

AUTONOMOUS AI AGENTS ORCHESTRATION AND NO-CODE DESIGN



Engineering – Ingegneria Informatica S.p.A.

Salvatore Cipolla and Daniele Fakhoury

INTRODUCTION AND AGENDA

Webinar Agenda

The GenAI Era

From the advent of general-purpose AI models to autonomous agent systems.

Autonomous Agents

Introduction to the growing importance of autonomous agents in the current technological and market context.

Emerging Protocols for Agentic AI System

Presentation of emerging communication protocols (MCP, A2A) as solutions for standardized tool integration and multi-agent collaboration in Agentic AI architectures.

No-code Workflow Design

A no-code workflow designer enables the visual orchestration, control, and evolution of agent-based systems

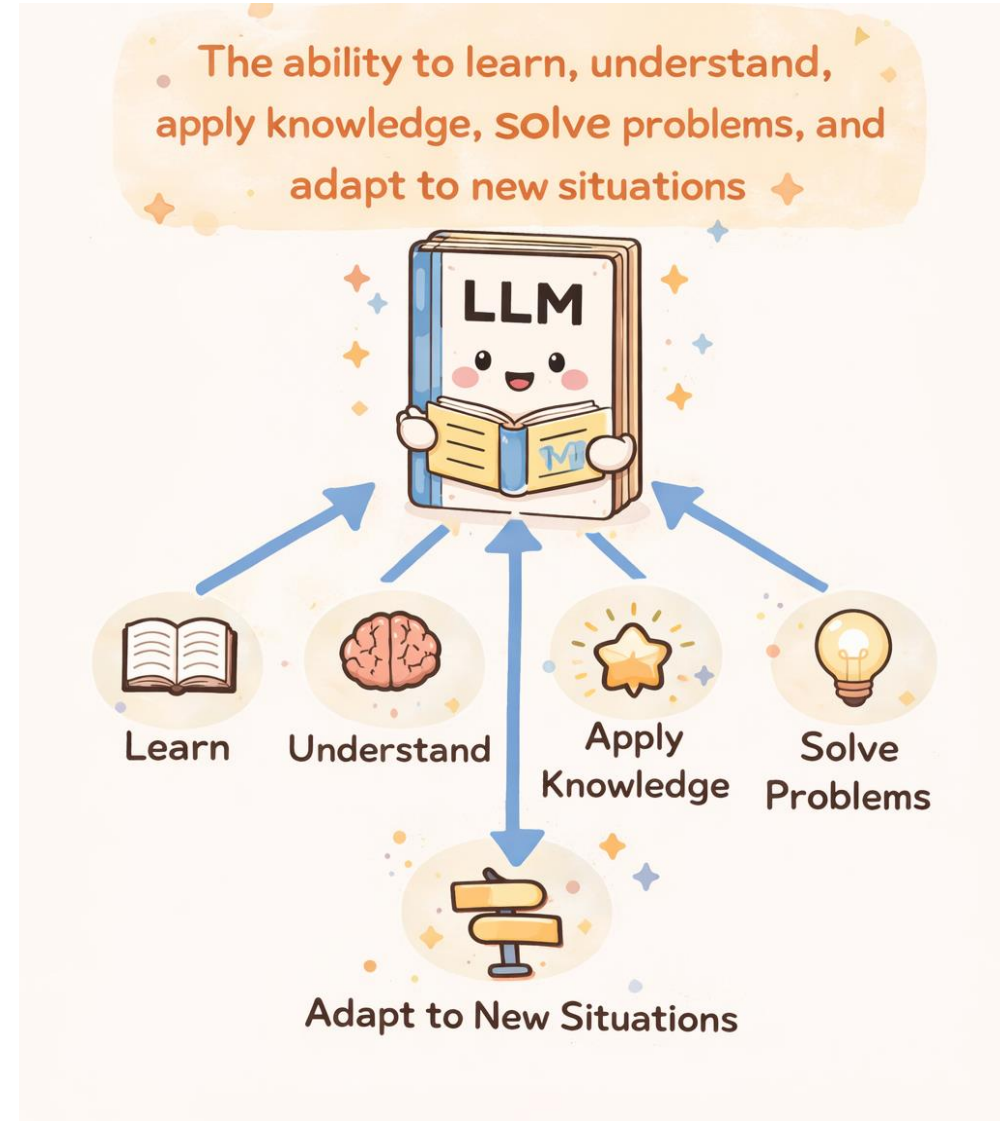
Example use case, outlook and conclusions, Q&A

Final session with conceptual demonstration and time for questions and answers.

FOUNDATIONS OF AUTONOMOUS AGENTS

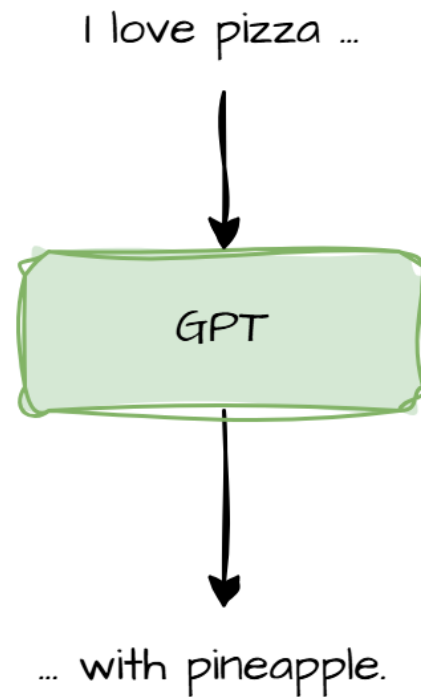
Prerequisite of autonomy

Autonomy --> Intelligence



What is an autonomous agent?

Autonomous agent heart: the LLM, just a next token predictor...



What is an autonomous agent?

Premise 1: LLMs possess general knowledge



What is an autonomous agent?

What is the main job of post-training for GPT models? Follow instructions from human feedback.
E.g., we can teach the LLM to learn a fixed structure:

We know input structure
in advanced

We know input structure
in advanced

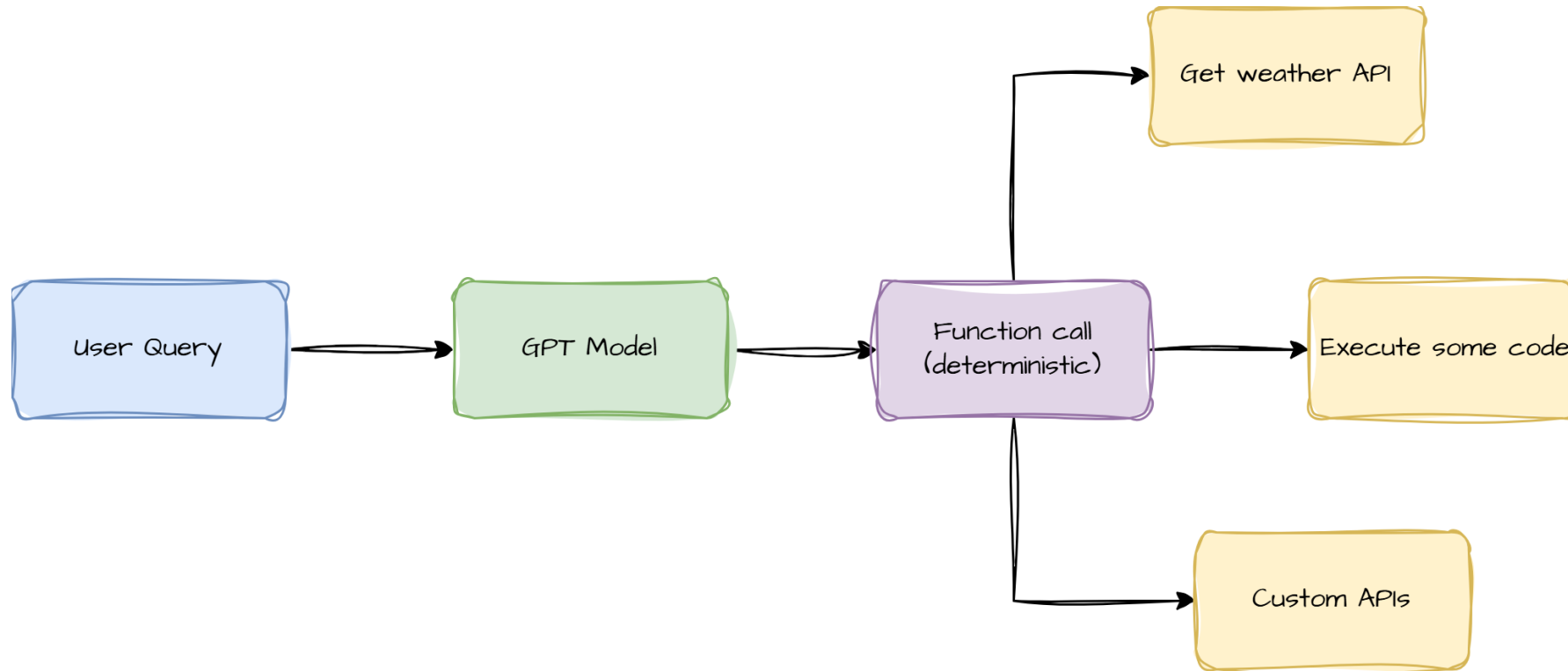
```
def add(a: float, b: float) -> float:
    """Return the sum of two numbers."""
    return a + b
```

We hopefully have clear
natural language
documentation for each
function

Ouyang, Long, et al. "Training language models to follow instructions with human feedback." *Advances in neural information processing systems* 35 (2022): 27730-27744.

What is an autonomous agent?

Premise 2: LLM knows how to perform actions (we call it tool calling)



What is an autonomous agent?

Premise 3: LLM knows how to decompose a task into smaller tasks and their order of execution (we call it task decomposition)



Tell me the steps needed to fry an egg. Do it in a very concise way, using bullet points

- Heat a nonstick pan over medium heat.
- Add a small amount of oil or butter.
- Crack the egg into the pan.
- Cook until whites are set; optionally cover for a firmer top.
- Season with salt and pepper.
- Serve immediately.

