Discovery and Enactment of Declarative Interaction Protocols in Dynamically Open Settings

Extended Abstract

Andrei Ciortea
University of St. Gallen
St. Gallen, Switzerland
Inria, Université Côte d'Azur, CNRS, I3S
Sophia Antipolis, France
andrei.ciortea@unisg.ch

Kai Schultz
University of St. Gallen
St. Gallen, Switzerland
kai.schultz@student.unisg.ch

Amit K. Chopra
Lancaster University
Lancaster, United Kingdom
amit.chopra@lancaster.ac.uk

Ganesh Ramanathan
Siemens AG
Zug, Switzerland
ganesh.ramanathan@siemens.com

ABSTRACT

We address the problem of enabling agents to interact flexibly in settings where protocols, participants, and the deployment context must be discovered at run time. Our approach synthesizes two complementary ideas: First, we leverage information protocols for flexible multi-agent interaction. Second, we adopt hypermedia-driven interaction to decouple agents from their deployment context. Agents use hypermedia to discover protocol specifications and participants in-context. Once a suitable protocol is found, agents use metaprotocols for dynamic role binding, and hypermedia controls then guide the protocol enactment. We demonstrate our approach in a scenario where agents start from a single entry URI to join an open marketplace, discover available items, protocols, buyers, and sellers, and successfully enact a protocol for buying and selling items. This shows that integrating information protocols with hypermedia-driven interaction enables highly flexible multi-agent coordination in dynamically open settings, such as the Web.

KEYWORDS

Information Protocols, Hypermedia Multi-Agent Systems, Engineering Multi-Agent Systems, Software Architecture

ACM Reference Format:
Andrei Ciortea, Kai Schultz, Amit K. Chopra, and Ganesh Ramanathan. 2026. Discovery and Enactment of Declarative Interaction Protocols in Dynamically Open Settings: Extended Abstract. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 3 pages. https://doi.org/10.65109/UBWR1103

1 INTRODUCTION

Recent initiatives like the Agent-to-Agent (A2A) protocol [13] and the Model Context Protocol (MCP) [1] have renewed interest in agents that can interact flexibly in open settings. Our focus is on *dynamic openness* [11], where agents enter or leave systems freely, and the deployment context itself may change at run time [14].

Dynamic openness poses challenges for multi-agent interaction. Traditional approaches to specifying protocols based on message orderings can over-constrain the interaction [3]: the temporary unavailability of a single agent can unnecessarily block progress. Instead, *information protocols* in BSPL [12] specify information dependencies among messages, enabling flexible interaction and recovery mechanisms [4]. Still, information protocols are enacted in well-formed systems once the interaction participants are known. In a dynamically open system, agents require mechanisms to discover other agents, their contacts, and available protocols. A successful example of such a mechanism is *hypermedia-driven interaction*, more formally known as *Hypermedia As The Engine Of Application State (HATEOAS)* [9]. HATEOAS underpins the design of the Web: User agents use hypermedia to discover resources and their affordances at run time, allowing for continuous change. This principle has already been applied to design dynamically open *Hypermedia Multi-Agent Systems (MAS)* [7].

We propose a framework for *situated protocol enactment* that synthesizes two complementary ideas: (1) information protocols for flexible multi-agent interaction, and (2) hypermedia-driven interaction to decouple agents from their deployment context.

2 DISCOVERY AND ENACTMENT

We structure our approach into three phases: First, agents use hypermedia workspaces to discover candidate protocols and participants in the context of their activities. Second, once an initiating agent selects a protocol and candidate participants, the agents enact a metaprotocol to negotiate role bindings. Third, information dependencies and hypermedia controls guide the decentralized enactment of the protocol.

2.1 In-Context Discovery

To illustrate in-context discovery, consider online marketplaces. Most platforms follow similar buying and selling protocols, yet each differs: eBay uses auctions, whereas Amazon Marketplace uses

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. https://doi.org/10.65109/UBWR1103

fixed prices. Users explore each platform's website to determine which protocols apply and to identify relevant sellers or buyers. We propose a similar design for *situated protocol enactment* in MAS. Building on the metamodel for Hypermedia MAS in [8], we organize systems into *hypermedia workspaces*—bounded environments that encapsulate a domain of activity (e.g., a marketplace) to provide a coherent hypermedia context for discovery and interaction. Just as well-designed websites guide and support human users, well-designed hypermedia workspaces guide and support agents—and we assume workspace designers ensure that advertised protocols are safe, live, and available just-in-time for agents.

