K. Mustapha, E. Tranvouez, B. Espinasse, A. Ferrarini (2010), « An Organization-oriented Methodological Framework for Agent-Based Supply Chain Simulation », IEEE-RCIS 2010, 5th International Conference on Internet and Web Applications and Services, Nice, France, May 19-21, 2010.

An Organization-oriented Methodological Framework for Agent-Based Supply Chain Simulation

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

***Abstract* — The SC organizational structure, and related management policies, is a crucial factor that can be adjusted to improve the SC performance, which consequently has to be taken into account in the SC modeling and simulation. This paper addresses a new methodological framework in the context of an agent-based SC simulation, which permits modeling and simulation of such SC organizational aspects, allowing observables of different levels of details. This methodological framework is structured according to two main abstraction levels, a conceptual level and an operational level. For each of these levels, different models are proposed and presented in detail. This methodological framework is associated with a multi model and multi-paradigm software architecture adapted to the SC simulation.**

Keywords-component; Agent Based Simulation, Multi-Agent Systems, Supply Chains, Organization.

I. INTRODUCTION

The Supply Chain (SC) domain is a rich playground for complex studies. The dimensions of the problem are numerous, and the conceptual and architectural challenges, that SC dedicated simulation-based decision support systems raise, imply a heavy workload. The simulation aims at experimenting and understanding (in a controlled environment) the economical, human and environmental consequences of decisions related to the organization, the management policies and the design of the production facilities. We propose to study the efficiency of production organization decisions which supposes to: i) describe the SC organization; ii) model and simulate the behaviors and decisions of its actors and iii) implement these decisions and see their local and global effect on the SC, iv) support each step with specific conceptual and software support. Our work involves a methodological and architectural framework which will assist the SC experts in producing and experimenting with distributed simulation of their SCs based on a multi-agent perspective.

Multi-agent or Agent Based Simulation (ABS) contribution to SC studies is established [1] [2] [3]. Agents are exploited for the design and/or the simulation of complex systems, as autonomous entities have the ability to perform their functions without the need for continuous interaction from the user. Agent-Based Simulation allows the focus on the behaviors of the various SC's actors.

LSIS previous work [4] proposed the basis of a methodological framework for helping SC experts to design their models in their own language (domain models), as well as transitional agent-based models which are used to produce the distributed simulation model on which experiments are conducted. The current work aims at taking into account the impact that an SC's organizational structure has on its performances by providing a methodological framework which support ranges from the domain model analysis to running the simulation. In line with our former works, the methodological framework considers the design phase according to a conceptual and the operational level. This framework has to facilitate the realization of the SC simulation with gradual processes that begin by defining the needs of the user prior to arriving to the implementation of the system while satisfying the initial requirements. This methodological framework requires software architecture that is adapted to the need of SC simulation. The proposed framework should take into account the objectives of different modeling paradigm, as well as heterogeneous (simulation) software environments to coexist. Thus, final users would avoid any loss of previous expertise in modeling language which they have chosen for legitimate scientific and comfort in use. This paper focuses on the main models of this methodological framework and introduces general software architecture.

Firstly, we define in section 2 our research problematic, which concerns modeling and simulation of SC with their organizational aspects. In section 3, we introduce an organizational-oriented methodological framework permitting to take into account these organizational aspects, according to a modeling at a conceptual and an operational abstraction levels. Then in section 4, we present the different models related to the conceptual modeling of this methodological framework: the Conceptual Role Organizational Model (CROM), and its refinement in the Conceptual Agent Organizational Model (CAOM). Section 5 describes the operational modeling of this framework, centered on the Operational Agent Model (OPAM) obtained by translation from the CAOM. Section 6 presents an illustrative example applying our methodological framework to the modeling of golf club manufacturing, with models related to the conceptual and operational modeling. Finally, we conclude by drawing the future step of our research.

II. ORGANIZATION AND SUPPLY CHAIN SIMULATION

Based on agent-oriented approach,as our work aims to take into account organizational aspects of SC, this section develops this research issues in an Agent-Based Simulation context, then exposes some challenges related to ABS. Finally, we briefly

introduce how organizational aspects can be taken into account in an SC modeling and simulation.

A. SC Modeling and simulation problematic and challenges

Our objectives are to propose a SC simulation methodological approach which allows describing an SC organization, identifying the observables and designing the simulation model. Observables are data and information ongoing decision processes, which need to be highlighted in the simulation results for particular study. Therefore, the main goal is to reproduce the SC behavior according to the level of details required to produce the user desired observables. The observables describe simple or aggregated (at different hierarchical levels), values or indicators describing the states of the SC entities or performance as well as processes or decision processes (scheduling strategies, 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, designed to observe periodic changes in 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 compare their status at different dates.

In order to achieve this objective, we consider the following requirements (at a functional or software level) which have to be met [5] [4] [6] [7]:

- *Multi-level modeling*: SC complexity requires describing a SC as composed of different organizational levels with various degree of detail. Each level must focus on its specific observable and behavioral representation needs, but also connect levels between themselves. Therefore, details can go down to the organization of a production cell or a machine efficiency, or at a higher level a transport company transport fleet management or a buying-selling strategy of a company of the SC.
- *Multi-scale simulation*: enacting the former requirement requires to be able to simulate these different behaviors. However, all the components along the different scale of the SC may not be relevant or efficient from a modeling complexity or technical point of view. Therefore, modeling and simulation must allow "pruning" the organization structure and propose meddling different scale of simulation : one production company may be completely described along all its organizational level, whereas another can be summarized in only one simulation entity, and at the same time allows both companies to interact.
- *Multi-paradigm modeling*: Behaviors of the SC entities can be coupled and may require relatively high level of description/modeling capabilities (to reproduce / validate negotiation or planning processes or protocols) or low level of description (simple behavior such as a simple production machine, a truck, etc.). It is related to the above requirements.
- *Managing different temporal scale*: as a consequence of the multiscale simulation, SC entities can either undertake activities in (simulated) real-time (for example monitoring a production machine or a truck) or have longer duration (e.g. a rescheduling process). The simulation must therefore deal with local schedulers (each dedicated to an organizational level or a group of simulated entities) while ensuring consistent global behavior of the SC (in terms of time constraint and causality).
- *Openness* to modeling or simulation *legacy software*: This interoperability is related to the reuse of important modeling and/or learning efforts previously done. It can cover the compatibility with previous research/simulation results.

Because of the SC nature and its simulation requirements, distributed software architecture is needed. Two main approaches are possible:

- Generic (homogeneous) agent based architecture (with dedicated modeling language) [5] [8].
- Coordinate separate simulations (particularly when different paradigms are used) through interoperability mechanisms and protocol as HLA [9].

We are aiming at achieving convergence using the above two approaches, in respects with the previous modeling and simulation requirements listed earlier. First of all, it is constructed by using an organizational oriented individual-based modeling approach that is simple enough to be related to the domain-dedicated modeling language, and also by producing models which afterward can be translated into other modeling paradigm and language. Secondly, it is maintained by proposing an agent based framework that keeps different models and simulations consistent independently of the software environment in which they are implemented in (as in [6] in an environmental decision support context). Moreover, this software framework must be sufficiently open to other simulation software environments. This paper will focus on the first part of the problem and show the main outlines of the second part towards operationalization and software architecture.

