Scaling AI with model context and agent to agent protocols

February 2026

Introduction

Generative AI is evolving from relying on a single model to leveraging coordinated teams of specialised AI agents that can review information, retrieve data, and act. Though this could yield enhanced results, without guiding principles on how agents should communicate with one another, share context, remember previous interactions, and hand over tasks, the complexity of the process goes up while the performance of the agent system drops.

Two new standards could help define these guiding principles:

- Model context protocol (MCP) provides a common format that allows models to use tools, and access data and memory across different systems
- Agent to agent (A2A) protocols allow the agents to discover each other, communicate messages, derive roles, and perform actions

Together, MCP and A2A create a unified operating layer for multiagent systems. MCP standardises how models access capabilities, while A2A defines how agents communicate and collaborate. When combined, they allow AI systems built on different platforms to work together smoothly instead of operating in isolation. This makes it easier for people to see how decisions are made, rely on consistent outcomes, and stay in control of how these systems behave. This article discusses how scalable, cooperative AI models can be developed by layering MCP and A2A and ensure that the agents can interact with each other.

Understanding the building blocks

MCP is an open standard which describes how AI models, agents, and developer tools can connect with external data, tools, and memory systems in a uniformly interoperable manner.

Some of the key competencies of MCP are:

01 **Context exchange:** MCP uses a standardised, structured format to define how an AI client and server exchange contextual information, such as available tools, data sources, and prompts. This shared structure ensures that both sides have a consistent understanding of the context, allowing them to coordinate reliably and reduce ambiguity during interactions. To support this exchange, MCP leverages the JSON-RPC 2.0 protocol over transport layers such as web communication sockets, enabling smooth operation across different runtimes and environments, even when multiple interactions occur concurrently.

02 **Tool invocation and discovery:** MCP servers expose tools along with descriptive metadata, including input and output formats defined through schemas and human-readable descriptions written in non-technical language. Using standard request-and-response patterns, clients can query the server to discover which tools are available and understand the functions and parameters they support. The decision to invoke these tools, whether automatically by the model or only with explicit user approval, is governed by client or application policies, ensuring flexibility while maintaining user control.

03 **Resource and memory integration:** MCP allows AI clients to access resources including but not limited to files, documents, database records, items stored in vector databases, or anything that would provide continuous context and knowledge to AI systems. MCP does not dictate how the user should implement memory in a certain way, rather it enforces a common interface across different systems, so they can describe, access, and share resources expressively that pertain to memory from clients to servers.

A2A protocol

A2A is a protocol in which AI entities communicate and cooperate with one another. A2A defines communication, task assignment, intention, capability, status, and result such that entities may locate each other, coordinate tasks, assign tasks, and combine their effects. A2A communication is conducted on a common communication channel where several specialised entities can cooperate. The strategy allows the systems to be scalable, efficient, and reliable and is one of the reasons why it has an advantage over models which require one single agent to perform every task.

Some of the key principles of an A2A protocol are:

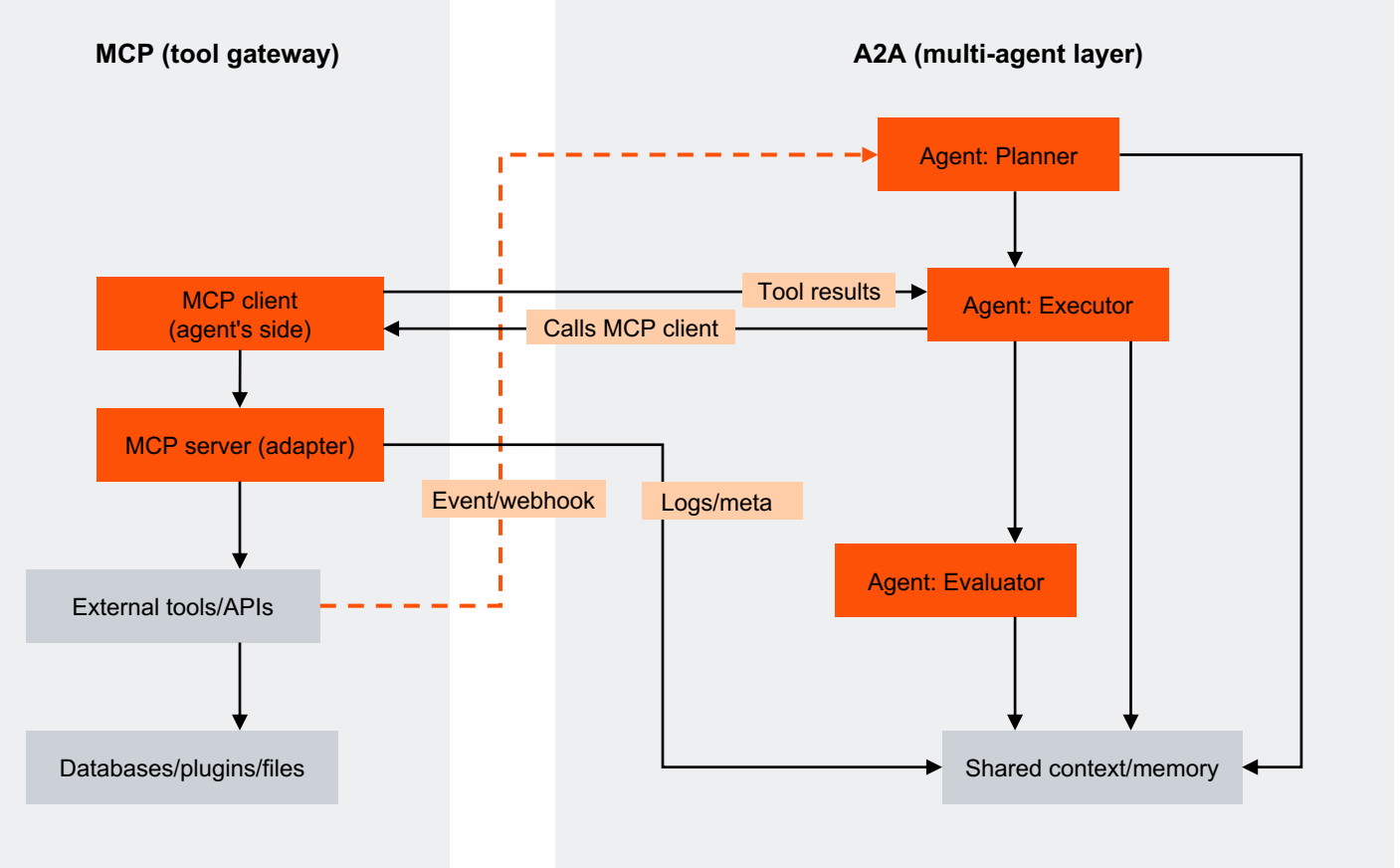
01 **Interoperability:** Although it is not necessary that agents should be developed using the same language or framework, to ensure that agents communicate easily, they should adhere to message format and protocol.

