

# SimStudio: a Next Generation Modeling and Simulation Framework

Mamadou K. Traoré  
LIMOS CNRS UMR 6158, Université Blaise Pascal  
Campus des Cézeaux  
63177 Aubière Cedex  
+33 473 405 046  
traore@isima.fr

## ABSTRACT

SimStudio is an operational framework that must serve to capitalize theoretical advances in Modeling and Simulation (M&S) as well as to gather M&S tools and make them accessible through a web browser. From a software perspective, SimStudio is a middleware for the federation of simulators and the collaborative building of simulations. From a hardware perspective, SimStudio is a mean to aggregate intensive computing resources through the http protocol.

## Categories and Subject Descriptors

I.6.7 [Simulation and Modeling]: Simulation Support Systems – Environments.

## General Terms

Standardization, Design.

## Keywords

Framework, Plug-in.

## 1. INTRODUCTION

Intensive efforts are done in North America to come up with a simulation-based engineering science whose kernel is a computation science made of a theory, an operational M&S framework and supporting computing infrastructures [1][2]. Meanwhile, in France and across Europe, many questions are raised about the integration of large scale hardware infrastructures and high-level software architectures [3][4]. These preoccupations make clear the fact that there is a mutual influence between: (1) gaining technological overhangs, (2) understanding and mastering M&S hardware and software architectures, and (3) achieving mutual M&S knowledge and products. It's this intertwining that SimStudio tries to crystallize, not by a pragmatic approach nor by exclusive and specific manipulations of existing or new codes, but by erecting an environment in which theoretical operations can be mapped onto practical ones and any manipulation can be symbolically tracked.

Simulation is coming to such a maturity that an axiomatic theory supported by an operational framework is very much needed to break its complexity barriers. Current grand challenges (model interoperability, reuse...) raise the need to treat simulation models as algebraic entities which can be manipulated symbolically as well as be easily translatable into operational objects.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. SIMUTOOLS 2008, March 03-07, Marseille, France  
Copyright © 2008 ICST 978-963-9799-20-2  
DOI 10.4108/ICST.SIMUTOOLS2008.3214

## 2. REQUIREMENTS FOR A NEXT GENERATION M&S FRAMEWORK

Any problem solving process is the fruit of the activity of a decision making system. This latter makes use of model(s) and data. Simulation is required to increase the capacities of such a system, in particular when the problems to solve are so complex that only computers can deal with them, i.e. they include various objectives and often hide quantities of interactions between their components. Models that are built to figure these problems and to manipulate them through artifacts are inaccurate in essence, and their quality has an impact on the accuracy of the decision made. Therefore, model design is a crucial process. What are the concepts required to drive such a process? What are the effective methodologies? Which kind of software system must be built? These are some major questions that the Simulation theory must answer in a generic way (i.e. application context free). We see this theory as an axiomatic mathematical and computational approach which must be supported by a technological operational framework that allows rapid prototyping and can be used for "in silico" experimentations.

The main general issue that we deal with is the status of a model as an approximation. From our point of view, this issue federates all major M&S aspects and bringing an answer to each of the questions it raises can drive to a complete Simulation theory. These questions are various variants of the same global concern: simulation model properties. From there a metrics base can be built, including credibility (verification, validation and error estimation), reuse (and by extension: substitutability, equivalence, composability), space-time complexity (speed-up and memory consumption, for sequential and parallel and distributed simulations), interoperability (interface compatibility, compliance to standardized communication protocols)... In the context of the SimStudio project, all these concerns are split into four research axes:

- An algebraic axis (specification): here, we deal with the formal semantics of simulation models, i.e. *what are the objects of the domain and what relationship do they entertain?* The increasing complexity of systems and the existence of multiple perspectives (continuous/discrete flows, deterministic/stochastic phenomena, variable scales, evolutionary/combinatorial approaches...) make hard the unification of M&S concepts and their universal formalization. Moreover, the constant changes of modeling

objectives for the same model makes it necessary to formalize the context in which this model is used [5].

