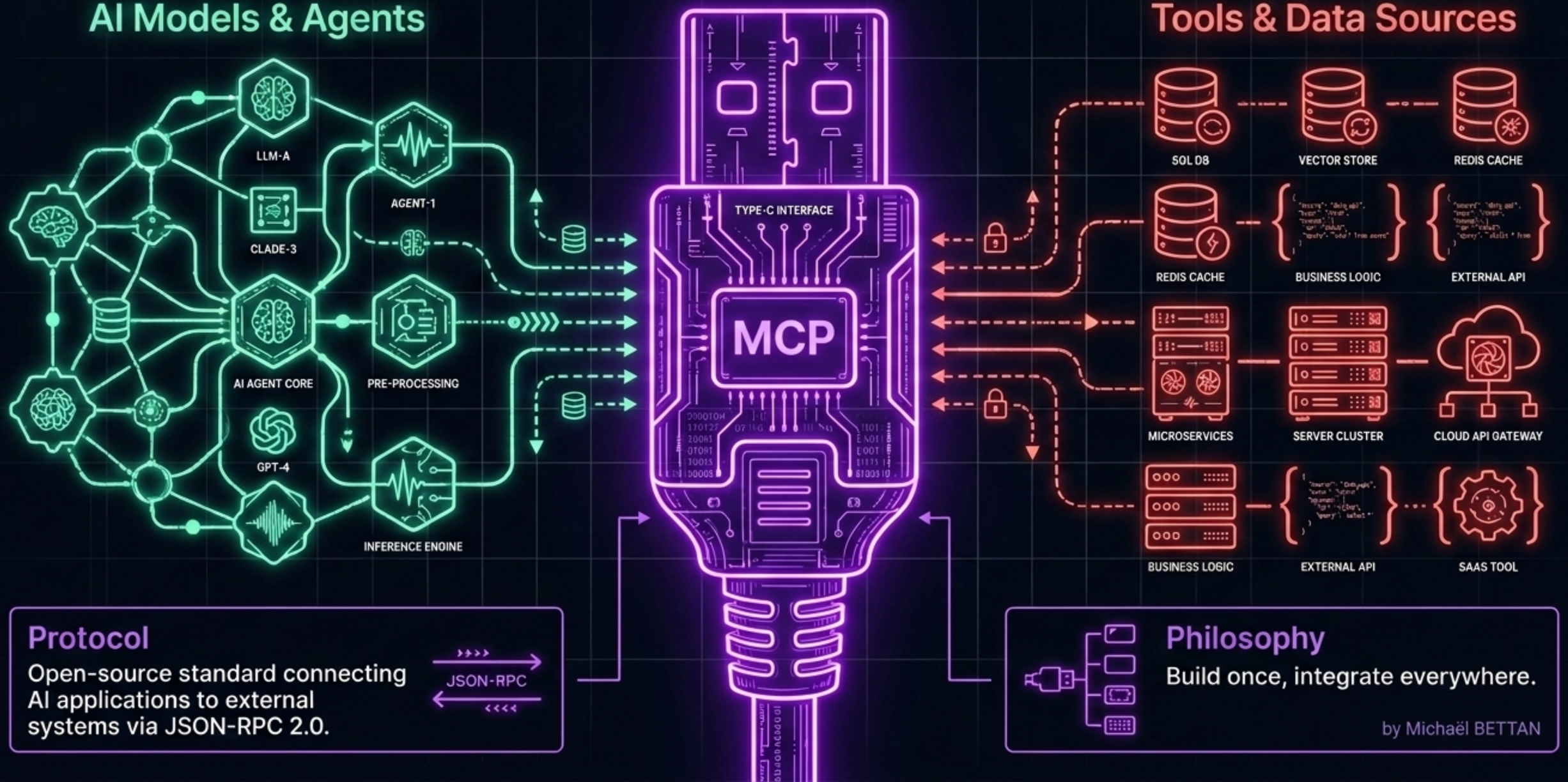
The Universal Adapter for AI Applications



Protocol

Open-source standard connecting AI applications to external systems via JSON-RPC 2.0.

