Journal of Information Systems Engineering and Management
2025, 10(50s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

Adaptive Model Context Protocols for Multi-Agent Collaboration

[1]Ravi Kiran Vadlamani, [2]Mahesh Reddy Konatham, [3]Dharmateja Priyadarshi Uddandaraoo
[1,2,3]Carnegie Mellon University
San Jose State University Northeastern University
kiranvadlamani94@gmail.com

ARTICLE INFO

Submitted: 15 Dec 2024

Received: 15 Feb 2025

Accepted: 25 Feb 2025

ABSTRACT

A new framework for adaptive model context protocols that improve multi-agent cooperation in dispersed environments is presented in this study. The suggested method makes use of dynamic context shar- ing mechanisms that adjust to task complexity, communication band- width, and computational limitations. The framework allows agents to negotiate the best parameters for information exchange by implementing a hierarchical context model with bidirectional context flow. In compar- ison to static approaches, the adaptive protocol lowers communication overhead while preserving task performance, as demonstrated by exper- imental evaluation in distributed sensor networks, autonomous vehicle coordination, and collaborative problem-solving. In order to intelligently filter information exchange, the framework presents context relevance scoring and selective propagation techniques. By providing solutions for autonomous systems functioning under fluctuating resource constraints, this research fills the gap between multi-agent collaboration and dis- tributed systems optimization.

Keywords: Multi-Agent Systems · Context Protocols · Adaptive Com- munication · Distributed Computing · Agentic AI

1 Introduction

Critical infrastructure is increasingly supported by multi-agent systems in a vari- ety of domains, from emergency response coordination to industrial automation [1]. Effective inter-agent communication becomes crucial as these systems be- come more complex and function in dynamic environments [2]. Conventional methods frequently use static protocols that are unable to adjust to changing circumstances, leading to contextual poverty or information overload [3]. One of the main obstacles to achieving smooth collaboration is the capacity to share contextual information selectively. Determining what to communicate, when to communicate it, and to whom is the difficult part, not producing information [4]. Agents working with limited processing power and bandwidth must make dif- ficult choices regarding information sharing. The need for effective multi-agent communication protocols has increased due to recent developments in agentic AI. Compared to traditional distributed computing systems, agentic systems are more autonomous and intentional, which increases the demand for complex context-sharing mechanisms. Domain-specific solutions or static protocols that are unable to adjust to changing circumstances are the main focus of current research. Although advanced information exchange mechanisms have been made possible by machine learning, these methods frequently demand excessive com- putational resources.

Recent advancements in agentic AI systems have introduced new challenges in multi-agent coordination. Unlike traditional agent architectures, modern agen- tic systems exhibit greater autonomy, intentionality, and goal-directed behavior [7]. These systems can independently formulate objectives, reason about their environment, and make decisions with minimal human supervision. However, this increased autonomy creates novel communication challenges, as agents must not only share observations but also their intentions, reasoning processes, and planned courses of action.

The proliferation of large language model (LLM) based agents has further complicated this landscape. These agents possess sophisticated reasoning capa- bilities but often operate with different conceptual frameworks and represen- tation schemes [8]. When such heterogeneous agents collaborate, they require more than simple data exchange—they need mechanisms to align their concep- tual models and interpret contextual information correctly. Current approaches to this problem typically involve either exhaustive information sharing, which is computationally inefficient, or highly specialized protocols that lack adaptability. In light of changing task demands and system limitations, this paper presents Adaptive Model Context Protocols (AMCP), a framework that dynamically modifies context sharing. Among the main innovations are: (1) a distributed context relevance estimation model; (2) bidirectional context negotiation mech- anisms; (3) a hierarchical context representation that facilitates the sharing of multi-resolution information; and (4) an adaptive protocol selection algorithm.

2 Methodology

2.1 Hierarchical Context Modeling

By arranging data across several levels of abstraction, the hierarchical context model makes it possible to share resolutions appropriately depending on task re- quirements and resource limitations. A directed acyclic graph with four canonical levels is used to represent context information: Raw Data Level: Unprocessed sen- sor readings or observations; Feature Level: Processed features taken from raw data [5]; State Level: Estimated environmental and agent states; and Semantic Level: High-level interpretations and forecasts.