02 **Capability and intent routing:** Each agent defines the capabilities it possesses to perform certain operations. It is the routing layer that identifies the agent to process the received request based upon the capabilities the agents possess.

03 **Shared memory and context:** A common memory or knowledge area can be available to all the agents in the vector database to ensure that the information is consistent across systems.

04 **Asynchronous collaboration:** Agents don't have to wait for each other and can work on the task concurrently, pass messages to each other, and build the result step by step.

Architectural overview: Layers of a scalable AI ecosystem



Source: PwC analysis

It usually takes five different layers to build robust AI systems that can bridge key gaps such as inconsistent context sharing, limited interoperability, and weak coordination between models with the help of MCP and A2A.

01 **Agent layer:** Several AI agents are involved in different operations of data extraction, reasoning, and verification of the output.

02 **Routing layer:** This layer determines which agent is handling what and ensures that the tasks are routed to the right place while also tracking the progress of the conversation.

03 **Gateway layer:** The universal connector provided by MCP is the MCP gateway layer, where the MCP methodology provides the agents with a standardised way of connecting to all the available tools along with API's without having the agents write their own code to connect with the tools.

04 **Shared memory layer:** This is the shared space in which agents can store and remember information and don't have to start from the beginning each time.

05 **Governance layer:** This is also the control centre, the orchestration and the governance layer which monitors events, ensures that regulations are followed, handles security, and alerts the user about any anomalies or errors in the process.

Scaling AI by implementing MCP and A2A

By implementing MCP and A2A in collaboration, users can distribute work across specialised agents. Some of the advantages of implementing both in unison are:

01 **Task distribution:** With this communication, the user can utilise one agent for focusing on a complex task while others handle the rest of them. For instance, one agent retrieves information, another summarises it, and a third checks the overall progress.

02 **Shared memory and persistent context:** Since MCP handles storage, the agents can share information creating copies and sending those copies back and forth.

03 **Parallelisation:** A2A allows the running of the agents simultaneously, instead of lining up. During peak traffic, the user can increase the number of spawned agents and scale back when the traffic is low.

04 **Resilience and redundancy:** If an agent goes down, the next one takes over the workload. This way, the system can continue to function despite the failure of individual components.

05 **Explainability and governance:** Since everything must go through MCP and A2A, the user can trace exactly what happened, who accessed which information, and which agent made which decisions which enable clearly defined audit trails.

Use cases

Some of the areas where MCP and A2A agents can be implemented collaboratively are:

Enterprise document intelligence

Task-optimised models are utilised by the distributed agents which perform document classification, extract entities, validate and summarise them, and perform redactions. This process is useful for large document ingestion operations with different schemas, all orchestrated through MCP connectors to enterprise content systems, cloud storage services, and data verse which are governed through centralised orchestration.

01

Customer support automation

Intent-driven routing sends category-based queries-billing, technical, product-to specialist responder agents, uses common conversation memory to maintain context across agents. It also allows parallel retrieval from multiple knowledge bases, CRM integration using MCP, and audit trails which further help the user in meeting compliance requirements without latencies.

