METHODOLOGICAL APPROACH FOR DISTRIBUTED SIMULATION : GENERAL CONCEPTS FOR $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$

Stphane Galland, Frdric Grimaud and Jean-Pierre Campagne
Systmes Industriels Coopratifs laboratory[*]
Ecole Nationale Suprieure des Mines
158, Cours Fauriel, Saint-Etienne, 42023 Cedex 2, France
e-mail: {galland,grimaud,campagne}@emse.fr

Additional reference: In *14th European Simulation Multiconference*, eds. Rik Van Landeghem, Society for Computer Simulation International (p. 77-82). Ghent, Belgium, May 2000. ISBN 1-56555-204-0.

KEYWORDS

Distributed simulation, Multi-agent systems, Methodological approach, Life cycle

ABSTRACT

We are located in the context of the simulation of complex industrial systems and distributed in geographical, decisional and informational term.

This paper is divided into two parts. In the first part, we position us within the studying framework of a modeling and specification methodology suggested by (Galland et al. 1999). We briefly expose the problems as well as the considered solutions. In the second part we present general concepts attached to our methodological approach $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$: a simple approach for distributed development and a simulation model life cycle. We finish this part by some explanations on the four major life cycle phases : analysis, specification, conception and implementation.

1 INTRODUCTION

Simulation is a recognized tool and is adapted to modern industrial problems. It takes into account the dynamic aspects during production system behavioural studies. The physical, informational and decisional distributions are seldom managed within same tools. Moreover modern simulation tools are rarely accompanied by complete and adapted methodologies. In (Galland et al. 1999), we propose a multi-agent approach for simulation ($\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$ [1]),which taking the three distribution aspects into account. Thus we iteratively develop the various properties of our methodology and its simulation environment.

In this article we present the general concepts used by $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$. We think that a methodology integrating physical, informational and decisional distributions of simulation models must be itself distributed *i.e.*, submodel developments can be carried out in parallel. This aspect is made possible by the use of simple project management rules. The simulation model life cycle is divided into two parts. The first one is composed by the four traditional methodological phases: analysis, specification, conception and implementation. The second part is a set of specific simulation project phases : experimental plan designs, experiments and result validation. We wish to concentrate our work on the four first stages. Indeed, we think that our contribution (management of physical, informational and decisional distributions) is concentrated on those.

The following section is a short outline of the context in which our work is located. In the section 3, we present general concepts of our methodological approach: distributed development, life cycle and analysis, specification, conception and implementation stages. Finally we conclude and expose our perspectives.

2 CONTEXT

The advent of the simulation technics brought to the industrialists the possibility to model the behavior of their systems in much more realistic ways. Indeed other existing technics *e.g.*, the arithmetic approach, operational research or the computer-assisted production control; make it difficult to support all the behavioral aspects of an industrial system. Simulation is one of the rare approaches integrating the dynamics of a system and usable in an industrial context.

However we would like to concentrate our work on the following problems: the integration of distribution since the first stage of the model's construction, the fact that accompaniment of simulation tools by a methodology is still rare and the need for modularity and reutilisability of the aforementioned models.

To partly solve the problems of the decentralization of information and knowledge as those of times in modelling

[*]SIC : Cooperative Industrial Systems
[1]**M**ulti-**A**gent **M**ethodological **A**pproach for **S**imulation

and simulation, the scientific community has considered the implementation of distributed simulations according to two major approaches: data-processing model *e.g.*, synchronization problems (Filloque 1992) and knowledge distributions *e.g.*, company representation in a world context (Burlat 1996). Distributed simulation, which is composed of these two axes, makes it possible to take into account the international characteristics of companies *i.e.*, problems of technical culture, knowledge and geographical distributions can be supported by distributed simulation models. However this field has not become fully developed yet. Indeed no methodology is truly adapted to these aspects.

We propose to conceive a methodological approach based on the multi-agent concepts. We use the vowel approach (or AEIO) defined by (Demazeau 1995) : a multi-agent system is defined according to four major axis : **A**gents, **E**nvironment, **I**nteractions and **O**rganization. Multi-agent systems enable to support the three distribution aspects of an industrial system :

physical distribution The autonomy and the interaction capacities of agents allows to carry out at the same time the distribution within a data-processing network, and the distribution of the various industrial system parts by associating each agent with the one of them;

informational distribution The cognitive and interactional agent capacities make it possible to distribute information;

decisional distribution The cognitive mechanisms composing the agents permits to set up the decision-making processes.