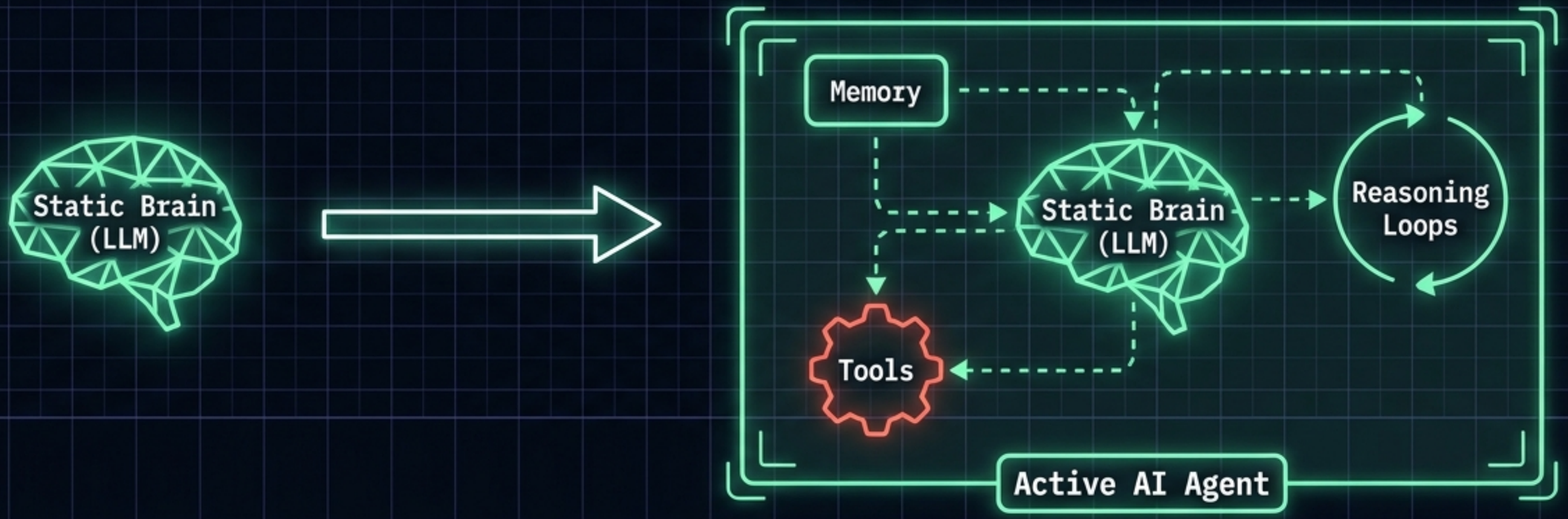
JSON-RPC

Philosophy

Build once, integrate everywhere.

