USING JDEVS FOR THE MODELING AND SIMULATION OF NATURAL COMPLEX SYSTEMS

Jean-Baptiste Filippi
Frederic Chiari
Paul Bisgambiglia

UMR CNRS 6134
University of Corsica
B. P. 52, 20250 Corte, France.
Web : http://spe.univ-corse.fr/filippiweb/
Email [filippi, chiari, bisgambi]@univ-corse.fr

ABSTRACT

This paper describes the JDEVS modeling and simulation environment. JDEVS has been developed for over a year to serve as an experimental framework for natural systems modeling techniques. It enables discrete-event, general purpose, object-oriented, component based, GIS connected, collaborative, visual simulation model development and execution. The sample models implementation shows that this experimental environment can be used for solving any complex problems solvable by discrete-event simulation but is especially suited for natural system simulation.

Currently only hierarchical block and cellular models are modeled and simulated but the current works is the development of a multi-layered modeling paradigm for spatially distributed systems (with vector and cellular models) that will eventually be implemented in the environment.

1 INTRODUCTION

The research in defining and creating an object oriented natural system modeling and simulation environment began 9 years ago at the SPE lab of the University of Corsica.

It has already been shown in (Delhom 1996) that the Object Oriented Modeling Formalism, based on DEVS can be used to solve any problems that can be solved by discrete event simulation.

1.1 Natural system modeling specific problem

The problematic is now to adapt the OOMS environment to the field of natural complex modeling. To do so, several modules have been developed to enable a DEVS simulation engine to be used by an ecosystem modeler.

The JAVA language has been chosen in that purpose. Several tools have been implemented around the core java simulation engine: A graphical modeling interface, an easy to use models library, a cellular simulation panel and a connection to a GIS (Geographic Information System). The Neuro-DEVS methodology is also proposed to implement easily Artificial Neural Networks (ANN) into OO models.

Artificial Neural networks (ANNs) are widely used in the field of natural system modeling since they can simulate systems that are not well defined and when a large amount of empirical data is available.

The paper presents the different modules, along with sample implementation of cellular and hierarchical block diagram models. The toolkit is still under development, the current work is the development of a topological vector based modeling and simulation formalism that will enable the propagation of models in a different manner than with cellular models (and avoid space sampling).

1.2 OOMS environment

The OOMS (Object Oriented Modeling and Simulation) environment is based on the DEVS (discrete events system specification) formalism defined by B.P Ziegler (Ziegler 1990). To be able to give the most general description of any system, the DEVS's description hierarchy has been extended with two other concepts in the OOMS

- The C. Oussalah's work (Oussalah 1989), which introduces the abstraction hierarchy.
- The J. Euzenat's work (Euzenat 1994) which proposes the granularity to defines a time hierarchy.

Currently only the time and description hierarchy are implemented in JDEVS. Integration of those concepts allows the automatic generation of a simulator from the model as defined in the DEVS formalism.

Within the formalism are specified basic models from which larger ones are built and the connection between them.

A basic model is called an atomic model and has specifications for the dynamic of the model.

A coupled model is the container for sub models (atomics or coupled) and links between sub models.

2 NATURAL SYSTEM MODELING

A natural system model is a model that intends to mimic the behavior of an ecosystem to study new interactions. The non exhaustive list of aims of such model contains forecasting future events, planning operations, simulating new scenarios with new construction and finding the equilibrium states for a new situation.

In the area of natural systems modeling and simulation, models behaviors are complex and often subject to many influences. Most systems that have to be modeled are of two kinds:

Hierarchical block diagram models, composed of interconnected components. Such models are used to simulate mathematical models that are built in this way. Both benefits from the DEVS formalism, mathematical models are decomposed in several reusable components to compose a hierarchical blocks system.

Cellular models (Wainer and Giambiasi 2001) can be simulated on continuous time based thus saving a lot of computational cost. Cellular models can also be stored in models libraries because of the genericity of description of DEVS models.

2.1 Hybrid methodology, coupling neural networks, OOMS and GIS

The complexity of natural systems raises some problems into the modeling process. Models have to be simplified and cannot match real systems. In fact, the lack of understanding in the system dynamic can lead to a model that is not realistic enough to be validated.

The ability of Neural Networks to generalize from empirical data often provides a better alternative than OO models because they can handle this lack of knowledge in the system dynamics.

Starting from these observations, an hybrid methodology has been developed, encapsulating ANN in OO Models to benefit from the two modeling techniques. Based on OOMS formalism, the hybrid methodology define interactions between OO and ANN's models. Full description of the methodology will be found in (Filippi et al. 2001).