Moreover the modularity generated by the use of multi-agent systems allows us to answer another crucial point : the reuse of knowledge and already installed tools in industrial companies.

These various points led us to propose the methodological approach $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$ whose realization, illustrated by the figure 1, is iterative (Galland et al. 1999). This approach permits to make gradually evolve our methodology and our simulation environment. We present the first results of the $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$ development in the following section.

3 MULTI-AGENT METHODOLOGICAL APPROACH

In this section, we expose the first results of our work on $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$: the support of the distributed development, the simulation model life cycle and its various major phases.

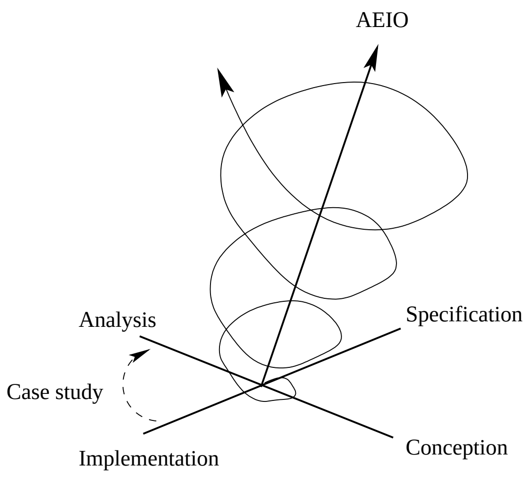

Figure 1: Problem solving evolution

3.1 Distributed Development

The concept of distributed modeling is introduced by implementing several parallel development processes.

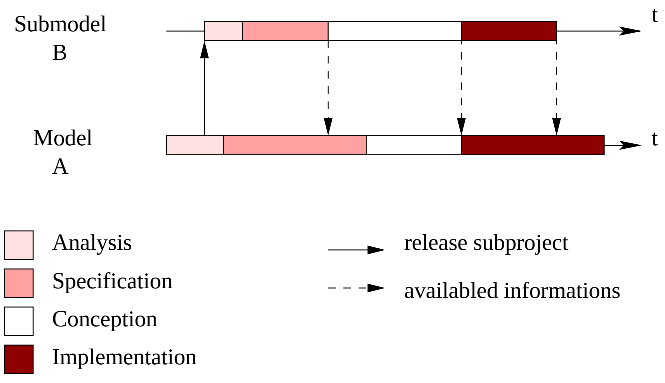

Figure 2: Example of distributed modelling with $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$

For example, let us take the modelling process illustrated by the figure 2. During the requirement specification of the main model A, actors realize that it is necessary to use a distant submodel B. Then the modelling process of B is started and the two projects are developed in parallel. The principle is that a phase of the project A cannot be finished if the same phase of B did not arrived in its term too *i.e.*, developed and validated.

Informations placed at the disposal of the distant models, have various types and are in function of the phase:

- an “distant” analysis must have a description of elements, which are shared;
- the “distant” conception must place at the disposal descriptions of agents associated sharable elements;
- the implementation phases of the various distributed models are completely independent. Only one con-

straint is necessary : all running simulation models must be finished before the experimental phase.

3.2 Life Cycle

In this section, we define the life cycle used by $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$. Let us attach to the definition for only one development team. The figure 3 illustrates the simulation model life cycle. It is closed to the "water-fall" definition (Sommerville 1998) and is especially based the Conical Methodology (Nance 1981). In addition to analysis, specification, conception and implementation phases, we bring specific simulation characteristics : experiment plan design, experiments and simulation result validation. We focus our work on the first four phases. Indeed, we think that our contribution, based on the multi-agent systems and UML, appears during these phases. The Petri net formalism, used in the figure 3, permits to represent the behavior of our methodological approach. By associating each mark to a development team activity, we can model the distributed developments of the simulation models. Our life cycle allows returns to previous stages in case of validation failures (transitions t_{12}, t_{13}, t_{14}, t_{14}, t_{15} and t_{16}).