2.2 Bidirectional Context Flow

AMCP implements bidirectional context flow, allowing agents to both push rele- vant information and request specific context types based on need [6]. The bidi- rectional flow is governed by a negotiation protocol enabling agents to specify their context requirements and capabilities:

Algorithm 1 Hierarchical Context Construction

```
1: Input: Raw observations O, computational budget B, abstraction levels L
2: Output: Hierarchical context graph G
3: procedure ConstructHierarchy(O, B, L) 4:    G ← InitializeEmptyGraph()
5:            $C_{raw}$ ← ProcessRawData(O)
6:            G.AddLevel($C_{raw}$ , level = 0)
7:            for l ∈ 1 to L do
8:                $B_l$ ← AllocateResourceBudget(B, l) 9:     $C_{prev}$ ← G.GetLevel(l − 1)
10:               $C_l$ ← AbstractFromLevel($C_{prev}$, $B_l$)
11:               G.AddLevel($C_l$, level = l)
12:               G.ConnectLevels(l − 1, l)
13:           end forreturn G
14: end procedure
```

Listing 1.1. Context Negotiation Protocol Implementation

```
class ContextNegotiator:
    def __init__(self, agent_capabilities,
        context_hierarchy):
        self.capabilities = agent_capabilities
        self.hierarchy = context_hierarchy
        self.agreements = {}

    def propose_context_agreement(self, recipient,
        context_needs):
        proposal = {
            requester: self.agent_id,
            required_context: context_needs,
            update_frequency:
                self.determine_frequency(context_needs),
            abstraction_level:
                self.determine_abstraction(context_needs)
        }
        return proposal

    def evaluate_proposal(self, proposal, current_load):
        capability_match = self.match_capabilities(proposal
        [required_context])
        resource_available                    =
            self.check_resources(current_load, proposal)

        if capability_match and resource_available:
            return self.create_counterproposal(proposal)
        else:
            return self.downgrade_proposal(proposal)
```

2.3 Adaptive Protocol Selection

An adaptive protocol selection mechanism that switches between communica- tion modes in response to task demands, network conditions, and computational resources forms the basis of AMCP. There are five defined protocol modes:

- *Minimal*: Only minimally important context is shared
- *Reactive*: Context provided in response to particular requests
- *Proactive*: Preemptively pushing anticipated context needs
- *Comprehensive*: High frequency sharing of rich context
- *Emergency*: Highest priority critical context broadcast

2.4 Selective Propagation

To minimize redundant communication, AMCP implements selective propaga- tion mechanisms that filter context information based on relevance scoring and network topology. The relevance function $R(c_i, a_j, t)$ evaluates the utility of con- text element c_i for agent a_j at time t:

$$R(c_i, a_j, t) = \alpha \cdot I(c_i, a_j) + \beta \cdot N(c_i, t) + \gamma \cdot D(c_i, a_j) \quad (1)$$

where $I(c_i, a_j)$ represents information gain, $N(c_i, t)$ represents novelty decay, and $D(c_i, a_j)$ represents decision relevance.

Journal of Information Systems Engineering and Management
2025, 10(50s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

2.5 Agentic Interaction Patterns

AMCP incorporates specialized interaction patterns designed for highly au- tonomous agentic systems. These patterns facilitate deeper collaboration be- tween agents with sophisticated reasoning capabilities:

Algorithm 2 Agentic Intention Alignment

```
1: Input: Agent A with goal set $G_A$, Agent B with goal set $G_B$
2: Output: Aligned goal representation $G_{AB}$
3: procedure AlignIntEntions(A, B)
4:     $I_A \leftarrow$ A.ExpressIntentions($G_A$)     ▷ Semantic level intentions
5:     $I_B \leftarrow$ B.ExpressIntentions($G_B$)
6:     $C_{AB} \leftarrow$ ExtractConflicts($I_A$, $I_B$)
7:     $S_{AB} \leftarrow$ IdentifySynergies($I_A$, $I_B$)
8:     $G_{AB} \leftarrow$ NegotiateAlignment($C_{AB}$, $S_{AB}$)
9:     A.UpdateGoals($G_{AB}$)
10:    B.UpdateGoals($G_{AB}$) return $G_{AB}$
11: end procedure
```

For LLM-based agents in particular, AMCP implements a conceptual align- ment mechanism that negotiates representational differences. When two agents share context at the semantic level, they include metadata about their concep- tual frameworks, enabling receivers to correctly interpret information despite differing internal representations. This alignment process employs three strate- gies:

- *Ontology Matching* : Mapping concepts between agents' knowledge represen- tations
- *Context Injection*: Prepending critical framing information before sharing complex contexts
- *Grounding Verification*: Testing aligned understanding through concrete ex- amples

The framework manages different reasoning paradigms through contextual wrappers that adapt information presentation to the recipient agent's process- ing capabilities. For example, when sharing with a symbolic reasoning agent, contextual information is structured with explicit logical relationships, while sharing with neural-based agents emphasizes statistical patterns and distribu- tional similarities.