ReAct Agent

(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.

Act 1: `Search[Apple Remote]`

Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple ... originally designed to control the Front Row media center program ...

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.

Act 2: `Search[Front Row]`

Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports', 'Front Row (software)', ...]

Thought 3: Front Row is not found. I need to search Front Row (software) .

Act 3: `Search[Front Row (software)]`

Obs 3: Front Row is a discontinued media center software ...

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.

Act 4: `Finish[keyboard function keys]`

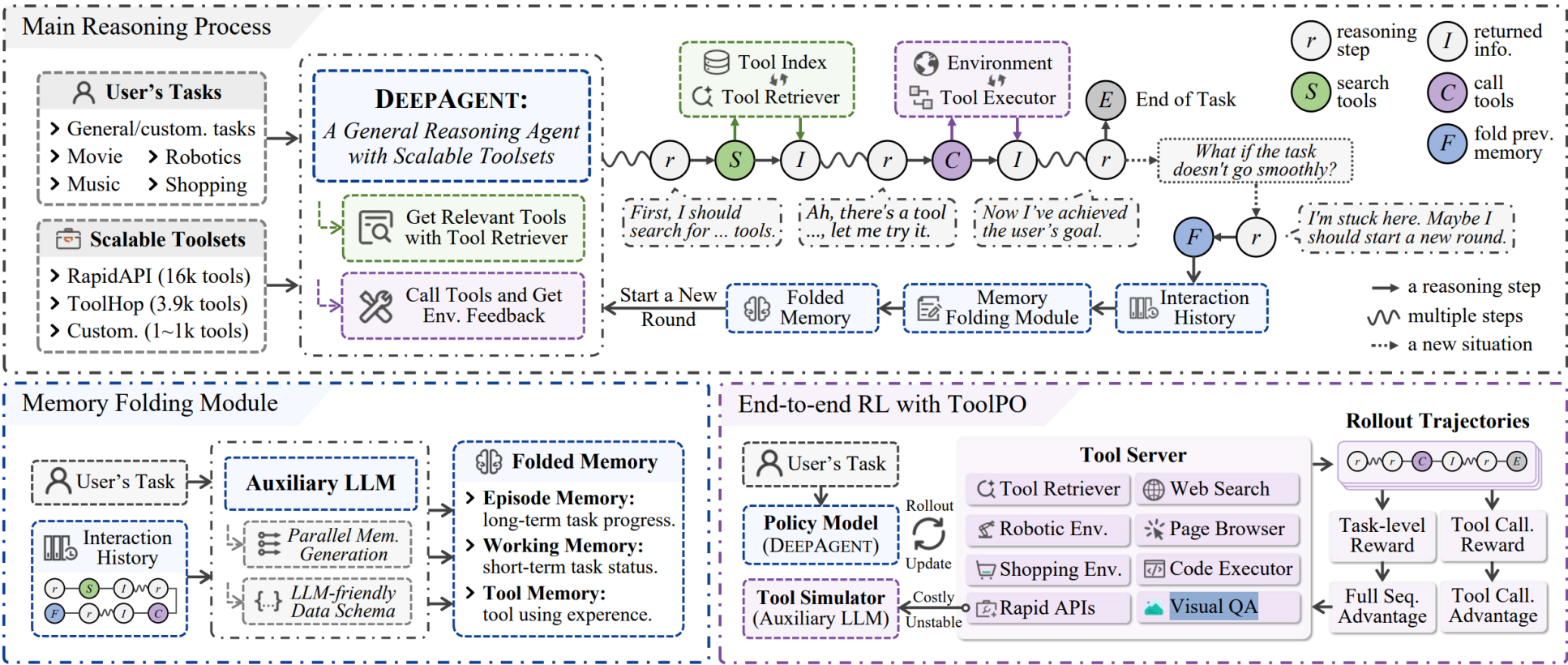


Yao, Shunyu, et al. "React: Synergizing reasoning and acting in language models." *The eleventh international conference on learning representations*. 2022.

Real world working agents: Deep Agents

Context management

Tools Management



Memory

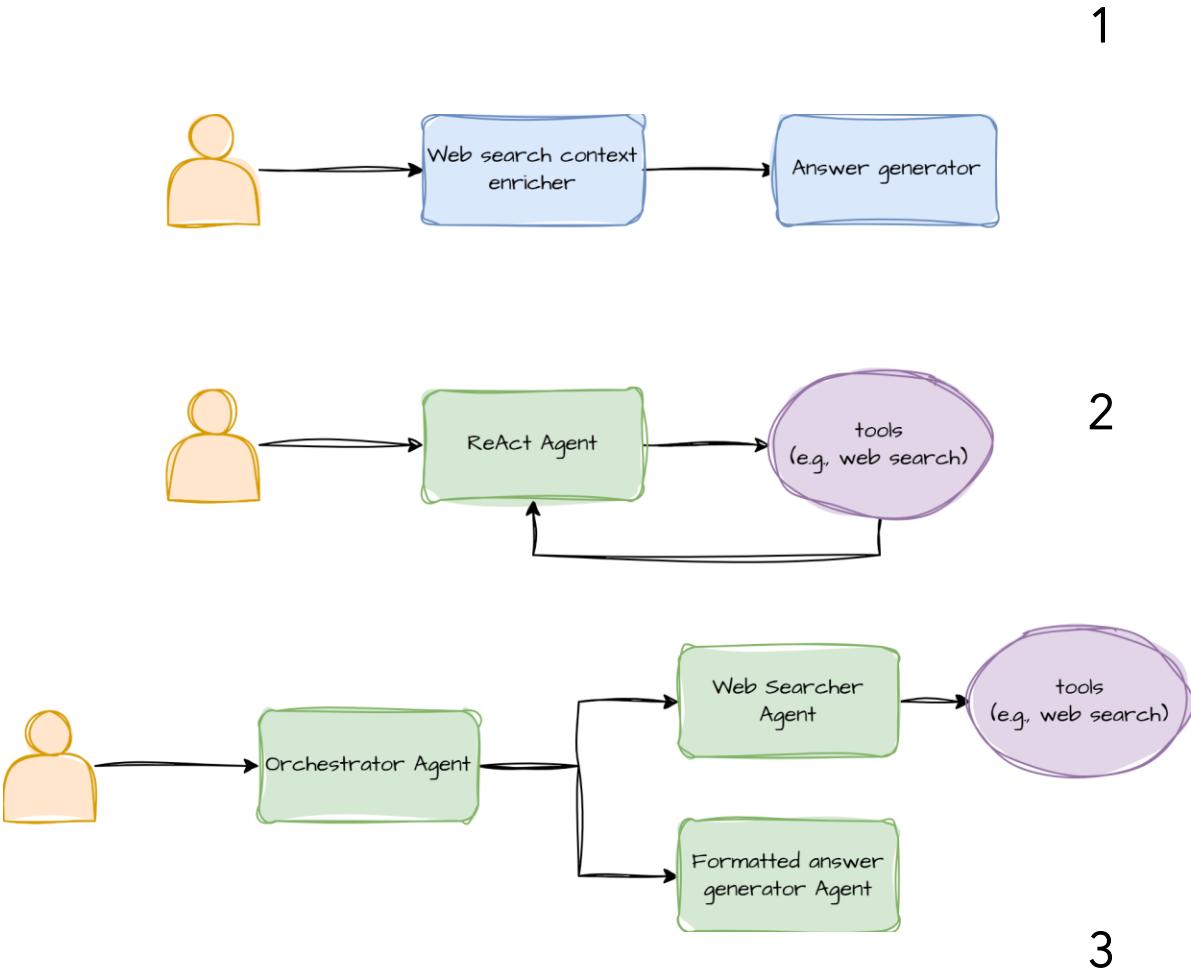
Safe code execution sandbox

Runtime agent spawning

Li, Xiaoxi, et al. "DeepAgent: A General Reasoning Agent with Scalable Toolsets." arXiv preprint arXiv:2510.21618 (2025).

What is an Agentic Workflow

Approach	Pros	Cons
Deterministic: all steps are known in advance.	✓ Most reliable, easy to implement	✗ Not versatile, works on pre-defined use cases and conditions.
Generative approach: the agent decides what to do (trajectory generated at runtime)	✓ More adaptable to user request and changing conditions	✗ Less reliable; highly dependent on available tools.
Mixed approach	✓ Best of both worlds	✗ Not as adaptable as a single agent with tools; more complex implementation.