B. Related works

ABS allows the understanding of different dynamic models, which are composed of entities with different complexity levels (from very simple entities or reactive agents to more complex such as deliberative agents). Another interest related to the ABS is the facility offered to the modeler to manipulate different levels of representations, such as individuals and groups of individuals. Agent-based modeling allows capturing the dynamic nature of SCs, facilitating the study of numerous resources coordination that is associated with the interaction of multiple companies [3].

Few researchers have proposed a general framework to support both the design and the realization of the SC simulation. We have looked at an agent oriented software engineering methodologies (among those rare Holonic compliant methods) in order to find conceptual and operational solutions, as organizational issues were to be added to the actor approach [7]. Methods like GAIA [15], CRIO [5], or MOISE+ [6] provided part of the solution of our required objectives.

We have established a comparison between the main multi-agent existing methodologies, in particular those incorporating the implementation phase. We have analyses these methods based on the requirements identified previously in subsection A. Table 1 below synthesizes this analysis.

TABLE I. COMPARISON BETWEEN THE METHODOLOGIES

	A: ANALYSIS , **D:** DESIGN, **High :** Support Code Generation , **Medium:** framework 'pattern'						
	Methodologies / Criteria	**CRIO**	**GAIA**	**AGR**	**ADELFE**	**AGENT – ACTOR**	**MOISE (A&A)**
Life cycle	Coverage of the life cycle	A & D	A	A & D	A & D	A & D	D
Design	Organizational model	Yes	Yes	Yes	No	Yes, ***Domain model-NetMan***	Yes
	Multi-level	Yes	Yes	Yes	No	Yes	Yes
	Multi-paradigm modeling	No	No	No	-	Yes	No
	Ontology support	Yes	-	No	No	Yes, Domain model	No
	Graphical notation	UML	UML-AUM	UML	UML-AUML	AUML - RCA	No
Development	Agent Development Envrionment	Yes ***Janus***	No	Yes ***MadKit***	Yes ***Adelfe***	Yes ***Majorca+ Anylogic***	No (not integrated)
	Multi-scale simulation	Yes	No	Yes	No	No	No
	Managing different temporal scale	Yes	No	Yes in Mimosa	No	Partial with Anylogic	No
	Implementation support	High	No	High	High	Medium	Medium (dedicated language)

Most approaches use roles in order to promote the flexibility of the developmental process, even with different abstraction or hierarchical levels. They have exploited the roles to either decompose or contextualize the behaviors of the agents or even add constraints to them (through rules or norms). Moreover, time is not an issue (apart for CRIO) as homogeneous agent granularity or type is mostly involved. When deliberative agents and reactive agents reproduce behaviors of different time horizon, then time synchronization becomes a hard requirement to be identified at the modeling phase and eventually control it at the software level (and maintained at the intermediate translation steps).

The description of the organization must be included from the beginning of the modeling approach, in order to propose the suitable observables of its components, as the organization of the systems pre-exists the agent model. Finally, cooperative behaviors are the basic tools to reproduce cooperation situation in a "real" SC as well as a way to deal with disrupting events, giving it the adaptability to the SC [12]. The deliberative/reactive agent architecture results directly from the need of validating such cooperative behaviors. Previous results presented in [4] did propose meddling different kind of behaviours allowing to study how local behaviors could impact the SC in a whole. However, organizational aspects were implied and only taken into account at a software level. Related observables therefore were not studied from the start as the modeling and simulation process was guided by the nature of the logistic processes (duality physical / decisional) and not the SC organizational structure.

III. A METHODOLOGICAL FRAMEWORK FOR SC ORGANIZATIONAL ASPECTS MODELING

The complexity of the modeling process as well as its implementation, lead us to propose as in [4] a modeling approach based on an incremental structure, in which different models are developed. According to this approach, the real system is first represented by a domain modeling of SCs (e.g. a NetMan model [4], an UEML model [1] etc.) allowing to represent the organizational aspects. The methodological framework which takes into account SC organizational aspect is structured according to two main abstraction levels, a *conceptual level* and an *operational level*. This allows the development of a conceptual and an operational organizational modeling of SC. The different models and the transition to agent-oriented modeling and simulation in our methodological framework are presented in Figure 1. The Problem could be summed up by the modeling and simulation of SCs, which is subjected to the environment dynamics and the scope that encompasses the conceptual and operational modeling phase.

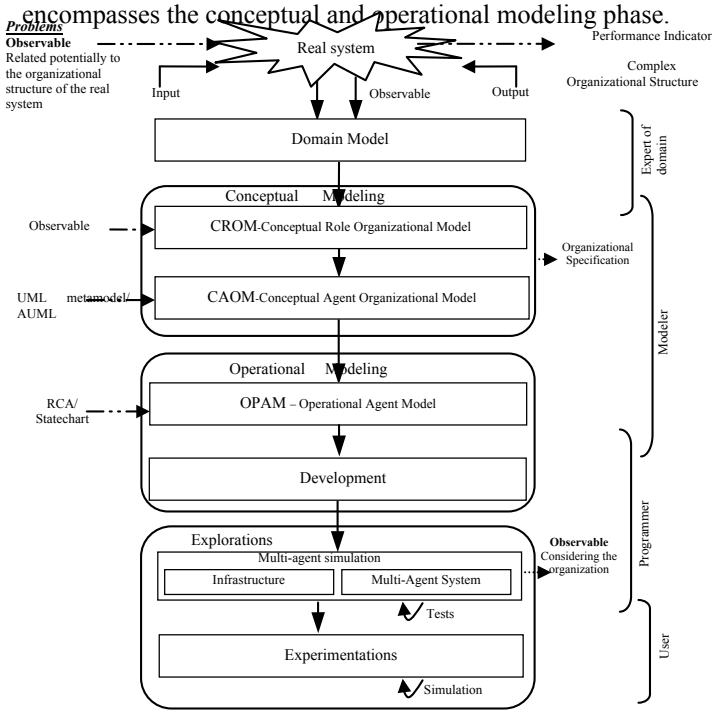


Figure 1. Methodological for the modeling and simulation oriented agents

The *Conceptual* (Organizational) *Modeling* level engages through a dialogue between the domain expert and an agent knowledgeable modeler. Initially, a CROM model (Conceptual Role Organizational Model) is produced by identifying the active entities and their organization from the domain model according to the concept of roles and group. This stage highlights the organizational structure of the SC as wells as the structural and dynamic relations between the entities composing this SC. This model is then transposed into the agent world through a Conceptual Agent Organizational Model (CAOM). This model is defined on the basis of observables which the user needs to obtain from the simulation preparing the implementation of the simulation. The important key of this step is to precisely identify the agents defined at the conceptual level, in order to make them operational according to an Operational Organizational Modeling.