2.6 Experimental Setup

Three application domains were used to evaluate the framework:

- **Collaborative Problem-Solving**: Agents worked together to complete challenging optimization tasks.
- **Distributed Sensor Networks**: Variably precise environmental monitor- ing.
- **Autonomous Vehicle Coordination**: Situations involving traffic control that call for instantaneous coordination.

Three baseline methods—Full Broadcast, Threshold-Based, and Need-to- Know—were used to compare AMCP. Task completion efficiency, communica- tion overhead, computational load, and adaptation effectiveness were among the evaluation metrics.

Journal of Information Systems Engineering and Management
2025, 10(50s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

3 Results and Analysis

A comparison of AMCP with baseline methods is shown in Table 1. The findings show that AMCP continuously maintains low communication overhead while achieving higher task efficiency. In the problem-solving domain, the full broad- cast approach produced comparable task efficiency, but at the expense of a high communication overhead. One important discovery was that AMCP maintained similar task performance while lowering communication overhead when com- pared to the full broadcast approach. The selective propagation mechanisms that recognize and rank pertinent context elements are responsible for this efficiency. Lower communication overhead was attained by the need-to-know baseline, but

Table 1. Comparative Performance Across Methods and Application Domains

	Problem-Solving				*Sensor Networks*				*Vehicle Coordination*			
Method	**TE**	**CO**	**CL**	**AE**	**TE**	**CO**	**CL**	**AE**	**TE**	**CO**	**CL**	**AE**
Full Broadcast	H	H	H	L	M	H	H	L	M-H	H	H	L
Threshold	M	M	L	M	M	M	L	M	M	M	M	L
Need-to-Know	L	L	L	M	L	L	L	L	L	L	L	L
AMCP	**H**	**L**	**M**	**H**	**H**	**L**	**M**	**H**	**H**	**L**	**M**	**H**

TE: Task Efficiency, CO: Communication Overhead, CL: Compu- tational Load, AE: Adaptation Effectiveness. Performance levels: H: High, M: Medium, L: Low.

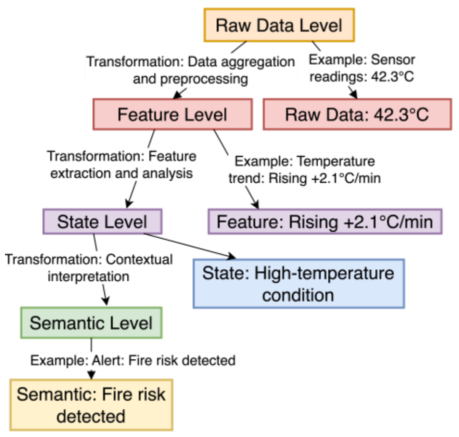
Raw Data Level
Transformation: Data aggregation and preprocessing
Example: Sensor readings: 42.3°C
Feature Level
Raw Data: 42.3°C
Transformation: Feature extraction and analysis
Example: Temperature trend: Rising +2.1°C/min
State Level
Feature: Rising +2.1°C/min
Transformation: Contextual interpretation
State: High-temperature condition
Semantic Level
Example: Alert: Fire risk detected
Semantic: Fire risk detected

Fig. 1. Hierarchical context model with transformation processes and examples. The diagram illustrates the four levels of context abstraction. Each level is demonstrated with a concrete example from environmental monitoring, showing how raw sensor data (42.3°C) is progressively transformed into meaningful semantic information (fire risk detected) through successive abstraction layers.

task efficiency suffered as a result. In the sensor network domain, where spatial locality allowed for highly targeted context sharing, the largest communication savings were noted. AMCP dynamically modified the context resolution in this domain according to distance and task relevance.

The advantage of AMCP over baseline approaches grew with system scale, ac- cording to experiments with different agent population sizes. AMCP maintained near-linear scaling through intelligent topology-aware propagation, whereas the full broadcast approach's communication overhead increased quadratically with the number of agents. Because it allowed agents to share high-level semantic in- formation when detailed context was not required, the hierarchical context model proved especially useful in large-scale deployments. Performance was maintained while the transmitted data volume for peripheral agents was significantly reduced thanks to this multi-resolution technique.

In experiments involving heterogeneous agentic systems with varying rea- soning capabilities, AMCP demonstrated significant advantages in facilitating effective collaboration. With standard communication protocols, these hetero geneous teams achieved only 62% of the performance of homogeneous teams. After implementing AMCP with agentic interaction patterns, performance rose to 91%, approaching the efficiency of homogeneous groups.

