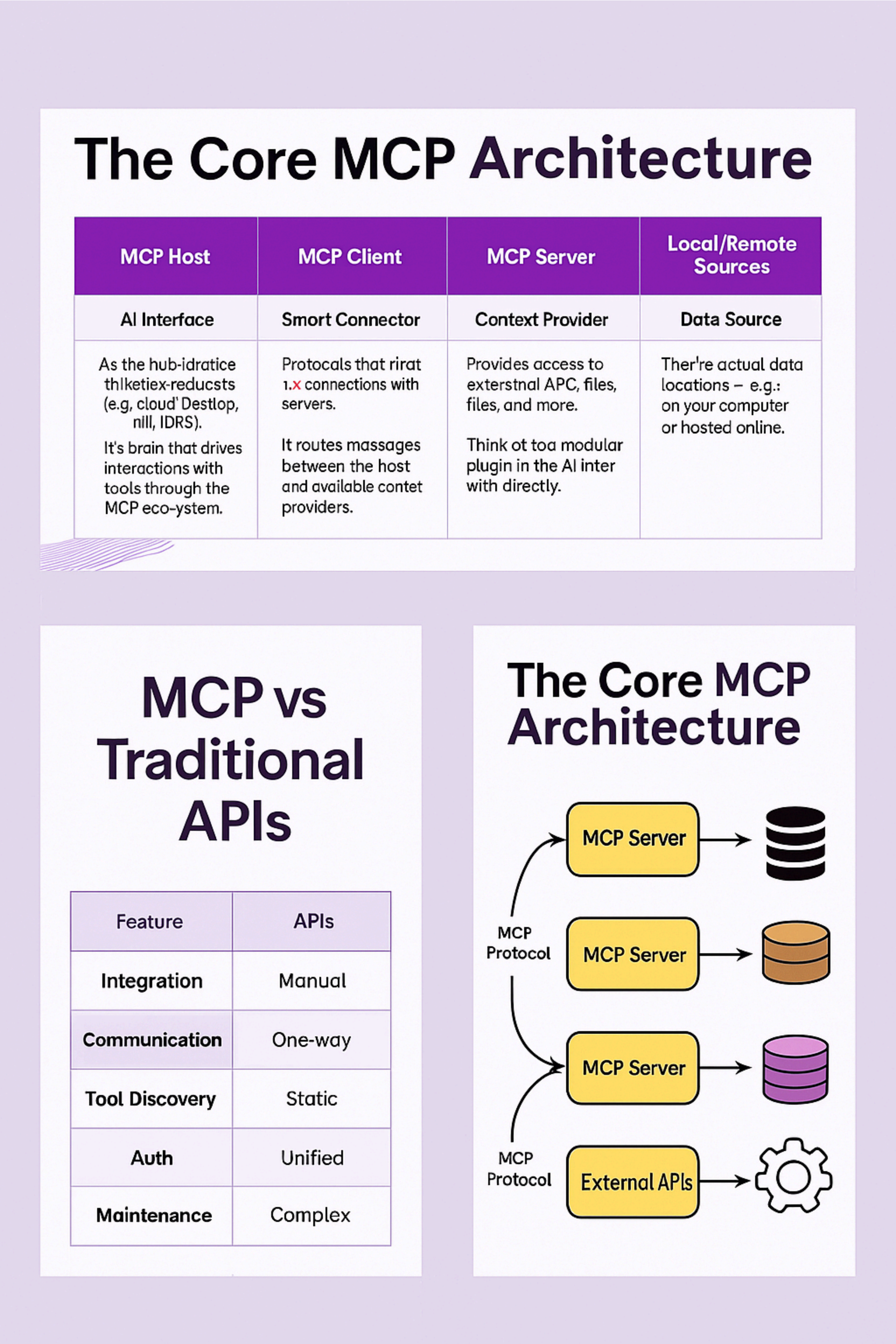
🧠 Model Context Protocol (MCP)



The Core MCP Architecture

MCP Host	MCP Client	MCP Server	Local/Remote Sources
AI Interface	**Smort Connector**	**Context Provider**	**Data Source**
As the hub-idratice thlketiex-reducsts (e.g, cloud' Destlop, nlll, IDRS). It's brain that drives interactions with tools through the MCP eco-ystem.	Protocals that rirat 1.x connections with servers. It routes massages between the host and available contet providers.	Provides access to exterstnal APC, files, files, and more. Think ot toa modular plugin in the AI inter with directly.	Ther're actual data locations – e.g.: on your computer or hosted online.

