Discussion Topic: Agent Communication Languages

by Pavlos Papachristos - Sunday, 14 September 2025, 2:47 PM

Number of replies: 1

Discussion Topic: Agent Communication Languages

One advantage of the ACLs like KQML is their interoperability that allows KQML to act as a language for agent communication by defining how agents can ask questions, share information and request actions from one another.

This supports the development of open multi-agent systems where heterogeneous components collaborate (Labrou & Finin, 1997). Another benefit is expressiveness. By encoding communicative acts, ACLs allow interactions such as negotiation, brokering, or subscribing to event notifications.

The ACLs however, have disadvantages due to their complex implementation and maintenance and the required agreement on ontologies and content languages. Moreover, while KQML specifies communication performatives, it does not prescribe precise semantics for content, leading to potential ambiguity. In practice, many projects have found method invocation approaches (e.g., Java RMI or Python RPC frameworks) to be simpler and more efficient when agents are homogeneous and tightly coupled. Direct method calls also provide better performance because they avoid the overhead of parsing and interpreting message structures.

In comparison, method invocation in Python or Java is best suited for tightly integrated systems, where all components share a common runtime environment. ACLs, on the other hand, are more appropriate for loosely coupled, distributed, and heterogeneous systems, particularly where autonomy and flexible coordination are required. Thus, the choice depends on whether the system prioritises ease of integration and semantics-rich interaction (favouring ACLs) or efficiency and simplicity (favouring direct method invocation).

References

Finin, T., Labrou, Y. and Mayfield, J., 1994. KQML as an agent communication language. Proceedings of the Third International Conference on Information and Knowledge Management. ACM, pp.456–463.

Labrou, Y. and Finin, T., 1997. A semantics approach for KQML—a general purpose communication language for software agents. Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS), pp. 447–454.

In reply to Pavlos Papachristos

Peer Response

by Martyna Antas - Sunday, 28 September 2025, 9:13 PM

Pavlos, thank you for your meaningful contribution to this discussion. I found your post a well-balanced comparison of ACLs and method invocation. I agree with your observation that interoperability and expressiveness are key advantages of ACLs, particularly in heterogeneous and distributed environments (Labrou and Finin, 1997). Your point about the ambiguity in content semantics is also important; KQML was deliberately underspecified in this regard, which makes flexibility possible but also increases the potential for misinterpretation.

That said, I would have liked to see you explore more of the current directions in this area. For example, Ehtesham et al. (2025) show how newer interoperability protocols such as the Model Context Protocol (MCP) and the Agent Communication Protocol (ACP) attempt to retain the flexibility of ACLs while addressing modern requirements like decentralised identity and cross-platform negotiation. These represent attempts to overcome precisely the implementation and maintenance challenges you highlight.

Another interesting perspective comes from Singh (1998) who argued that ACLs should be grounded not in unverifiable mental states but in social commitments between agents. Although an older idea, this social semantics has influenced later frameworks and could have been a useful lens for your point about ambiguity.

Finally, I would be curious to know your views on the rise of natural language as a communication medium for agents. Recent work shows that large language model-based agents can coordinate through dialogue in ways that bypass formal ACLs altogether (Yan et al., 2025). This raises a provocative question: will ACLs evolve alongside these trends or will they gradually be replaced?

References:

Ehtesham, A., Singh, A., Gupta, G.K. and Kumar, S. (2025) *A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-*

Agent Protocol (A2A), and Agent Network Protocol (ANP). arXiv preprint. Available at: **https://arxiv.org/html/2505.02279v1** (Accessed: 28 September 2025).

Labrou, Y. and Finin, T. (1997) *A semantics approach for KQML—a general purpose communication language for software agents*. Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS), pp. 447–454.

Singh, M. (1998) *Agent communication languages: Rethinking the principles*. IEEE Computer, 31(12), pp. 40–47. Available at: **https://www.researchgate.net/publication/221454944** (Accessed: 28 September 2025).

Yan, B., Zhou, Z., Zhang, L., Zhang, L., Zhou, Z., Miao, D., Li, Z., Li, C. and Zhang, X. (2025) *Beyond self-talk: a communication-centric survey of LLM-based multi-agent systems*. arXiv preprint. Available at: **https://arxiv.org/abs/2502.14321** (Accessed: 28 September 2025).

