K. Mustapha, E. Tranvouez, B. Espinasse, A. Ferrarini (2011), « An Agent-based and Organization oriented Software Architecture for Supply Chains Simulation », IEEE-ETFA 2011, 16th International Conference on Emerging Technologies and Factory Automation, Toulouse, France, 5-9 sept. 2011

An Agent-based and Organisation oriented Software Architecture for Supply Chains Simulation

Karam MUSTAPHA, Erwan TRANVOUEZ, Bernard ESPINASSE, Alain FERRARINI
LSIS UMR CNRS 6168 – Aix-Marseilles University
Domaine Universitaire de Saint-Jérôme, 13397 MARSEILLE CEDEX 20, France
{karam.mustapha, erwan.tranvouez, bernard.espinasse, alain.ferrarini}@lsis.org

Abstract

The Supply Chain (SC) organizational structure and related management policies are crucial factors that can be adjusted to improve the SC performance, and tested through simulations. To facilitate the design of these simulations, we have proposed an agent-based methodological framework for SC modelling, taking into account observables of different levels of details and related to these SC organizational aspects. This paper describes an agent-based software architecture, based on a mediator, enacting this methodological framework at a software level, to allow SC organisational-oriented simulation. This architecture can be seen as the interaction between different simulation platforms.

Keywords: *Agent Based Simulation, Multi-Agent Systems, Supply Chains, Organization, Indicator,*

1. Introduction

The Supply Chain (SC) domain raises numerous conceptual and architectural challenges. Because of its complexity, developing SC simulation-based decision support systems implies a heavy workload. Simulation aims to experiment and understand (in a controlled environment) the economic, human and environmental consequences of decisions related to the design and the organization and the management policies of production facilities. Multi-agent or Agent Based Simulation (ABS) contribution to SC studies is well established [5] [23] [15]. As autonomous entities with the ability to perform their functions without the need for continuous interaction from the user, agents are used for design and/or simulation of complex systems. ABS also allows focusing on the behaviours of the various SC's actors.

The Supply Chain (SC) organizational structure and related management policies are a central factor that can be adjusted to improve the SC performance, which consequently has to be taken into account in the SC modelling and simulation. However, most of the related various research works does not allow to study the efficiency of organizational related decisions. Such a study supposes to: i) describe the SC organization; ii) model and simulate the behaviours and decisions of its actors and iii) implement these decisions and exhibit their local and global effects on the SC, iv) support each of these steps with specific conceptual and software support.

The global objective of the present research is to give a focus on the impact of a SC's organizational structure performance by providing a methodological framework which ranges from domain model analysis to running the simulation. In line with our previous works [14] on SC simulation, and in order to consider these organizational aspects of the SC, we have proposed a specific agent-based methodological framework [16] allowing, from modelling to simulation, the production of observables at different levels of details related to a SC organization. This framework aims to facilitate the realization of the SC simulation with gradual processes. It begins by defining the needs of the user prior to arriving to the implementation of the system while satisfying the initial requirements. This methodological framework relies on a software architecture adapted to the needs of SC simulation, such as heterogeneous simulation software environments integration.

This paper proposes an agent based software architecture that supports the simulation of supply chains in operation, taking into account its organizational structure and allowing the study of the impacts of this organization, based on a methodological framework. This architecture can be seen as the interaction between different simulation platforms.

Section 2 exposes our research problematic, which concerns modelling and simulation (M&S) of SC with their organizational aspects. In section 3, we briefly present our organizational-oriented methodological framework that takes into account SC's organizational aspects, with models at a conceptual and operational abstraction level. Then in section 4, we detail the agent-based software architecture in line with the proposed methodological framework, to simulate SC's organizational aspects. This architecture is instantiated on an illustrative example. Finally, we conclude by drawing the future step of our research.

2. Agents and Organisation Oriented SC Modelling and Simulation

Agent Based Simulation (ABS) allows the understanding of various dynamic models, as composed of entities with different complexity levels (from very simple entities or reactive agents to more complex ones such as deliberative agents). Another interest of ABS is the ability offered to the modeller to manipulate different levels of representations, such as individuals and groups of individuals. Agent-based modelling allows capturing of the dynamic nature of SCs and facilitates the study of numerous resources coordination associated with the interaction of multiple companies [15].