[1] Unified Enterprise Modeling Language - www.ueml.org

The *Operational Modeling* level provides a solution to implement an executable system (simulation). The software designer details the CAOM model by associating a conceptual agent with a software agent architecture (for example BDI) and specifying their behaviors (for example a UML state chart for a reactive agent) and interactions (UML sequence diagram), resulting in an Operational Agent Model (OPAM). The implementation of these models in a simulation(s) environment leads us to the Agent-Based Simulation system which is then executed. The last stage of the conception requires the realization of many tests for the validation of the multi-agents system.

In our previous works [4], 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 prior to implementation it is necessary to describe them to previous levels (conceptual and operational level). A second objective of this work is to propose a business model that is adequately open to different software platforms in order to facilitate the process from translation into implementation. This requires software architecture, multi-model multi-paradigm and a methodological framework to facilitate the construction of simulation of a SC. Next section will focus on investigating all these approaches, concepts and simulation models for multi-agents.

IV. CONCEPTUAL ORGANIZATIONAL MODELING

A methodology should provide an appropriate set of abstractions to identify, develop and describe a problem and propose any potential solutions. The methodological framework proposed covers the dotted area in Figure 1. Then, we suppose an existing domain model, which have to be progressively transformed into a running simulation with heterogeneous observables.

A. Conceptual Role Organizational Model (CROM)

1) Founding concepts

This model allows highlighting the organizational structure of the SC as well as the structural and dynamic relations between the entities that make up this SC. The founding concepts of our modeling approach are defined by a role oriented meta-model based on the existing methodologies evoked in section 3. This meta-model has to precisely define all the concepts used in the development process. Our Conceptual Role Organizational Model or CROM meta-model extends to the Agent Actor model first [4] and then adds the organizational representation capacity. We have integrated the concept of a group included in AGR that incorporates the concept of hierarchy. The recursive description of an organization hierarchy of CRIO helped us to apprehend its implication from a conceptual and architectural point of view. As an interface between the agent and the role, the notion of capacity represents an interface between two adjacent abstraction levels in the hierarchy of the system.

A system is described as a successive hierarchical layer, denoted by level, regrouping a set of roles of the same hierarchical decomposition level of the domain model. A level is in general characterized by a single time horizon (real time …). The CROM model integrates the notions of actor, group, role, service and relation (see example in figure 7):

- An *actor* is an active entity in the organization.
- A *group* represents a set of roles in the organization sharing a common goal (derived from the domain model).
- A *role* represents the functional position played by an actor in the context of a group.
- A *service* is a function performed by the role of an actor [13].
- A *relation* is an interaction between entities.

As these concepts are used to translate SC domain models, a CROM model is associated to domain ontology i.e. in the present case is SC ontology. Such ontology can propose a library of role hierarchy collecting the different roles a particular SC uses, as well as predefined groups type (for example different production service organization …).

2) CROM Modeling concepts

The meta-model in Figure 2 shows how these concepts constitute the building blocks of a CROM model.

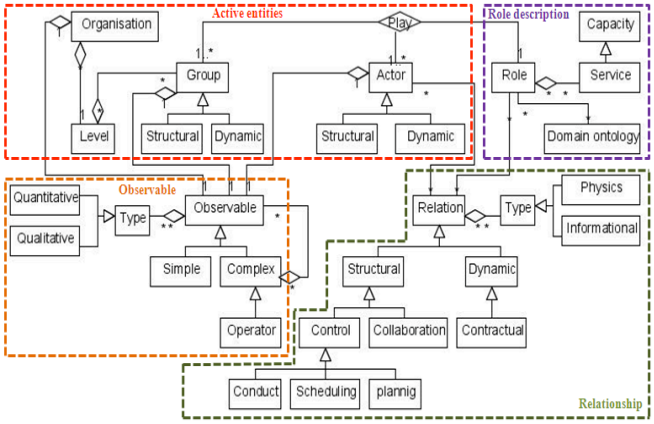


Figure 2. CROM Metamodel

.We consider that an organization is composed of (hierarchical) levels that hold one or several groups, each group containing actors playing roles. An actor plays one role in one group, but can also play the same role in different groups. Conversely, the same role can be played by several actors. Organization, group, and actors can generate observables (quantitative or qualitative). A role provides services to other roles of the same group, while a service may require capacities (as defined in the domain model). Relations connecting actors may hold between actors and/or roles. They represent a flow of information and/or physic (products, semi finished products, raw material…).

There are two type of groups: *structural* (an isomorphic description of the SC organization) or *dynamic* (i.e. characterized by a time duration or a goal shared by actors from different structural groups). Structural and functional relation sub-types relates to the same distinction. A structural

group thus holds only structural relation. The different relations mentioned earlier are detailed in Table2.

TABLE II. EXAMPLES OF RELATION TYPES

Type of relation	Categories of relation	Flow Type	Graphical notation	Description
Structural	Collaboration	information		Represent collaborative process between actors of a same sub-structure such as cooperative scheduling (implies some autonomy)
	Control	information		An actor has (hierarchical) control over another i.e. he can order others to do specific tasks.
	Scheduling	information materials		Organize the realization of tasks, taking in consideration time constraints (deadlines, …)
	Planning	information		Planning is the implementation of objectives over time.
Functional	Contractual	information or materials		General interaction between actors belonging to different dynamic groups. The relation can be limited in time. e.g. trades aspects such as command passing between a. customer and its supplier

The observable (see fig. 3) is characterized by the activity it monitors (productivity, quality, cost…), its quantitative or qualitative nature which requires defining its measuring units and the authorized values (whole or real number if quantity, list of values if qualitative) and finally its dated value.

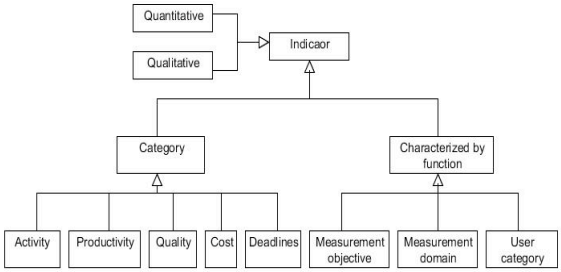


Figure 3. Observables classification

Based on CROM concepts (actors, groups, and hierarchical levels), each level is defined by its groups, their actors and their roles. The roles are characterized by a service that they set in motion. Thus, the structure of the group is defined like a quintuplet: $G_i=\{Ac_i,R_i,S_i,Re_i,T\}$, where Ac_i is the set of actors represented by the group G_i, R_i is the set of the roles played by Ac_i, Si is the set of the services of the roles, Re_i is the set of the relations between the actors, and T denotes the time. Every level is characterized by a temporal horizon {short term, medium term, long term} which the simulation must respect.

B. Conceptual Agent Organizational Model (CAOM)

The Conceptual Agent Organizational Model (CAOM) is a translation of the CROM in "conceptual agent" modeling. Agents are not detailed as they will be described in the design operational agent model (see Figure 1 for more details) according to specific agent architecture. This classical step in Agent Oriented Software Engineering associates roles with the agent according to the chosen agent modeling approach.

