Initial post – Comparing agent communication languages vs method invocation

Agent Communication Languages (ACLs), such as KQML (Knowledge Query and Manipulation Language), have been instrumental for the development of structured and complex interactions in multi-agent systems, and continue to play a crucial role in distributed systems (Finin *et al.*, 1994). Some of the main positive features of ACLs include flexibility and interoperability, semantic richness, and asynchronous communication (Ignise and Vahi, 2024). ACLs provide a standardised and agnostic communication method, which allows multiple, disparate agents to communicate regardless of their internal architecture or implementation language (Kravari and Bassiliades, 2015; Savaglio *et al.*, 2020). Languages like KQML also offer a wide range of performatives (communication acts) which can express complex intentions, beliefs, and queries, enabling nuanced and context-aware interactions (Challenger, Kardas and Tekinerdogan, 2016). Finally, ACLs typically support asynchronous message passing, which is well-suited for distributed systems where agents may operate independently and at different speeds (Bellifemine, Caire and Greenwood, 2007).

Despite these unique advantages, ACLs also present certain challenges when compared to standard method invocation in languages like Python or Java, mostly due to language complexity, performance challenges, and learning curves. First, ACLs can be more complex to implement and use compared to simple method invocation, requiring additional message parsing and interpretation (P. Müller and Fischer, 2014). Similarly, the need to encode and decode messages in a specific format can introduce performance overheads, especially in systems with high-frequency interactions (Li *et al.*, 2024). Finally, developers familiar with widely-used programming languages may face additional hurdles if attempting to work with ACLs (Bergenti *et al.*, 2017). By contrast, method invocation in Python or Java offers simplicity and direct control flow, while providing provides strong type checking and immediate feedback on errors, which facilitates code reading and debugging (Matveeva, 2023). However, this approach is limited to local or tightly coupled systems employing similar architectures, and lacks the semantic expressiveness of ACLs (Woolridge, 2009).

In conclusion, while ACLs offer powerful capabilities for agent-based systems, their adoption needs to be carefully considered based on the specific requirements of the application and trade-offs between flexibility and complexity.

References

Bellifemine, F., Caire, G. and Greenwood, D.P.A. (2007) *Developing Multi-Agent Systems with JADE*. Chichester: Wiley (Wiley Series in Agent Technology). Available at: https://www.wiley.com/en-us/Developing+Multi-Agent+Systems+with+JADE-p-9780470058404 (Accessed: 10 March 2025).

Bergenti, F. *et al.* (2017) 'Agent-oriented model-driven development for JADE with the JADEL programming language', *Computer Languages, Systems & Structures*, 50, pp. 142–158. Available at: https://doi.org/10.1016/j.cl.2017.06.001.

Challenger, M., Kardas, G. and Tekinerdogan, B. (2016) 'A systematic approach to evaluating domain-specific modeling language environments for multi-agent systems', *Software Quality Journal*, 24(3), pp. 755–795. Available at: https://doi.org/10.1007/s11219-015-9291-5.

Finin, T. *et al.* (1994) 'KQML as an agent communication language', in *Proceedings of the third international conference on Information and knowledge management*. New York, NY, USA: Association for Computing Machinery (CIKM '94), pp. 456–463. Available at: https://doi.org/10.1145/191246.191322.

Ignise, A. and Vahi, Y. (2024) 'Collaboration and Coordination in Multi-Agent Systems', *TechRvix* [Preprint]. Available at: https://doi.org/10.36227/techrxiv.172902764.41177581/v1.

Kravari, K. and Bassiliades, N. (2015) 'A Survey of Agent Platforms', *Journal of Artificial Societies and Social Simulation*, 18(1). Available at: https://doi.org/10.18564/jasss.2661.

Li, X. *et al.* (2024) 'A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges', *Vicinagearth*, 1(1), p. 9. Available at: https://doi.org/10.1007/s44336-024-00009-2.

Matveeva, A. (2023) *A Guide to Debugging Python code (and why you should learn it)*, *Medium*. Available at: https://medium.com/@yellalena/a-guide-to-debugging-python-code-and-why-you-should-learn-it-ae30d20419b7 (Accessed: 10 March 2025).

P. Müller, J. and Fischer, K. (2014) 'Agent-Oriented Software Engineering', in *Application Impact of Multi-agent Systems and Technologies: A Survey*. Berlin Heidelberg: Springer-Verlag, pp. 27–53. Available at: https://link.springer.com/chapter/10.1007/978-3-642-54432-3_3 (Accessed: 10 March 2025).

Savaglio, C. *et al.* (2020) 'Agent-based Internet of Things: State-of-the-art and research challenges', *Future Generation Computer Systems*, 102, pp. 1038–1053. Available at: https://doi.org/10.1016/j.future.2019.09.016.

Woolridge, M. (2009) *An Introduction to MultiAgent Systems, 2nd Edition | Wiley*. 2nd edn. Chichester: John Wiley & Sons. Available at: https://www.wiley.com/en-us/An+Introduction+to+MultiAgent+Systems%2C+2nd+Edition-p-9780470519462 (Accessed: 11 February 2025).