Agent based SC simulation is now frequently used, but few researchers have proposed a general framework to support both the design and the realization of the SC simulation. Among those, the MASCF methodology (Multi-Agent Supply Chain Framework) [11] adapts the SCOR model to a structured generic methodology for multi-agent system development (Gaia). However, the organizational modelling is based on a management of a process metaphor that underrates the organizational structure. A more general study of agent oriented software engineering methodologies (among those rare holonic compliant methods), undertaken in order to find conceptual and operational solutions, has confirmed that organizational issues were to be added to the actor approach [14]. Methods like GAIA [27], CRIO [10], MOISE+ [13] or Luis Antonio work [1], provide only a part of the solution for the required objectives.

Almost all the previously referred approaches use the notion of *roles* in order to promote the flexibility of the design process, even with different abstraction or hierarchical levels. As an abstract view of the distributed organization, roles can be combined and associated to agents' specific architecture: from complex information processing units (i.e. with deliberating capacities) to more simple programmable units (reactive agents or state-machine like automata).

As the structure of the studied systems pre-exists, the description of the organization must be included from the beginning of the modelling approach, in order to propose the best suitable observables of its components. The *group* and the *holon* concepts meet this requirement. Finally, cooperative behaviours are needed to reproduce cooperation situation in a “real” SC, as well as a way to deal with disrupting events, adding adaptability to the SC [24]. The deliberative/reactive agent architecture results directly from the needs of validating such cooperative behaviours [14].

Moreover, in order to complete or reduce the M&S process, previous experimentations can be reused by exploiting “black-box” simulations already implemented (ie COTS - Components On the Shelves) in more “classical” modelling language and dedicated simulation environment. The final simulation must deal with deliberative agents, reactive agents and other simulations involving different time horizons. Time synchronization then becomes a hard requirement to be identified at the modelling phase and eventually controlled at the software level (and maintained at the intermediate translation steps).

Observables that have to be taken into account are SC data and information on on-going decision processes, which need to be highlighted in the simulation results. Therefore, the main goal is to reproduce the SC behaviour according to the level of details required to produce the user desired observables. The observables represent simple or aggregated (at different hierarchical levels) values or indicators describing the states of the SC entities or performance, as well as physical or decision processes (scheduling plans, stock management strategies, etc.) and their consequences (performance evaluation of their outcomes on the SC). An indicator is usually defined as selected information associated with a phenomenon and designed to observe periodic changes by the light of objectives. Therefore, it is a quantitative data that characterizes an evolving situation (an action or consequences of an action) in order to evaluate and to compare their status at different dates [16].

Another objective of the presented work is to propose software architecture to execute the obtained simulation models. In order to achieve this objective, we consider the following requirements (at a functional or software level) which have to be met [8] [10] [14] [25]: Multi-level modelling, Multi-scale simulation, Multi-paradigm modelling, Managing different temporal scale, and Openness to modelling or simulation legacy software [16]. Due to the SC nature and its simulation requirements, specific distributed software architecture is needed. Two main approaches are possible:

- *Propose a generic (homogeneous) agent based architecture* (with a dedicated modelling language) [10] [12] that requires either modelling from scratch or translation of the models into a new/other simulation environment; resulting in both case in development cost rise (time and expertise). Moreover, modelling decision process requires AI-like (Artificial Intelligence) behaviours hindering such approach [6].
- *Coordinate separate simulations* (particularly when different paradigms are used) through interoperability mechanisms and protocols as HLA (High Level Architecture) [17] that provide a more open approach to integrate previous simulations. Some agent-Based HLA implementation has been proposed, but HLA architecture has suffered some critics about its complexity and performance (level of data exchange). [2] as well as imposing significant modification in the simulators architectures.

The solution, proposed in this paper, is to combine these two approaches, by using an organizational oriented individual-based modelling approach that is simple enough to be related to the domain-dedicated modelling language, and also by producing models which afterward can be translated into other modelling paradigm and simulation language. The simulation of this model is ensured by an agent-based framework coupling distributed simulations potentially implemented in different environments (as in [8] in an environmental decision support context), while respecting the temporal and data dependencies between all the simulations.

This paper focuses on the software architecture, but in order to apprehend the global simulation process, next section presents briefly our methodological framework.

3. A Methodological Framework for SC Organizational Aspects M&S

The complexity of SC modelling and simulation process as well as implementation support, lead us to propose a modelling approach based on an incremental process, relying upon models with gradual increasing details. The real system is firstly represented by a domain model of SCs (e.g. a NetMan model as in [14], an UEML model -Unified Enterprise Modelling Language - etc.) to represent the organizational aspects. We propose a structured organization-oriented methodological framework according to two main abstraction levels: a *conceptual* and an *operational* level. Using the domain model provided by the domain expert, a simulation model is built step by step. The conceptual level proposes concepts and models helping to grasp the complexity of the SC and its simulation objectives, whereas the operational level prepares the implementation of the simulation model including software integration issues. The different models and the transition to agent-oriented M&S in our methodological framework are presented in Figure 1 (refer to [16] for more details).