1) CAOM Modeling Concepts

CAOM objective is to specify the behavior of each CROM actor. It involves "filtering" the CROM model to only retain actors which have some interest for the simulation as the desired level of detail of their behavior. Thus, it highlights the observable values quantitative / qualitative data relevant to the objectives of system representation

The following meta-model in Figure 3 shows these concepts which form the building blocks of a CAOM model. As synthesized in the meta-model, an organization is composed of (hierarchical) levels that contain one or multiple groups; in which it contains agents playing roles. An agent plays a single role in one group. An agent is capable of playing the same role in different groups and the same role can be played by several agents. Organization, group, and agents can generate observables (quantitative or qualitative).

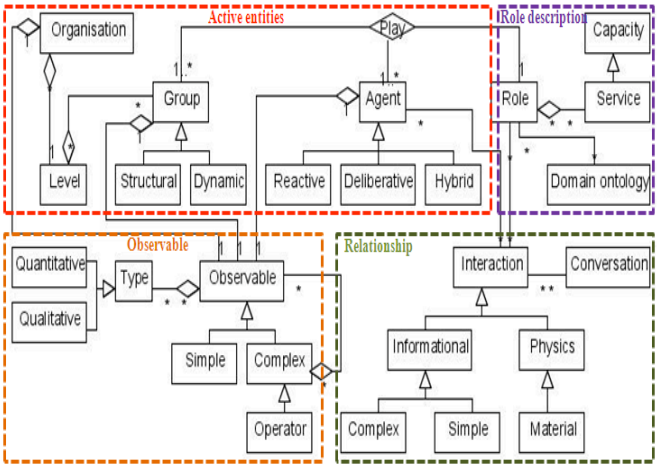


Figure 4. CAOM Metamodel

A role provides services to other roles of the same group, while a service may require capacities (as defined in the domain model). There exists domain ontology for each role. Interactions connecting agents may hold between agent and / or roles. There are two types of groups: structural (it is an isomorphic description of the supply chain organization) and dynamic (i.e. characterized by a time duration or a goal shared by actors from different structural groups). Informational and physics interaction sub-types relates to the same distinction. The different interactions mentioned earlier are detailed in Table 3.

TABLE III. INTERACTIONS TYPE

Type of interaction	Categories of interaction	Graphical notation	Description
Physical	Material		Exchange of material between actors is represented by the reactive agents. i.e. material delivery (Stock Truck)
Informational	Simple		Simple exchange of information to achieve tasks, i.e. allocation of tasks, knowledge sharing.
	Complex		Suppose that agents must coordinate their actions in order to provide all their skills to solve more complex tasks. For example, industrial activities that require a distributed approach, such control systems, design and manufacture of industrial products, distributed control. [13]

As a CAOM model is defined by a set of agent, groups and levels (themselves defined by groups and their agents), the structure of the group can be defined as the quintuplet:

$G_i=\{A_i,S_i,I_i,T\}$,where A_i is the sets of agents belonging to the group G_i, S_i is the set of the services of the agents, I_i is the set of interactions between the agents, and T denotes the time simulation identifying the time scale {short term, medium term, long term} as defined in the CROM. This model is inspired by AGR [13] and CRIO [5].

The principle translation task is to decide about the role that should be included in the CAOM model. Roles can be combined into one or several agents, according to the kind of behavior which is expected to be studied (simple or "intelligent" machine, workshop global or internal behavior…). Table 4 summarize different criterion used to decide on the translated method of a role in a CAOM model. For example, in the Agent-Actor model "mechanical" roles are described with reactive agents, (implemented in AnyLogic[2]) whereas role with complex behaviors are enacted by deliberative agent (Majorca platform [4], [14]). In the case CRIO agent/holon architecture is chosen to support the CAOM model, then roles can be described with hybrid agent.

TABLE IV. ACTORS TO AGENTS

CROM	CAOM	
Roles of the actor	**Agent <Type>**	**Expected behavior (translation criteria)**
	Reactive agent <RA>	If simple behavior is required, a stimuli-response behavior type is sufficient. This can be described later on with a UML state chart diagram. (for example a production machine)
	Deliberative agent <DA>	If decision-making and negotiation is needed then capacities will require a deliberative agent to perceive its environment and other agent behavior. Example: production manager (CROM Example)
	Hybrid agent <HA>	Reactive and deliberative behaviors are required. For example an "intelligent" machine capable of cooperating with other machines when disrupting events while occurring.

The second task of this translation to CAOM model concerns the transformation of the relationship between the CROM actors. Thus, CROM relationships are transposed in agent world as interactions while keeping their classification (cf. table 3).

Figure 4 shows an example of a UML sequence diagram used to model the communication between agents during the realization of a cooperative disruption resolution strategy. Initially, Company 1 sends a request to cooperate with Company 2 and Company 3 (e.g. sending the deadline of receipt or task delivery after using stocks) (Figure 4 (1)).

Proposed solutions results from the cooperation process. If the reception message is associated "with impossible switching task" then the proposed solution should be removed from the list of tasks. If sending a message requires time then "Stop calculations", in case of reception of the last message "End of calculations" and then begin the decision phase (Figure 4 (2)).

[2] www.xjtek.com

"Decision phase" consists in selecting an option from the list of potential solutions then sending message "Switching confirmed with the task" with single resolution adopted. If the message is received "Permutation confirmed with the task" then if it is always valid, it will then send message "confirmation of cooperation with the task" if not, it is terminated by "Cooperation impossible" (Figure 4 (3)).

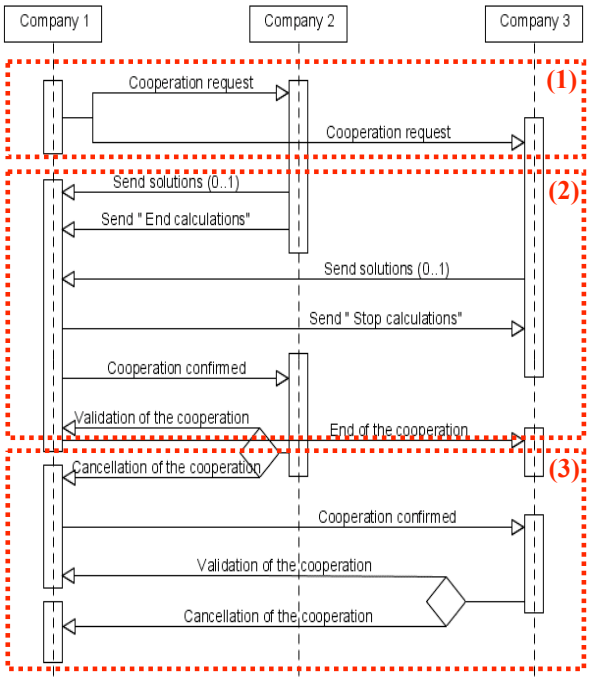


Figure 5. Communication modeling

V. OPERATIONAL MODELING

This section provides a description on the necessary tasks used to create the operational agent model. An approach to transform the conceptual model (CAOM) to an operational model is presented. Also, we propose an agent architecture which defines the composition of the interacting agents' environments, cognitive agent environment and the environment of the reactive agents.

A. Operational Agent Model (OPAM)

The operational model provides a solution for implementing of the conceptual model (CAOM). This step has led to the development of an operational model agent including the choice of agent architectures.