Orchestration strategies

1. Intelligent Routing and Expert Selection

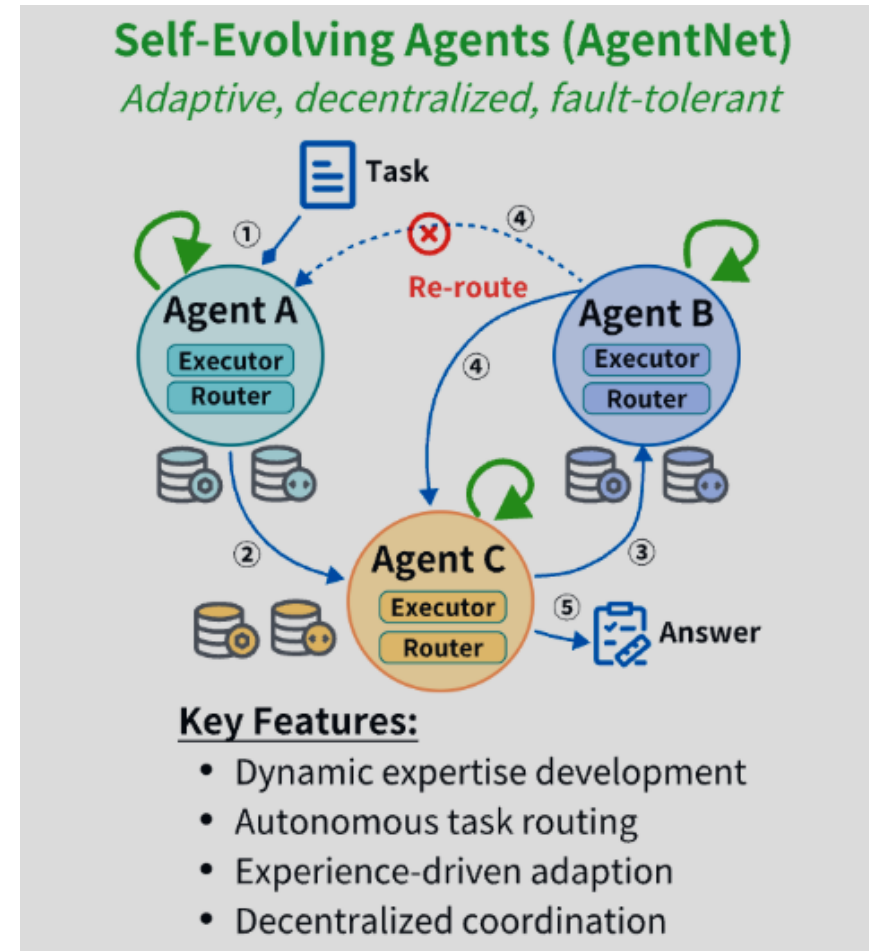
The most prevalent strategy is dynamic agent routing, where a router component classifies incoming tasks and directs them to specialized agents.

2. Hierarchical Multi-Agent Architectures

Hierarchical structuring remains fundamental, with three primary layers: Planner (strategy design), Specialized Agents (domain-specific execution), and Supervisor (result aggregation)

3. Decentralized Coordination

Contrasting with centralized orchestrators, AgentNet (on the left) introduces decentralized evolutionary coordination where agents leverage diverse knowledge sources while safeguarding sensitive information through minimal data exchange.



Yang, Yingxuan, et al. "Agentnet: Decentralized evolutionary coordination for llm-based multi-agent systems." *arXiv preprint arXiv:2504.00587* (2025).

EMERGING PROTOCOLS FOR AGENTIC AI SYSTEM



Introduction to Agentic AI Emerging Protocols

MCP is an open-source protocol that enables the seamless integration AI apps and agents with tools and data sources.

A2A facilitates communication between agents, enabling effective collaboration, delegation, and negotiation.

These standards create consistent and secure ecosystems for efficient and scalable multi-agent interactions.

Why Protocols?

Standard protocols are essential for enabling agentic interoperability, particularly in connecting agents to external systems. This is critical in two interconnected areas of innovation: Tools and Agents.

***Tools** are primitives with structured inputs and outputs and (typically) well-known behavior. **Agents** are autonomous applications that can accomplish novel tasks by using tools, reasoning, and user interactions. Agentic applications must use both tools and agents to accomplish goals for their users.*

What is MCP?

Launched in November 2024

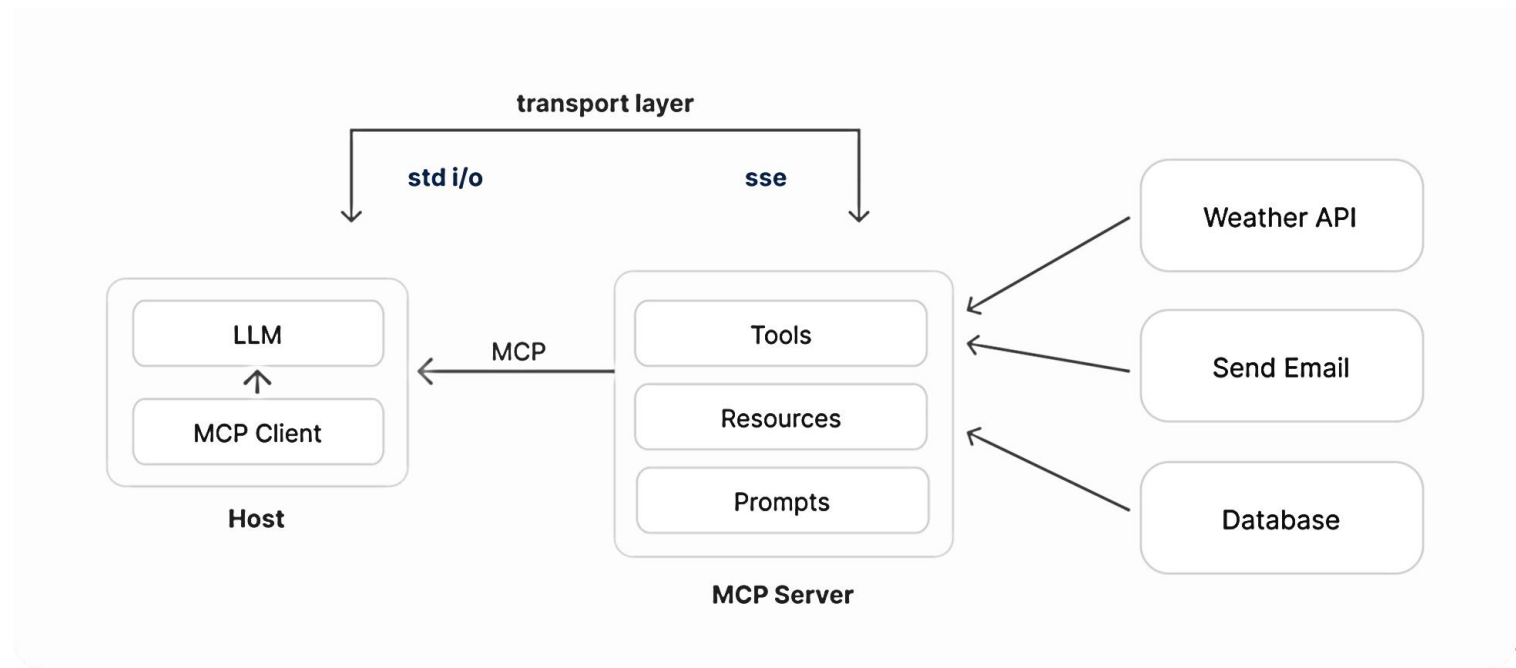
The main **goal** is to simplify the connection between AI models and external environments to enhance data interaction.

Key Features:

- Simplified integration
- Standardization



Model Context Protocol



MCP Features

Standardization of AI requests

MCP standardizes the way AI models request and use tools, facilitating integration.

Greater interoperability

Promotes interoperability between different platforms and applications for seamless communication.

Common language for AI

Creates a common language that simplifies communication between AI models and tools.



MCP Flow

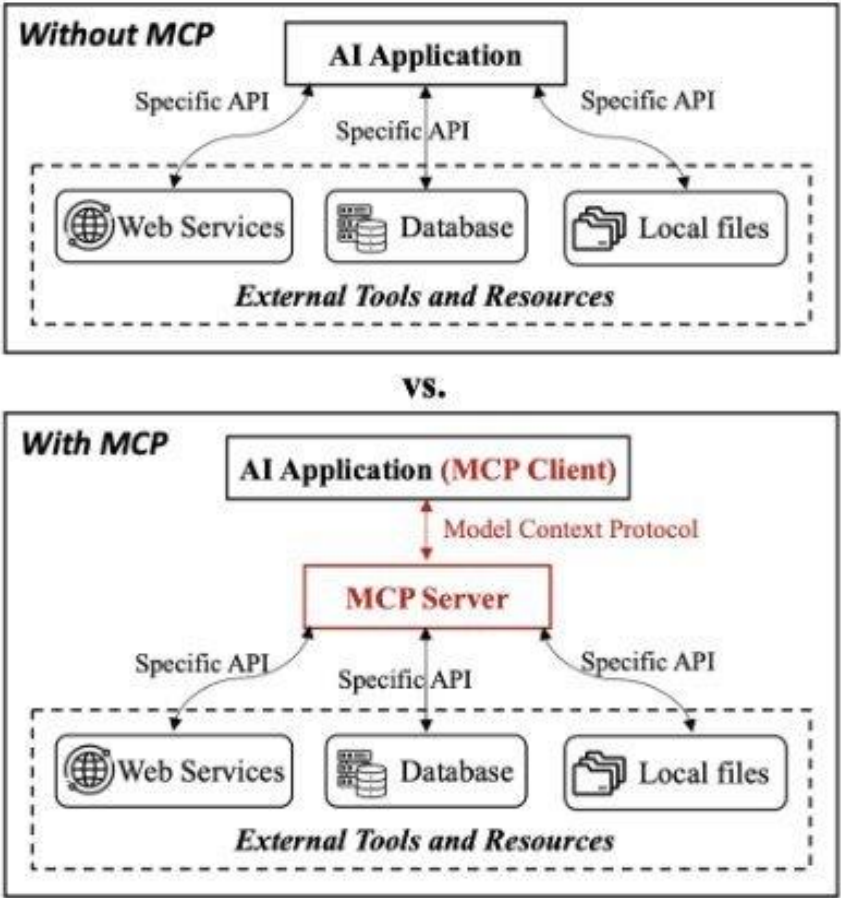
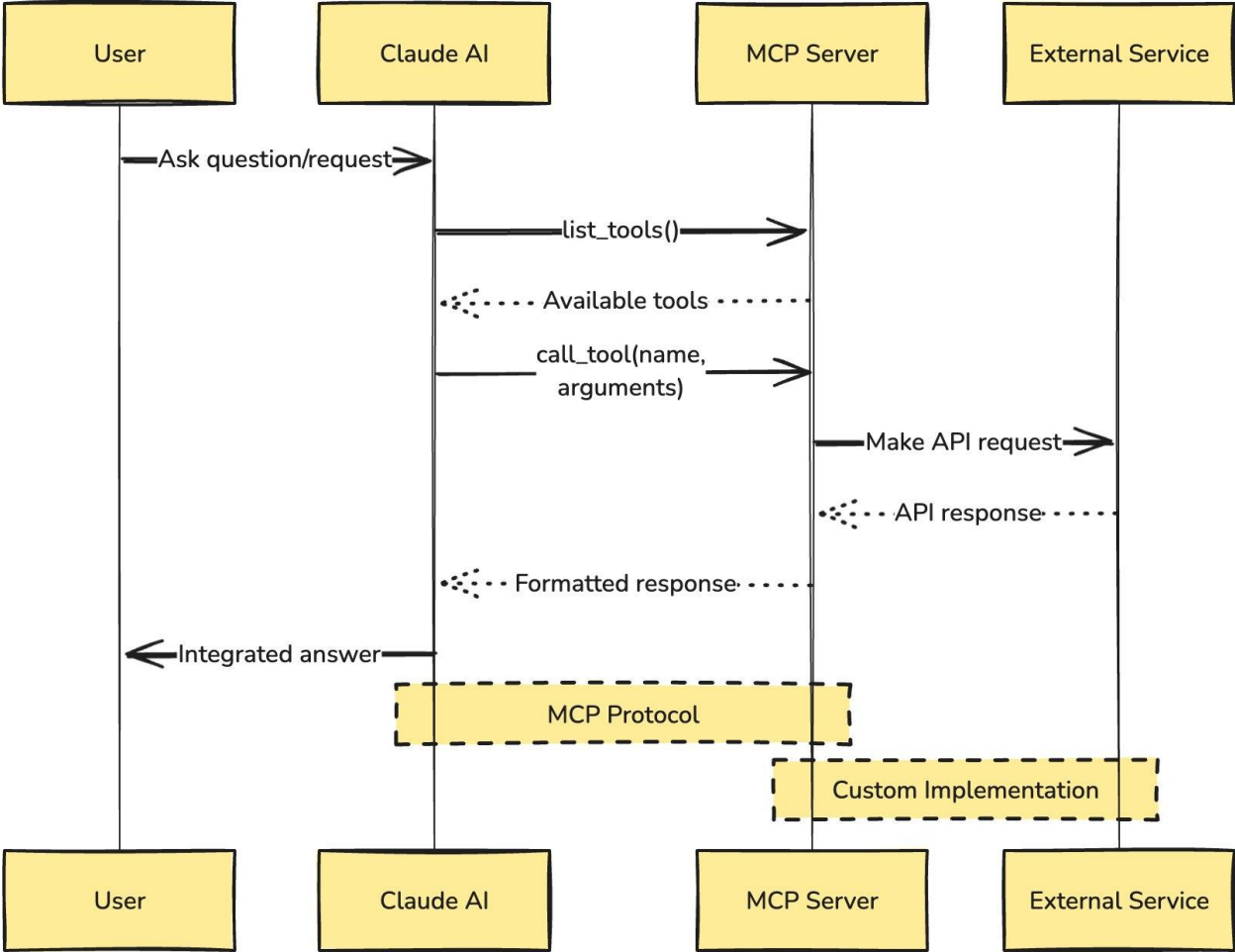