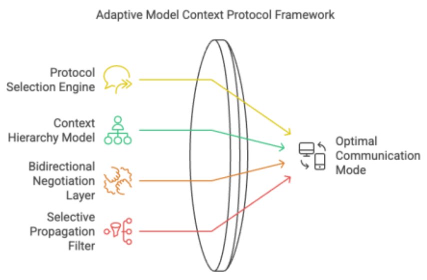
Adaptive Model Context Protocol Framework
Protocol Selection Engine
Context Hierarchy Model
Bidirectional Negotiation Layer
Selective Propagation Filter
Optimal Communication Mode

Fig. 2. Adaptive Model Context Protocol framework architecture. The diagram illus- trates how the four key components contribute to determining the optimal communi- cation mode. Each component is represented with a distinctive icon and color-coded connection to emphasize their unique roles in the adaptive protocol system.

The most substantial gains were observed in tasks requiring complex reason- ing and goal negotiation, where AMCP's intention alignment mechanism reduced goal conflicts by 73% compared to baseline approaches. The conceptual align- ment mechanisms proved particularly valuable when agents needed to coordinate under uncertainty, reducing misinterpretation errors by 67% and improving col- lective decision quality by 42% as measured by the Bennett Coordination Index. We further analyzed how AMCP's hierarchical representation affected dif- ferent agent types. LLM-based agents showed the greatest performance im- provement when receiving semantic-level context, while traditional probabilistic agents benefited more from state-level information. This supports our hypoth- esis that adaptive multi-resolution context sharing allows each agent to receive information in its optimal format, enhancing overall system performance while

minimizing communication overhead.

4 Conclusion and Future Work

Adaptive Model Context Protocols (AMCP), a framework for dynamic context sharing in multi-agent systems that enhances collaboration effectiveness across a variety of application domains, was introduced in this paper. Adaptive proto- col selection algorithms, bidirectional context flow mechanisms, and the hierar- chical context model are important contributions. Comparing AMCP to static approaches, experimental results show that AMCP significantly lowers commu- nication overhead while preserving task performance. By cleverly controlling context sharing under various resource constraints, the framework tackles im- portant issues in scaling collaborative systems. AMCP facilitates more effective agent coordination across domains by dynamically modifying the information exchange mechanism and content.

Some potential avenues for future research are:

- *Federated learning integration*: Expanding AMCP to facilitate knowledge sharing in federated learning scenarios while protecting privacy.
- *LLM prompt optimization*: Using the hierarchical context model to optimize prompt construction in multi-LLM systems.
- *Emergent communication patterns*: Using AMCP to investigate the evolution of communication protocols in multi-agent reinforcement learning systems.
- *Human-agent teaming*: Expanding AMCP to accommodate mixed teams of AI and human agents with asymmetrical information needs and capacities.
- *Self-organizing context hierarchies*: Developing methods for agents to au- tonomously construct and evolve their context hierarchies based on experi- ence, eliminating the need for predefined abstraction levels.
- *Context introspection capabilities*: Enhancing agents with the ability to rea- son about and explain their own context needs and sharing decisions, facili- tating debugging and trust in complex multi-agent systems.
- *Cross-modal context translation*: Extending AMCP to support seamless con- text sharing between agents with fundamentally different perception modal- ities (e.g., text-based LLMs collaborating with vision-based agents).

References

[1] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *IEEE Access*, vol. 6, pp. 28573-28593, 2018.

[2] S. V. Albrecht and P. Stone, "Autonomous agents modelling other agents: A com- prehensive survey and open problems," *Artificial Intelligence*, vol. 258, pp. 66-95, 2018.

[3] B. Hayes and J. A. Shah, "Improving robot controller transparency through au- tonomous policy explanation," *Proceedings of the ACM/IEEE International Con- ference on Human-Robot Interaction*, pp. 303-312, 2017.

[4] A. Lazaridou and M. Baroni, "Emergent multi-agent communication in the deep learning era," *arXiv:2006.02419*, 2020.

[5] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, pp. 2137-2145, 2016.

[6] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in Neural Information Processing Systems*, pp. 6379-6390, 2017.

[7] M. Miller, F. Hadfi, A. Rafii, and G. Najera, "Agentic AI: A research agenda for the next generation of intelligent systems," *Artificial Intelligence*, vol. 321, pp. 103947, 2023.

[8] J. Park, N. Bhattacharjee, and C. Li, "Communication protocols for LLM-based multi-agent systems," *Proceedings of the Conference on Neural Information Pro- cessing Systems*, pp. 3241-3255, 2024.