For the representation of agents and their behavior at the operational level, we propose a modeling approach which allows differentiation between agents which is guided by the observables. This modeling approach is based on two environmental agents, one for reactive agents and the other for the cognitive agents.

Agents present in the cognitive environment act independently to achieve their goals. They have an explicit representation of the environment and have reasoning abilities. The cognitive agents can play several roles in the multi-agent system by the implementation of multiple plans. Agents in the reactive environment act in response to environmental stimuli.

1) Specifying the behavior of agents

The development of the model OPAM consists of a comprehensive description of the behavior of agents and their interactions content. The choice of agent architecture has been made at the conceptual level to specify how they could ensure roles and services (i.e. plans which allow them to perform this role).

Each conceptual agent is represented by an agent model which consists of two software architectures, reactive and cognitive. These distinctions, in terms of software architecture of agents, lead us to consider two agent platforms. Each platform includes the same type of agents which have specific simulation environments.

The design of the agent business model defines the agent models involved in the multi-agent system. This modeling step includes the specification phase of agents and relies on the formalisms of the adapted representation. It aims to represent the behavior of agents; therefore we retain the use of two specification behavior formalisms. The behavior of reactive agents is specified using the modeling language AUML (Agent Unified Modeling Language) [15]. The cognitive behavior of the agents is specified using the RCA formalism (representation of the behavior of agents [14]). An illustration is given as part of a re-scheduling in Annex.

2) Specifying the interactions between agents

The interaction between cognitive and reactive agents within the Operational Agent Model is formalized by sending and receiving messages. The refinement of the interactions description is to characterize the type (message versus signal) and the content of these messages. The interactions between agents can be declined by three possibilities: (i) the interaction between cognitive agents, (ii) the interaction between cognitive agents and reactive agents, defined in terms of accountability relationships, and (iii) the interactions between reactive agents.

TABLE V. INTERACTION DESCRIPTION

Type of interaction	Graphical notation	Description
Message	◄——►	Cognitive agents communicate by exchanging messages. Each cognitive agent is associated with an instance of the Jess inference engine.
Signal	·····►	Reactive agents interact by exchanging signals.

The interactions between cognitive agents are defined as messages. While the interaction between cognitive agents and reactive agents require a transformation of the relevant messages to the two chosen environments for the development and implementation of the system (using sequence diagrams defined in UML). As for the interactions between the reactive agents, it is identified in terms of signals. Table 5 details the different types of interactions.

B. Software infrastructure

The simulation of the operational model, which was produced after several stages of initial conceptual model refinement, assumes the existence of a software infrastructure that supports both production models (ontology). In addition, it ensures the integrity of the distributed simulation (of two software environments) while providing the desired simulation data (observable).

The design of a simulation of SC can be based on a methodological approach coupled with a dedicated software infrastructure. In terms of the software architecture agents, we were led to consider two environments, where each environment contains the same type of agents with specific simulation environments. Initially, we present the development environments retained for successful implementation of cognitive agents and reactive agents. Then, we describe the general architecture of infrastructure simulation integrating platforms for development agents.

1) Selection techniques: distributed simulation

The design of multi-agent system requires the use of implementation languages in order to program agents and their organization. This work is performed by the software engineer (Figure 1) and relies on the exploitation of computer languages related to platforms of a specific development. For the environments implementation of cognitive and reactive agents, we consider two development environments that ensure the inclusion of specific agent architectures, MAJORCA and Anylogic.

For the development of cognitive agents, we choose MAJORCA, a platform that provides a development environment for multi-agent systems based on the concept of behavioral plans specified using the RCA formalism. [2]. The implementation phase of reactive agents concerns the programming of these diagrams for modeling and simulation. Therefore, we have used the software Anylogic which is a software platform for development and simulation of complex discrete systems, continuous and hybrid. Anylogic offers an environment of discrete event simulation developed in Java.

The choice of software Anylogic was further motivated by the possibilities offered in terms of interoperability (Java classes, databases, etc...). It incorporates the notion of time and favors as the relation between cognitive agents and reactive agents.

2) Architecture of the simulation environment

The proposed simulation environment oriented agent is based primarily on interoperability between two platforms. It also defines the accessible interfaces of users to design models, scenarios, visualize the evolution of the system and its components and finally to use simulation results. Based on the above, we propose a simulation architecture environment consisting of the following elements: (i) MAJORCA for the implementation of cognitive agents, and to instantiate a Jess inference engine for each agent defined in the operational model (ii) Anylogic for the execution of reactive agents, (iii) Mediator to ensure interoperability, and (iv) a database to record the parameters of the backup scenarios and simulation results.

The database provides the interface between Anylogic and MAJORCA user with concern of exploitation of model data. The database is used to capture the model parameters, record simulation data and display the results. Figure 5 illustrates the

general architecture of the simulation platform of the oriented agents.

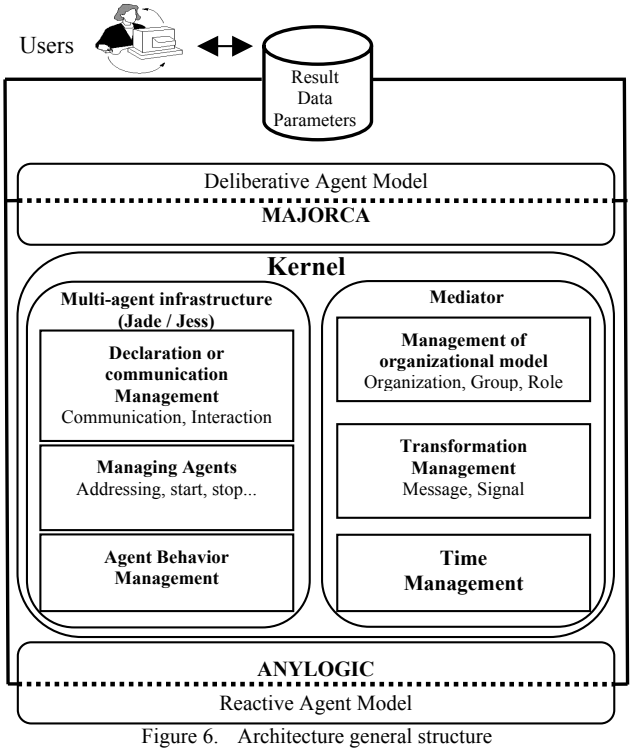


Figure 6. Architecture general structure

Our Mediator provides a set of essential services to interoperability. These services are grouped into six categories:

TABLE VI. SERVICES DESCRIPTION

Services	Description
Management of organizational model	This module manages organizations. It also provides mechanisms for acquisition and release roles.
Agents management	The kernel also provides all necessary tools to manage the lifecycle of agents: addressing, start, stop, etc...
Transformation management	Message processing vs. signal and signal vs. message.
Declaration or communication management	Services that enable the environment to communicate their data and the offered interactions.
Time management	Services concerning the time management and in particular the synchronization between different platforms.
Agent behavior management	This module provides an instrumentation-based sensor enabling an observer role for another role. It is a model of concurrent execution.

