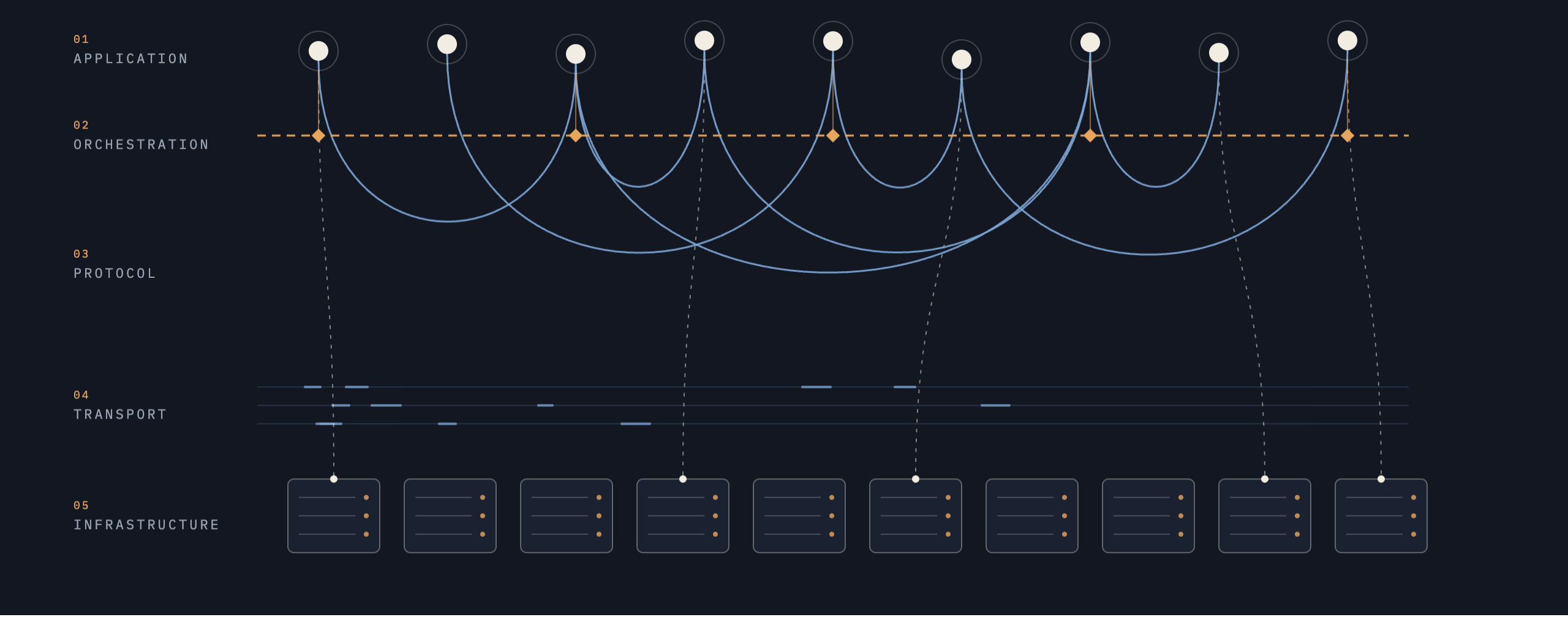


Mai Khoi Tieu

AI Agent Protocols

arXiv:2504.16736v3 [cs.AI] 21 Jun 2025

A Survey of AI Agent Protocols

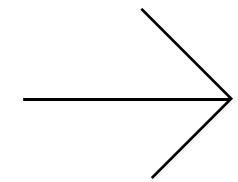
Yingxuan Yang, Huacan Chai, Yuanyi Song, Siyuan Qi,
Muning Wen, Ning Li, Junwei Liao, Haoyi Hu, Jianghao Lin[*],
Gaowei Chang[†], Weiwen Liu, Ying Wen, Yong Yu, Weinan Zhang

Shanghai Jiao Tong University, [†]ANP Community
{zoeyyx, chiangel, wnzhang}@sjtu.edu.cn