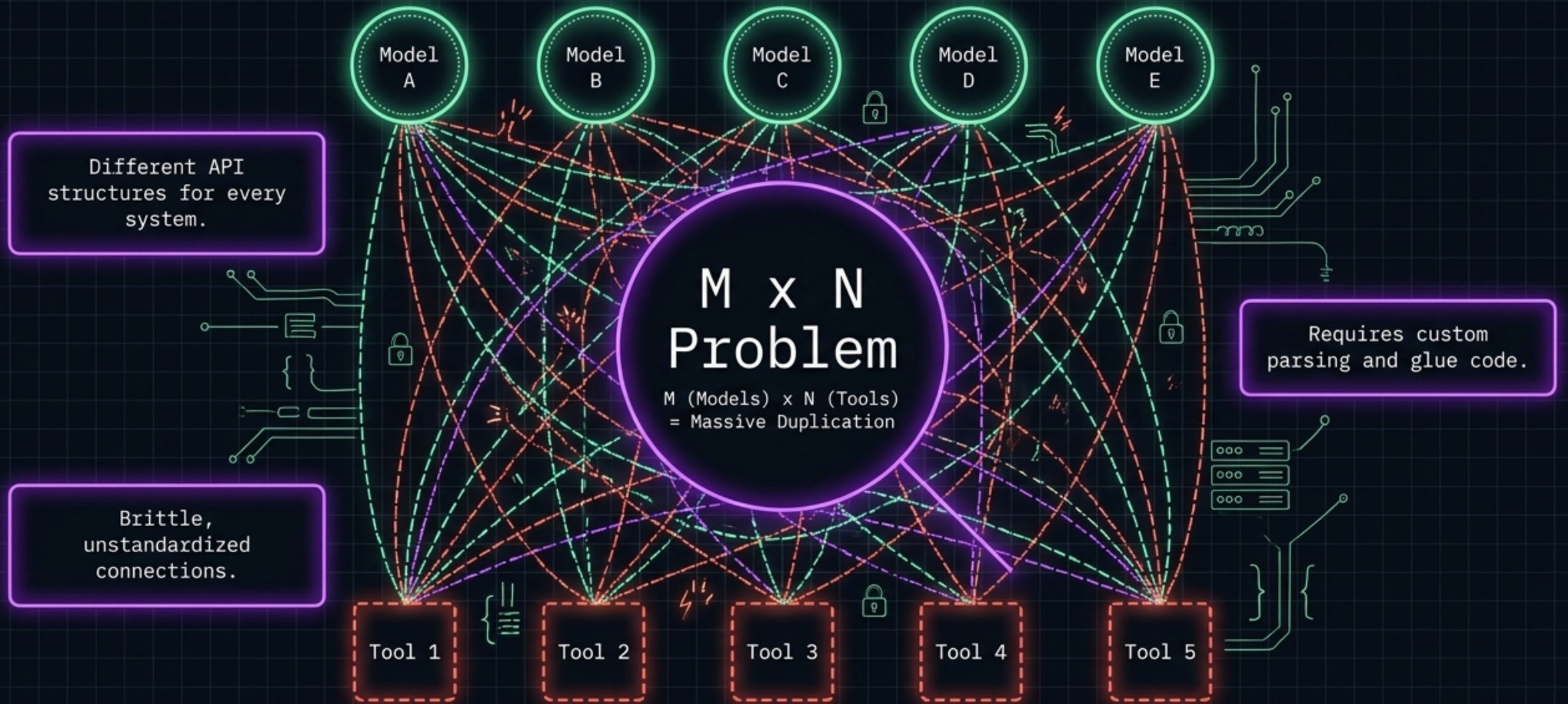
by Michaël BETTAN

Evolution from Static Brains to Active Agents

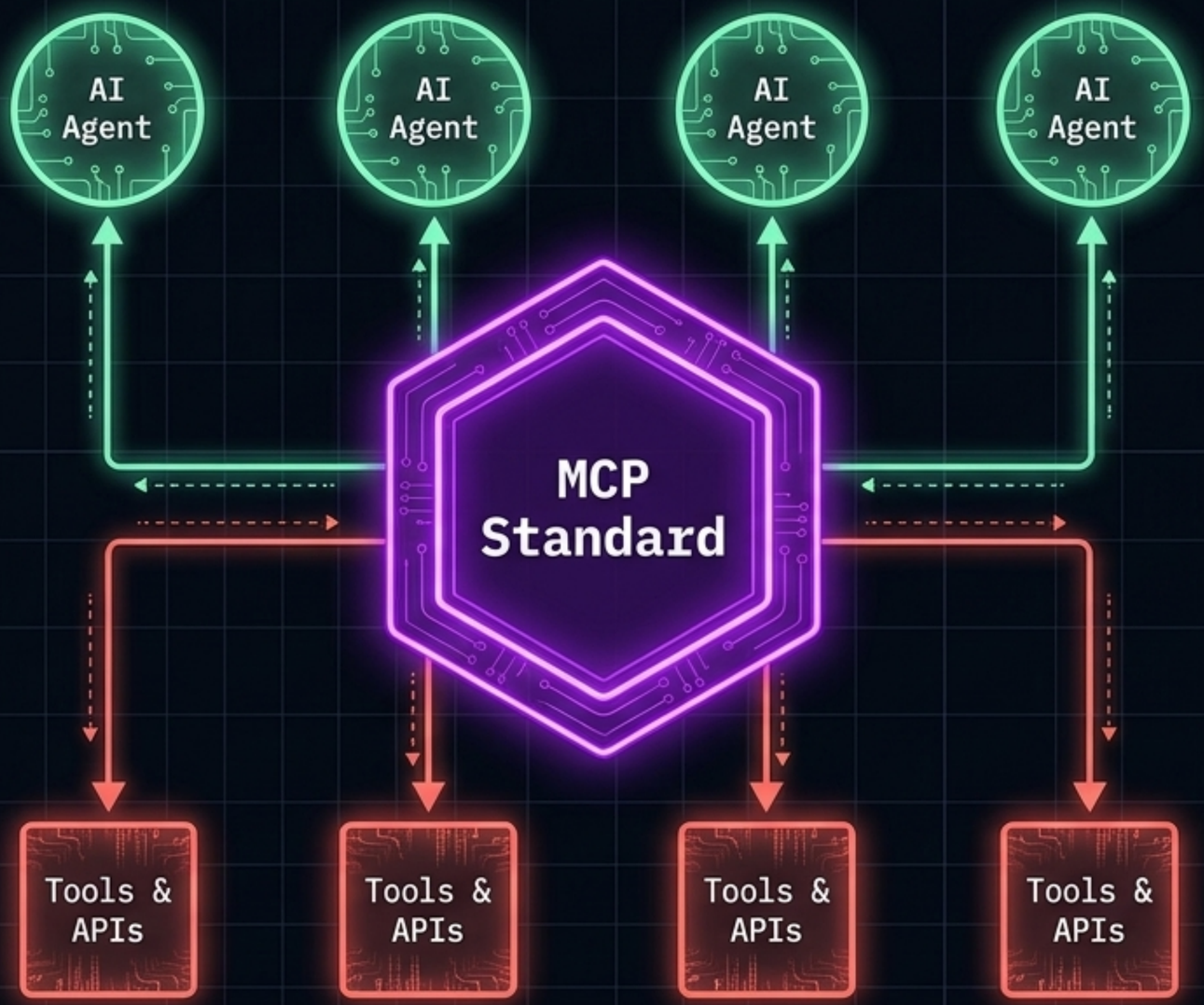


The Intelligence Matrix		
Entity	Large Language Model (LLM)	AI Agent
Capabilities	Generates text, images, video natively	Interacts with 3rd-party APIs, retrieves data, acts
Actionability	Isolated; cannot ‘do’ things	Dynamic; can scan code, book flights, execute tasks
Examples	Gemini, GPT-4, Claude	Claude Desktop, ChatGPT, VS Code Copilot, Cursor, Cline

The Unscalable Nightmare of Custom Integrations



Standardizing the AI-to-Tool Handshake



For Developers

Zero custom integration glue code. Build one MCP server to reach all supported agents.

