The City College of New York

CSC 59866-E: Senior Project I

AI Agents for Decision Making in the Real World

By Saptarashmi Bandyopadhyay
Email: sbandyopadhyay@ccny.cuny.edu, sbandyopadhyay@gc.cuny.edu
Assistant Professor of Computer Science
City College of New York and Graduate Center at the City University of New York

May 13, 2026 CSC 59866

Advanced Topics: Standardized Protocols for Real-World Agents

Today's Agenda

1. **Standardized Tool Protocols (MCP).**
2. **Standardized Peer Protocols (A2A).**
3. **Industrial Governance (Physical AI Agent Standards).**

Standards for Agent-to-Agent Communication

Standardized Protocols for Agent Communication

Circa **2024**, every single AI model needed to be trained with its own Python tool; one for Slack, one for Jira, etc.

Today, we don't need to have all of that. The name of the game is *interoperability*: We deal not with individual tools with their own APIs but universal *protocols* where AI Agents can autonomously discover new capabilities.

Model Context Protocol (MCP)

MCP is an open-source industry standard for **Agent-to-Tool communication** developed by Anthropic, the creators of Claude, Claude Code, etc.

The main use of MCP is to provide a universal connector for different tools so that AI Agents can easily use and understand them without prior context.

Here's the structure of a typical MCP setup:

- **MCP Client:** The Agent.
- **MCP Server:** A wrapper around *any* source of data (e.g. Gmail, Dropbox, Github)

How MCP Mechanics Work

Standardized Schemas: The MCP Server advertises its capabilities via standardized JSON schemas.

The Agent dynamically reads this schema and instantly knows how to use the tool, without the developer writing specific prompt instructions.

JSON-RPC 2.0: All communication (queries, tool executions, resource fetching) happens over JSON-RPC.

The Result: You can plug a completely new, proprietary corporate database into an LLM, and via MCP, the agent can immediately query it with zero-shot accuracy.

Agent-to-Agent Protocol (A2A)

While MCP connects agents to *tools*, the **A2A Protocol** connects agents to *other agents*.

Developed for horizontal, peer-to-peer multi-agent coordination (e.g., an Inventory Agent talking to a Logistics Agent).

The 3-Step Sequence:

1. **Discovery:** Agents read "Agent Cards" (manifests) to find the right peer for a sub-task.
2. **Authorization:** Agents verify cryptographic identity and grant scope.
3. **Communication:** Dispatching tasks and streaming state via Server-Sent Events (SSE).

A2A Task Bidding

- When a primary agent needs a sub-task completed, it doesn't just pick randomly. It broadcasts a Call for Proposals (CFP) to the A2A network.
- Peer agents calculate their estimated cost C_i and expected success probability P_i .
- They submit a bid based on Expected Utility:

$$U_i = R_{task} \cdot P_i - C_i$$

- The primary agent selects the peer that mathematically maximizes the global reward. This converts chaotic LLM chatting into rigorous, market-based distributed computing.

Industrial Control for AI Agents

Industrial Governance & Deployment

Moving from theoretical software to real-world deployment requires strict governance.

"The Industrial AI Agent Manifesto" (Digital Twin Consortium, 2026) outlines the "Ten Laws for Trustworthy Autonomous Operations."

If your agent controls a physical supply chain, a power grid, or financial assets, it is no longer just a chatbot—it is an industrial controller subject to regulatory auditing.

Law 1: Deterministic Validation and Execution

LLMs are inherently probabilistic. Industrial systems require deterministic outcomes.

The Law: An AI agent must never directly execute a critical action. Its output must be parsed and validated by a deterministic, rules-based safety envelope.

If the agent hallucinates a command to "Set furnace temperature to 10,000 degrees," the deterministic boundary catches the physical violation and rejects the payload.

Law 2: Physics-Aware and Process-Aware Intelligence

Industrial agents cannot operate purely on semantic language; they must respect the physical constraints of their environment.

The Manifest Requirement: Agents must be grounded in physics-based Digital Twins.

Before executing an A2A logistics route, the agent must query a digital twin simulator (via MCP) to verify that the physical truck actually has enough fuel and time to complete the route.

Law 3: Reproducible Context and Audit Trails

In traditional software, if a bug occurs, you check the stack trace. How do you debug an LLM hallucination from three weeks ago?

Reproducible Context: The system must freeze and log the exact memory vectors, the exact Zettelkasten notes, and the exact system prompt present at the moment of decision.

Auditability: Without reproducible context, when an industrial agent makes a costly error, you cannot prove *why* it made that decision to regulators.

Summary of Modern Agent Stacks

Memory: Agents use Zettelkasten-style frameworks (A-Mem) to dynamically link, evolve, and prune context, bypassing the quadratic compute cost of massive context windows.

Tools: The Model Context Protocol (MCP) standardizes how agents discover and execute external APIs.

Coordination: The A2A protocol allows agents to dynamically discover and negotiate with peers using market-bidding mathematics.

Governance: The Industrial Agent Manifesto ensures these probabilistic systems are bounded by deterministic safety constraints and rigorous audit trails.

Questions?