- A logical axis (analysis): here, the matter is to identify underlying logical semantics that allow to reason about the structure and the behavior of simulation models. *How can a simulation model be amenable to formal analysis?* The classic post-simulation way to deal with such an issue can face some operational qualities (speed-up, confrontation with real data...), while an ante-simulation approach could allow to cope with algebraic qualities (formal verification, applicability of a model in a given context, model composability, model equivalence...). This latter way is a novel and very hard one [6][7].
- An executive axis (code generation): here, we deal with automatic code synthesis and automatic trajectory output. *How transparency can be brought in the specification-to-code process? And in the reverse process?* The clear separation of modeling activities from simulation activities (code synthesis, mapping to hardware, trajectory generation, communication and interoperability of software components...) is still a challenge in M&S, especially for large scale problems (federation of huge codes, coupling of non contemporary existing simulators...). The issue has to be tackled both at algorithmic and technological level.
- An application axis (use): *how to support the scale crossing of application codes?* In one hand, it is necessary to define generic simulation-based problem solving schemes, and in the other hand, there is a need to integrate software components in real-time environments.

A unique operational framework has to concretize all the theoretical efforts presented above. This framework is named **SimStudio**. To summarize, the SimStudio project aims at building a next generation modeling and simulation (M&S) framework, which must serve both as:

- A virtual lab for the experimentation and the study of M&S advanced concepts.
- A collaborative and community-focused platform for the mutualization of M&S resources.

Therefore, SimStudio would drive to the erection of standards in simulation models design, implementation, analysis and integration. Like similar projects in other domains (J2EE, Apache...), it takes advantage of concurrent contributions from its users communities, including:

- M&S research teams: the framework is a way for them to experiment with their new M&S tools and to fix them using the other tools available in the framework (assuming that their tools comply to the plug-in principle of SimStudio). Also, it becomes very easy for them (especially for teams not really specialized in M&S or scientific computing) to design, analyze and experiment with simulation models without being obliged to deal with long and costly prototyping phases. A third interest of the framework is the potential to access to large models libraries.
- M&S consuming business and public institutions: they deal specifically with de facto standards, graphical user interfaces, and computing performances.

- Education: the framework is a pedagogic tool for establishments that involve M&S in their curricula. It can gain from the feed-back provided by these establishments through study cases and training courses.

The concept of M&S framework (in the meaning of Apache and J2EE) drives to a concentration of knowledge, which is a factor for the emergence of a widely accepted theory (for researchers and users). Today, we're far from this situation: groups are formed apart according to the type of M&S approach they adopt, or real system/problem they study (communities for DEVS, Multi-agents and Artificial Intelligence, manufacturing systems, ecosystems...), and they convey different views of the same activity, making it hard to come up with a fundamental transversal scientific recognition of M&S. We state that the building of such design, implementation, experimentation, archiving and reuse frameworks is inescapable in the very near future, as suggested in [8].

### 3. SIMSTUDIO

The framework is a multi-layer platform composed by the following parts (figure 1):

- A **Modeling layer** to integrate and use graphical and textual specification plug-ins, opening the way to the building of large models and frames libraries.
- An **Analysis layer** to integrate and use formal analysis plug-ins, allowing the evaluation of model algebraic and simulator operational metrics (uncertainty vs. accuracy of simulations, models properties within a frame...).
- A **Simulation layer**, a middleware for the unified execution and the transparent deployment of simulation codes over mutualized resources (grid, internet, specific clusters...).
- A **Visualization layer** to integrate and use simulation visualization and animation plug-ins, with the ability to link together users, simulations and animations (for training purposes, virtual reality...).
- **APIs** to offer managing services (user's account, authentication, workspace customization, collaborative tools, models archiving and extraction, plug-in installation and management). Any authorized user can add his own plug-ins to the platform and make it accessible or not to the community.
- An **Automation layer** for translation and interfacing, allowing to map multi-perspective models into the universal formal semantics of the platform, to generate simulation codes and trajectories and, to allow analysis and visualization plug-ins to operate on models, simulators and simulation traces.
- **Containers** for the platform key entities. A generic model component technology allows to build standardized models and model libraries and to aggregate existing models. In a similar way, there is a technology for building standardized simulators components and another one for building standardized trajectories.

To be effective, the framework requires that interfaces be defined to glue its different layers. In this paper, we show how a specific modeling plug-in is interfaced with specific inner simulation

components, and how these latter are interfaced with a specific visualization plug-in. The plug-ins and the simulation components

have been independently realized by separate teams.