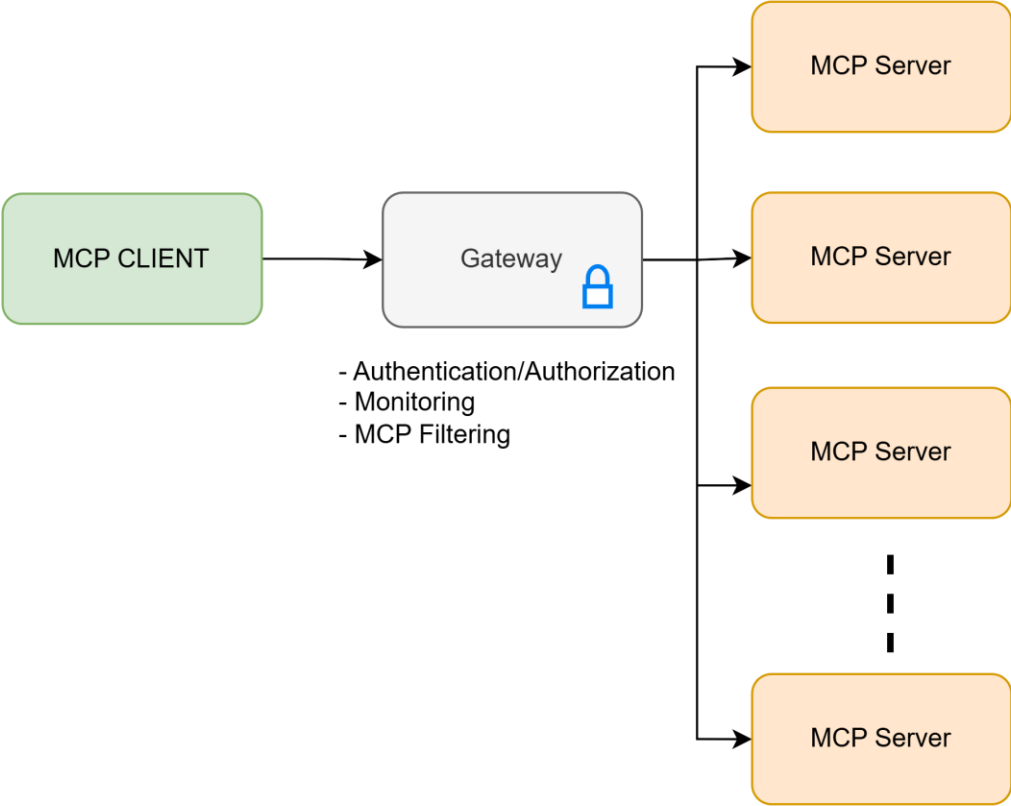
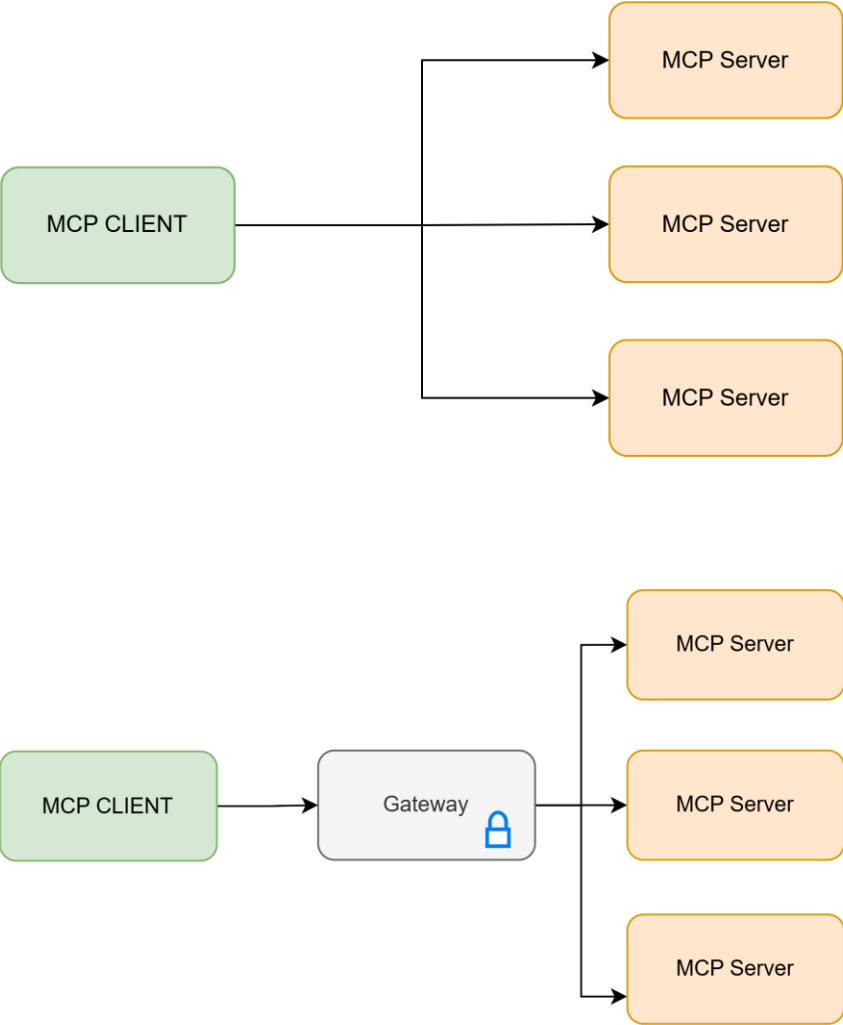
Figure 1: Tool invocation with and without MCP.



MCP Servers

- **Brave Search** - Web and local search using the Brave Search API
- **Puppeteer** - Browser automation and web scraping
- **EverArt** - AI image generation using various models
- **Filesystem** - Secure file operations with configurable access controls
- **Git** - Tools for reading, searching, and manipulating Git repositories
- **GitHub** - Repository management, file operations, and integration with the GitHub API
- **GitLab** - GitLab API, enabling project management
- **Google Drive - File** access and search capabilities for Google Drive
- **Google Maps** - Location services, directions, and place details
- **Memory** - Persistent memory system based on knowledge graphs
- **PostgreSQL** - Read-only database access with schema inspection
- **Slack** - Channel management and messaging capabilities
- **Sqlite** - Database interaction and business intelligence capabilities
- **Time** - Time and time zone conversion capabilities
- [GitHub - modelcontextprotocol/servers: Model Context Protocol Servers](#)

MCP - Potential Design





WHAT IS A2A COMMUNICATION?

Coordination between agents

A2A communication allows agents to coordinate activities efficiently and collaboratively.

Shared goals

Agents share goals and objectives to create a collaborative and dynamic ecosystem.

