From simulation to development in MAS: A JADE-based Approach

João P. C. Lopes

Thesis developed under supervision of *Henrique Lopes Cardoso*

June 23, 2014

---

1. Abstract

Multi-agent systems (MAS) present an interesting approach to the efficient development of modular systems, composed autonomous, interacting computational units called agents. Frameworks exist that aid the development of this class of systems and they range from mostly general-purpose frameworks to domain-specific in an array of different domains.

1.1. Motivation

Multi-agent-based simulations (MABS) are sometimes used on the course of development of a full-featured MAS – for instance, for testing purposes – for the potential gains in performance when simulating MAS. However, most platforms for MAS development are not well suited for MABS development due to scalability limitations[1]. Some related works were studied that demonstrate some interest in solutions that make MABS more easy to be created using MAS development frameworks. Furthermore, an opportunity exists to partially automate the development of robust MAS from a previously tested simulation.

JADE [2], a very popular MAS development framework allows the creation of seamless distributed agent systems and complies with FIPA standards for agent interaction. Unfortunately, its multi-threaded architecture falls short in delivering the necessary performance to run a local simulation with a large number of agents.

Repast [3] is an agent-based simulation toolkit that allows creating simulations using rich GUI elements and real time agent statistics. It can easily handle large numbers of agents in a single simulation. Unlike JADE, though, Repast lacks much of the infrastructure for agent creation and interaction.

The main motivation for this thesis lies in the potential performance gains when using agent simulation frameworks to produce a simulation of a MAS more complex than those typically developed with such frameworks. Some works [4, 5] propose, as a solution for bridging the gap between MAS development and simulation, an integration of JADE and simulation features by extending this framework with a simulation layer created from scratch, or by integrating another framework, such as Repast.

1.2. Goals

This thesis proposes the development of an integrated solution for bridging the domains of simulation and MAS. In order to do that, two main sub-goals were identified.

1. **First**, the creation of an adapter or API that would allow developers to abstract from simulation frameworks' features. It does so by reimplementing from scratch many JADE features, including its implementation of FIPA specifications for agent interaction and management. Since the API's architecture is very close to JADE's, conversion becomes more straightforward.

2. **Second**, the development of a code conversion mechanism. By abstracting from the simulation tools and creating a MAS-like MABS, it becomes possible and more straightforward to engineer a tool that performs the automatic conversion of these MABS into equivalent MAS.

1.3. SAJaS

The **Simple API for JADE-based Simulations (SAJaS)** provides a set of features present in JADE; those features were reimplemented from scratch in an attempt to simplify their internal complexity, increasing its overall performance but preserving JADE-like external execution. Among these JADE-based features are the agent management, messaging and agent interaction FIPA specification.

One of the main conceptual challenges when developing SAJaS was in adapting JADE's asynchronous execution to Repast's tick-based synchronized environment. Except when executed during their setup or takedown, agents' actions in JADE are encapsulated in Behaviours which run concurrently or in parallel. In SAJaS, behaviours are executed sequentially. Furthermore, to emulate JADE's asynchronous communication

in SAJaS, messages sent to an agent are kept in its message queue until needed, and processed only then – in contrast, the arrival of a message in JADE triggers the appropriate behaviour right away.

SAJaS's behaviours follow the same life cycle as their JADE counterpart, implementing the methods `action`, `onStart`, `done` and `onEnd`. It also currently includes an implementation of some JADE interaction protocols, namely FIPA Request and FIPA Contract Net.

A concious effort was made to keep SAJaS fairly generic in regard to its dependency to Repast, opening doors for future integration with other simulation tools without the need to modify the existing framework.

1.4. MASSim2Dev

The **MAS Simulation to Development code conversion tool (MASSim2Dev)** is the tool that, using SAJaS, closes the gap between MAS simulation and development.

JDT was chosen for the development of this tool. Some of its most interesting features are the automatic cloning of projects, the handling of classes, imports, methods and fields as objects and the possibility of doing complex manipulation tasks without parsing the code. It also allows the use of a high level AST for a more direct manipulation of the source code.

After conversion, no dependency to the previous platform exists in the generated project. Naturally, this is not true is JADE features that are not yet available in SAJaS are used in the MAS. The same happens when using internal features from SAJaS that are not present in JADE.

1.5. Validation

Three tests were designed to demonstrate that the behaviour of the currently implemented JADE features in the API is the same as the analogous JADE ones. Furthermore, using the API it is possible to achieve very significant performance gains.

The first example consists in a simple contract net between one buyer and multiple sellers. In the second example, multiple contract nets run concurrently and some of the buyers include available computational trust about sellers. The third example is a board game called Risk developed prior to this thesis. The goal of this last example was to test SAJaS in a "real" example of a previously developed MAS, one that had not been developed specifically for this thesis.

These scenarios covered all SAJaS features. It was possible to demonstrate that bringing JADE and Repast together is not only feasible using the developed tools, but also provides increased performance when compared with JADE MAS.

1.6. Conclusions

The study of the available literature demonstrated an interest in developing tools for simulation of MAS due to limitations in existing MAS development frameworks.

The proposed integrated solution, composed of SAJaS and MASSim2Dev, allows the development of simulations using familiar JADE features in a simulation framework like Repast.

Presently, a subset of JADE features are present in SAJaS. However, the validation tests created show that these with features it is possible to create MAS with some complexity and that their conversion using MASSim2Dev preserves the original functionality.

Some suggested future work includes expanding the subset of JADE features supported by SAJaS, extending native support to more simulation frameworks, enhancing the plugin with user configurations and enabling support for charts and displays in SAjaS.

References

[1] D Mengistu, P Troger, L Lundberg, and P Davidsson. Scalability in distributed multi-agent based simulations: The jade case. In *Future Generation Communication and Networking Symposia, 2008. FGCNS'08. Second International Conference on*, volume 5, pages 93–99. IEEE, 2008.

[2] Fabio Luigi Bellifemine, Giovanni Caire, and Dominic Greenwood. *Developing multi-agent systems with JADE*, volume 7. John Wiley & Sons, 2007.

[3] N Collier. Repast: An extensible framework for agent simulation. *The University of Chicago's Social Science Research*, 36, 2003.

[4] E García, S Rodríguez, B Martín, C Zato, and B Pérez. Misia: Middleware infrastructure to simulate intelligent agents. In *International Symposium on Distributed Computing and Artificial Intelligence*, pages 107–116. Springer Berlin Heidelberg, 2011.

[5] J Gormer, G Homoceanu, C Mumme, M Huhn, and J Muller. Jrep: Extending repast simphony for jade agent behavior components. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 02*, pages 149–154. IEEE Computer Society, 2011.