The Conceptual Organizational Modelling engages through a dialogue between the domain expert and an agent-knowledgeable modeller. An actor model is produced by identifying the active entities and their organization from the domain model according to the role concept. The modeller has to translate/abstract the domain model into a Conceptual Organizational Modelling based on (hierarchical) levels, actors, roles and groups named Conceptual Role Organizational Model (CROM). This stage highlights the organizational structure of the SC as wells as the structural and dynamic relations between the entities composing this SC. Then, a conceptual agent-based model is produced on the basis of observables which the user needs to obtain from the simulation building up the route toward the implementation of the simulation. This model is transposed into the agent world (at a conceptual level) concluding the phase of "specification" with a multi-agent and organization model named Conceptual Agent Organizational Model (CAOM) ready to be described at an architectural and software design level.

The important key of this step is to precisely identify the agents defined at the conceptual level in order to develop them adequately at the operational level. The Operational and Organizational Modelling provides a solution to implement an executable system to perform simulations based on the previous conceptual models. This step involves the choice of agent architectures, depending on the complexity of the behaviours needed to be simulated. This process is guided by the observables selected earlier by the domain expert.

The software designer details the CAOM by associating a conceptual agent with a software agent architecture (e.g. BDI - Believe, Desire, Intention - [20]) and specifying their behaviours (e.g. an UML - “Unified Modelling Language” - activity diagram for a reactive agent) and interactions (e.g. AUML - “Agent Unified Modelling Language” - sequence diagram [18]), resulting in an OPerational Agent Model (OPAM). The implementation of these models in a simulation(s) environment results in an ABS system which can be executed. The refining process currently follows *ad hoc* rules. As experience with the models increase rule generalisation can be defined and then automated through model transformation engines.

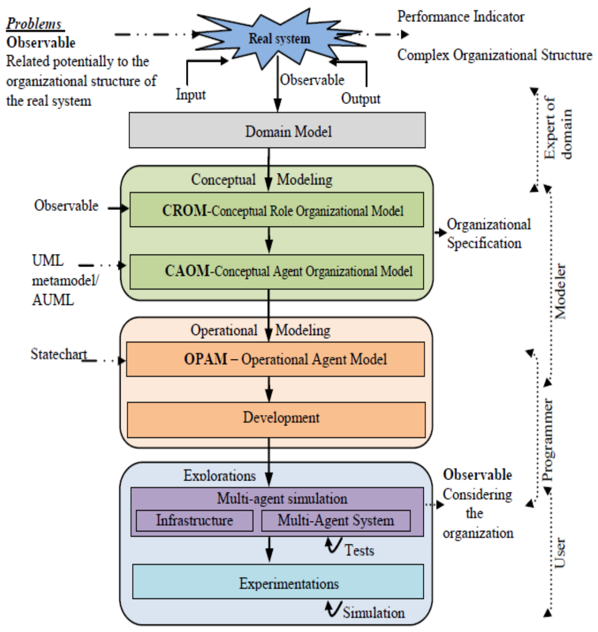

Figure 1. Methodological framework cycle

In [14] the observables, potentially related to the organizational structure of the real system, are not described in the design model. They are only mentioned in the multi-agent system model, i.e. only one step before implementation. As they may induce different simulation needs, it is necessary to describe them earlier in the modelling process (at a conceptual and operational level). Moreover, as observables can describes phenomena at different level of the SC (an actor state or activity, a group of actor ie a production cell or a company, a cooperation process…), the organisation modelled can be studied along different points of view and modelled according to these observables.

A second objective of our work is to propose a software model that is adequately open to different software platforms in order to facilitate the translation process (model to implementation) as well as reuse the previous simulation models. In the next section, we present the proposed general software architecture for modelling and simulation.

4. An Agent-based Software Architecture for SC Organizational Aspects Simulation

The simulation of the operational model, produced after several stages of models refinement, assumes the existence of a software infrastructure that supports heterogeneous simulation models. In addition, it should ensure the integrity of the distributed simulation (of two or more software environments) while providing the desired simulation data (observable). In this section, firstly we present what requirements rise up from these objectives, before introducing the general architecture of an agent and the organizational oriented simulator.

4.1. Architectural requirements

This section addresses simulations integration and interoperability issues, viewed as the management of data and event dependencies between simulators. Considering the complexity of such task, we combine different integration approach: FIPA (Foundation of Intelligent Physical Agents) specifications on agent-based software integration [9], HLA specification on distributed simulation integration [7], in order to redefine initial ad-hoc Actor simulation architecture [14].