Abstract

The rapid development of large language models (LLMs) has led to the widespread deployment of LLM agents across diverse industries, including customer service, content generation, data analysis, and even healthcare. However, as more LLM agents are deployed, a major issue has emerged: there is no standard way for these agents to communicate with external tools or data sources. This lack of standardized protocols makes it difficult for agents to work together or scale effectively, and it limits their ability to tackle complex, real-world tasks. A unified communication protocol for LLM agents could change this. It would allow agents and tools to interact more smoothly, encourage collaboration, and triggering the formation of collective intelligence. In this paper, we provide the first comprehensive analysis of existing agent protocols, proposing a systematic two-dimensional classification that differentiates context-oriented versus inter-agent protocols and general-purpose versus domain-specific protocols. Additionally, we conduct a comparative performance analysis of these protocols across key dimensions such as security, scalability, and latency. Finally, we explore the future landscape of agent protocols by identifying critical research directions and characteristics necessary for next-generation protocols. These characteristics include adaptability, privacy preservation, and group-based interaction, as well as trends toward layered architectures and collective intelligence infrastructures. We expect this work to serve as a practical reference for both researchers and engineers seeking to design, evaluate, or integrate robust communication infrastructures for intelligent agents. Ongoing updates and the latest developments will be maintained at https://github.com/zoe-yyx/Awesome-AIAgent-Protocol.

Key Words: AI Agent Protocol, AI Agent, Agent Protocol Evaluation, LLMs



Introduction

Application layer

Assistants, agent platforms, creation tools, entertainment, e-commerce

Protocol layer

MCP · A2A · ANP · ACP · LMOS · Agora

Infrastructure layer

HTTP/REST APIs · GPU and cloud · data centres

Problem. Agents from different vendors meet tools and each other with no shared rules, which limits interoperability and scale.

Analogy. The early Internet was just as fragmented until TCP/IP and HTTP connected it.

Promise. A standard could create a network of agents whose collective intelligence exceeds any single one.

What an LLM agent is

LLM agents combine an LLM's language abilities with autonomous decision-making: they reason over several steps, plan ahead, remember past interactions and use external tools to adapt their responses.

Foundation model	Memory	Planning	Tool use	Action execution
Large language model	Short-term		External Tools	API calls
Multimodal large model	Long-term		APIs	Database queries
				Other external system

What a protocol is

Standardized rules, formats and procedures for structured communication among agents, and between agents and external systems.

Manner	Scenario	Efficiency	Range	Standardized	AI-native
API	Server-to-server integration	✓✓	✗	✗	✗
GUI	Computer and mobile use	✗	✓	✓	✗
XML	Browser use	✗	✓	✗	✗
Protocol	Agent interaction	✓✓	✓✓	✓✓	✓✓

Current agent protocol landscape

	General-purpose	Domain-specific
Context-oriented	**MCP** agent <-> tools, data, services	**agents.json** websites declare agent interfaces
Inter-agent	**A2A · ANP** AITP, AConP, AComP	**PXP · LOKA · LMOS** human-, robot- and system-agent

Understanding agent protocols

1

Who or what is communicating?

The participants: an AI application, an agent, a tool provider, a remote agent

2

What can they exchange?

Capabilities, messages, data structures or units of work

3

How does it proceed?

Discovery, request, progress updates, results and errors

Protocol	Main relationship	Core concern	Unit of interaction
MCP	AI app ↔ capability provider	Tools, resources, prompts	Tool call, resource read, prompt
A2A	Agent ↔ agent	Delegation and collaboration	Message or Task, ending in Artifacts
ANP	Agent ↔ open network	Identity, discovery, communication	Identity, description, interaction

A tool is not automatically an agent; a remote agent is not just a tool endpoint; and discovering an agent does not make it trustworthy.