Responses to the Forum:

Initial Post

by Martyna Antas - Sunday, 28 September 2025, 8:40 PM

Number of replies: 1

Agent Communication Languages (ACLs) such as KQML were originally designed to allow autonomous agents to exchange not only data but also requests, goals and commitments. Rooted in speech-act theory, they emphasise communication about intentions and purposes rather than simple procedural execution (Finin et al., 1994). This makes ACLs well suited to multi-agent systems operating in heterogeneous, distributed environments.

Compared with method invocation in Python or Java, ACLs provide a richer, semantically grounded framework. Method calls are efficient but tightly coupled: the caller must know the exact interface, and the interaction is deterministic. By contrast, ACLs enable flexibility and negotiation, allowing agents to collaborate even with partial or evolving knowledge. More recent research shows that this need for interoperability is still critical. For example, a 2025 survey highlights how protocols such as ACP and A2A extend earlier ACL concepts for use in contemporary distributed systems, with decentralised discovery and identity mechanisms (Ehtesham et al., 2025).

At the same time, communication in multi-agent systems is being reimagined. Deep reinforcement learning research has shown how protocols can emerge between agents without predefined languages, raising questions about when structured ACLs are necessary (Zhu et al., 2024). Similarly, recent work on large language model-based agents explores

how natural language can itself become a medium of coordination, blurring the line between designed protocols and emergent dialogue (Yan et al., 2025).

The trade-off remains between expressiveness and efficiency. Method invocation provides speed and simplicity, whereas ACLs and modern communication frameworks offer adaptability and autonomy. The challenge for researchers and practitioners is to decide when the additional complexity of rich communication yields genuine value over conventional programming techniques.

Reference:

Ehtesham, A., Singh, A., Gupta, G.K. and Kumar, S. (2025) *A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP)*. arXiv preprint. Available at: **https://arxiv.org/html/2505.02279v1** (Accessed: 28 September 2025).

Finin, T., Fritzson, R., McKay, D. and McEntire, R. (1994) *KQML as an agent communication language*. Proceedings of the 3rd International Conference on Information and Knowledge Management. ACM. Available at: **https://cdn.aaai.org/Workshops/1994/WS-94-02/WS94-02-007.pdf** (Accessed: 28 September 2025).

Yan, B., Zhou, Z., Zhang, L., Zhang, L., Zhou, Z., Miao, D., Li, Z., Li, C. and Zhang, X. (2025) *Beyond self-talk: a communication-centric survey of LLM-based multi-agent systems*. arXiv preprint. Available at: **https://arxiv.org/abs/2502.14321** (Accessed: 28 September 2025).

Zhu, C., Dastani, M. and Wang, S. (2024) *A survey of multi-agent deep reinforcement learning with communication*. Autonomous Agents and Multi-Agent Systems, 38(4), 4. Available at: **https://link.springer.com/article/10.1007/s10458-023-09633-6** (Accessed: 28 September 2025).

In reply to Martyna Antas

Re: Initial Post

by Pavlos Papachristos - Friday, 17 October 2025, 11:16 PM

You've done a great job outlining how ACLs have evolved, especially in showing the core tension that runs through multi-agent system design. To me, what's often overlooked is how the speech-act roots of ACLs reveal a kind of philosophical stance: they treat agents almost as intentional beings, with beliefs and goals we can reason about (Dennett, 1989). That

perspective still matters because it lets us think about behaviour on a conceptual level rather than getting stuck in code or protocol details.

Your comparison with simple method invocation is a helpful one, though I'm not sure the line between them is quite as sharp as it once was. In practice, modern service-oriented architectures and microservices have started to blur the edges. Systems now carry semantic descriptions or discovery layers that make them a bit more conversational than purely procedural (Papazoglou et al., 2007). Even RESTful APIs, when enriched with metadata, borrow some of the flexibility that ACLs were originally known for, just without the full pragmatic depth.

The discussion of emergent communication protocols feels especially current. Research on differentiable communication has shown that agents can invent compact, purpose-built languages that sometimes beat hand-crafted ones in narrow domains (Mordatch & Abbeel, 2018). The downside, as you note, is interpretability—humans can struggle to see what those agents "mean," which becomes risky anywhere safety or auditability matter.