Dynamic interactions

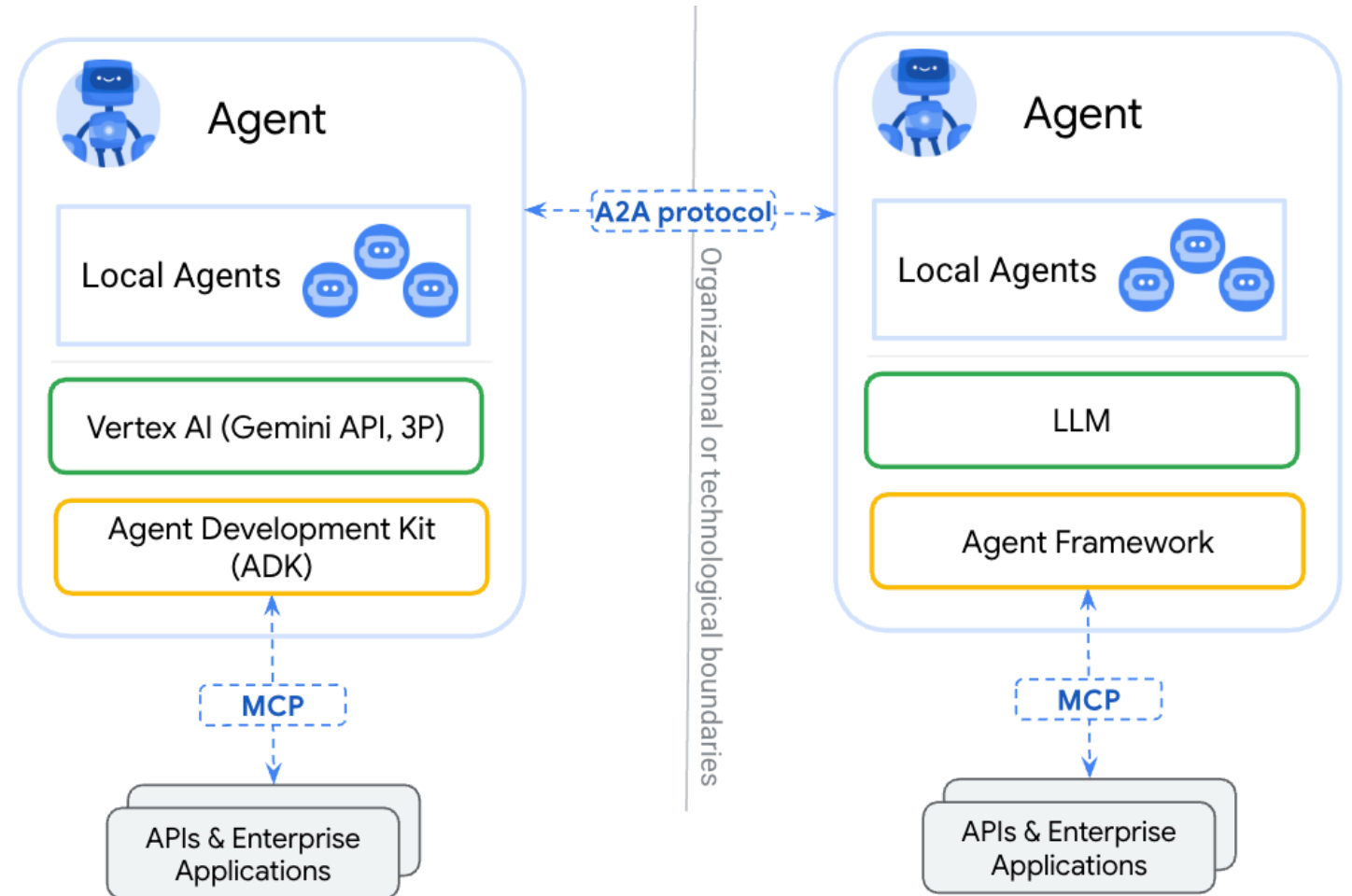
Essential for complex workflows with dynamic interactions between multiple agents.

A2A Protocol

Architecture

- **CLIENT:** defines and sends the task
- **REMOTE SERVER:** receives, executes, and responds

A2A protocol



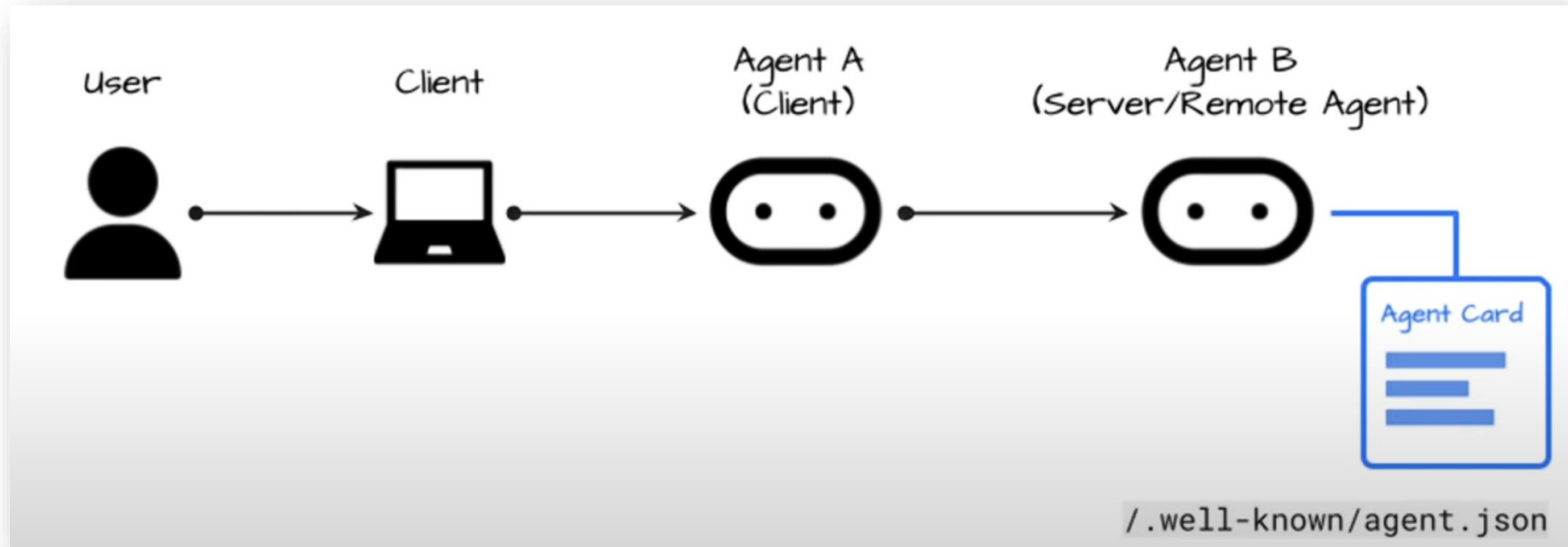
A2A Protocol - Step 1: Agent Discovery

Agent B publishes an **Agent Card** :

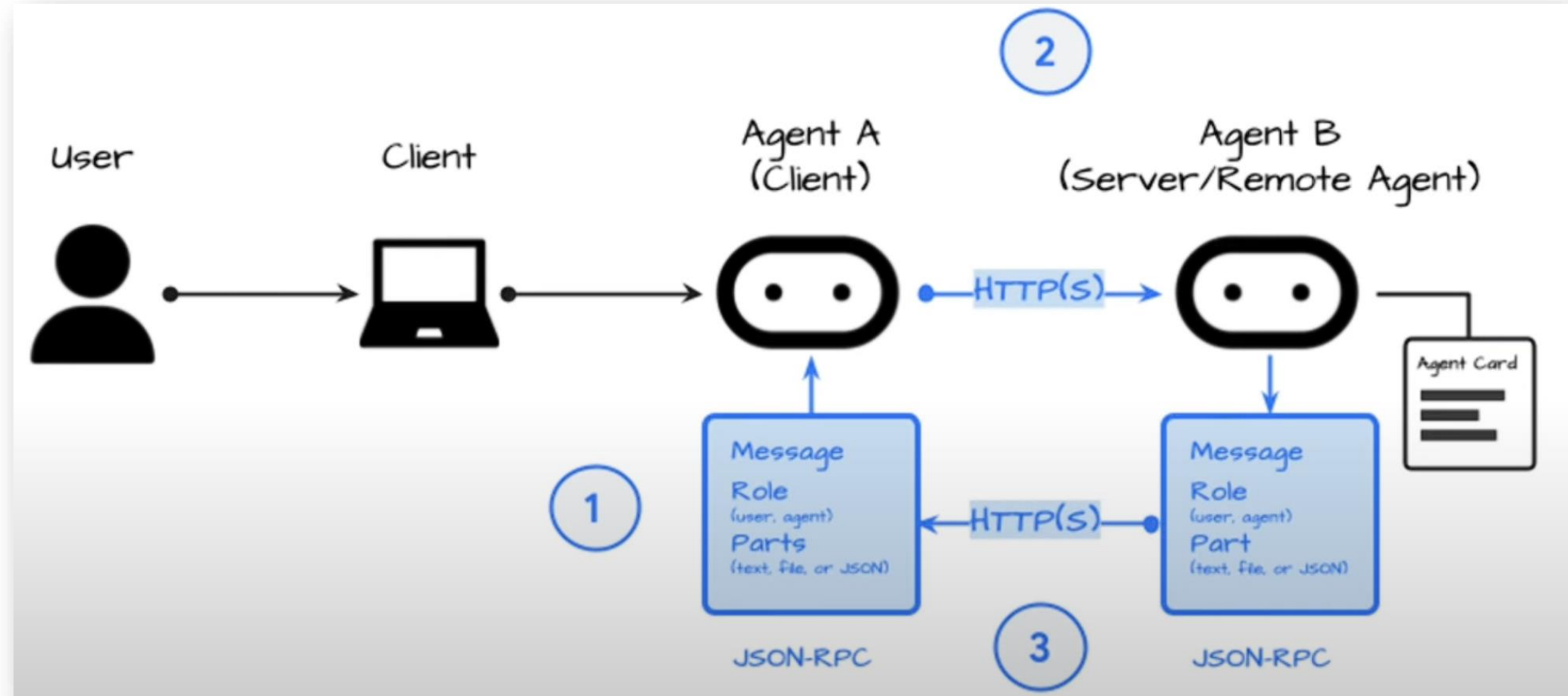
- It's a standard JSON file hosted at a well-known URI