MCP vs Traditional APIs

Feature	APIs
Integration	Manual
Communication	One-way
Tool Discovery	Static
Auth	Unified
Maintenance	Complex

The Core MCP Architecture

📌 What is MCP?

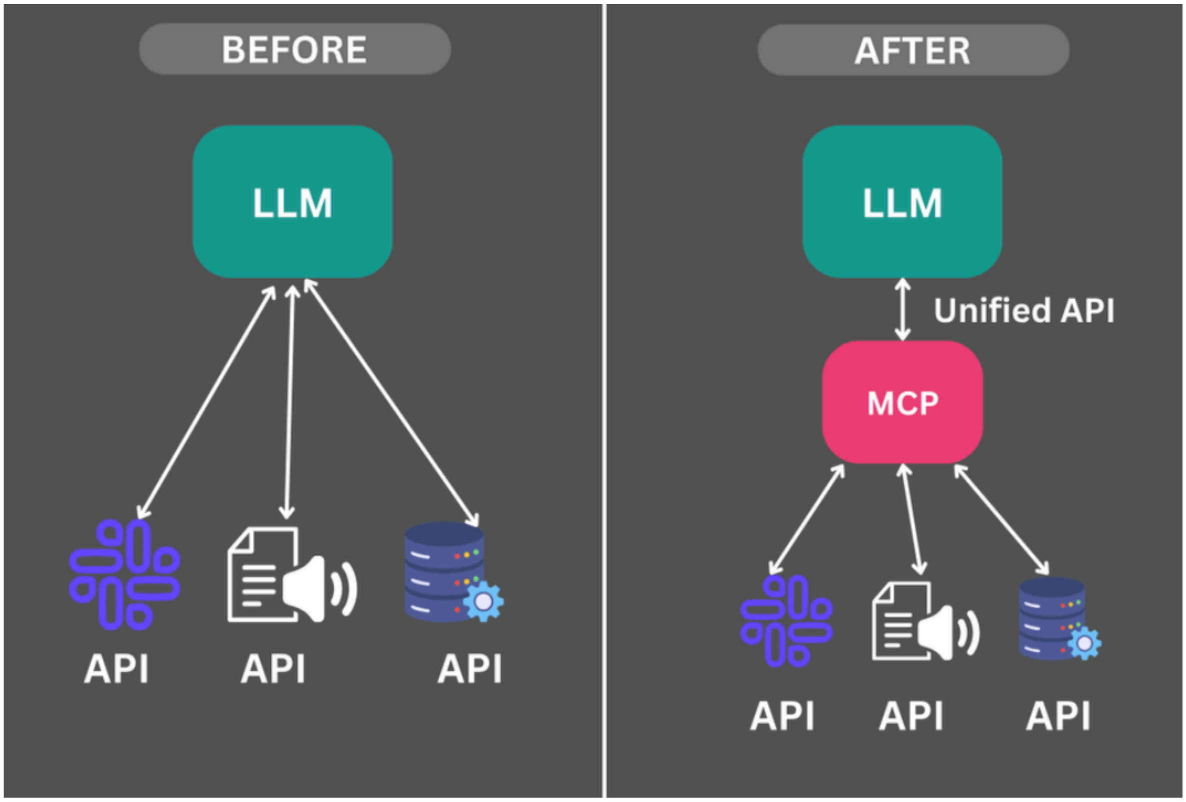
Model Context Protocol is a standardized open protocol to connect LLMs with external tools, data, and services. It's like a **USB-C for AI** — one unified connector that works dynamically and securely.