MCP

Host
The AI application: holds the model, context and policy

→

MCP Client
Inside the host; one client per server

→

MCP Server
Exposes tools, resources and prompts

→

Systems
Files, databases, APIs and services

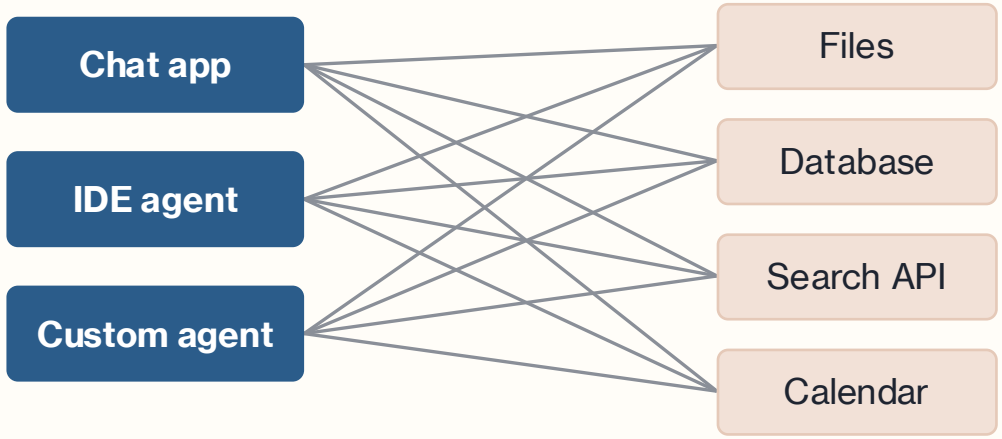
1 · INITIAL → DISCOVER
server/discover, then tools/list: versions, capabilities, tool schemas.

2 · REQUEST → CALL
The model picks a tool; the client sends tools/call.

3 · RESPONSE → RESULT
Content returns, or input_required and a retry.

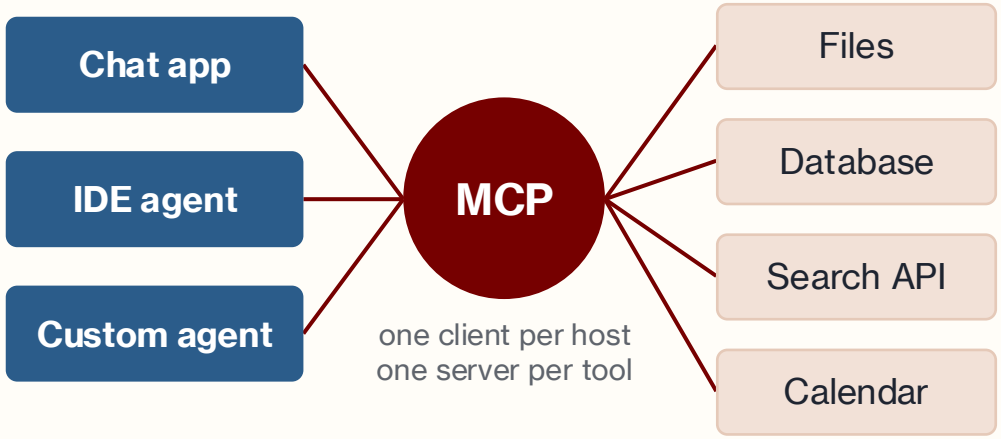
MCP

Without a shared protocol



3 hosts × 4 tools = **12 custom integrations**

With MCP



3 hosts + 4 tools = **7 implementations**

At 10 hosts and 50 tools: 500 one-off integrations without a protocol, 60 with MCP. Each host builds one MCP client; each tool ships one MCP server.

MCP

Wire: JSON-RPC 2.0 messages · transports: stdio (local server) and Streamable HTTP (remote server)

Stateless core

No sessions or Mcp-Session-Id; each request carries version and capabilities in _meta

Multi round-trip requests

A server returns input_required; the client retries with the answers

Explicit handles

State that must survive between calls is a value the model passes on

Header-based routing

Mcp-Method and Mcp-Name headers let gateways route without reading the body

Cacheable lists

List results carry a time-to-live and a cache scope

Extensions

Long-running tasks move to an extension; OpenTelemetry trace context in metadata

Authorization, built from OAuth standards

2024-11
No authorization framework

2025-03
OAuth 2.1 for remote servers

2025-06
Tokens issued for one server

2025-11
URL-based client identities

2026-07
Issuer validation

A2A: delegating work

What it is designed for

Google's Agent2Agent protocol (April 2025) lets independent agents discover each other, exchange messages and delegate work — a research agent handing statistics to a statistics agent, a support agent asking an order agent to check a transaction.