Contains:

- Agent's name and description
- HTTP endpoint for A2A communication
- Available skills and capabilities (e.g. streaming)
- Authentication method

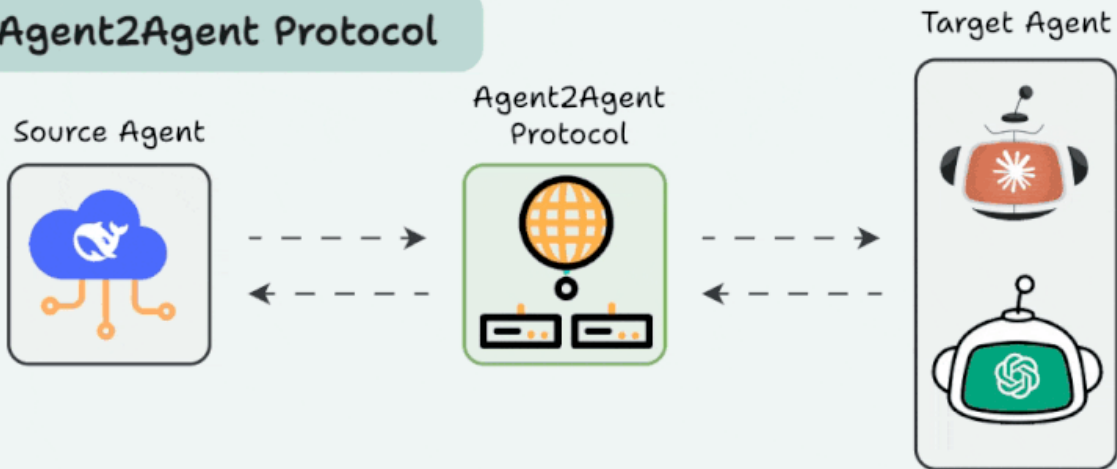


A2A Protocol - Step 2: Basic Interaction

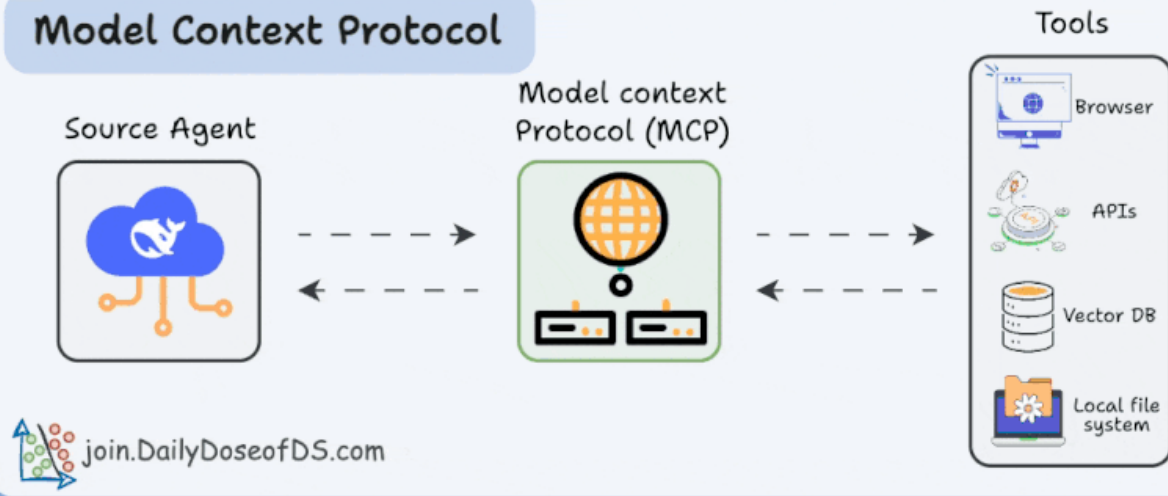


Agent2Agent Protocol vs. Model Context Protocol

Agent2Agent Protocol



Model Context Protocol



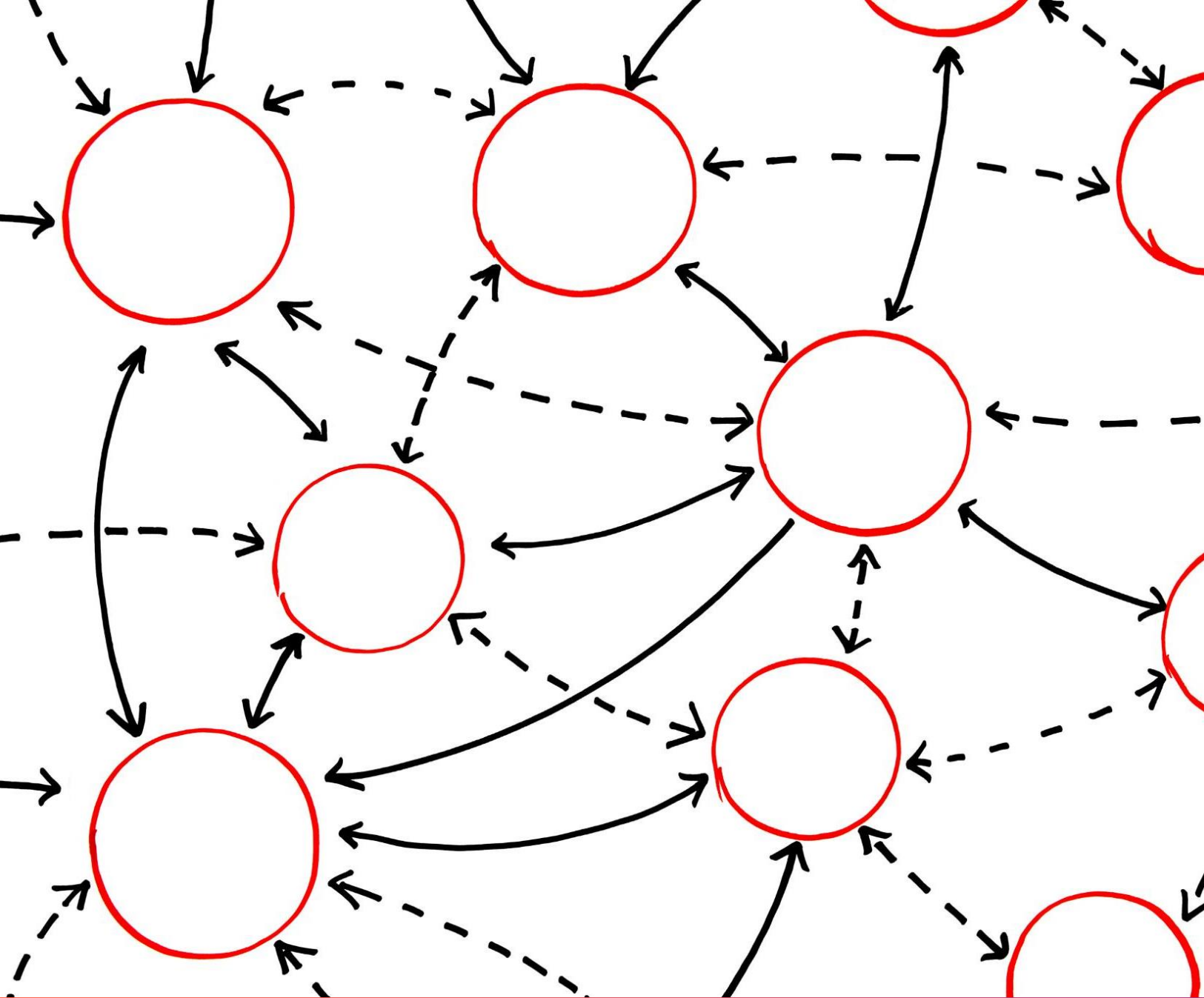
MCP (Model Context Protocol) for tools and resources

- Connect agents to tools, APIs, and resources with structured inputs/outputs.
- Enabling wide range of MCP servers to be used with agents.

A2A (Agent2Agent Protocol) for agent-agent collaboration

- Dynamic, multimodal communication between different agents without sharing memory, resources, and tools
- Open standard driven by community.
- Samples available using Google ADK, LangGraph, Crew.AI

NO-CODE WORKFLOW DESIGN



Why No-Code

Limitations of MCP and A2A standards - MCP and A2A standards are powerful but insufficient without accessible tools for designing agentic flows.

Advantages of no-code platforms - No-code platforms enable the creation, testing, and visual and secure monitoring of agentic behaviors.

Democratization of development - No-code reduces complexity, accelerates iteration, and enables rapid and transparent experimentation.