This architecture can be considered as an organizational architecture for the implementation and simulation of complex systems. Compared to the previously proposed architecture in LSIS, we propose a generic architecture which ensures interoperability between two or more simulation platforms (through a mediator). Moreover, modules have been added to manage explicitly the organizational model and the time synchronization between the different platforms.

VI. ILLUSTRATIVE EXAMPLE: THE GOLF CLUB MANUFACTURING

The example presented is concerned with the description of a manufacturer responsible for assembling golf clubs.

A. Implementation of the methodological framework

This section illustrates the implementation of Agent-Oriented Methodological framework on industrial case. This illustration is obtained by presenting models that are derived from various phases of design. We describe the Conceptual Role Oriented Model, Conceptual Agent Operating Model and Operational Agent Model. Each of these models is applied to an industrial case [4], the results of various modeling phases associated with methodological framework leading to the conduct of simulation.

A supply chain consists of several roles within the organization. Seven main roles were identified and need to be defined before proceeding further into the model SC. Below is a description of each role:

- Producer: Manufactures modules or parts with operations on materials;
- Processor: Performs operations on products or parts according to customer specifications;
- Assembler: Assembles products or modules from parts or modules delivered by suppliers;
- Customizer: Perform, custom commands from products and modules;
- Distributor: Receives, stores, prepares and ships products;
- Retailer: Performs commercial acts, receives, stores and prepares to meet product orders or requirements of customers.
- Transporter: handles goods between the centers under the orders of customers.

1) An Illustrative CROM model of a SC

The following example illustrates how a CROM model is used to describe the organizational structure of a given SC. In this example, the structure is divided into 3 levels. Each level consists of one or more group (structural or dynamic) of actors. In the first level, Company 1, Company 2, Company 3 and Company 4 are four actors connected by a collaboration relationship; each of these companies is represented by a group of actors playing roles at different hierarchical level.

For example, Actor Company 1 can negotiate with Actor Company 2 at the N1 level, while at level N2 it checks their respective production/assembly capacities with their production/assembly manager. The last actors take short term decisions on this level but they are responsible to enact these decisions and thus control in real time their execution on the third level. In the given example, a VMI (Vendor Management Inventory) process is described where company 2 uses company 1 as a stock resource when needed. Whereas the stock actor belongs to company 1 (structural relationship) plays the same role <Stock 1> in the dynamic group constituted with the assembly actor from company 2. Control relationship

specifies the flow of information with actors use in order to accomplish their objectives.

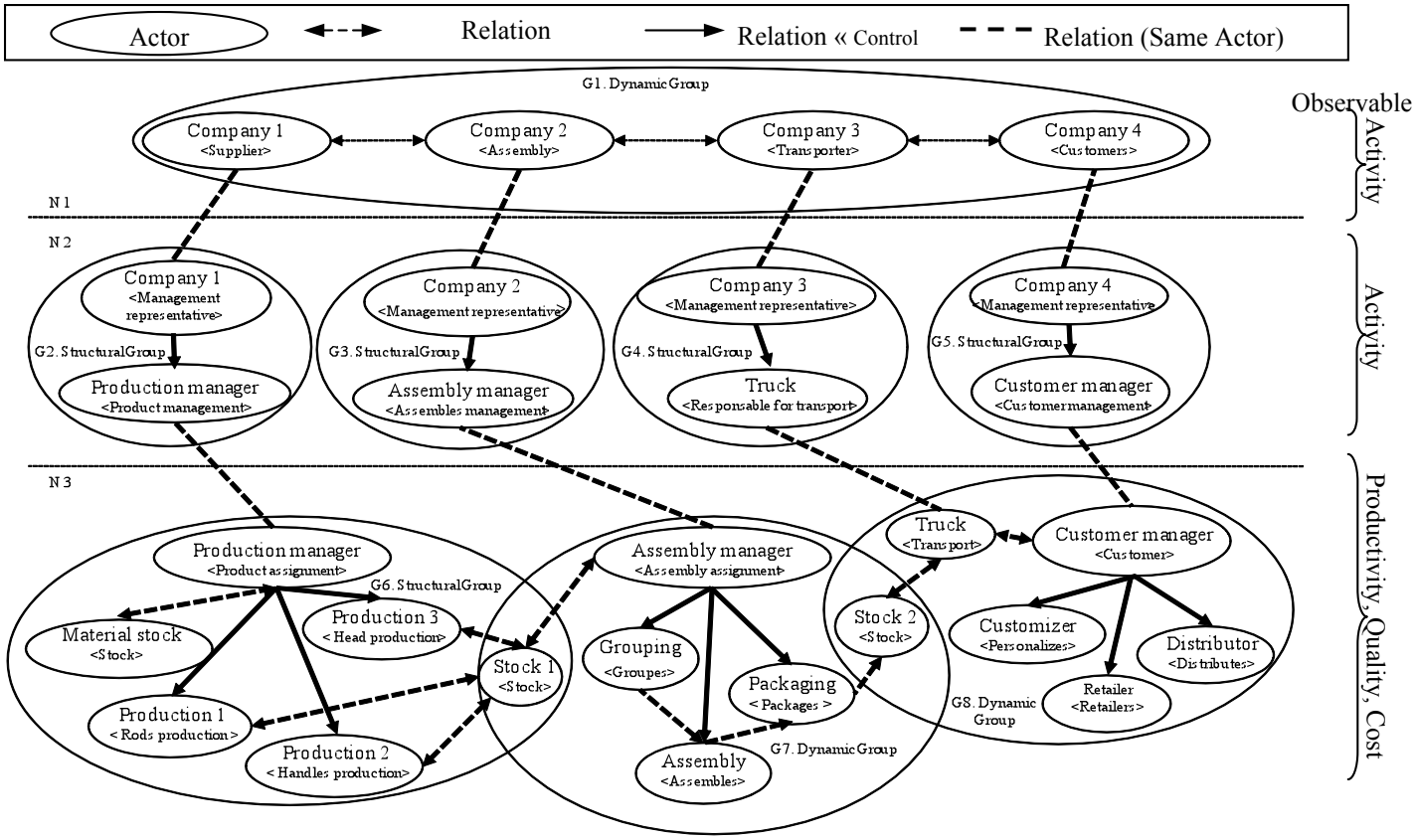
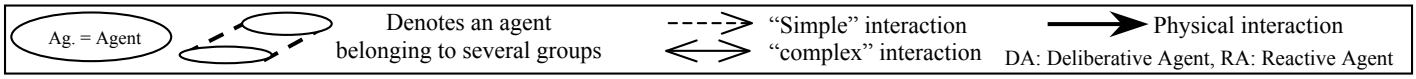


Figure 7. A CROM model of a Supply Chains

2) An Illustrative CAOM model of a SC

The following example, given in figure 8, illustrates a CAOM model describing the organizational structure of a given SC with reactive and deliberative agents.