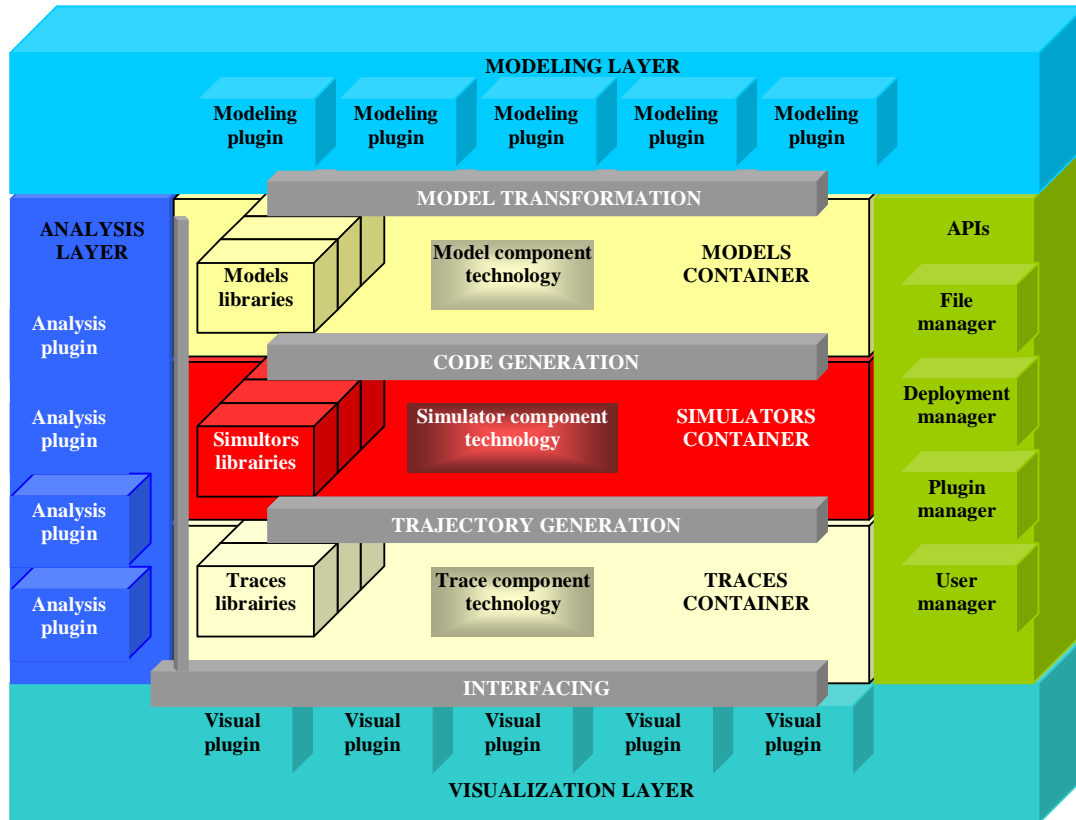






Figure 1. SimStudio M&S Framework

### 3.1 Modeling Plug-in

This tool allows to get from an intuitive graphical and textual specification, an XML document, ready for use by simulation and visualization tools. It combines Flash and XML technologies and uses the object-oriented programming paradigm. The application was created with version 8 of Adobe Flash. It is entirely based on the DEVS formalism [9]. Each specification element is a graphical object on which it is possible to act using the mouse (figure 2). To create a model, one must click on "File-> New..." and then type in the name of the model. A first box appears for the system to model. It may contain sub models that can be either "coupled" (i.e., containing other sub models), or "atomic" (with no sub models). To add a sub model, simply click on the  icon and choose the type of model as desired. Models have certain characteristics in common: each model can be removed by clicking on the  icon, and each model can have one or more input and/or output ports by clicking on the  icon. An input

port is associated with a variable type accepted by the model at its entrance. These ports can be either a simple type such as a string, a Boolean... or a complex type (mixing simple and/or complex types). Coupled models can be enlarged by clicking on the  icon. This allows to add sub models to them. For atomic models, it is required to specify their state, internal and external transition functions, and output and time advance functions. Sub models can be connected through their input and output ports. Simply click on one of the ports of a model, drag and drop to a port of another model. The link creation fails if the ports types (simple or complex) and their natures (Boolean, string...) don't match.

Once the entire system is modeled, the user can save it by clicking on "File" then "Save As...", which results in the creation of an XML file format, according to the DTD presented in Figure 3. He can also load an existing model by clicking on "File" and "Open" and select the XML-formatted model.