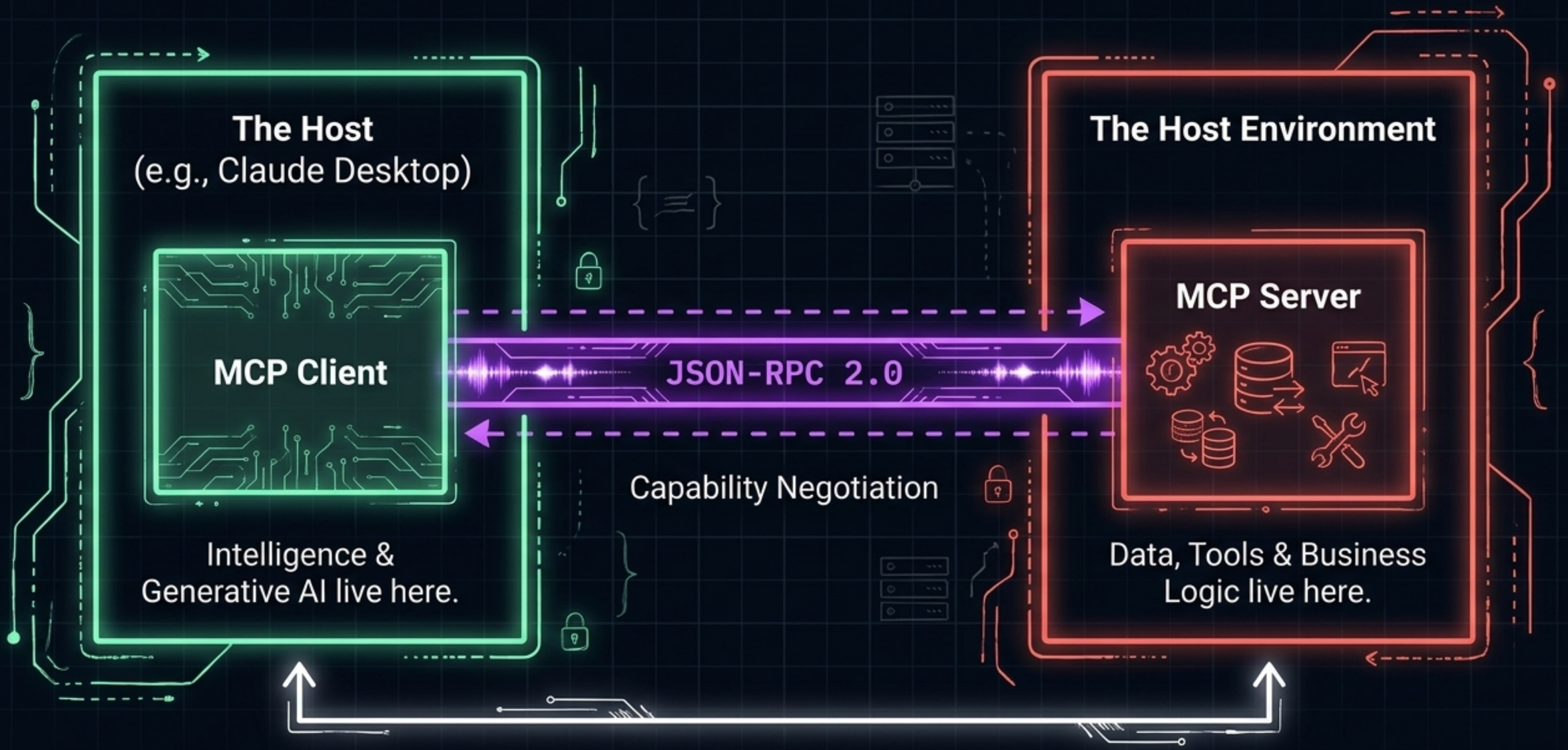
For AI Agents

Instant access to a massive ecosystem of external capabilities.

For End-Users

Personalized assistants that can securely hit APIs (like Google Calendar) and take real action.