Until this point we did not clearly introduce multi-agent systems and UML into our life cycle. The figure 4 presents dependences of the various produced documents with respect to multi-agent system and UML concepts. Our approach is based on the use of a UML metamodel. It defines the modelling elements, which are specific to production systems, as well as a formalism. This metamodel is used and respected by all the componants of our methodology. $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$ produces a set of documents specific to each life cycle phase. A multi-agent oriented UML model is built during the *modelling*. It is the formalized translation of informations contained in the requirement specification. It uses the modelling elements and the formalism defined in the UML metamodel. The main characteristic of the producted UML model is the respect of the structural constraints imposed by the metamodel *i.e.*, the using coherence of distributed modeling elements cannot be carried out directly. To resolve these problems, we define a specific multi-agent system. During the *checking* phase, this system checks the multi-agent oriented UML model. If no anomaly or error were detected, the checking system produces a checked UML model. This last is not an instance of a multi-agent model. It is made up only of conceptual modelling elements. The semi-automatic generation of the multi-agent simulation model is carried out by the application of translation rules and the use of preset software components (cooperation protocols, production facility agents, *etc.*). This *conception* permits to completely define a multi-agent model dedicated to simulation and independent of any platform and tool. User software components correspond to definitions of modelling elements and are defined by development teams *e.g.*, decision-making processes could be modeled with this kind of componants. The last phase of the model

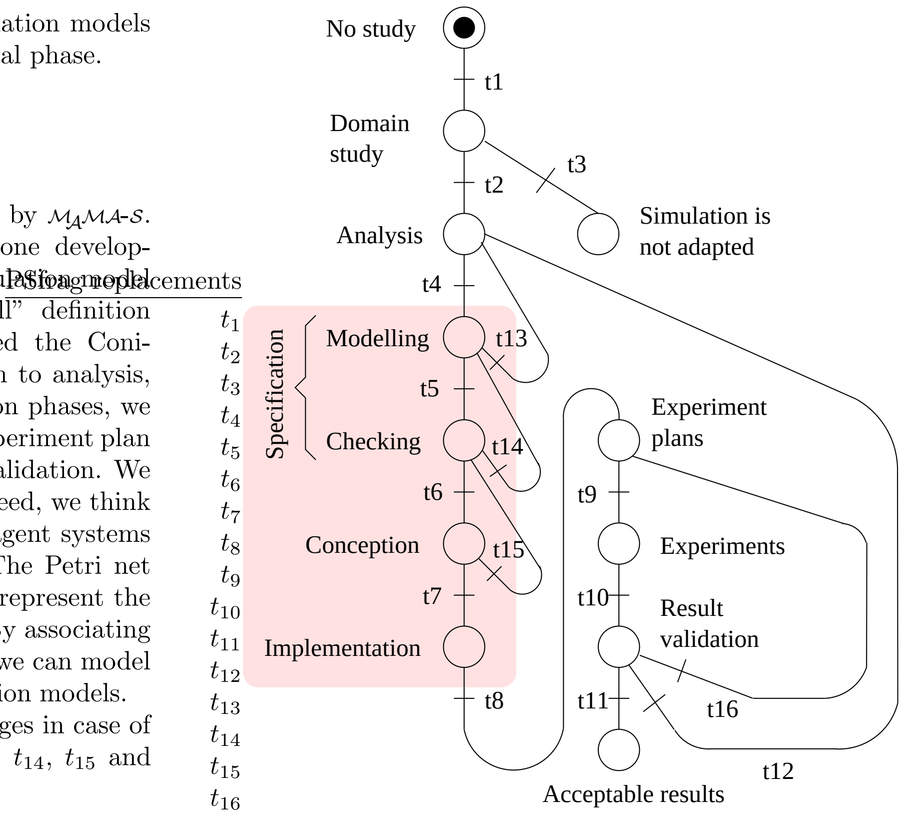

Figure 3: $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$ life cycle course

t_1	Problem is highlighted
t_2	Simulation is selected
t_3	Simulation is unadapted
t_4	Requirement specification is written
t_5	MAS-oriented UML model is built
t_6	UML model is checked
t_7	Instanciable MAS is ready
t_8	Instanced MAS is built
t_9	Experiment plans are ready
t_{10}	Experiments are realized
t_{11}	Result validation is reached
t_{12}	Requirement specification changes are needed
t_{13}, t_{14}, t_{15} and t_{16}	Validation failure is happened

build is the *implementation*. The multi-agent simulation model is made up with the checked UML model and with a whole of specific platforms and tools *e.g.*, multi-agent platform MAST and simulation tool ARENA®. The dedicated editor has a particular status. It allows to create and publish the various models of the $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$ lige cycle. For that, it exposes the formalism contained in the UML metamodel and permits to call upon the other methodology components.