FIPA proposes to agentify software services in order to separate the discovery and selection of services from the actual service call. Interaction protocols are defined to support the chain of actions that agents follow to track and execute software distributed over an open environment. It is a general software integration approach which, however, does not deal with data sharing and time synchronization at a conceptual or software level.

HLA, an IEEE standard, is totally dedicated to distributed simulation management As an integration specification, HLA does not propose a software implementation or consider the internal structure of Simulators (Federate). Its reckoning by the simulation community has resulted in numerous implementation and adaptation to different application domain, including SC simulation [17]. A Distributed Simulation is seen as a Federation of Simulators, coordinated by a central unit - the RTI (Real Time Infrastructure) – exchanging data and instantiating an Object Modelling Template (OMT) in respect with simulation rules which maintain the integrity of the global simulation (data format, time synchronization, events causality chain…).

Labarthe‘s architecture couples an agent-based simulation - simulating decision-making processes - with Anylogic (*www.xjtek.com*) a Discrete Event Simulation Software - simulating SC resources i.e. the SC physical system. Coupling is ensured by an agent scheduler who routes events from the physical systems to the decision system. As the simulation clock is used in the Anylogic models, simulation is driven by Anylogic. Additionally, organization structures are not explicitly described, and simulation data is centralized in Anylogic.

Our approach to SC simulation considers heterogeneity of agent behaviour as the consequences of the domain which expert observable choices and not necessarily the nature of the SC entities. Thus the simulation deals with heterogeneous complex behaviour which the simulation framework must integrate.

4.2. Software Architecture for Modeling and Simulation (SAMOS)

As a first step toward generalization, we have considered two simulation environments integrated through a mediator. The basic idea was to identify and isolate the simulation function which ensures the simulators integration. As shown in fig.2; the architecture is designed to be the more open to other simulators as possible with a mediator used to facilitate the interactions between them. However, while keeping in mind such objectives, we have chosen to test our propositions by beginning with two “specialized” simulation environments. SAMOS is thus currently composed of i) the JASON platform; ii) the JADE platforms and iii) a mediator.

The JASON platform is adapted to the development of BDI (ie deliberative) agents [3] [4]. It is an extended interpreter [25] of AgentSpeak [19] a BDI programming language allowing complex behaviour modelling. The *JADE platform* (Java Agent Development Framework) [22] is also a FIPA compliant Agent Oriented Software Engineering tool implemented in Java. It proposes a framework for agent management (agent directories, communication management …). Agent internal structure is open and left mostly to the programmer initiative. The *mediator* see [22] supports the simulations integration by proposing generic services the more independently as possible of the simulators architecture.

JASON is used to implement and simulate decision-making processes, whereas JADE deals with simple agent behaviours. Agents from both environments must interact; the mediator realizes the transmission of information (message, signals, objects, data…) while keeping simulation specific constraints respected (for ex., time synchronicity between both environments). A Database is also included to capture the model parameters, record simulation data and results analysis. It is accessed by the simulators and the mediator. Figure 2 summarizes the general architecture of SAMOS.

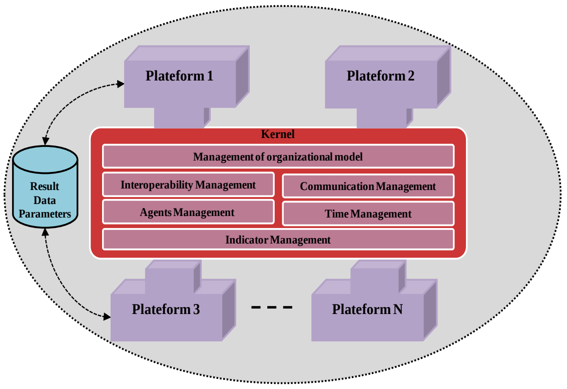

Figure 2. SAMOS General Architecture

The mediator role in the integration process is synthesized into five services presented in Table 1:

Services	Description
Agents Management	Classical agent life-cycle management, this module manages the birth and death of agents, …
Communication management	Also a basic service in Multi-Agents Systems, this module manage the agent directories (address and capabilities), as well as the logical routing of messages or events.
Organizational Model Management	This module manages the organization dynamics: group creation, subscribing and unsubscribing to groups…
Interoperability management	Responsible at the software level for interactions between simulations. It can rely on APIs to route physically message, events, data between the simulators, or clock synchronization signals.
Time Management	Ensures time is managed coherently in the simulators. Depending on the time management strategy it controls the execution of the simulator (for ex., pause a simulator while response is computed in another simulator).
Indicator Management	The aim is to produce the indicators characterizing the observable defined in the conceptual modeling of the supply chain studied.