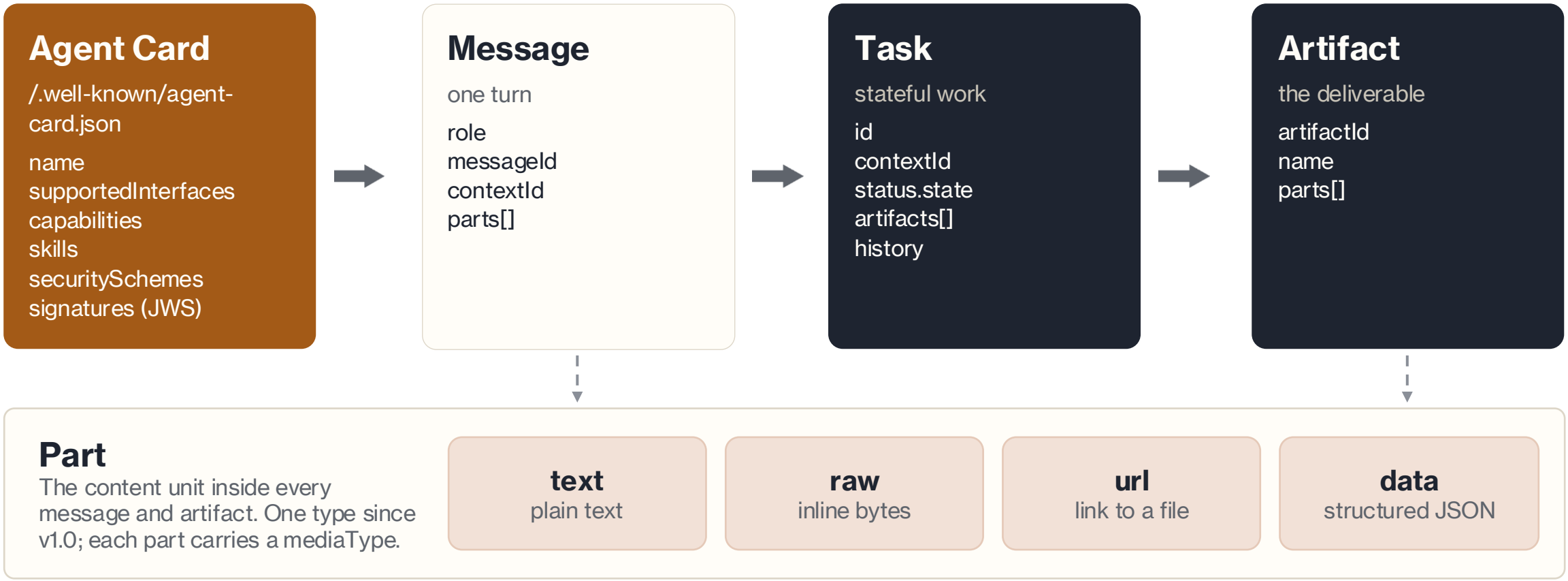
Opaque execution

The caller never sees the remote agent's model, prompts, memory, tools or reasoning — only its declared capabilities, how to send work, and how results come back.