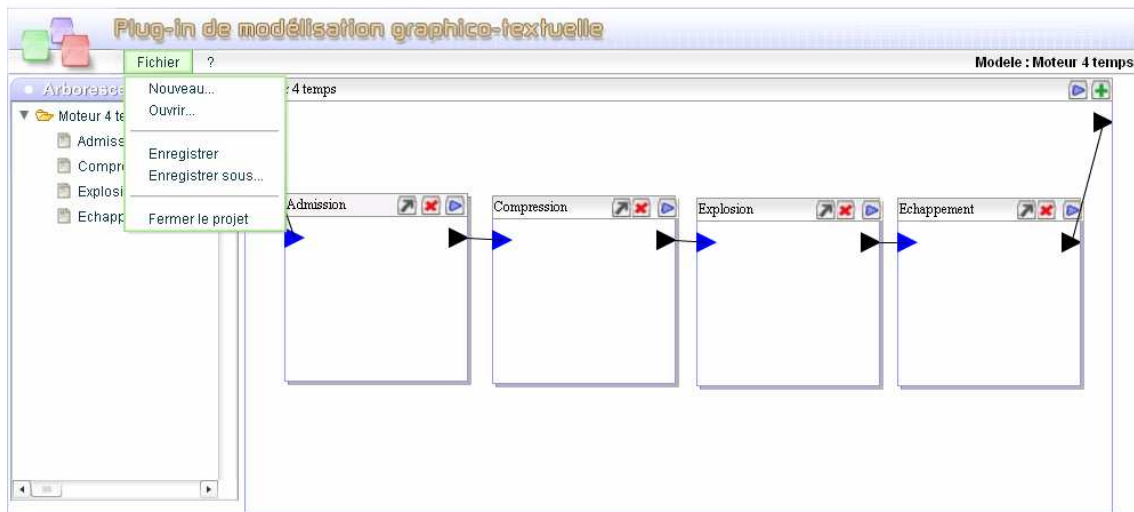


Figure 2. Screenshot of the DEVS Modeling Plug-in

```

<?xml version="1.0" encoding="UTF8"?>
<!ELEMENT plugin (type, model)>
<!ELEMENT type (simple, complex)>
<!ELEMENT simple (item*)>
<!ELEMENT complex (item*)>
<!ELEMENT item EMPTY>
<!ATTLIST item id ID #REQUIRED
name CDATA #REQUIRED
naturePortType IDREF #IMPLIED
PortType (0|1) #IMPLIED >
<!ELEMENT model (ports, links?, models?)>
<!ATTLIST model id ID #REQUIRED
name CDATA #REQUIRED
type (atomic|coupled) #REQUIRED
posX CDATA #REQUIRED
posY CDATA #REQUIRED
state CDATA #IMPLIED
internalT CDATA #IMPLIED
externalT CDATA #IMPLIED
outputF CDATA #IMPLIED
timeA CDATA #IMPLIED>
<!ELEMENT ports (input,output)>
<!ELEMENT input (item*)>
<!ELEMENT output (item*)>
<!ELEMENT links (link*)>
<!ELEMENT link EMPTY>
<!ATTLIST link id ID #REQUIRED
name CDATA #REQUIRED

```

```

upPort IDREF #REQUIRED
upPortMain CDATA #REQUIRED
downPort IDREF #REQUIRED
downPortMain CDATA #REQUIRED>
<!ELEMENT models (model)*>

```

Figure 3. DTD for the XML Files Generated

### 3.2 Visualization Plug-in

A viewer has been realized, resulting in a graphical representation of the traces of DEVS simulation models. It is inspired by the work done by *Hongyan Song* during his Master of Science at *McGill University* (under the direction of Professor *Hans Vangheluwe*), who defined an XML format that can represent simulation results. Our viewer is using a modified version of this XML format. It is based on the combination of SVG (Scalable Vector Graphics) and XSLT transformations. The fact that these two languages use an XML syntax provides maximum compatibility. The resulting DTD is given in Figure 4. The input and output files for the viewer are both in XML format. Therefore the XSLT transformation scheme can be applied to translate one in another. This transformation is made using the XSL library and the PHP language. The use of scripts ECMA Script provides access to the DOM SVG document, and allows to animate the image created. This solution has the good taste to work with all browsers supporting the display of an SVG document. A tool tip system has been put in place in order to give some information on the key parts of the image, all with a simple click.