Separation of Concerns in the MCP Architecture

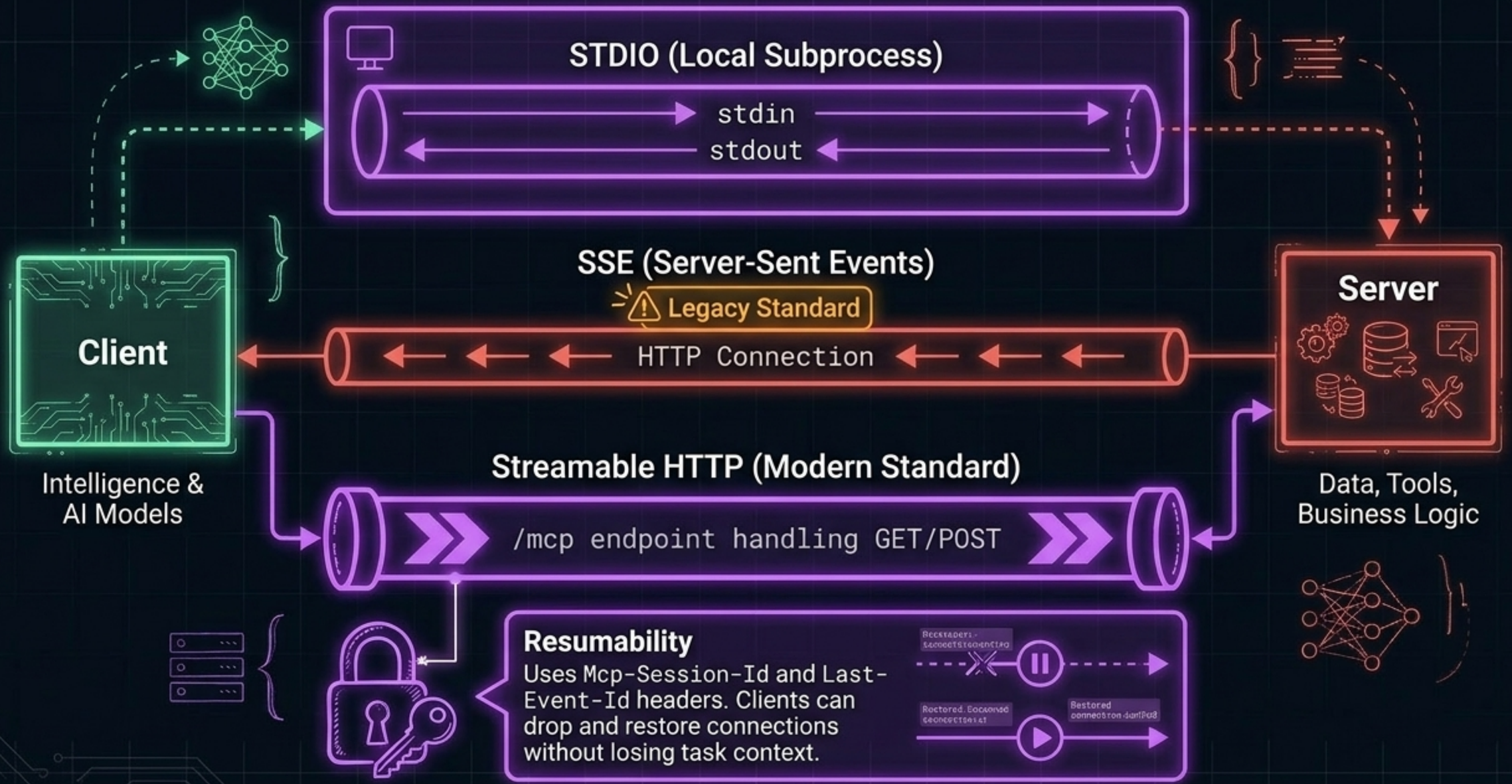


Key Rule: Clients request. Servers execute.

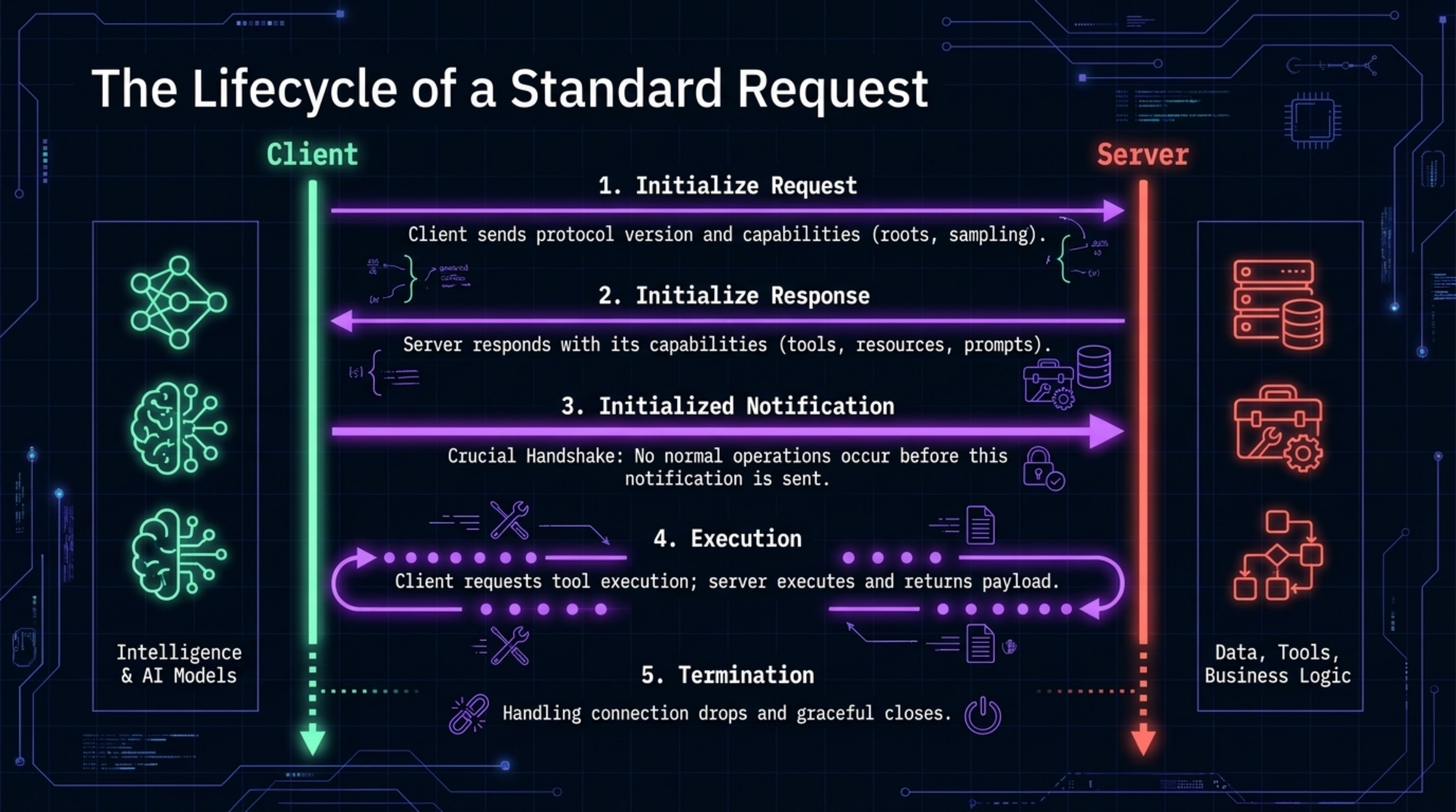
The Three Primitives of Server Capabilities