Upon possible applications, three important uses of hybrid systems can be highlighted.

The concurrent simulation where outputs of an ANN model are compared with results of another kind of model.

ANN as sub-components of a global model allows a partial lack of the model understanding.

Adaptive models, that would be able to adapt runtime to situations that hasn't been taken into account (D'Alch-Buc and Patillon 1999).

Data is needed to feed our natural system models. GIS seems to be a perfect data source since data that concerns the studied environment is stored at the same place using a geographic reference. It is the perfect information system to perform the "data fusion" (Clark and Yuille 1990) that will compose the "natural model" database .

3 THE JAVA DEVS TOOLKIT

These thoughts about an OOMS ecosystem oriented modeling environment ended up in the construction of a toolkit, where ecosystems oriented tools are extending an OOMS modeling and simulation engine. JDEVS will not be compared to other environments such as MOOSE (Lee and Fishwick 1999), Devsjava (Sarjoughian and Zeigler 1998), CD++ (Wainer and Giambiasi 2001) or the VSE (Balci et al., 1998), although those environments can be used for the same application than JDEVS the aim is different here as the focus is on natural systems modeling.

JDEVS toolkit is composed of five independent modules. They can interact with other modules that are already developed and some elements, including the java simulation kernel might be changed. The toolkit is public domain and freely available, with source code from the project website at "http://spe.univ-corse.fr/filippiweb/appli/". It has been developed by the first author.

3.1 Modeling and simulation kernel

The modeling and simulation kernel is a simple java implementation of the DEVS formalism (Ziegler 1990). Atomics and coupled models are described as follow.

3.1.1 Atomic DEVS models definition in JDEVS

Starting from the assumption that a modeler that made the effort to understand the DEVS formalism knows something about programming, the definition for the dynamic of atomic models is made directly in JAVA. To help him in this task, the GUI generates a java skeleton, stores it in a database and compiles it.

An atomic model can be formally described as: $M=<X,Y,S,\delta int,\delta ext,\lambda,ta>$, to be easily understood the java description should be as close as possible from this description.

The following code is a generated java skeleton for a DevsAtom1 $=<X\{in1\},Y\{out1\},S\{first\},\delta int,\delta ext,\lambda,ta>$

```
public class DevsAtom1 extends AtomicModel {
 // ports
  Port in1 = new Port(this,"in1","IN");
  Port out1 = new Port(this,"out1","OUT");
public DevsAtom1 () {
  super("DevsAtom1");
  states.setProperty("first",""); //S set
}
 public EventVector outFunction(Message m) {
        //@todo output function
        return new EventVector();
     }
 public void intTransition() {
       //@todo internal transition function
     }
 public EventVector extTransition(Message m){
       //@todo input function
       return new EventVector();
     }
 public int advanceTime() {
  //@todo time advance function
      return integer.MAXVALUE;
    }
}
```

The output and external transition functions are returning event vectors that will be appended to the job list.

One of the ecosystem modeling domain specific problem is the lack of understanding of some natural systems. This leads to the situation where self learning models such as neural networks are offering better results than physical models. Neuro-Devs (Filippi et al 2001) is a proposition for an implementation of Neural Networks in the DEVS formalism.

3.1.2 Neuro-DEVS, neural net model definition in JDEVS

An interface is created to separate the ANN construction from the modeling process to allow the neural network to be manipulated as a stand-alone object.

3.1.2.1 Neuro-DEVS applications

The concurrent simulation can be used to avoid an unexpected behavior of a neural network by comparing the neural network output with the output of a simple model to validate the result.

Adaptive models can be used to modify the neural network runtime according to an error feedback (Difference between the model's forecast and the real world data collected afterwards) (D'Alch-Buc and Patillon 1999).

ANN as a sub-component can be used if Neural Networks provides better results for only a piece of the whole system (like battery in an energetic system (Jungst et al. 2000)), this is also the application that is implemented in this paper.

3.1.2.2 Neuro-DEVS definition

ANN (Hecht-Nielsen, 1989) has input ports and output ports (X, Y) corresponding to its input and output neurons.

Inputs and outputs are arrays of values differently bounded, thus values for minimum and maximum of each arrays must also be stored to scale down the inputs and scale up the outputs (as natural systems perform operations in R).