There is a hierarchy among the sub models of a global model, in accordance with DEVS paradigm. Atomic sub models possess state variables whose values evolve during the course of the simulation, and coupled sub models are composed of other atomic and/or coupled sub models. The selection menu is used to display this hierarchy. If a model is atomic, its state variables are displayed. For example in Figure 5, the atomic model FEU has two variables COULEUR and VOITURE. A click on the name of a variable displays in a new window the trajectory of this variable.

```

<?xml version="1.0" encoding="UTF8"?>
<!ELEMENT trace (modelfile,event+)>
<!ELEMENT modelfile EMPTY>
<!ATTLIST modelfile file CDATA #REQUIRED>
<!ELEMENT event (model ,eventtime ,port*,state)>
<!ELEMENT model (#PCDATA)>
<!ELEMENT eventtime (#PCDATA)>
<!ATTLIST event kind (IN | EX) #REQUIRED>
<!ELEMENT port (message)>
<!ELEMENT message (#PCDATA)>
<!ELEMENT state (variable+)>
<!ELEMENT variable (name,type,value+)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT value (#PCDATA|variable)*>

```

```

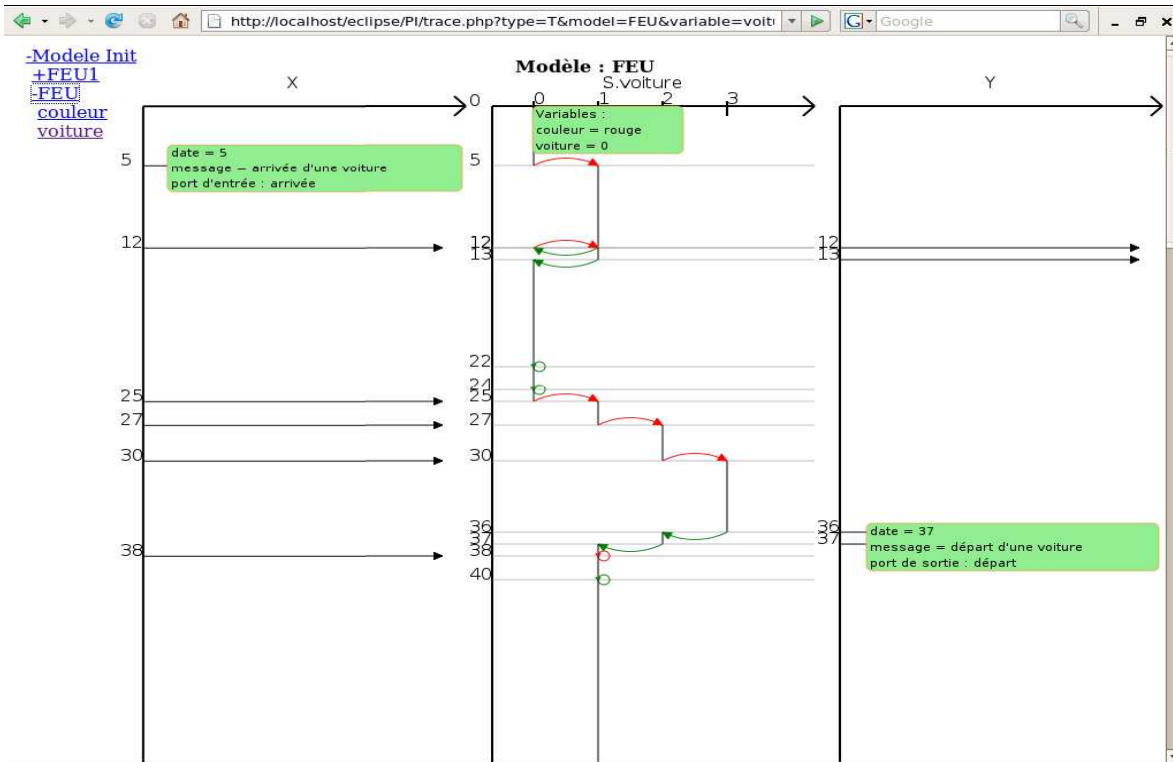
<!ATTLIST port name CDATA #IMPLIED>
<!ATTLIST port category (I|O) #REQUIRED>
<!ATTLIST variable category (P|C|PC|CC) #REQUIRED>

```

**Figure 4. DTD for the XML Trace Files**

The display of the trace of a model is only available for atomic models (the visualization of a coupled model is a set of visual frames, each for one of the atomic models that compose the coupled model). This is represented in two ways. The horizontal axis gives the possible values of a variable (selected by the user), the vertical axis representing time. On figure 5, we can see the arrows representing the transitions that took place in the model. The green arrows represent internal transitions, while red ones represent external transitions.

An HTML page is generated with the help of PHP. It assembles all the graphical elements. The scroll bar of the browser allows moving along the axis of time to see the entire trajectory.