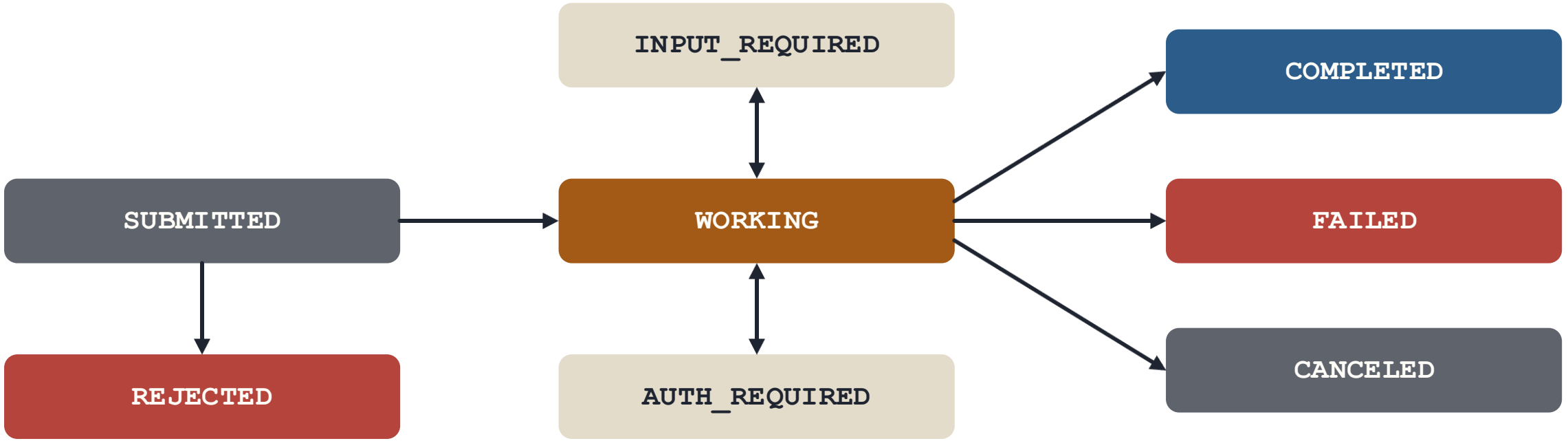
How agents collaborate

1. Read the Agent Card, from a well-known URL or a registry
2. Check skills, interfaces and security against the job
3. Send a message: SendMessage or SendStreamingMessage
4. The agent replies, or creates a task (it may reject)
5. Follow updates, answer INPUT_REQUIRED, validate the artifacts

A2A: core components



A2A: task lifecycle



INPUT_REQUIRED — the remote agent needs more from the caller

AUTH_REQUIRED — an extra authorization step comes first

REJECTED — the agent declines the task

ANP: open agent networks

What it is designed for

An open network where agents can be identified, described, discovered and contacted across organizations — without a central authority everyone must trust.

Document	Topic	Status
ANP-02, 03	DID authentication; did:wba	Released
ANP-07, 08	Description; discovery	Released
ANP-09	Messaging profiles	Group E2EE candidate
ANP-06	Meta-protocol	Draft