Table 1. Services description

4.3. Agent modelling and interoperability

Current SAMOS environment contains several type of agents: i) *Deliberative agent*, developed in JASON, implements SC decision-making processes i.e. SC entities whose behaviours produce complex observables; ii) *Reactive agents*, developed in JADE, implementing basic behaviours; and iii) *Service agents*, i.e. agents not directly concerned by the simulation models but supporting the simulation process.

Accordingly to the complexity degree of their behaviours, agents interact in SAMOS either by exchanging rich content message or by sending signals. As deliberative agents may conduct negotiation or coordination processes, the consequent interaction must be described and carried out by interaction protocols with AUML interaction diagram. As FIPA Agent Communication Language (ACL) has been chosen, message is structured in order to qualify semantically each bits of information exchanged (i.e. intent of the message, protocols required…). Reactive agents can interact directly by emitting signals (or events) - i.e. message with limited content (e.g.. "machine breakdown") - or indirectly by modifying the state of objects defining their environment (e.g. status of the product manufactured). The *Communication management module* delivers these messages and translates them to a suitable format understandable by another simulation environment if necessary. This module is composed, in SAMOS, of directory agents responsible for keeping and spreading the information about the agents (name, address, capabilities). The Interoperability Management module then may have to translate this message at the software level (for ex., call the adequate API function which may generate an event in the other simulator). Table 2 summarizes different types of agents and their roles in SAMOS, some of them are provided by the JADE Platform.

Agent	Description
AMS Agent Management System	Manage agent life cycle, as well as "white pages" directory, i.e. the list of the agents name and their communication address.
***DF* - Directory Facilitator**	Provide a "Yellow Page" service as it record agents roles, capabilities and may answer request for another agent directory needs.
***ACC* - Agent Communication Channel**	Routes messages from an agent to another, independently of the platform of both agents. Implements for this purpose the IIOP protocol.
***IAg* Indicator Agent**	Is associated to an indicator: it provides computational facilities to produce the value of aggregated indicators. Thus it agentifies the observables identified in the conceptual models. Indicator agents are also categorized depending of the type of indicator they represent (Activity, Productivity, Quality…).
***DSA* Data Source Agent**	Centralizes the source of data in a group of agents, an Indicator Agent is needed for exploiting its values. Also responsible for finding the agents that have the required information. Then, it regroups and sends these data to the right Indicator Agent.
***GMA* Group Manager Agent**	Manages a group i.e. allows an agent to play a role in the group, as well as represent the agents in the group for specific requests. For ex., if an IAg needs a particular type of data, the Group Manager will identify the agents producing that data.

Table 2. Agents description

Next section illustrates how a simulation can be conducted within the SAMOS architecture.

5. An illustrative architecture of SC and simulation

5.1. An illustrative SAMOS implementation

Figure 3 illustrates a software architecture supporting our methodological and "simulation-related" requirement. As exposed in the previous section, this SAMOS implementation contains Jade Agents, JASON agents and a mediator in charge of their interaction.

The "simulation model" agents seen in this figures, results from applying our methodological approach ie progressive translation of the CROM and CAOM models of the case study presented in [16]. It is composed of 2 groups describing a simplified SC organisation structure. Communication between JASON and JADE agents is done through messages. Therefore, a mediator layer (denoted Kernel) ensures the communication link between different platforms ("physical" interoperability is simulated in this case as both are FIPA compliant environment). Please note, that the mediator is presently developed as a group of specialised agents.

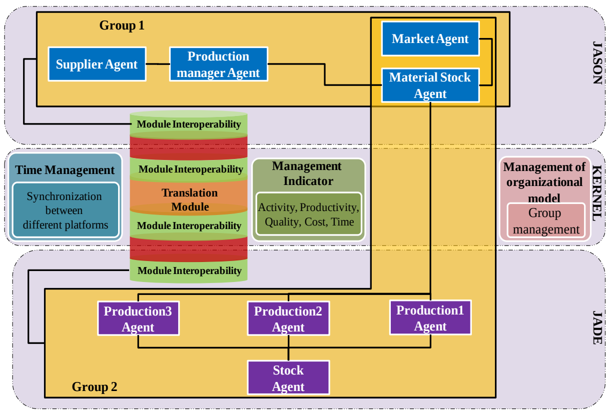

Figure 3. Architecture (Platforms: JASON and JADE, Mediator)

The role of each agent (either simulation-related or service-oriented) is explained in fig 4 (cf. appendix). This figure (edited for readability improvement) shows the exchange of message between these agents traced during a simple simulation. The simulation scenario (message – 1 to 22) begins with an initialization phase consisting in agents' requesting to play roles in groups to dedicated managers (repertories, group manager – "agent gestionnaire" in fig. 4 - …).