In the following sections we present the four main phases of $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$: analysis, specification, conception and implementation.

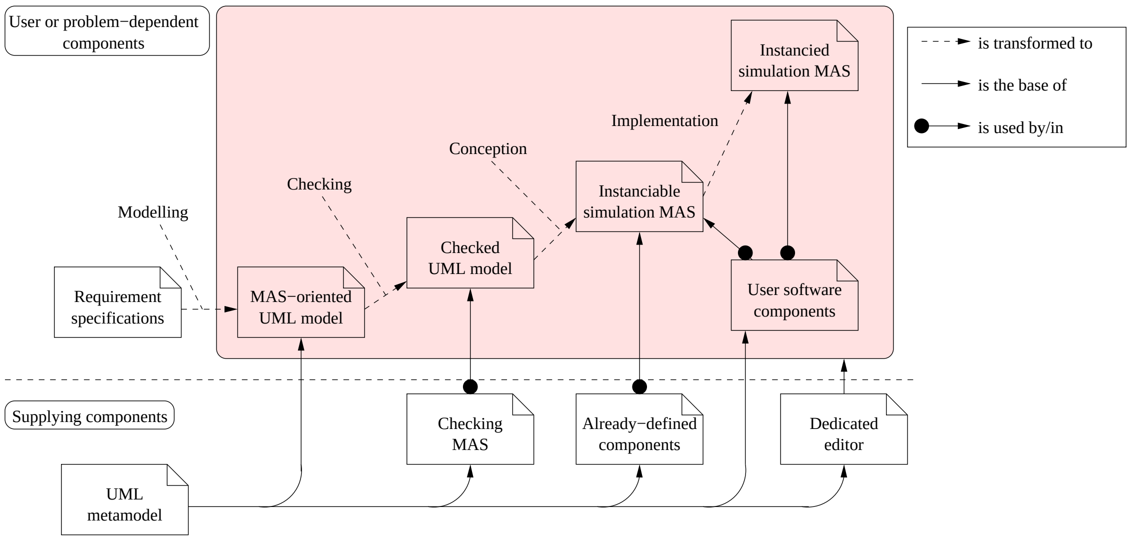

Figure 4: $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$ life cycle dependences

3.3 Analysis

The analysis or requirement definition is the process aiming to establish which functionalities the system must provide and which nonfunctional constraints it must satisfy.

It is necessary to specify the level of detail to which the needs must be expressed. A problem definition must be comprehensible and used to design and carry out simulation models.

It is important to make the difference between goals and needs *e.g.*, "easy-to-use system" (nonmeasurable property) is a goal whereas "all user commands are selectable with popup menus" (verifiable property) is a need associated to a goal.

The analysis phase must produce a document called *requirement specification.* According to (Sommerville 1998), it must satisfy the six following criteria:

- specify only the external system behavior,
- specify the realization constraints,
- easy to update,
- use as reference to the maintenance programmers,
- contain the indications concerning the later stages of the life cycle,
- specify the acceptable answers to the undesirable events.

We propose a structure for the requirement specification, which is based on the proposal of (Sommerville 1998) :

1. Introduction
   This part presents the reason for being of the simulation model, its context, a short definition of the awaited functions, a presentation of the requirement specification structure and notations;

2. A physical flow description
   This part describes the whole of the physical structures, which are in the simulation model. We especially find :
   - local and distant production-line descriptions,
   - operating times,
   - breakdown durations,
   - tool-change durations,
   - used resources (single or multiple),
   - management rules;

3. An informational flow description
   This part presents informational structures of the simulation model. Various informations and types of information could be describe :
   - informal product description (manufactured or not),
   - routing and nomenclature informations,
   - decisional flow description *i.e.*, the whole of decision categories, which exist into the system;

4. A decisional system description
   This part contains the whole of information necessary to the comprehension of the simulated system oganisation *e.g.*, the organisational structures and the decision-making rules;

5. Glossary and index.

3.4 Specification

According to the French Association of Standardization (AFNOR 1989), a specification is "a whole of activities, which define in a precise, complete and coherent way the user needs". We can affirm that the specification is a phase of the simulation model life cycle and corresponds to the definition of the essential characteristics that must have a simulated system. A specification is the requirement specification formalization and allows to validate the coherence of the brought solution with respect to the expressed needs.