Released 1.2 architecture, with the draft layer dashed

Application protocols

Agent Description (ANP-07), Discovery (ANP-08), application protocols

Meta-protocol (ANP-06) — draft

Optional negotiation of how to talk; outside the released 1.2 architecture

Identity and encrypted communication

did:wba identities, DID authentication, end-to-end encryption

ANP: interaction steps

1

Find

Via a .well-known path or a search agent

Application layer

2

Describe

Read its description: skills, interfaces and security needs

Application layer

3

Verify

Resolve the DID; check signatures with its keys

Identity layer

4

Interact

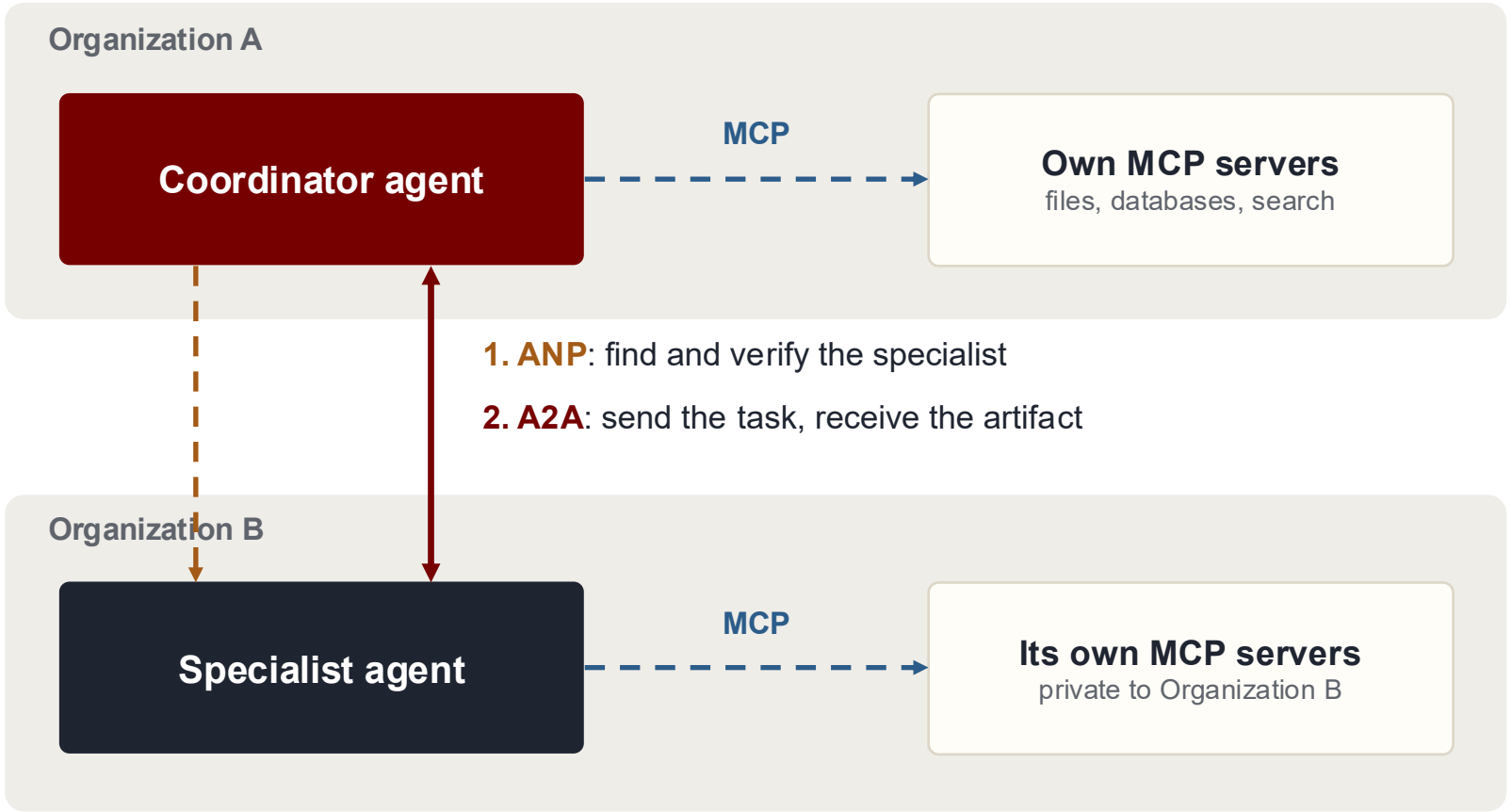
Exchange messages through the described interface

Application layer

Optional: where both agents implement the draft meta-protocol, they first negotiate a mutually supported interface.

A DID proves who controls a key — not whether an agent is trustworthy, competent or allowed to see a dataset.

How the protocols fit together



Most systems use part of it

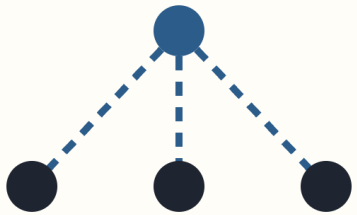
MCP alone
one agent and its tools

A2A between known agents
endpoints already configured

ANP, then A2A
discovery, then delegation

MCP inside every agent
whatever links the agents

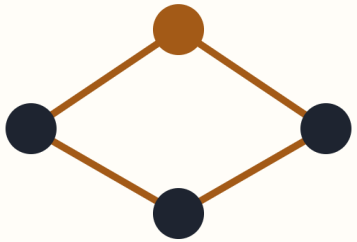
Use case



MCP

Star

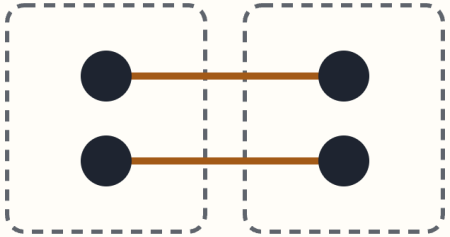
One travel client calls flight, hotel and weather servers. Simple; the centre is a bottleneck.



A2A

Departments

Specialists call each other directly; the planner mainly collects results.