Once registered, the simulation begins (numbers in parenthesis refers to fig.4 message number).

1- *Supplier Agent*: This agent needs some goods in a certain quantity triggering the SC dynamics to fulfill this need. Firstly it sends a order message (23) to the agent 'production_manager".

2- *Production Manager Agent*: Upon receiving *Supplier Agent* message, checks available space with agent *stock1* (26). If available space can hold the products ordered, production process starts. A command is sent to *Material Stock Agent* (28) to deliver all necessary materials to *Production* agents and *Stock1 Agent* is informed to expect deliveries. In case *Stock1* has no space available, production process is terminated.

3- *Production Agent 1, 2 and 3*: These agents receive materials from agent *Material Stock* (31-38), produce all product parts and deliver them to agent *Stock1* (40-43).

4- *Indicator Agent*: collects information from agents to compute quality, cost…. This process is done through a *Data Source Agent* (44, 49), which gathers data from other agents (e.g. 46).

These communication flows result from the conceptual models, i.e. they describe a business process, as well as how the software architecture enacts these flows considering the software environment in which these agents evolves.

6. Conclusion

In an agent-based SC simulation context, we have presented an organizational oriented methodological framework, for modelling and simulation of Supply Chain organizational aspects. It allows highlighting observables of different level of details while reproducing the SC behaviour according to the desired observables. This methodological framework is structured according to a conceptual and an operational abstraction levels. At the conceptual level, the modelling is based on a Conceptual Role Organizational Model (CROM), which is refined into a Conceptual Agent Organizational Model (CAOM). The operational level, modelling is mainly based on the Operational Agent Model (OPAM).

In this paper, we have focused on the proposal of an open software architecture supporting the transformation of the conceptual model into an operational model by generalizing the previous "hard wired" architecture [9] inspired by previous agent-based integration framework [3]. This architecture can be seen as the interaction between different simulation platforms. We have shown how different types of agents - deliberative and reactive agents - can interact during simulation as well as the role of some service agents (group manager, indicator and DataSource Agent) supporting this simulation. Development is currently based on the interaction between the JADE platform (for the reactive agent) and the JASON environment (for the deliberative agent).

As the modelling cycle relies on model refinement, based on particular simulation objectives, models can not be totally reused. However, because agents constitutes models sub-components, their behaviours may be reused and thus quicken simulation development. As development is still in progress, agents architecture are to be improved (in term of genericity) and then used to simulate a more complex SC case study already defined.

References

[1] Antonio, L., D'Amours, S., Frayret, J.M., "A methodological framework for the analysis of agent-based supply chain planning simulations", SpringSim '08: Proceedings of the 2008 spring simulation multiconference, Society for Computer Simulation International San Diego, CA, USA, (2008).

[2] Boer C.A., de Bruin A. and Verbraeck A. Distributed simulation in industry – a survey. Journal of Simulation, Volume 3, Number 1, March 2009 , pp. 3-16(14) Journal of Simulation, Volume 3, Number 1, March 2009 , pp. 3-16(14)

[3] Bordini, R. and Hubner, J,. An Overview of Jason. Association for Logic Programming Newsletter 19(3). DOI=http://www.cs.kuleuven.ac.be/~dtai/projects/ALP/newsletter/aug06/nav/articles/article5/fs.pdf, (2006).

[4] Bordini, R., Hubner, J., Vieira, R,. Jason and the Golden Fleece of Agent-Oriented Programming. Multi-Agent Programming. pp. 3-37, (2005).

[5] Bruekner S., Baumgaertel H., Parunak HVD, Vanderbok and Wilke . "Agent Models of Supply Network Dynamics: Analysis, Design and Operation, in The Practice of Supply Chain" Management: Where Theory and Application converge, Harrison, Lee and Neal Eds., (2003).

[6] Chatfield D.C., Hayya J.C., Harrison T.P., A multi-formalisme architecture for agent-based, order-centric supply chain simulation, Journal of Simulation Modelling : practice and theory, Vol 15, pp. 153-174, 2007

[7] DMSO: "High Level Architecture", (1998).