Large-language-model agents complicate things even more. Natural language gives them tremendous expressive power and makes their exchanges easier for us to follow, but it also brings ambiguity, higher computational cost, and new security issues like prompt injection (Greshake et al., 2023). The real design question now isn't whether to use ACLs at all, but which layers of a system should keep strict formal semantics and which can safely rely on looser, natural interaction.

In the end, it comes down to context. Systems that need verifiable correctness will keep leaning on formal ACLs, while open or adaptive environments benefit from more fluid, emergent communication. My sense is that the most robust architectures ahead will mix both approaches, choosing the right language for the job rather than betting on a single model.

References:

- Dennett, D.C. (1989) The Intentional Stance. MIT Press, Cambridge, MA.

- Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T. and Fritz, M. (2023) 'Not what you've signed up for: compromising real-world LLM-integrated applications with indirect prompt injection', arXiv preprint arXiv:2302.12173.

- Mordatch, I. and Abbeel, P. (2018) 'Emergence of grounded compositional language in multi-agent populations', Proceedings of the AAAI Conference on Artificial Intelligence, 32(1), pp. 1495–1502.

- Papazoglou, M.P., Traverso, P., Dustdar, S. and Leymann, F. (2007) 'Service-oriented computing: state of the art and research challenges', Computer, 40(11), pp. 38–45.

Initial Post

by Jaco Espag - Wednesday, 3 September 2025, 4:08 PM

Number of replies: 2

Agent Communication Languages (ACLs), such as KQML, enable agents to communicate both data and the intent behind messages through performatives like inform and request. This approach is rooted in speech act theory, allowing agents to coordinate more meaningfully than simple method calls.

One of the main advantages of ACLs is interoperability. Agents developed in different programming languages or running on distinct platforms can still interact as long as they share a communication protocol and a common ontology. This flexibility makes ACLs well-suited for open and dynamic multi-agent systems where components may be heterogeneous and distributed (Souza et al., 2016). However, this strict reliance on a common ontology may lead to errors if agents interpret terms differently. This makes achieving semantic alignment both essential and challenging at the same time.

Developing and maintaining shared ontologies along with their supporting components, parsers and semantic managers, adds complexity and overhead to the system (Berna-Koes, Nourbakhsh and Sycara, 2004; Fatras, Ma and Jørgensen, 2022). Method invocation in languages like Python or Java is more straightforward and efficient. It requires no additional protocol or semantic layer, but it only works reliably in tightly coupled, homogeneous environments where all components are built within the same framework or have deep knowledge of the internals of other elements of the system.

In summary, ACLs provide semantic richness and autonomy valuable for complex, distributed systems. For smaller or tightly integrated systems, the simplicity and speed of direct method calls often outweigh the benefits of ACLs.

References

Berna-Koes, M., Nourbakhsh, I. and Sycara, K. (2004) ‘Communication efficiency in multi-agent systems’, in IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004. IEEE International Conference on Robotics and Automation, 2004. ICRA ’04. 2004, pp. 2129-2134 Vol.3. Available at: https://doi.org/10.1109/ROBOT.2004.1307377.

Fatras, N., Ma, Z. and Jørgensen, B.N. (2022) ‘An agent-based modelling framework for the simulation of large-scale consumer participation in electricity market ecosystems’, Energy Informatics, 5(4), p. 47. Available at: https://doi.org/10.1186/s42162-022-00229-0.

Souza, M. et al. (2016) ‘Integrating Ontology Negotiation and Agent Communication’, in V. Tamma et al. (eds) Ontology Engineering. Cham: Springer International Publishing, pp. 56–68. Available at: https://doi.org/10.1007/978-3-319-33245-1_6.

In reply to Jaco Espag

Peer Response

by Linga Murthy Kanuri - Saturday, 6 September 2025, 5:35 PM

Your analysis fairly evaluates the advantages and disadvantages of ACLs. One of the most appealing features of ACLs, particularly in distributed and heterogeneous systems, is interoperability. Agents developed in different programming languages and contexts can interact meaningfully through ontological communication (Souza et al., 2016). Because diverse ontological interpretations may make communication difficult, semantic alignment remains challenging.