⚠️ How AI Accessed Tools Before MCP

- Manual API integrations (e.g., Google Drive, SQL)
- Platform-locked plugins (e.g., OpenAI-only systems)
- Agent frameworks (e.g., LangChain) needed custom setups
- Static context via vector DBs — no live interaction

🚀 Why MCP Matters

- Eliminates brittle, complex API setups
- Makes AI agents scalable and decoupled
- Supports real-time, two-way access to tools & data.



🧩 Core MCP Architecture

MCP Host

> The AI interface initiating context requests (e.g., IDE, desktop client)

MCP Client

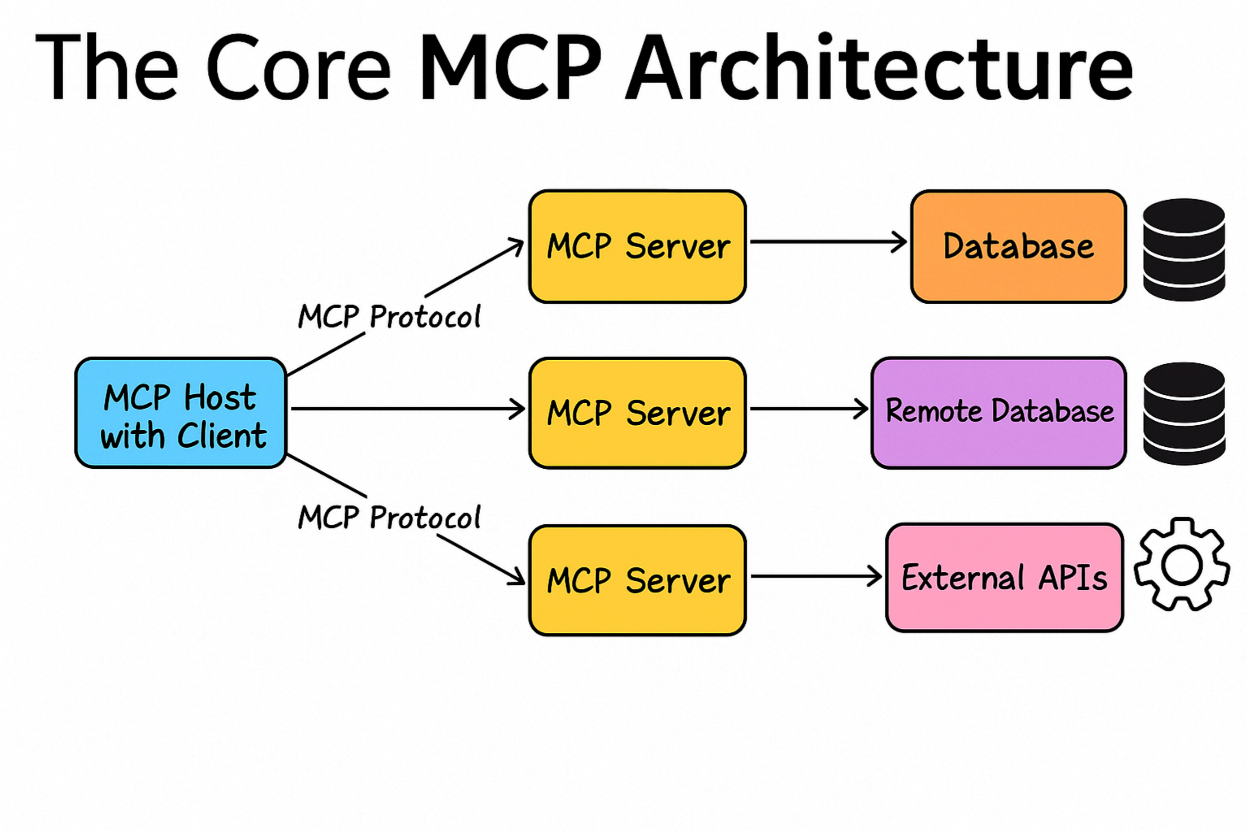
> Maintains communication between host and MCP servers