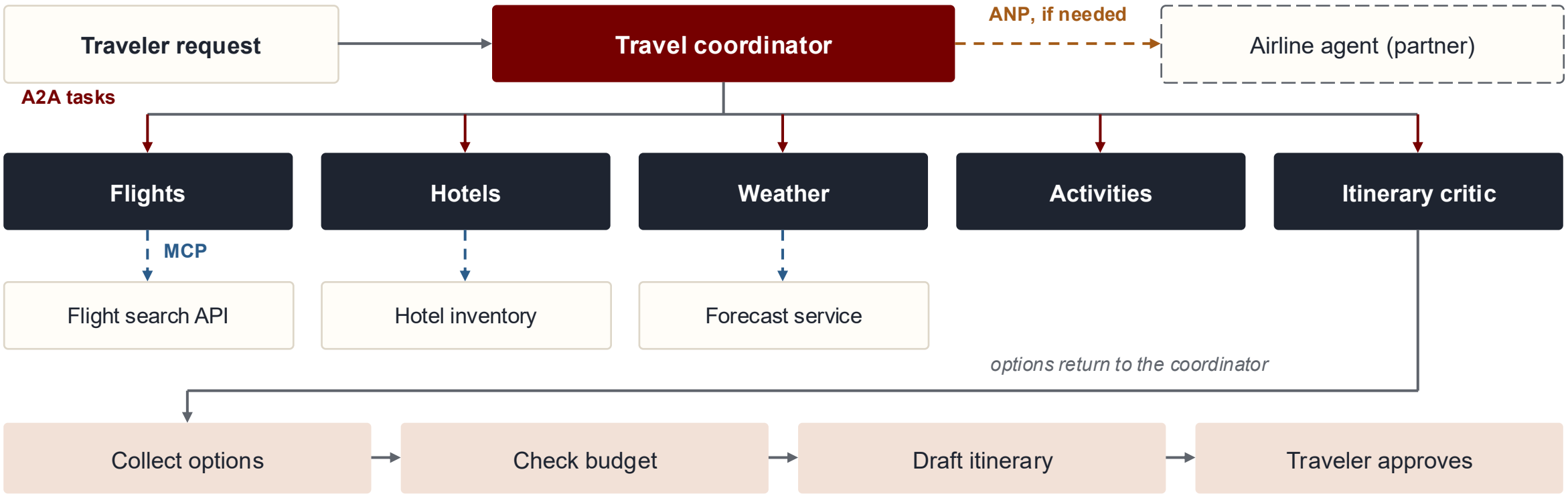
ANP

Cross-domain

Agents in separate organizations exchange formal requests across boundaries.

Task: plan a five-day trip from Beijing to New York (Yang et al., 2025).

Use case


Traveler request
Travel coordinator
ANP, if needed
Airline agent (partner)
A2A tasks
Flights
Hotels
Weather
Activities
Itinerary critic
MCP
Flight search API
Hotel inventory
Forecast service
options return to the coordinator
Collect options
Check budget
Draft itinerary
Traveler approves

Academic Outlook

SHORT TERM

From static to evolvable

Shared benchmarks and testbeds

Privacy-preserving protocols: role- or task-based access; share aggregates, not raw data

Agent mesh: group-chat style coordination

Protocols agents can learn and adapt

MID TERM

From rules to ecosystems

Protocol knowledge trained into models

Layered protocol architectures, with ethical and legal rules in the upper layers

LONG TERM

From protocols to infrastructure

Collective intelligence and scaling laws for agent populations

An Agent Data Network for machine-to-machine data

Conclusion

MCP

Standardizes how an agent reaches tools and data

A2A

Standardizes how independent agents delegate and track work

ANP

Standardizes how agents across organizations identify and find each other

Protocols make agents interoperable; making the whole system dependable is the job of the engineering around them.

Reference

1. Yang, Y. et al. (2025). A Survey of AI Agent Protocols. arXiv:2504.16736 (v3, June 2025).
2. Model Context Protocol maintainers (28 July 2026). The 2026-07-28 Specification. blog.modelcontextprotocol.io
3. A2A Project. Agent2Agent Protocol: What's new in v1.0, and specification. a2a-protocol.org.
4. Agent Network Protocol project. ANP 1.2 release and specification repository. github.com/agent-network-protocol/AgentNetworkProtocol.
5. Lopopolo, R. (11 February 2026). Harness engineering: leveraging Codex in an agent-first world. OpenAI.