The contrast you draw between ACLs and direct method invocation is also insightful. Method calls in languages such as Java or Python offer efficiency and simplicity, but they are best suited for tightly coupled and homogeneous environments (Berna-Koes, Nourbakhsh and Sycara, 2004). By comparison, ACLs provide richer semantics that support autonomy and flexible coordination. However, they introduce additional complexity, particularly in developing and maintaining parsers, semantic managers, and other supporting infrastructure (Fatras, Ma and Jørgensen, 2022).

Examining hybrid approaches in greater detail might be beneficial. While ACLs can handle interactions across more diversified or distributed infrastructures, direct method calls may be more appropriate for managing communication within homogenous subsystems. Performance and interoperability can be practically balanced using such a tiered strategy.

Overall, your piece successfully integrates theory and practice with solid backing from the well-chosen sources. You demonstrate how various communication techniques can be used based on the system's requirements and clearly outline the trade-offs.

References

- Berna-Koes, M., Nourbakhsh, I. and Sycara, K. (2004) ‘Communication efficiency in multi-agent systems’, *Proceedings. ICRA ’04. 2004 IEEE International Conference on Robotics and Automation*, 3, pp. 2129–2134. doi:10.1109/ROBOT.2004.1307377.
- Fatras, N., Ma, Z. and Jørgensen, B.N. (2022) ‘An agent-based modelling framework for the simulation of large-scale consumer participation in electricity market ecosystems’, *Energy Informatics*, 5(4), p. 47. doi:10.1186/s42162-022-00229-0.
-
- Souza, M. et al. (2016) ‘Integrating Ontology Negotiation and Agent Communication’, in Tamma, V. et al. (eds) *Ontology Engineering*. Cham: Springer, pp. 56–68. doi:10.1007/978-3-319-33245-1_6.

In reply to Jaco Espag

Re: Initial Post

by Pavlos Papachristos - Friday, 17 October 2025, 11:26 PM

Agent Communication Languages, or ACLs, such as KQML, were created to let software agents exchange not only information but also intent. Through performatives like inform or request, an agent can express what it wants another agent to know or do. The idea traces back to speech-act theory, where communication is viewed as action rather than mere data transfer. Because of that grounding, ACLs give agents a way to coordinate in a manner that feels closer to conversation than to simple function calls.

One clear strength of ACLs lies in how they promote interoperability. Even if two agents are written in different languages or live on separate platforms, they can still communicate effectively provided they share a protocol and a common ontology. That ability makes ACLs attractive for large, open multi-agent systems that need to evolve and plug in new components on the fly (Souza et al., 2016). At the same time, relying so heavily on shared ontologies brings its own problems. When two agents interpret the same term slightly differently, subtle but serious errors can creep in. Maintaining consistent meaning—semantic alignment—is therefore both essential and notoriously difficult.

Building and maintaining those shared ontologies, together with all the supporting tools such as parsers and semantic managers, adds a fair bit of complexity and runtime cost (Berna-Koes, Nourbakhsh & Sycara, 2004; Fatras, Ma & Jørgensen, 2022). By contrast, direct method calls in languages like Python or Java feel much simpler and faster. They don’t need extra protocols or semantic layers. The trade-off, of course, is that such calls only work well inside tightly coupled systems, where every component has been designed within the same framework—or at least knows quite a lot about how its peers operate internally.

In practice, the choice depends on context. ACLs offer richer semantics and greater autonomy, which are invaluable in large, distributed, and evolving environments. But for smaller or more uniform systems, the plain efficiency of a direct method call often wins out. There’s no single answer here; each approach carries its own balance of flexibility, overhead, and control.

References

- Bellifemine, F.L., Caire, G. and Greenwood, D. (2007) Developing Multi-Agent Systems

with JADE. John Wiley & Sons, Chichester.

- Euzenat, J. and Shvaiko, P. (2013) Ontology Matching. 2nd edn. Springer, Heidelberg.

- Robu, V., Noot, H., La Poutré, H. and van Schijndel, W.J. (2013) 'A multi-agent platform for auction-based allocation of loads in transportation logistics', Expert Systems with Applications, 40(8), pp. 3483–3491.

- Visser, P.R., Jones, D.M., Bench-Capon, T.J. and Shave, M.J. (1997) 'An analysis of ontology mismatches: heterogeneity versus interoperability', AAAI 1997 Spring Symposium on Ontological Engineering, pp. 164–172.