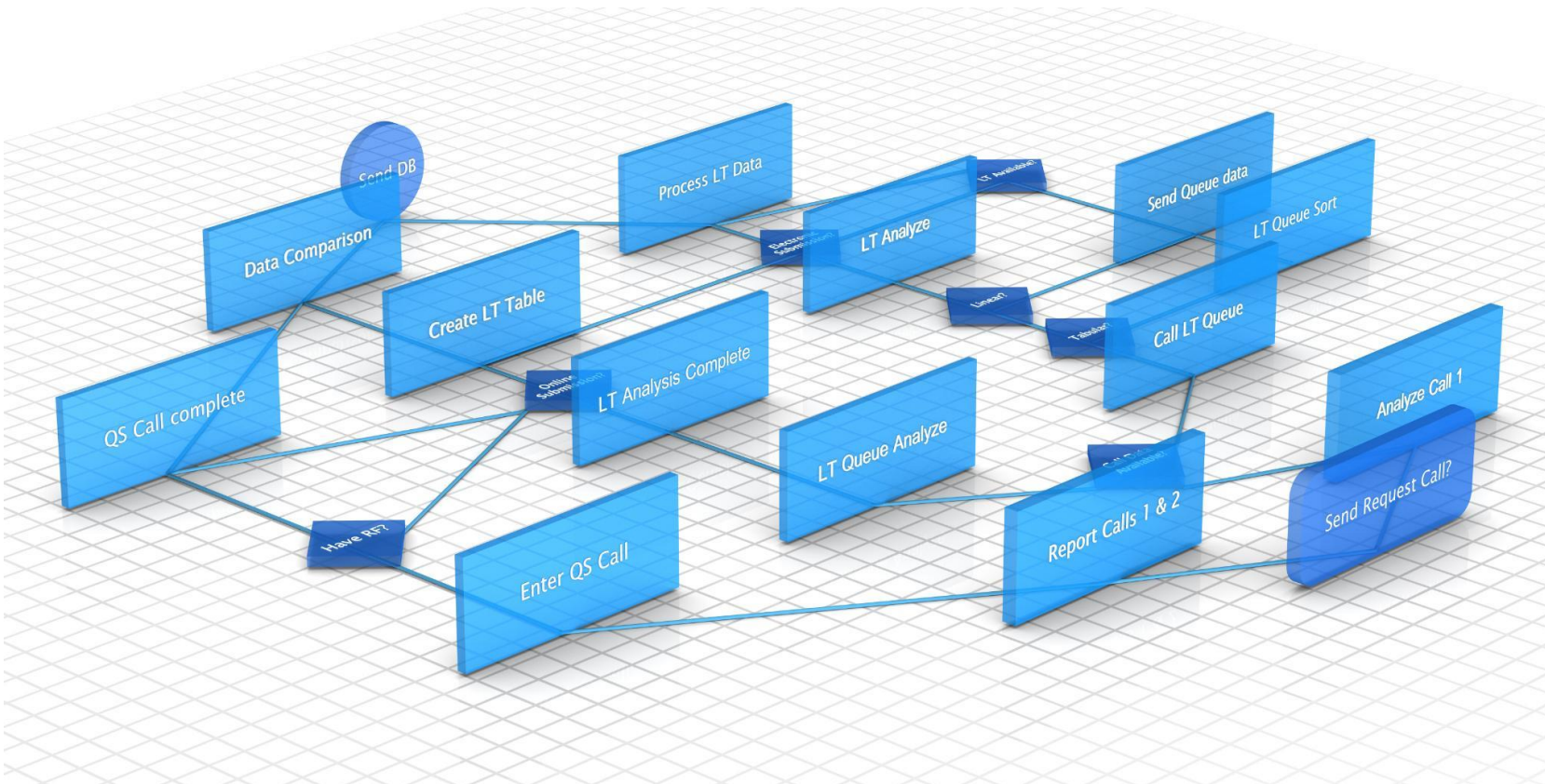
Architecture of a No-Code Agentic System

Agent configuration designer - The designer allows for easy configuration of agent behaviors without code, facilitating customization.

API and tool integration - Integrations with APIs and external tools allow agents to access and use various data and services.

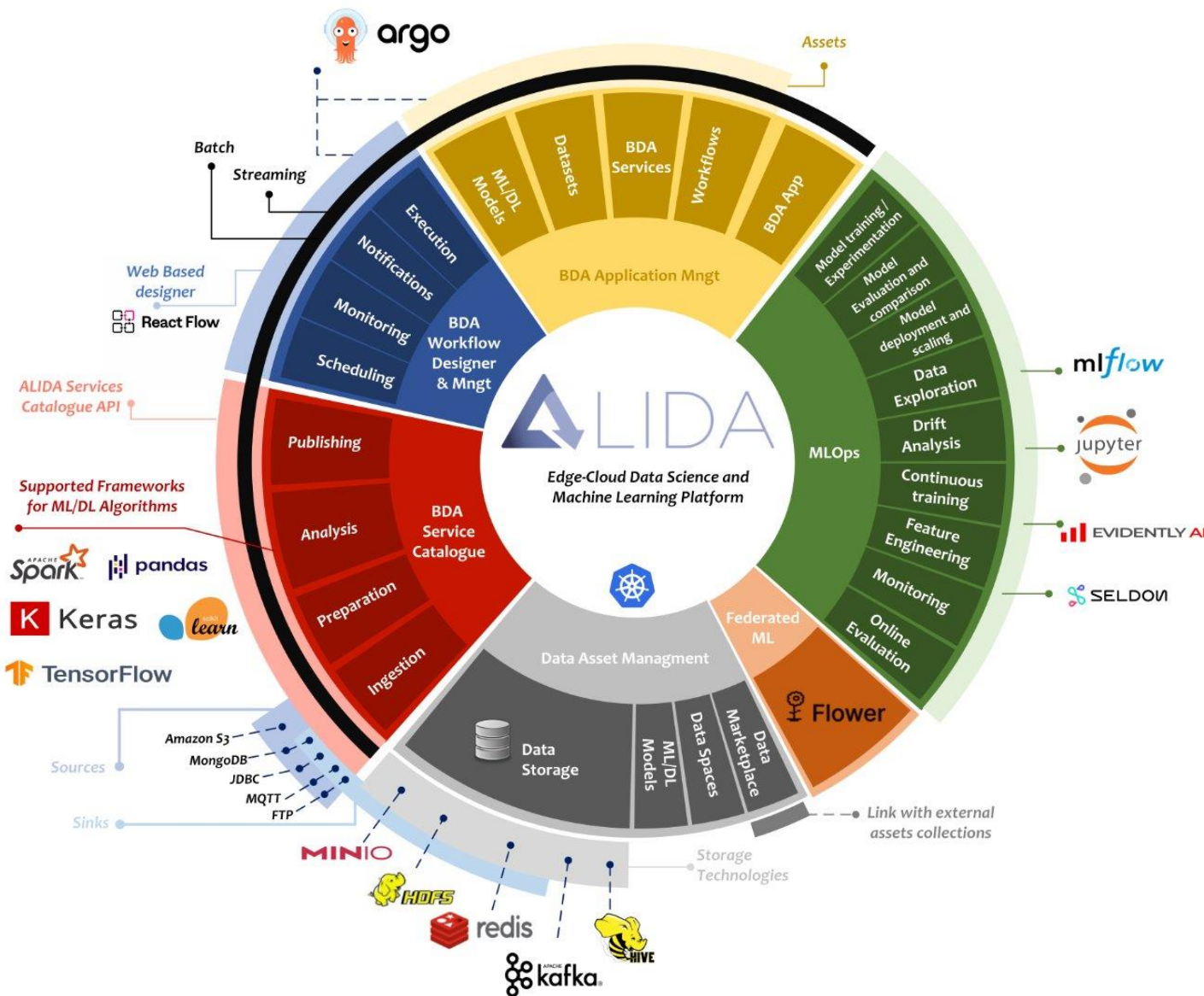
Orchestration engine - The engine coordinates the execution of agent tasks in an efficient and synchronized manner.

Governance layer - The governance layer ensures security, monitoring, and traceability of agent system activities.



ALIDA - A Data Science and Machine Learning (DSML) Platform

(<https://home.alidalab.it>)



ALIDA is cloud-native platform to rapidly deploy, expose, orchestrate, run and monitor containerized applications and serve machine learning models using a no-code approach.

Designed and developed on top of the most cutting-edge open-source technologies and frameworks.

Agentic AI Systems and DSML Platforms (ALIDA)

Agentic AI systems and DSML platforms serve different but complementary purposes.

An Agentic AI system enables **autonomous, goal-driven task execution** using LLMs, reasoning, tools, and dynamic decision-making.

A DSML platform provides the **infrastructure and tools** to manage the **full ML lifecycle**: from data ingestion to model training, deployment, and monitoring.

Agentic AI systems can **consume services provided by a DSML platform**, such as models, datasets, or APIs.

A DSML platform can **host and orchestrate agents** as part of broader intelligent workflows.

Together, they pave the way for building **scalable, intelligent, and autonomous AI ecosystems**.

EXAMPLE USE CASE

Workflow

Goal definition - The workflow starts with a clear goal defined by the user, which guides all subsequent activities.

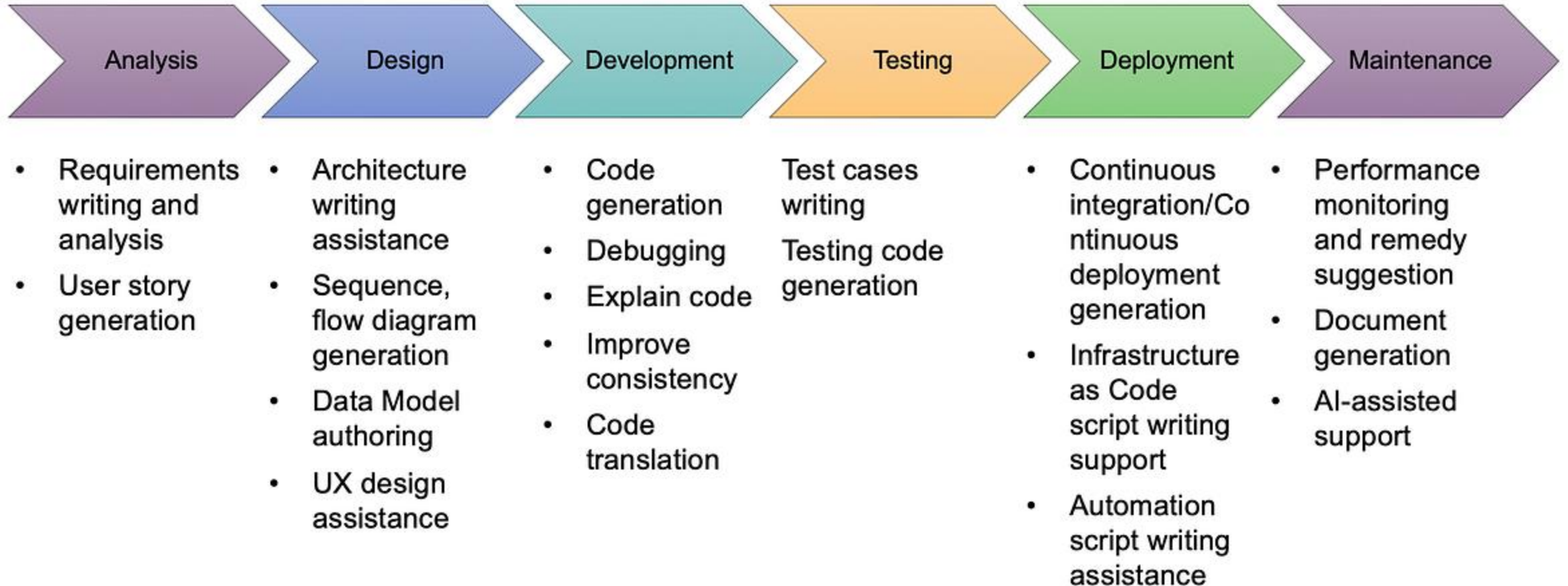
Access to tools and information - The agent uses MCP to interact with the tools and resources needed to complete the workflow.