	Tools (External Actions)	**Resources** (Data Sources)	**Prompts** (Workflows)
Control	Model-controlled.	Application-controlled.	User-controlled.
Function	Executable functions the server performs on behalf of the LLM. Inputs generated via libraries like Zod.	Static data/context provided to empower the LLM. Can be fixed (`config://app`) or templated (`settings://{type}`).	Server-hosted templated messages guiding interactions. Automatically injects server-side context.
Analogy/ Example	Action Pattern (e.g., `search_flights`, `execute_query`).	RAG Pattern (Retrieval-Augmented Generation).	Workflow Template (e.g., '`write product description`').

The Transport Layer Matrix



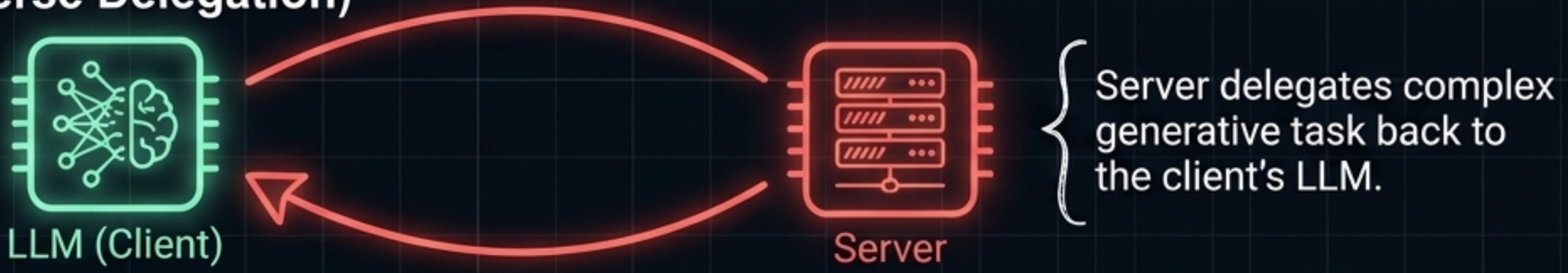
The Lifecycle of a Standard Request



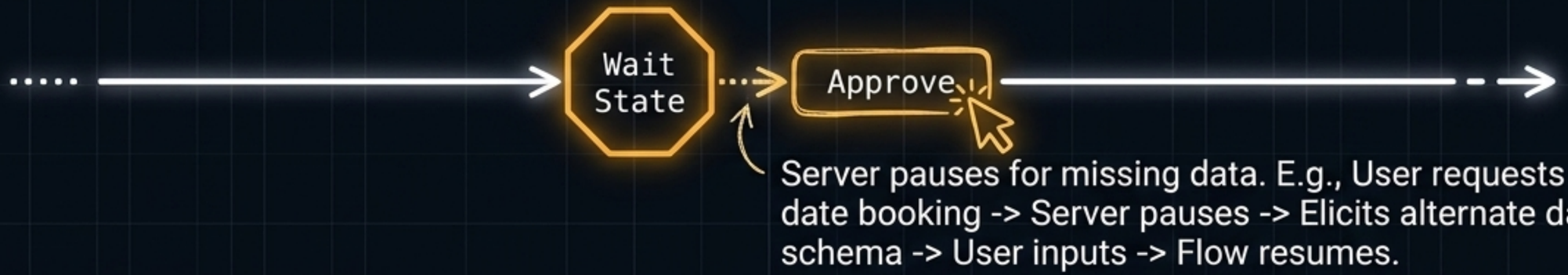
Advanced Interactive Workflows

Circuit 1 - Notifications

Circuit 2 - Sampling (Reverse Delegation)