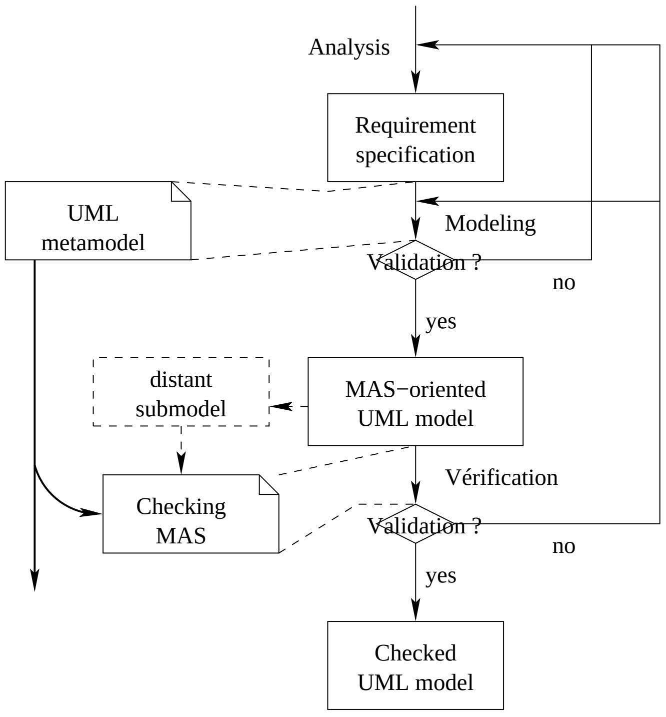

Figure 5: Specification phase

Within the framework of our methodological approach, we propose a formal definition of the specification progress, which is illustrated by figure 5. Two concepts are used during this specification phase : multi-agent systems and the UML metamodels (Muller 1997).

We use a multi-agent system to check the coherence of model with respect to distant model definition *e.g.*, a resource, which is defined in the model M_2, must be used by the model M_1 according to its definition in M_2. But the whole of constraint attach to distributed modelling elements does not allow to carry out a complete checking by the multi-agent system *during* the conceptual model construction *e.g.*, if a modelling componant, which corresponds to a distributed resource R, is "put" in the simulation model before the machine M, which uses it, the multi-agent system is incompetent to carry out the correlation between the resource use and its definition. To solve this problem, let us split the specification phase into two successive stages : the modelling and the checking.

The modelling is based on the UML metamodel use. It defines construction rules for the conceptual simulation model. Thus the UML mechanisms enables to create models, which respect a static preset formalism. Thus developers are able to put modelling elements in an unspecified order. They use the construction rules enacted in the UML metamodel. Then we obtain a static conceptual model.

The checking is carried out by a specific multi-agent system. It permits to check the use of the distributed componants according to their definitions. During this stage, our methodological approach should indicate if resources are misused, if production chains exist, or if entities are sent to properly simulation models.

The result of this phase is a conceptual model, which respects at the same time the formalism enacted by the UML mtamodel and well-formed communications with the distant modeling elements (checking by the specific multi-agent system).

3.5 Conception

The figure 6 illustrates the *conception* phase progress. From the conceptual model (or checked UML model), the development team works on the generation of a multi-agent model equivalent to the conceptual one. This phase is partly automated by the use of a whole of translation rules from conceptual modelling elements to agent societies. Developers can modify this set of rules in order to enter in adequacy with the simulation problem solving. The resulting model is multi-agent platform-independent.

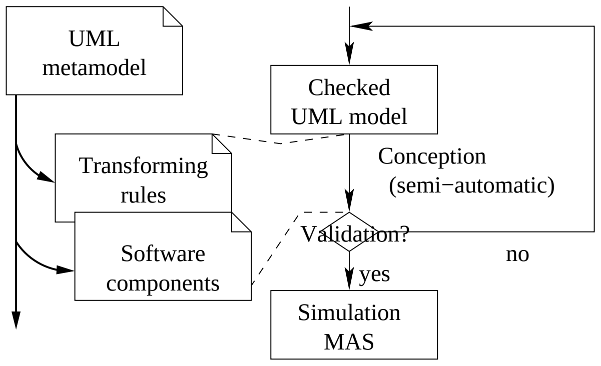

Figure 6: Conception stage

3.6 Implementation

We considere the implementation as an entirely automated phase. Indeed, it is easy to pass from a generic multi-agent model to a particular implementation on a multi-agent platform as SWARM (Burkhart 1994), ARÉVI (Duval et al. 1997) or MAST (Boissier et al. 1998). In addition, the transforming rules can take into account the use of commercial simulation software as Arena® or Simple++®.