02

Code generation and review systems

Agents autonomously perform code generation, static analysis, test creation, vulnerability scanning, and documentation. MCP tools integrate the integrated development environments (IDEs), repositories, and continuous integration/continuous deployment pipelines, thus enabling parallel validation through automated code and review workflows, and explainable code changes.

03

Advantages and challeges

Based on the above use cases, some of the advantages of using multi-agent architecture are:

- **Specialisation over generalisation**: Work can be divided into smaller pieces, with each piece being specifically designed for an agent/model to work at its optimal level. This can be done more effectively than relying on a general-purpose system.
- **Independent scaling:** All the components can be scaled as per their load requirements based on the highest load or demand level.
- **Fault isolation:** The failure of one agent or service does not affect the whole system, and other components can still be up and running, and the process gets more graceful.
- **Technology flexibility:** It can integrate different models, frameworks, or vendors as required, thereby allowing teams to choose the most suitable tool for each task.
- **Context reuse:** By sharing memory or state across agents, it eliminates redundant lookups and helps minimise token usage, making the process more efficient and cost effective.

At the same time some of the challenges of using a multi-agent architecture are:

- **Orchestration overhead:** Multiple agents' coordination increases latencies for every interaction due to message passing and synchronisation.
- **Higher complexity:** With more agents and components, the overall system becomes harder to debug, monitor, and maintain, as teams must track a larger set of interactions, message flows, and states across the environment.
- **Network dependencies:** Since it is based on several distributed services, it inherits the usual risks related to networks such as timeouts and connection failures.
- **State consistency:** Maintaining consistent shared context or memory across asynchronous agents requires careful design and is highly prone to errors.
- **Cold start delays:** Working with multiple specialised agents can cause latencies at the startup stage compared to running a single always-on model.

Governance, security, and observability

Responsible scaling of intelligence must ensure that autonomy doesn't lead to anarchy. MCP and A2A support this by embedding governance, security, and observability features directly into the architecture. These capabilities ensure that every agent's action is authenticated, monitored, traceable, and aligned to the defined policies with:

01 **Authentication and access control:** Each agent and source of data operate under valid credentials so that communication is secure.

02 **Auditable context flow:** All operations like tool invocation, message exchange and communications, or memory updates are logged for traceability.

03 **Policy enforcement:** Governance agents can dynamically monitor activity, detect violations, and apply compliance rules.

04 **Observability:** With trace correlations based on IDs, system operators can see the flow of reasoning across agents and layers, making it easier to debug and optimise performance.

This combination of visibility and control can ensure that scale never sacrifices safety, or that intelligent systems remain accountable and explainable.

Operational efficiency and cost optimisation

Beyond coordination, MCP and A2A provide financial benefits. By distributing work among specialised agents, organisations can match the right computer cost to the right task. Lightweight agents handle simple reasoning locally, while heavyweight reasoning models are invoked only when necessary. Catching and shared context reduces redundant API calls, while asynchronous workflows improve resource utilisation. Therefore, MCP and A2A could make intelligence scalable in both performance and economics, which could be a critical advantage in the production of AI systems.

Way forward

The convergence of MCP and A2A is setting the stage for the next level of AI architecture which will be distributed, interoperable, and self-regulating. The following factors are accelerating this shift:

01 **Federated agent networks:** Systems where independently owned agents collaborate securely across organisational boundaries.

02 **Semantic routing:** Assigning tasks to the right agent based on meaning and intent, rather than relying on fixed rules.

03 **Self-optimising workflows:** Agents that analyse their own performance metrics and optimise their workflow accordingly.

04 **Edge-integrated intelligence:** Extending MCP interfaces to edge devices for enhanced privacy and low-latency collaboration.

As these trends continue to evolve and stabilise, AI could transition from a toolset into a comprehensive digital ecosystem that reflects and represents the same cooperative structure as human intelligence.

The future of scalable AI does not lie in developing bigger models but in developing and architecting systems that think with one another. MCP sets up the ground for shared understanding, while A2A protocols bring structure to communication and collaboration. Together, these form the architecture of intelligence—a scalable, secure, interoperable framework for the next era of AI. Just as the Internet connects information, MCP and A2A are connecting intelligence by turning isolated models into coordinated entities which can solve complex problems which individual models cannot resolve.

About PwC

We help you build trust so you can boldly reinvent

At PwC, we help clients build trust and reinvent so they can turn complexity into competitive advantage. We're a tech-forward, people-empowered network with more than 364,000 people in 136 countries and 137 territories. Across audit and assurance, tax and legal, deals and consulting, we help clients build, accelerate, and sustain momentum. Find out more at www.pwc.com.

PwC refers to the PwC network and/or one or more of its member firms, each of which is a separate legal entity. Please see www.pwc.com/structure for further details.

Contact us

Debankur Ghosh
Partner, Emerging Technologies
PwC India
debankur.ghosh.in@pwc.com

Abhishek Verma
Associate Director, Emerging Technologies
PwC India
abhishek.verma@pwc.com

Contributors
Alik Sen
Sachin Prasad
Srija Nag
Bhanu Prakash Reddy

Editor: Rubina Malhotra
Design: Prerna Rajpal

PR/February 2026 – M&C 51502