Three characteristic functions are needed to perform operation : the activation function, the propagation function, and the learning function. The neuro atomic model is a subclass of the atomic model, it can be formally described as :

NAM=<X,Y,S,NN,ta,init,δint,δext,λ,learn,act,prop>

With NN = link to neural network object. The use of an interface authorizes to manipulate objects trough an object broker (such as CORBA), thus separating the ANN construction from the modeling process.

It is then up to the modeler to use these function in its code to define how and where the neural network should be used.

If the ANN object is not used as an adaptive model, it should have already "learned" the patterns, as initial weights must be defined before the simulation starts.

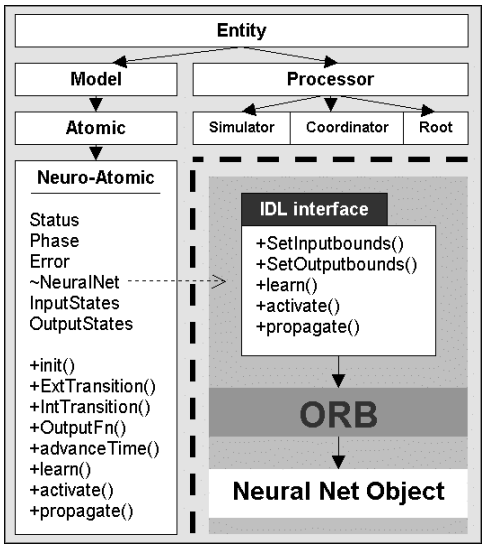

Figure 1 : Class hierarchy and interface definition

In the DEVS formalism, the structure is described in the coupled models. Coupled models have no definitions for the dynamics of the models, but define the links between its atomics and sub-coupled models.

3.1.3 Coupled models description in JDEVS

If the user wants to interact directly with the simulation engine, the coupling between models can be made directly in a java class. However, with the use of the GUI, it is possible to graphically define interactions between models. The structure is then saved in the XML format (Bernardi et al. 2001).

Coupled model can be formally defined as CM=<X,Y,CHILD,EIC,EOC,IC,SELECT>

Part of the resulted XML document type definition for a coupled model is:

```
<!ELEMENT MODEL (TYPE, NAME, BOUNDS?, INPUT*,
OUTPUT*, CHILD*, EIC?, EOC?, IC?)>
```

With TYPE defining the kind of coupled model (Cellular, kernel, coupled...), NAME the name of the model, BOUNDS the position of the model on the screen (used only by the GUI for a representation), INPUT the set of input ports, OUTPUT the set of output ports, CHILD the index for the components of the coupled model (in the priority order), EIC is the external input coupling, EOC the external output coupling and IC the internal coupling.

Each coupled model is stored in a different XML file and, when loaded, the parser automatically instantiates the models and creates the links between them.

3.2 Graphical user interface

The graphical user interface is the modeling front-end of the toolkit, using this front end, the user can graphically create, compile, link and store atomic and coupled models, debug the resulting model and perform the simulation.

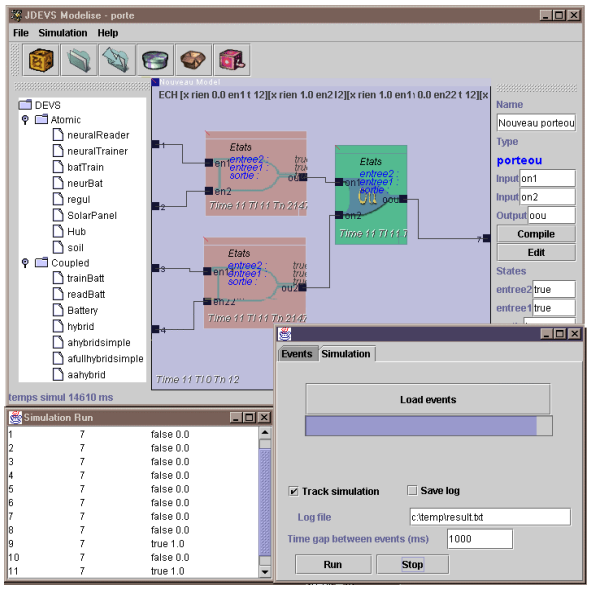

Figure 2 : JDEVS Hierarchical block modeling and simulation interface

The GUI also allows distributed modeling, if different modelers works on sub-coupled models and store them in the same library, it is possible to federate those models in another graphical modeling client. (Figure 2) shows the modeling and simulation interface. At the left stands the library of models (with atomic and coupled models), with a mouse click the selected model is added to the selected coupled model.