3.7 Instanciation Example

Within the iterative development framework of our methodological approach, we designed a simple multi-agent model. It is composed of agents, which are intermediaries between various simulation models. There last are built in a specific simulation software (Arena®). This multi-agent model permits to build simulation models, which take into account physical and decisional flow distributions. However we limit the modeling capabilities of this multi-agent model to flowshop cases *i.e.*, neither cycle nor flow junction.

Within a validation framework, we apply this first model on a teaching case. Learners deals with a physical or decisional parts of the simulation model. Each learner group manages his part independently and parallely to other groups. This application highlights local decision-making problems within production systems and allows to create interesting teaching situations.

We present and develop this first multi-agent model and its application in (Galland et al. 2000).

4 CONCLUSION AND PERSPECTIVES

Simulation is recognized as a tool adapted to the study of industrial system behaviors. But even if this technic takes into account dynamic aspects, it seldom allows physical, informational and decisional distributions. In addition, modern tools are rarely accompanied by adapted methodologies. With this observation, we had proposed in (Galland et al. 1999) a methodological approach for the simulation based on the multi-agent concepts : $\mathcal{M}_{\mathcal{A}}\mathcal{MA}$-$\mathcal{S}$. An iterative development of this methodology was adopted *i.e.*, the various functionalities evolve gradually during our work.

In this article we present the first results on the development of our approach. We think that a methodology integrating physical, informational and decisional distributions must be itself distributed *i.e.*, the various sub-models necessary to a simulation model could be developed in parallel. Once to expose our point of view on this problem, we define the simulation model life cycle. It is made up primarily by the traditional phases of analysis, specification, conception and implementation. We add to it the three phases of experimental plan design, expriments and simulation result validation, which are more specific to simulation problems. We consider that our contribution is in the level of the four first phases.

In the future, we will defined the various concepts used and usable during the analysis (requirement specification), the specification (Uml metamodel, checking multi-agent system), the conception (multi-agent simulation model) and the implementation (multi-agent platforms). We will checked our theories on industrial and teaching case applications.

REFERENCES

O. Boissier, P. Beaune, H. Proton, M. Hannoun, T. Carron, L. Vercouter, and C. Sayettat. 1998. “The Multi-Agent System Toolkit”. Technical report. SIC/ENSM-SE.

R. Burkhart. 1994. “The Swarm Multi-Agent Simulation System”. In *OOPSLA Workshop on ”The Object Engine”*.

P. Burlat. 1996. “Contribution à l'Évaluation Économique des Organisations Productives : vers une modélisation de l'entreprise-compétences”. Phd thesis Université Lyon 2 (Jan.).

Y. Demazeau. 1995. “From Interactions to Collective Behaviour in Agent-Based Systems”. In *European conference on cognitive science* (Saint-Malo, France, Apr.).

T. Duval, S. Morvan, P. Reignier, F. Harrouet, and J. Tisseau. 1997. “ARéVi : Une Boîte à Outils 3D Pour Des Applications Coopératives”. *“La coopération”* 9, no. 2 (June): 239-250.

J.-M. Filloque. 1992. “Synchronisation Répartie sur une Machine Parallèle à Couche Logique Reconfigurable”. Phd thesis Institut de Formation Supérieure en informatique et Communication - Université de Rennes 1 (Nov.).

S. Galland, F. Grimaud, P. Beaune, and J.-P. Campagne. 1999. “Multi-Agent Methodological Approach for Distributed Simulation”. In *Simulation in Industry - 11th European Simulation Symposium* (Erlangen - Germany, Oct.), G. Horton, D. Möller, and U. Rüde eds, 104-108.

S. Galland, F. Grimaud, and J.-P. Campagne. 2000. “Multi-agent architecture for distributed simulation : Teaching application for industrial management”. In *Simulation and Modelling : Enablers for a better quality of life – 14th European Simulation Multiconference* (Ghent, Belgium, May), R. Van Landeghem eds, 756-762.

P.-A. Muller. 1997. *Modélisation objet avec UML*. Eyrolles.

R. Nance. 1981. “Model Representation in Discrete Event Simulation: The Conical Methodology”. Technical report. Department of Computer Science, Virginia Polytechnic Institute and State University, Blackburg, USA.

I. Sommerville. 1998. *Le Génie Logiciel et ses applications.* InterEditions.