[8] Espinasse, B., Serment, J., Tranvouez, E., "An Agent Integration Infrastructure for the Development of Environmental Decision Support Systems based on Simulation", in: AIS-CMS International modeling and simulation multi-conference, Buenos Aires - Argentina. ISBN 978-2-9520712-6-0, (2007).

[9] FIPA, FIPA Contract Net Interaction Protocol Specification, Foundation for Intelligent Physical Agents, www.fipa.org/specs/fipa00029/, (2002).

[10] Gaud, N., Galland, S., Koukam, A., Towards a Multilevel Simulation Approach based on Holonic Multi-agent. Published in the 10th International Conference on Computer Modeling and Simulation (EUROSIM/ UKSiM'08), pp. 180–185, England. April 1–3, (2008).

[11] Govindu, R., Chinnam, R.B., MASCF: A generic process-centered methodological framework for analysis and design of multi-agent supply chain systems. Pergamon Press, Inc., NY, USA, (2007).

[12] Hubner, J.F., Vercouter, L., Boissier, O., Instrumenting Multi-Agent Organizations with Artifacts to Support Reputation Processes, Sixth European Workshop on Multi-Agent Systems, Bath, UK, 18-19 December 2008.

[13] Hubner, J. F., Sichman, J. S., Boissier, O. Developing organized multi-agent systems using the MOISE. International Journal of Agent-Oriented Software Engineering, 1(3/4), 370-395, (2007).

[14] Labarthe, O., Espinasse, B. , Ferrarini, A., Montreuil B., Toward a Methodological Framework for Agent-Based Modeling and Simulation of Supply Chains in a Mass Customization Context, in: Simulation Modeling Practice and Theory International Journal (SIMPAT), vol. 15, n° 2, pp. 113-136, February (2007).

[15] Monteiro T., Anciaux D., Espinasse B., Ferrarini A., Labarthe O., Roy D. , "Chapter 6. The Interest of Agents for Supply Chain Simulation", in: Wiley-ISTE (Ed.), ``Simulation for Supply Chain Management'', C. Thierry – A. Thomas – G. Bel, septembre (2008).

[16] Mustapha, K., Tranvouez, E., Espinasse, B., Ferrarini, A., An Organization-oriented Methodological Framework for Agent-Based Supply Chain Simulation, 4th International conference on research challenges in information sciences, IEEE, Nice, France (2010).

[17] Ounnar, F., Archimède, B., Pujo, P., Charbonnaud, P., HLA Distributed Simulation Approaches for Supply Chain", in: Hermès Science Europe Ltd (Ed.), "Simulation for Supply Chain Management", Hermès Science Europe Ltd, (2008).

[18] Odell, J., Parunak, H.V.D., Bauer, B. 'Representing agent interaction protocols in UML', Proceedings of the First International Workshop on Agent- Oriented Software Engineering, CIANCARINI, P. and WOOLDRIDGE, M. (Eds), (2001).

[19] Rao, A,. S.,AgentSpeak(L): BDI Agents speak out in a Logical Computable Language. In W. Van de Velde and J Perram, editors, Proceedings of the Seventh Workshop on Modeling Autonomous Agents in a Multi-Agent World (MAAMAW'96), Jan. 22-25, Eindhoven, Netherlands, no. 1038 in LNAI, pp. 42-55, Springer-Verlag, London, U.K (1996).

[20] Rao A. S., M. P. Gorgeff. Modeling rational agents within BDI-Architecture.in J. Allen & al Ed., Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning. San Mateo, USA, Morgan Kaufmann, Pub, p. 473-484, (1991)

[21] Rimassa G., Bellifemine f., Poggi A., JADE - A FIPA Compliant Agent Framework, PMAA`99, p. 97-108, Londres (1999).

[22] Serment J., Espinasse B, Tranvouez E., An Agent Integration Infrastructure for the Development of Environmental Decision Support Systems based on Simulation, AIS-CMS International modeling and simulation multiconference, Buenos Aires - Argentina, 2007 ISBN 978-2-9520712-6-0

[23] Shen, W., Hao, Q., Yoon, H. J. and Norrie, D. H. ‘Applications of agent-based systems in intelligent manufacturing: An updated review’, Advanced Engineering Informatics, vol. 20, pp. 415-431, (2006).

[24] Tranvouez, E., Ferrarini A., Espinasse B., Cooperative Disruption Management In Industrial Systems: A Multiagent Approach, 12th IFAC Symposium on Information Control Problems in Manufacturing - INCOM'2006, EMSE, Saint Etienne, Mai (2006).

[25] Vangheluwe, H., et al. An introduction to multi-paradigm modelling and simulation. School of Computer Science, McGill University, Montréal, Canada, (2002).