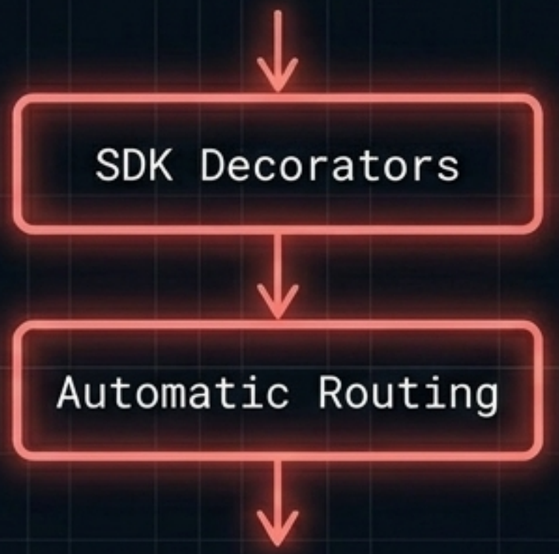


Circuit 3 - Elicitation (Human-in-the-loop)



Server Architecture and Implementation Constraints

High-Level API



Fastest path for simple servers.

Low-Level API

Required for manual handling of Sampling and Elicitation.

Context Managers

Python SDK constructs that precisely allocate and clean up server resources (like DB connections) safely, even on error.

Client Configuration (mcp.json)

Host environments map servers via command (node, python) and args.

Warning: Never hardcode API keys here—inject via environment variables.

Production Security and Operations

Basic Auth (Low Security)

Simple API keys sent via HTTP headers.

JWT (Stateless Scopes)

JSON Web Tokens mapping specific permissions (e.g., User.Read) to enable granular Role-Based Access Control (RBAC).

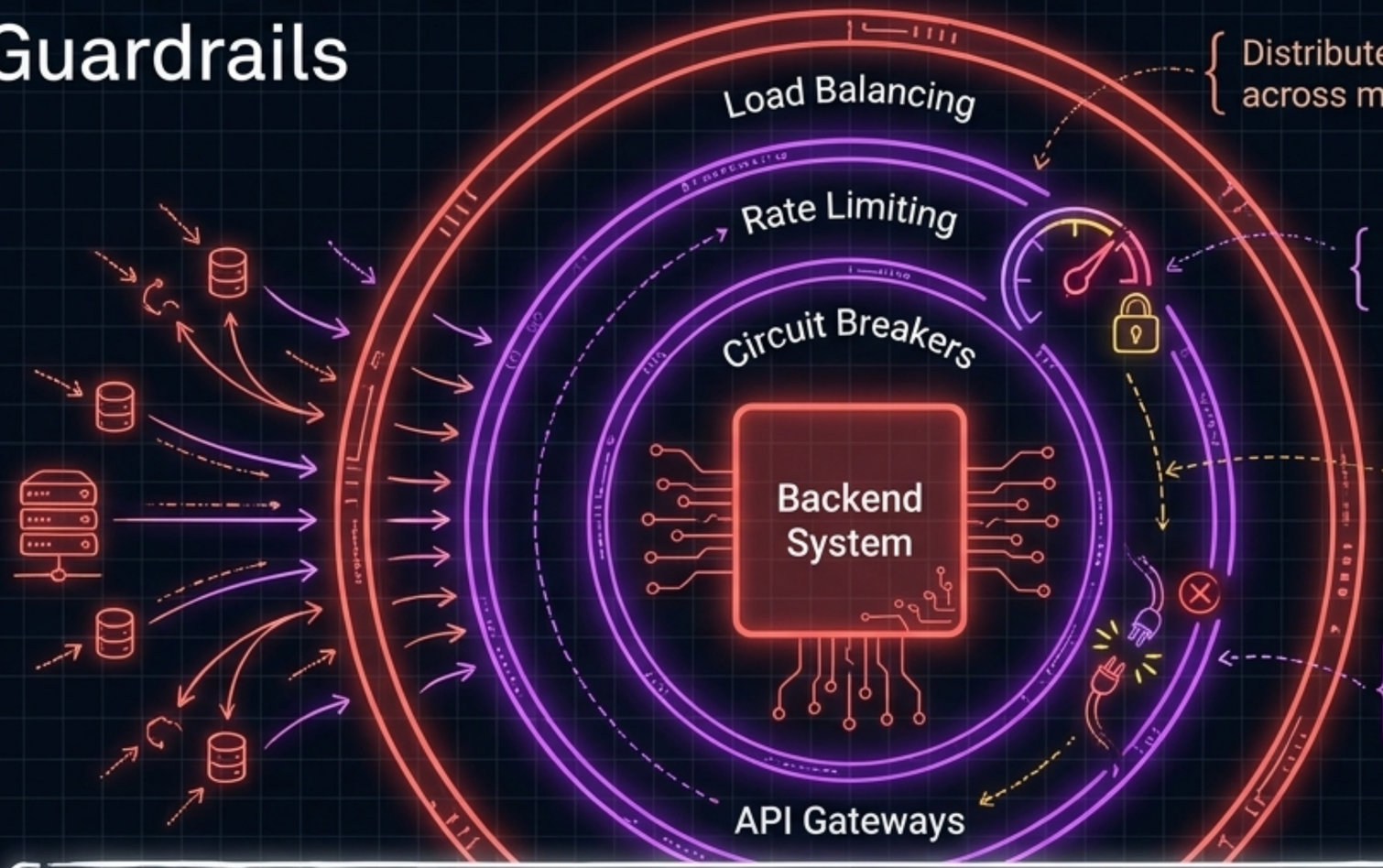
OAuth 2.1 (Robust Decoupling)

Uses external Identity Providers (IdP) to issue tokens, fully removing credentials from the resource server.