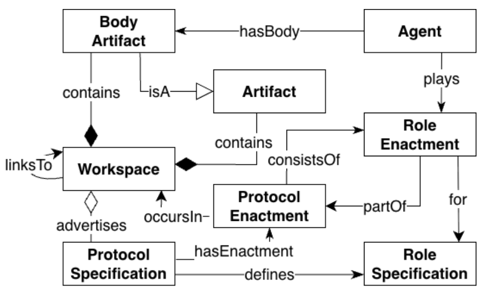

Figure 1: Metamodel for situated protocol enactment.

Figure 1 illustrates our metamodel. Agents can join or leave workspaces freely and are reified through *body artifacts* [2] that others can perceive and interact with. When an agent joins, the workspace representation is updated with a hyperlink to the agent's body artifact, which provides metadata such as supported protocols, roles, capabilities, and hypermedia controls for interacting with the agent—a practice from the early FIPA standards [10] to A2A Agent Cards [13]. Workspaces also expose hyperlinks to declarative protocol specifications in BSPL. A joining agent can then explore the workspace to discover protocols, candidate participants, and other resources such as product catalogs.

2.2 Dynamic Role Binding

Once agents have discovered applicable protocols and candidate participants, the next challenge is coordinating role assignments. A protocol enactment begins when an initiating agent decides to instantiate a protocol. To assign participants to roles, the initiator uses *metaprotocols* [5]—protocols that specify communication about another protocol's elements, such as role bindings.

```
RoleNegotiation {
   roles  Initiator , Candidate
   parameters out protocolName key, out systemName key, out sender, out
        proposedRole, out accept , out reject , out enactmentSpecs

    Initiator  -> Candidate: OfferRole [out protocolName key, out systemName
        key, out sender, out proposedRole]
   Candidate -> Initiator : Accept[in protocolName key, in systemName key,
        in proposedRole, out accept ]
   Candidate -> Initiator : Reject [in protocolName key, in systemName key,
        in proposedRole, out reject ]
    Initiator  -> Candidate: SystemDetails [in protocolName key, in
        systemName key, in accept , out enactmentSpecs]
}
```

Listing 1: A simple metaprotocol for role negotiation.

Listing 1 illustrates a simple metaprotocol for role negotiation: the initiator proposes a role to a candidate, who can accept or reject. The initiator instantiates this metaprotocol until all roles are filled. We assume agents implement such metaprotocols, which is reasonable given that they are domain-independent and reusable across workspaces. However, note that it is not possible to circumvent the need for domain knowledge: agents must still understand protocol semantics and assess role suitability, whether that knowledge is represented explicitly or encoded in weights of foundation models.

2.3 Dynamic Enactment

After the roles are assigned, agents enact the selected protocol. In dynamically open settings, message exchanges must remain loosely coupled: agents cannot rely on predefined contacts or rigid message sequences. Instead, contacts are discovered via hypermedia controls exposed by body artifacts in the shared workspace (see Section 2.1). The protocol enactment is then driven by the information dependencies among messages. Together, information protocols and hypermedia-driven interaction yield *situated protocol enactment*: agents decide what to do next based on the local information they hold and the hypermedia controls they perceive, while global progress follows from the protocol's information dependencies.

```
Buy {
  roles Buyer, Seller
  parameters out buyID key, out itemID key, out item, out money
  Buyer -> Seller : Pay[ out buyID key, out itemID key, out money]
   Seller  -> Buyer: Give[ in buyID key, in itemID key, in money, out item ]
}
```

Listing 2: A simplified protocol for purchasing items.

3 EXPERIENCE AND CONCLUSIONS

We implemented a `bazaar` workspace where agents can buy or sell rugs.[1] The workspace advertises the `Buy` protocol shown in Listing 2. Seller agents join the workspace to discover potential buyers and their contacts. Buyer agents explore the workspace to discover product offers, identify potential sellers, and retrieve the `Buy` protocol specification. To enact the protocol, a buyer looks up participants who advertise that they can play the `Seller` role in their descriptions and uses the metaprotocol in Listing 1 to offer the role binding. Once a seller accepts, the transaction proceeds.

