Task-oriented Coordination Requirements for AI Agent Protocols

Yong Cui, Chenguang Du

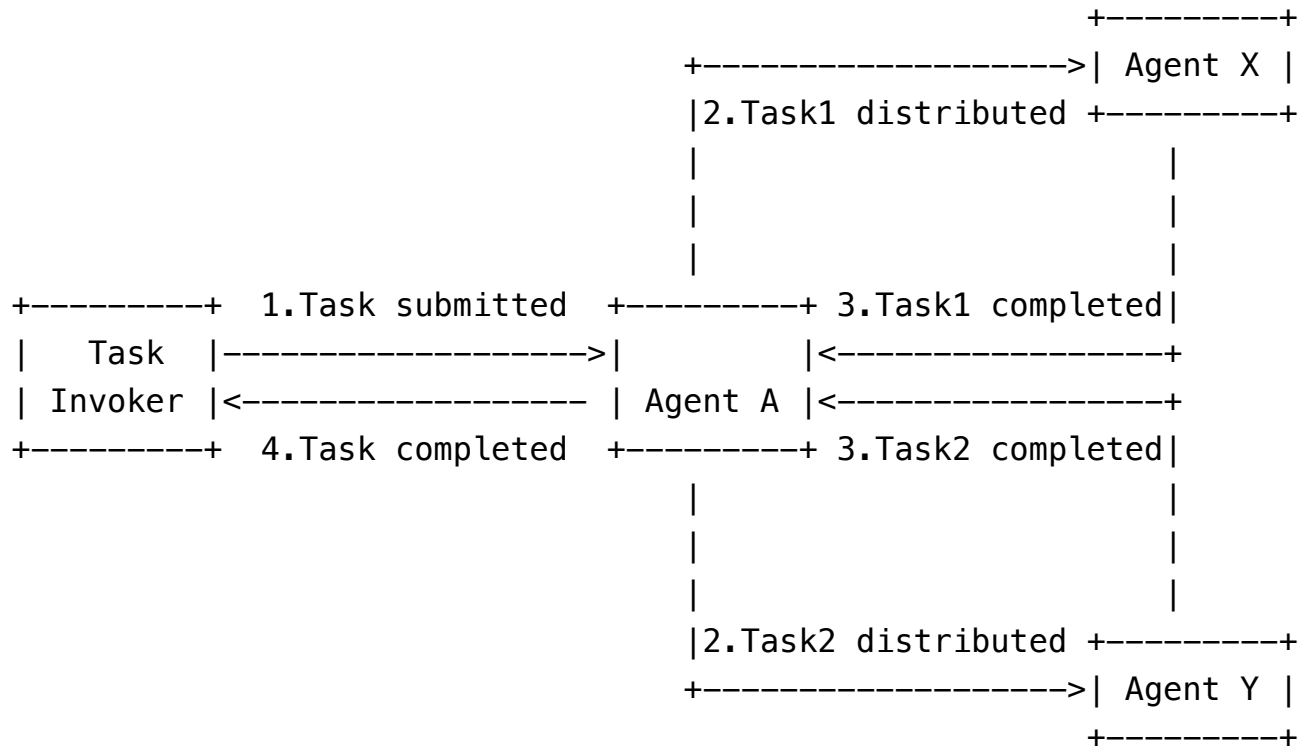
Tsinghua University, Beijing Zhongguancun Laboratory

July, 2025

Motivation

- Challenges
  - Agents need to work together to address complex problems
  - Existing protocols are designed for general use
  - Maybe different tasks need different ways to communicate.
- Goals: Provide requirement guidance for AI agent communication protocols
  - ✓ Classify agent task scenarios
  - ✓ Identify what protocol capabilities are required for task coordination

Task Coordination Workflow



1.Task submission: The invoker submits a task to coordinator A

2.Task decomposition and dispatch: Coordinator A decomposes and dispatches subtasks to executors X/Y

3.Result aggregation and return: Executors report back to A, which aggregates and returns the final result

Task Categories

Different types of tasks have different requirements for communication, performance, and reliability.

- High-throughput Tasks
  - ✓ Handle many requests at the same time ==> need a lot of bandwidth and processing power
  - ✓ Robots and drones work together
- Low-latency Tasks
  - ✓ Need to be done very quickly ==> strict time requirements
  - ✓ Sending home alarm messages right away when something happens
- High-reliability Tasks
  - ✓ Must be done correctly and safely ==> need to work even when something wrong
  - ✓ Important control commands in smart factories

Protocol Requirements

- Task Description
  - ✓ Precisely define: Goal, constraints, and success criteria
  - ✓ Minimize context: Dispatch only necessary information
- State Management
  - ✓ Support state transitions: submitted, running, completed/failed and canceled
  - ✓ State synchronization mechanism： track these changes effectively
- Communication Mechanisms
  - ✓ Multiple modes: Request/Response、Publish/Subscribe、Broadcast
  - ✓ QoS prioritization: Control messages are more urgent than big data transfers

Protocol Requirements

- Context Sharing & Privacy
  - ✓ Standardized format: JSON-LD, Protobuf, etc.
  - ✓ Access control: Role-based access control for least privilege
- Exception Handling
  - ✓ Automatic retry and degradation: Retry or degrade gracefully on network hiccups/timeouts
  - ✓ Error notification: Clear error codes and recovery guidance

Discussion

➢ Future Work

- Refine task types: Cover broader task scenarios
- Analyze requirement differences: Analyze requirements for different task types

➢ Additional Thinking

- Limited and Simple Use Cases： Need more real-world agent collaboration use cases ==> understand the requirements and challenges
- Maybe develop into a layered architecture:
  - Lower layers: agent discovery and connectivity
  - Upper layers : context sharing, privacy protection, state management

Thanks!
We welcome collaborators!

Q&A

Chenguang Du
ducg@zgclab.edu.cn