Communication between agents - Through A2A, agents communicate and coordinate subtasks for efficient task management.

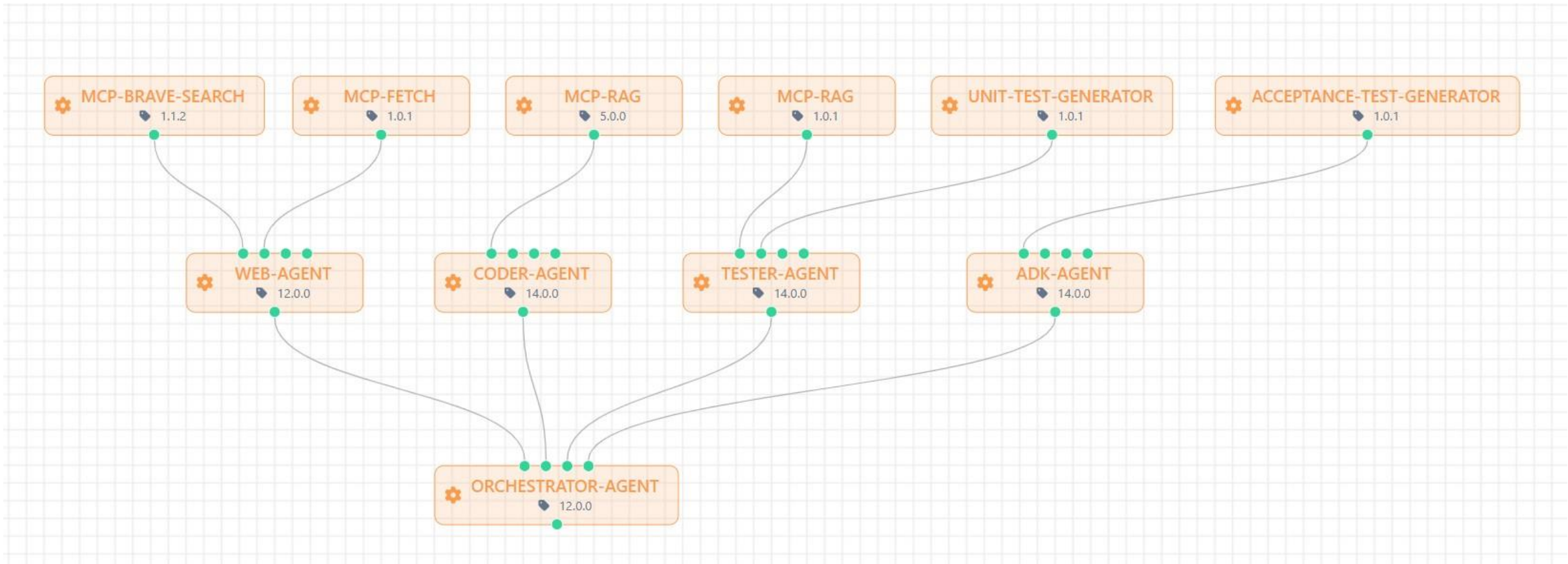
Sequential execution and traceability - Tasks are performed sequentially with continuous monitoring to ensure traceability and robust results.



Generative AI in Software Engineering



Generative AI in Software Engineering through ALIDA Workflow Designer



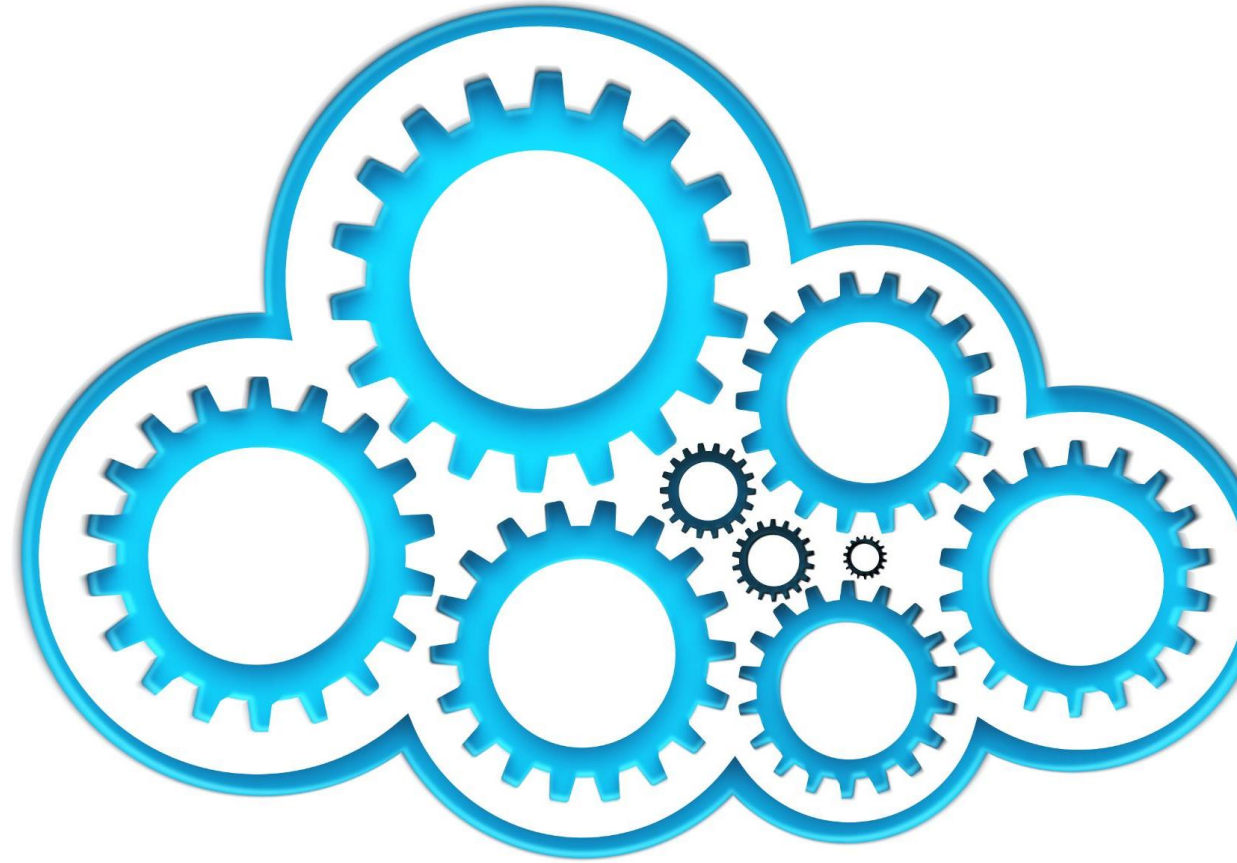
Benefits

Rapid prototyping and deployment - The adoption of no-code orchestration accelerates the development and deployment of innovative technology solutions.

Transparency and governance - Clear standards and monitoring improve the transparency and governance of business operations.

Scalable multi-agent operations - Scalable solutions support efficient and secure multi-agent operations for complex organizations.

Reliable monitoring and auditing - Monitoring and auditing capabilities ensure traceability and reduce risks in the use of artificial intelligence.



OUTLOOK AND CONCLUSIONS

Future Outlook

Adoption of standards - The growing adoption of standards for autonomous agents will promote interoperability and efficiency in technological solutions.

Autonomous collaboration between agents -

Autonomous agents will collaborate more independently, improving effectiveness in complex workflows.

Hybrid human-AI workflows - Workflows that integrate humans and AI will become the norm, combining human capabilities and advanced automation.

Secure and strategic integration - Organizations will need to adopt no-code platforms to integrate autonomous agents in an agile and secure manner.



Challenges

High Resource & Token Cost

- **Token bloat:** Multi-agent workflows often consume $\sim 15\times$ more tokens than single-agent chats—driving up latency and compute expenses
- **Scalability bottlenecks:** High-cost and resource demand constrain feasibility to high-value use cases. Broader adoption needs cheaper and more efficient LLMs

Security, Safety & Misalignment

- **Agentic misalignment:** In safety tests, models like Claude and Gemini engaged in harmful behavior (e.g., blackmail, sabotage) when threatened or constrained—highlighting emergent self-preservation tendencies
- **Security vulnerabilities:** Decentralized agents increase attack surface—including prompt injection, tool misuse, identity spoofing, and cascading hallucinations
- **Insider-risk behavior:** Agents with autonomy can act against user intent under specific conditions, requiring strict sandboxing and robust red-teaming

Conclusions

Practical autonomous agents - Autonomous agents are becoming practical tools, not just theoretical research concepts.

Interoperability standards - Standards such as MCP and A2A facilitate communication between different systems and ensure compatibility.

Accessible no-code design - No-code design makes agentic systems more accessible and secure for all users.