This scenario operationalizes the key elements of our approach: agents use hypermedia exploration for in-context discovery of protocols and participants within a shared workspace; dynamic role binding is realized through a metaprotocol; and agents interact via hypermedia controls discovered at run time. Agents require only minimal prior knowledge: the core Hypermedia MAS ontologies [8] and domain-specific logic for the marketplace scenario. A remaining challenge is enabling agents to reason about their suitability for roles in protocol specifications discovered at run time. We leave it as future work to explore the use of Large Language Models (LLMs) to further reduce reliance on engineered knowledge.

[1] The prototype uses the BSPL Python library for information protocols (https://gitlab.com/masr/bspl), the Yggdrasil framework for Hypermedia MAS (https://github.com/interactions-HSG/yggdrasil), and agents written in Python using the Kiko programming model [6]. Prototype available at: https://github.com/Interactions-HSG/HypermediaProtocols

REFERENCES

[1] Anthropic. 2024. Model Context Protocol (MCP). https://modelcontextprotocol.io/. Accessed: 28.04.2025.

[2] Matteo Castellucci, Samuele Burattini, Andrei Ciortea, Jérémy Lemée, Danai Vachtsevanou, Alessandro Ricci, and Simon Mayer. 2025. Towards Agents' Embodiment in Hypermedia Multi-agent Systems. In *Multi-Agent Systems*, Rem Collier, Alessandro Ricci, Vivek Nallur, Samuele Burattini, and Andrea Omicini (Eds.). Springer Nature Switzerland, Cham, 361–381.

[3] Amit K Chopra, Munindar P Singh, et al. 2020. An evaluation of communication protocol languages for engineering multiagent systems. *Journal of Artificial Intelligence Research* 69 (2020), 1351–1393. https://doi.org/10.1613/jair.1.12212

[4] Samuel H. Christie, Amit K. Chopra, and Munindar P. Singh. 2022. Mandrake: multiagent systems as a basis for programming fault-tolerant decentralized applications. *Autonomous Agents and Multi-Agent Systems* 36, 1 (08 Feb 2022), 16. https://doi.org/10.1007/s10458-021-09540-8

[5] Samuel H. Christie, Amit K. Chopra, and Munindar P. Singh. 2022. Pippi: Practical Protocol Instantiation. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems* (Virtual Event, New Zealand) *(AAMAS '22)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 281–289.

[6] Samuel H. Christie, Munindar P. Singh, and Amit K. Chopra. 2023. Kiko: Programming Agents to Enact Interaction Models. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems* (London, United Kingdom) *(AAMAS '23)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1154–1163.

[7] Andrei Ciortea, Olivier Boissier, and Alessandro Ricci. 2019. Engineering World-Wide Multi-Agent Systems with Hypermedia. In *Engineering Multi-Agent Systems*, Danny Weyns, Viviana Mascardi, and Alessandro Ricci (Eds.). Springer International Publishing, Cham, 285–301.

[8] Andrei Ciortea, Matteo Castellucci, Kai Schultz, Jérémy Lemée, Danai Vachtsevanou, Fabien Gandon, Simon Mayer, Valentin Berger, Samuele Burattini, Alessandro Ricci, and Olivier Boissier. 2025. Yggdrasil: An Artifact-based Framework for Hypermedia Multi-Agent Systems. In *Agents and Multi-Agent Systems Development – Platforms, Toolkits, Technologies*, Rem Collier, Viviana Mascardi, and Alessandro Ricci (Eds.).

[9] Roy Thomas Fielding. 2000. *Architectural styles and the design of network-based software architectures*. Ph.D. Dissertation. University of California, Irvine.

[10] Foundation for Intelligent Physical Agents. 2002. FIPA Agent Management Specification. http://fipa.org/specs/fipa00023/SC00023K.html. Document number: SC00023K.

[11] Onn M Shehory. 1998. *Architectural properties of multi-agent systems*.

[12] Munindar P. Singh. 2011. Information-driven Interaction-oriented Programming: BSPL, the Blindingly Simple Protocol Language. In *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 2* (Taipei, Taiwan) *(AAMAS '11)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 491–498. http://dl.acm.org/citation.cfm?id=2031678.2031687

[13] Rao Surapaneni, Miku Jha, Michael Vakoc, and Todd Segal. 2025. Announcing the Agent2Agent Protocol (A2A). https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/. Accessed: 28.04.2025.

[14] Danny Weyns and Fabien Michel. 2015. Agent Environments for Multi-agent Systems – A Research Roadmap. In *Agent Environments for Multi-Agent Systems IV*, Danny Weyns and Fabien Michel (Eds.). Springer International Publishing, Cham, 3–21.