**Figure 5. Screenshot of the Visualization Plug-in**

### 3.3 Simulation Cement

The DEVS formalism is well established and much implemented [10], and still a standard for its implementation is an opened issue. A library of DEVS simulation components has been built in SimStudio. For portability and easy integration reasons, this library has been developed in Java.

The library is divided into 6 different packages which are: Model (where are defined DEVS basic models, i.e. atomic and coupled models, as well as all the elements that are closely related to them, such as ports and state variables), Simulator (implementing the executable components associated to atomic and coupled models), Type (simple and complex predefined types), Exception (predefined exception handlers), Message (initialization, scheduling, input and output messages) and Util (various useful

classes). Each package has a particular role that we present hereafter:

Automating the link between modeling and simulation is realized by a script that gets as input the name of the XML file created by the modeling plug-in, then generate the files containing the definition of Java classes, and then compiles them, and finally opens a window for simulation sessions. The output of the Java classes is a set of XML file, one for each atomic sub model's trajectory. The user can decide whether or not he wants to generate a record for a given sub model.

#### 4. CONCLUSION

There is a tremendous number of M&S tools, a few of them takes benefit of the Internet, which would allow to share their use in a common platform. To meet this increasingly important need, the concept of a framework composed of modeling, analysis, simulation and visualization plug-ins has been considered. It aims at combining several M&S tools and at making them accessible through a Web browser. In addition, many management tools can be added for the benefit of the users. At a software level, it becomes possible to federate applications codes via a standardized global middleware. At an hardware level, it becomes possible to provide access to computing farms, via the http protocol. Therefore, SimStudio is a vast lab for experimentation and capitalization of scientific insights, whose content and development are the work of the community (one could do comparisons in terms of contribution and philosophy to concepts such as the wiki web in the domain of collaborative platforms, the J2EE and DOT NET frameworks in the domain of software engineering...). We believe it belongs to the next generation M&S tools.

One of the major technological locks lies in the plug-in based solutions integration principle, because it raises both the issue of standard (not yet resolved neither at specification level, nor at middleware level) and separation of concerns (not yet fully implemented for a better understanding of the field). These locks are integral parts of the complexity barriers that M&S must face today. As a matter of fact, SimStudio is an effort for convergence towards M&S standards, through the consolidation of a complete

theory. Because it is a generic and ambitious infrastructure, it could be a catalyst for further progress in M&S advanced aspects.

#### 5. REFERENCES

- [1] NSF. 2006. Simulation-based Engineering Science. Revolutionizing Engineering Science Through Simulation, NSF Report. May.
- [2] President's Information Technology Advisory Committee. 2005. Computational Science: Ensuring America's Competitiveness, Report to the President.
- [3] Groupe Conseil Général des Technologies de l'Information. 2005. La Politique Française Dans le Domaine du Calcul Scientifique, Rapport n° II.B.14.2004, Mars.
- [4] Groupe Simulation – Académie des Technologies. 2005. Enquête sur les Frontières de la Simulation Numérique en France. La Situation en France et Dans le Monde. Diagnostic et Propositions, Rapport de l'Académie des Technologies, Mai.
- [5] Traoré M.K. and Muzy A. 2006. Capturing The Dual Relationship Between Simulation Models And Their Context, Simulation Modelling Practice and Theory (SIMPRA), 14(2), 126-142.
- [6] Traoré M.K. 2006. Analyzing Static and Temporal Properties of Simulation Models, Proceedings of the 38th Winter Simulation Conference. Monterey, California, USA, December 3-6, pp. 897-904.
- [7] Traoré M.K. 2006. Making DEVS Models Amenable to Formal Analysis, Proceedings of the Spring Simulation Multiconference (SpringSim'06). Huntsville, Alabama, USA, April 2-6, pp. 33-39.
- [8] RNTL. 2001. Rapport du groupe de travail RNTL sur le logiciel libre. 14 Novembre.
- [9] Zeigler B.P., Praehofer H. and Kim T.G. 2000. Theory of Modeling and Simulation. Integrating Discrete Event and Continuous Complex Dynamic Systems, Academic Press.
- [10] DEVS Standardization Group. 2008. DEVS Tools. Available at < <http://www.sce.carleton.ca/faculty/wainer/standard/>>. Last access on February, 21, 2008.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*SIMUTools'08*, March 3–7, 2008, Marseille, France.

ISBN 978-963-9799-20-21