[26] Vieira, R., Moreira, A., Wooldridge, M. AND Bordini, R. H, (2007). On the formal semantics of speech-act based communication in an agent-oriented programming language. In Journal of Artificial Intelligence Research 29, pp. 221-267 (2007).

[27] Zambonelli, F., Jennings N., Wooldridge, M.. Developing multi-agent systems: the GAIA methodology. ACM Trans. on Software Engineering and Methodology, 12(3), (2003).

7. Appendix: Communication between the agents (JASON and JADE)

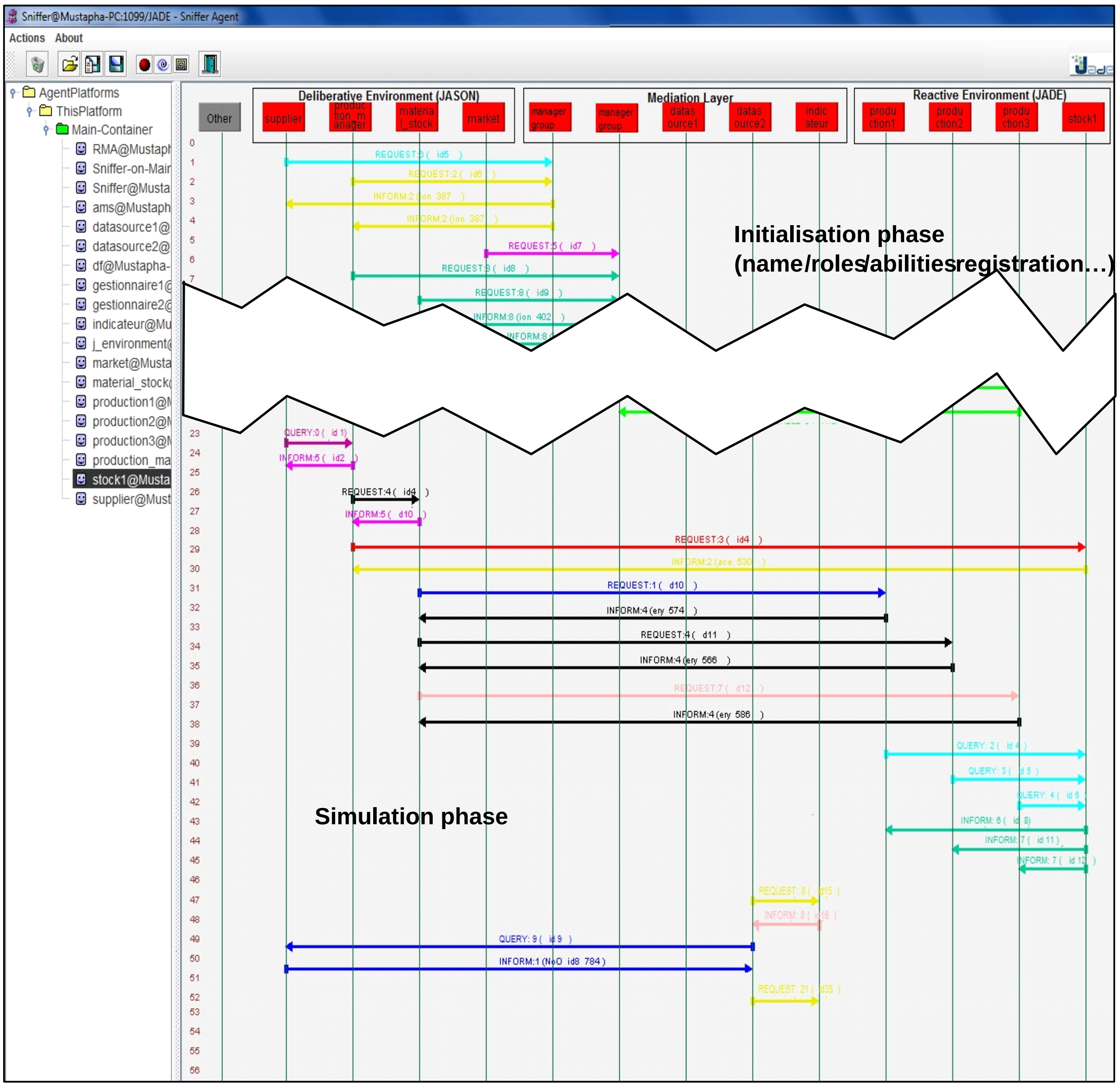

Figure 4. Simulation trace with interaction between Simulation agents and Service agents