Summary post

In my initial post, I discussed how Agent Communication Languages (ACLs), specifically KQML, can facilitate structured interactions within multi-agent systems, emphasizing their strengths in flexibility, interoperability, and asynchronous communication (Finin *et al.*, 1994). I also addressed their downsides, notably complexity, performance challenges, and their learning curve compared to method invocation in languages like Python and Java (P. Müller and Fischer, 2014). Georgios echoed these points, underscoring the advantages of ACLs while suggesting that more advanced programming languages (including hybrid models) are increasingly favoured for their simplicity and performance (Bennett, Farmer and McRobb, 2016). Noora agreed with my analysis and noted the trade-off between expressiveness and complexity in ACLs, proposing that a hybrid approach using both method invocation and ACLs could offer balanced solutions for various applications.

The hybrid model suggested by my colleagues presents a promising direction for multi-agent systems. By integrating functionality from both method invocation and ACLs, developers can leverage straightforward control flows and rapid debugging when implementing core logic using languages like Python and Java, while utilizing ACLs like KQML for rich, high-level communication. Frameworks such as JADE and SPADE provide powerful structures for this hybridization, using distributed architectures and adhering to protocols like FIPA ((Bellifemine, Caire and Greenwood, 2007; Bergenti *et al.*, 2017; Palanca *et al.*, 2020). Additionally, the advanced performatives introduced by CG-KQML can enhance cooperative tasks while maintaining the efficiency of method invocation, allowing for semantic expressiveness in agent interactions (Bouzouba, Moulin and Kabbaj, 2020).

In conclusion, while ACLs present significant advantages for agent communication, evaluating their integration with traditional programming methods in a hybrid model can lead to more effective solutions for complex, multi-agent environments. This balanced approach holds the potential to maximize the benefits of both paradigms, enhancing the design and operational capabilities of intelligent systems.

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).

Bennett, S., Farmer, R. and McRobb, S. (2016) *Object-Oriented Systems Analysis and Design Using UML*. 4th Edition. McGraw Hill. Available at: https://www.mheducation.co.uk/object-oriented-systems-analysis-and-design-using-uml-9780077125363-emea (Accessed: 7 April 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.

Bouzouba, K., Moulin, B. and Kabbaj, A. (2020) 'CG-KQML+: An Agent Communication Language and its use in a Multi-Agent System', *CEUR Workshop Proceedings*. [Preprint].

Finin, T. *et al.* (1994) 'KQML as an agent communication language', in *Proceedings of the third international conference on Information and knowledge management - CIKM '94. the third* *international conference*, Gaithersburg, Maryland, United States: ACM Press, pp. 456–463. Available at: https://doi.org/10.1145/191246.191322.

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).

Palanca, J. *et al.* (2020) ‘SPADE 3: Supporting the New Generation of Multi-Agent Systems’, *IEEE Access* [Preprint]. Available at: https://doi.org/10.1109/ACCESS.2020.3027357.