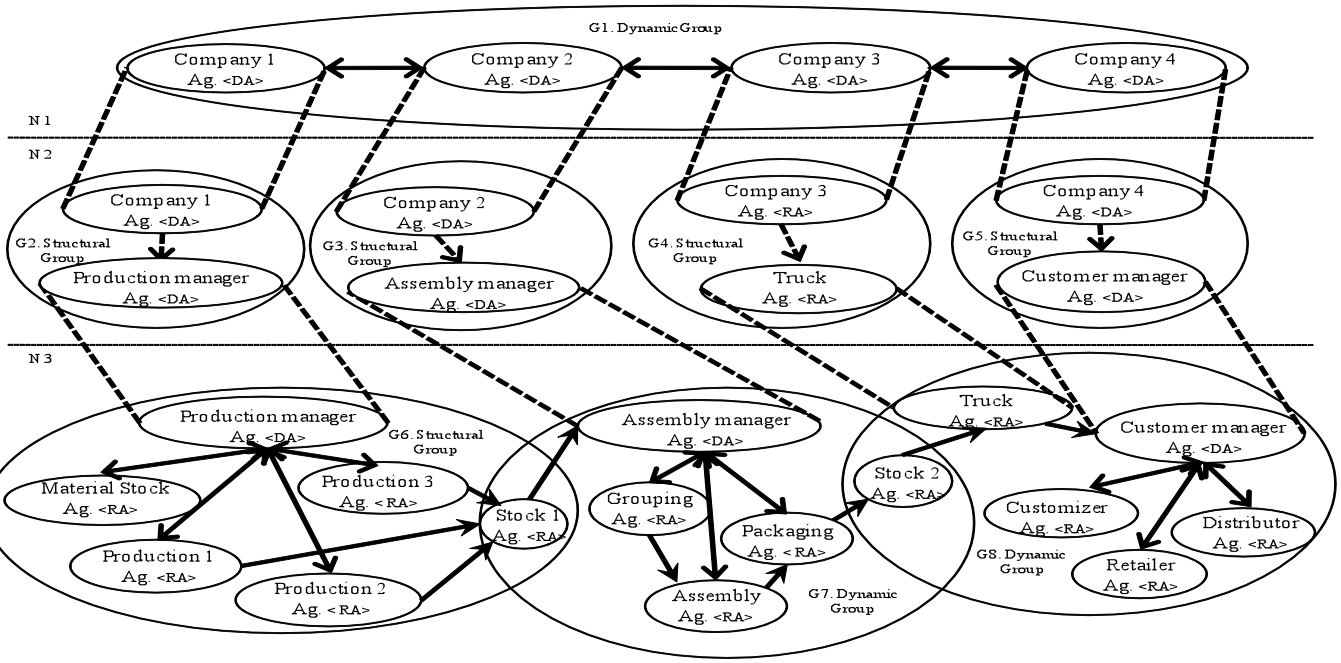


Figure 8. A CAOM model of a Supply Chains

The structure is divided into 3 levels. Each level consists of one or more group (structural or dynamic) of agents.

. In the first level, Agent Company 1, Agent Company 2, Agent Company 3, and Agent Company 4 are four agents connected by an interaction (complex). Each of these companies is represented by a group of agents playing roles at different hierarchical levels... Also, a VMI (Vendor Management Inventory) process is described, where Company 4 relies upon Company 3 for its products transportation <Tuck agent>. Company 3 has no physical stock through its owner but manages virtually the <Stock 2> capacity of the Company 2. Physical interaction specifies the flow of information with agent use in order to accomplish their objectives.

3) An Illustrative OPAM model of a SC

Figure 9 gives an example where the structure is divided into two environments, one environment for cognitive agents and the other for reactive agents.

For example, Agent Company 1 and the Agents Production manager are connected by an interaction through messages (deliberative environment). Agent Company 3 is a hybrid agent since it plays two different roles, one cognitive in the cognitive environment and other reactive in the reactive environment.

Deliberative Environment

Reactive Environment

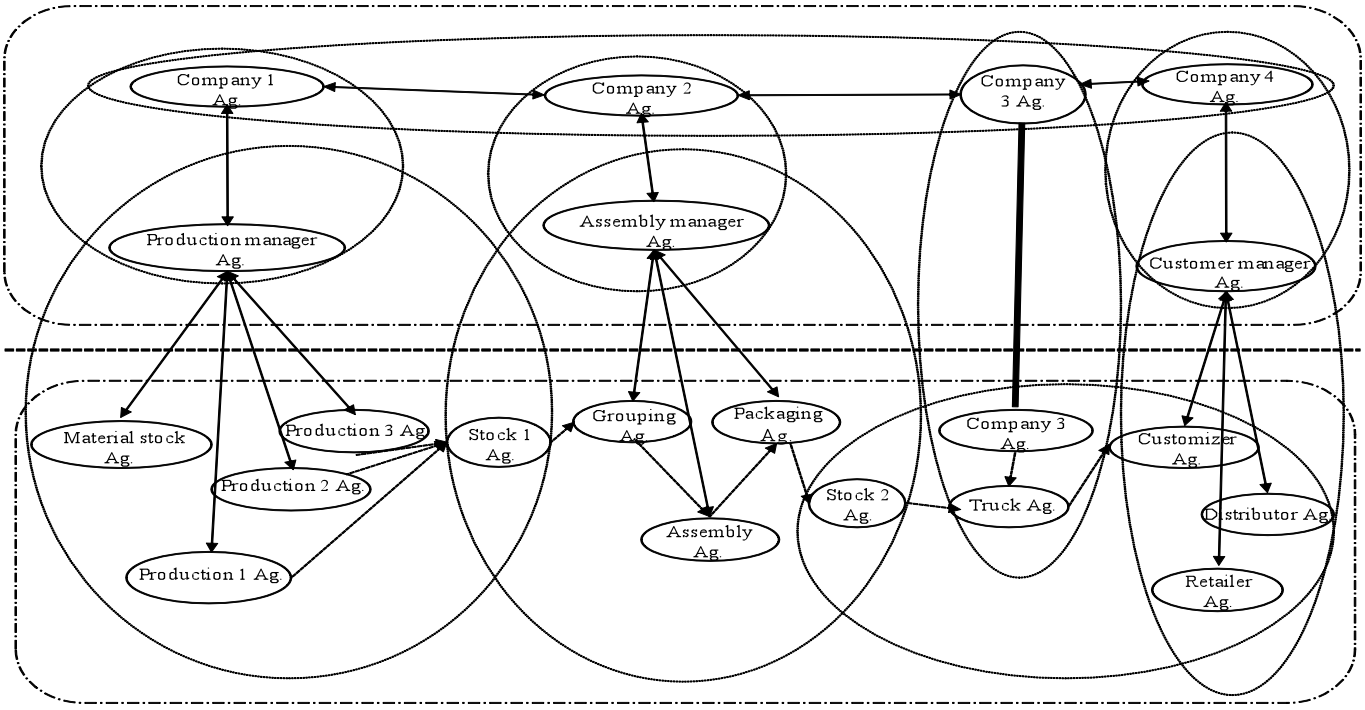


Figure 9. An OPAM model of a Supply Chains

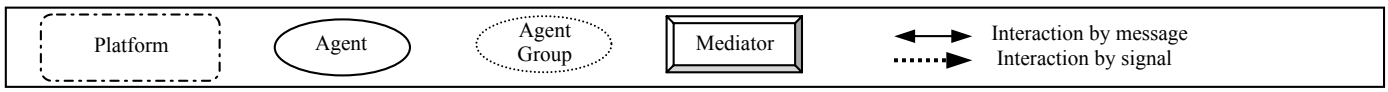
4) An Illustrative architecture of SC

Figure 9 illustrates a software architecture supporting the previous models. It is divided into two platforms, one for deliberative agents (MAJORCA) and the other for reactive agents (ANYLOGIC). The communication between MAJORCA elements is done through messages while the communication between ANYLOGIC elements is done through signals. Therefore, there exists a mediator layer which has the role of transforming messages to signals and vice versa.