MCP Server

> Interfaces with external APIs, files, databases (context providers)

Data Sources

> Actual data endpoints, locally or remotely hosted



🔁 How MCP Works

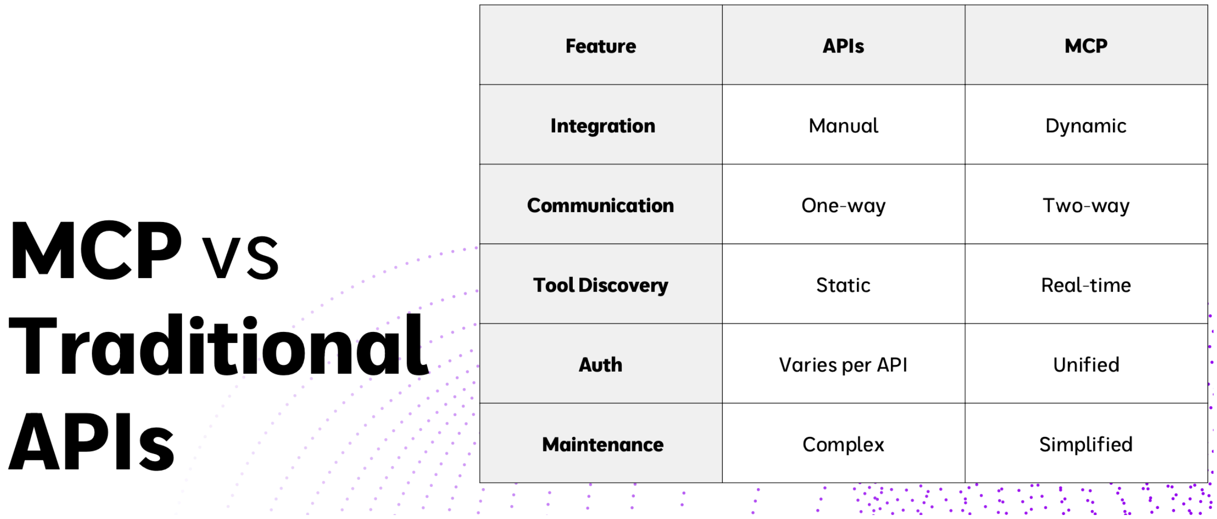
1. Host acts as the central communication hub

2. Client connects with multiple servers
3. Servers fetch, format, and return external data/tools
4. Results are processed and returned to the AI agent

🔄 Communication Model

- MCP uses **JSON-RPC 2.0** for message formatting
- Compatible with HTTP, WebSocket, etc.
- Enables real-time, ongoing conversations between LLMs and tools

📊 MCP vs Traditional APIs



MCP vs Traditional APIs

Feature	APIs	MCP
Integration	Manual	Dynamic
Communication	One-way	Two-way
Tool Discovery	Static	Real-time
Auth	Varies per API	Unified
Maintenance	Complex	Simplified

Feature	Traditional APIs	MCP
Integration	Manual	Dynamic
Communication	One-way	Two-way
Tool Discovery	Static	Real-time
Auth	Varies per API	Unified
Maintenance	Complex	Simplified

🤖 MCP in Agentic Workflows

- Makes AI workflows smarter and more dynamic
- Allows multi-step operations, context-based decisions, and live tool updates
- Enables agents to truly collaborate with real-world tools

✅ When to Use MCP

Use When:

- Building context-rich AI agents
- Needing dynamic access to tools, APIs, or files
- Creating scalable, modular agent systems

Avoid When:

- System demands tight latency and deterministic outcomes
- Working in restricted, secure-only environments

🔮 What's Next for MCP

- More prebuilt connectors (MCP servers)
- Integrated orchestration with **CrewAI**, **LangGraph**, etc.
- No-code/low-code UI tools for agent setup