Observability Panel

Observability: Implement distributed tracing, token budgeting metrics, and structured logging to monitor AI behavior.

Resilience Mechanisms and AI Guardrails



Distributes traffic evenly across multiple nodes.

Distributes traffic evenly across multiple nodes.

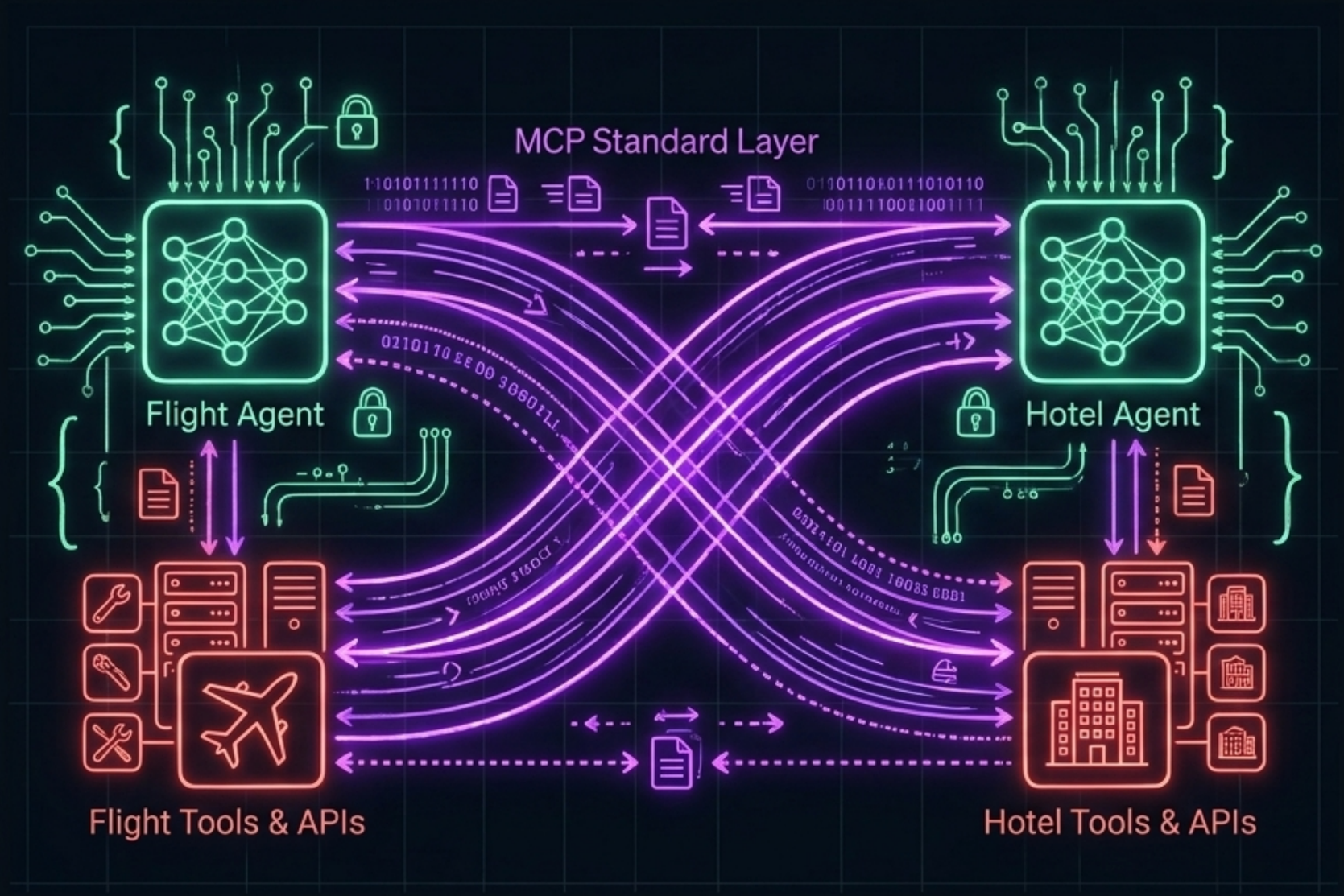
Prevents token/cost abuse by aggressive agents.

Fails gracefully when downstream services go offline.

Testing Guardrails

AI behavior is non-deterministic. CI/CD pipelines must run Adversarial AI testing to catch prompt injections and hallucination regressions, ensuring HIPAA/GDPR compliance.

The Synthesis: Multi-Agent Collaboration



The Insight

The Problem with Monoliths: One bloated agent trying to do everything is brittle and error-prone.

The Agent-to-Agent Model: Pioneered by Google, specialized agents use standard MCP protocols to dynamically discover each other's capabilities.

The Result: They independently assign tasks, share context, and solve complex problems collaboratively without centralized hardcoding.