Company 1 agent (Production) communicates with Production manager agents in order to perform a production (Produces). Production manager agents communicate with Production Agents through the mediator. Company 3 agent (Transporter) transports the product through Truck agent which transports the product from Stock Agent to Company 4 agent (Customer).

5) Conclusion: towards simulation

This work proposes an open software architecture; this architecture will integrate two different simulation platforms permitting a distributed simulation, based on two specific simulation environments, one for cognitive agents and one for reactive agents, using a mediator to ensure their interoperability, the organizational and time managements.



MAJORCA

Mediator

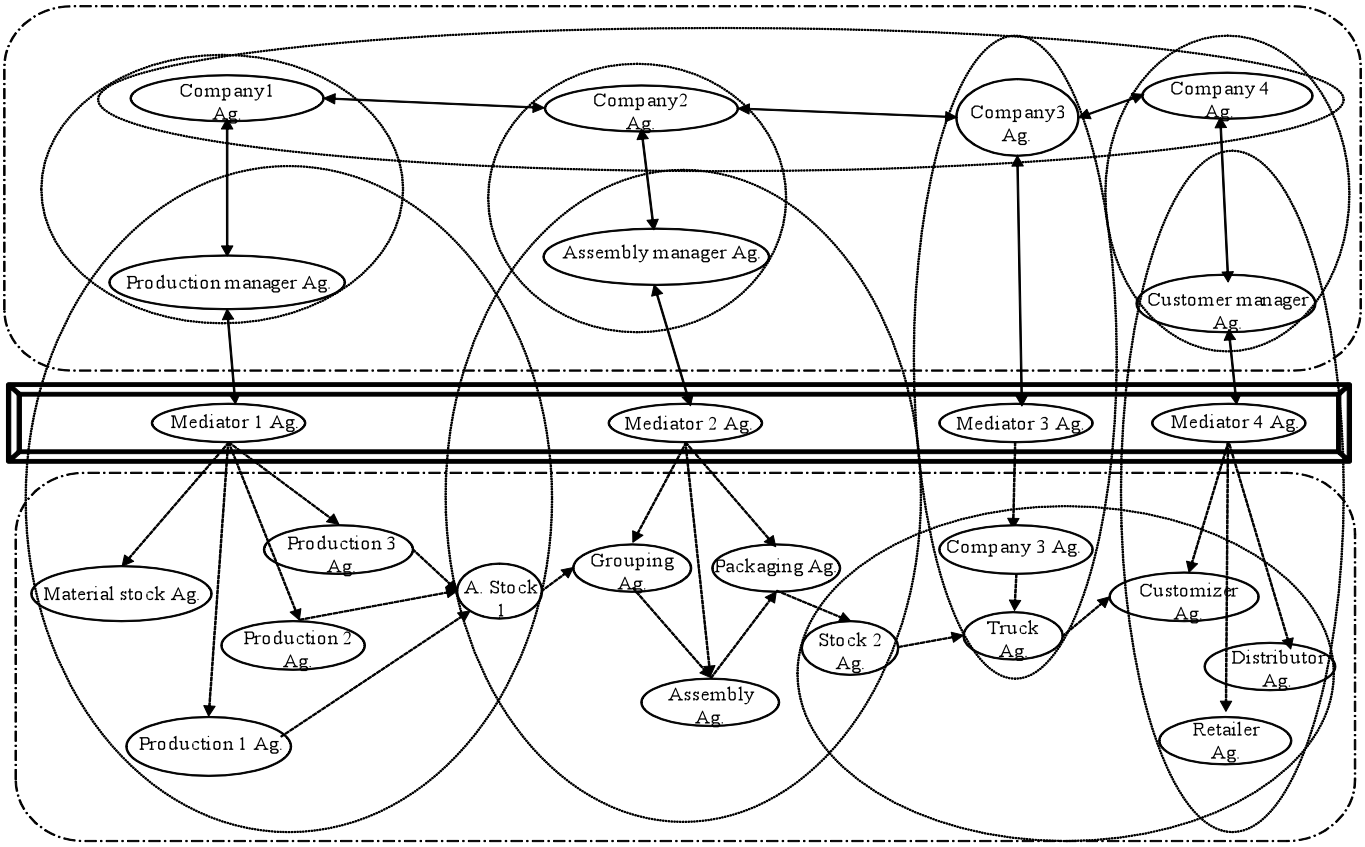


Figure 10. Software architecture of a Supply Chains

VII. CONCLUSION

In the agent-based SC simulation context, we have presented in this paper an organizational oriented methodological framework, which permits modeling and simulation of SC organizational aspects. It allows observables of different level of detail while reproducing the SC behavior according to desired observables. This methodological framework is structured according to a conceptual and an operational abstraction levels. At the conceptual level, the modeling is based on a Conceptual Role Organizational Model (CROM), which is then refined into a Conceptual Agent Organizational Model (CAOM). At the operational level, modeling is mainly based on the Operational Agent Model (OPAM).

This framework has to permit the study of the impact of a specific SC organizational structure and its related management policies on SC performance. Based on a SC expert modeling of a particular SC, an organization/role oriented (CROM) and an agent-oriented (CAOM) conceptual model helps in designing a simulation model, which will reproduce the SC global and local behaviors. These conceptual models are defined independently of particular agent architecture or even on specific software architecture but propose transitional steps to guide their development.

Current work is looking forward at defining translation rules from CROM to CAOM model, taking into account the type and level of details of desired observables, while respecting the organization structure and the temporal constraints in which different time horizons produce. Future work will propose an open software architecture supporting the transformation of the conceptual model into an operational model by generalizing the previous “hard wired” architecture [7] inspired by previous agent-based integration framework [2]. This architecture can be seen as the interaction between different simulation platforms, an interaction that can be resolved in two ways: by means of signals and a mediator.

Different types of agents will be used: deliberative agents and reactive agents. Their development will be based on the interaction between the Anylogic platform (for the reactive agent) and the JADE / JESS environment (for the deliberative agent).

VIII. References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

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

[8] 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.

[9] 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.

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

[11] 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

[12] Tranvouez, E., Ferrarini, A., "MultiAgent Modelling of Cooperative Disruption Management in Supply Chains", in: IEEE – International Conference on Service System and Service Management (ICSSSM'06), Troyes, France, October 2006.

[13] Ferber, J. & al. Towards an integral approach of organizations in multi-agents systems: the MASQ approach. Semantics and dynamics of organizational models in Virginia Dignum, 2009.

[14] 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, Ecole des Mines de St Etienne, mai 2006.

[15] Odell, J., Parunak, H.V.D. and 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

Appendix