In the center stands the hierarchical block diagram representation of the model, all components can be moved with the mouse, the linking between models is performed by a click from the origin port to the destination port.

On the right stand the properties of the selected component, if it is an atomic model, the user can edit and compile it from this properties panel.

At the bottom of (Figure 2) stands the simulation panel, here the user can load the input event lists and run the simulation to the screen or to a file.

To debug the model, it is possible to track the simulation. In this mode, a chosen delay is put between the processing of each event. The diagram representation of the models is then showing the job queue at any time as well as the states for all properties of all models.

3.3 DEVS models library

A complete description of the library can be found in (Bernardi et al. 2001). The implementation of the library description in JDEVS is resulting in a module in the GUI. This module presents models according to its domain and sub-domain, all classified in a tree like architecture.

3.4 GIS interconnection

To create cellular models that will reproduce some piece of land for propagation models, we needed to define a way to export data from the GIS and a procedure that will initialize the cells according to the spatial data collected in the GIS.

Since no open standards like (openGIS 2001) or (SEDRIS 2001) have yet been became standards, the choice has been made to use simple ASCII files to transfer data from the database to the simulation engine and back. Nevertheless part of the current work in JDEVS is concerning data interchange format for multimodels (Lee and Fiswwick 1999) simulation results. To perform the coupling, the user has to select a zone in a GIS, then rasterize the zone and export the resulting map in an ASCII file.

To enable a 3d visualization of the model, the same cut has to be made in and elevation grid map (at any resolution) then exported aside as an ASCII elevation grid.As the simulation will eventually be displayed and imported back in a GIS, the ASCII map can be refined with a link to a background image, the cell size and the absolute coordinates of the map; those data are simply stored in the header of the file and are not used by the simulation engine. The interconnection module rasterize the selected map in the GIS according to the cell size and then generates a set matrix that is used by the simulation panel to initialize the model.

3.5 Cellular simulation panels

This module allows the user to perform (and debug) simulation of a cellular model. The user can directly interact with the simulation, as he can send events using the mouse. The outputs of the cellular simulation panel are timed maps that are imported back into the GIS.

A general architecture shown in (Figure 3) has been adopted to model cellular systems.

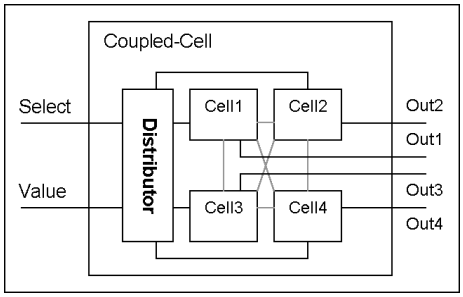

Figure 3 : Cellular models architecture

It is composed of a distributor and cells in a cellular coupled model. The general inputs are connected to the inputs of the distributor, then the distributor will send them to either all the cell or to the cell that would have been selected by an event in its “Select” port.

All cells are connected to the general output. During the initialization, the cellular coupled model is loading the GIS generated file, calculates the number of cells needed (width*height) and then instantiates and performs the coupling for every cell.

The simulation result is a set of discrete events, the cellular coordinator is logging changes in states that occurs for all cells. This discrete events set is then converted

to a discrete time, two dimensional states maps that are imported back in the GIS.

(Figure 4) shows the propagation of bugs in an orchard in the JDEVS 2D simulation Panel. Using this tool, the user can interact runtime with the simulation, with a click on the map it is possible to select a cell and send a specific event. A 3D simulation panel (Figure 5) has also been developed to enable a better visualization of phenomena. This tool is especially useful to visualize water flows (as in Figure 5) or 3 dimensional cellular models.

4 IMPLEMENTATION OF A 2D CELLULAR MODEL: BUGS PROPAGATION IN AN ORCHARD.

Bugs are devastating 25% of the Corsican fruit production each year, to minimize their effect it is necessary to study their propagation. Using a DEVS description of a bugs colony behavior according to the fruits maturity, the temperature and the kind of fruits a simulation of bugs.

The conceptual model along with the complete study can be found in (Faure 2001). Propagation has been made in an orchard with 7 kind of fruits that has different maturation dates. A cellular model representing a bug colony has been developed based on observations made by entomologists. There is no interest here to explain the model dynamics, but the use of JDEVS shown a very high interest for the entomologists as they were able to visualize easily the state of the bugs colony of the studied zone during the simulation (Figure 4).

The model is now being validated using comparisons with some real-life observation. Afterwards the software can be used for any other orchards providing that the information exists in the GIS.

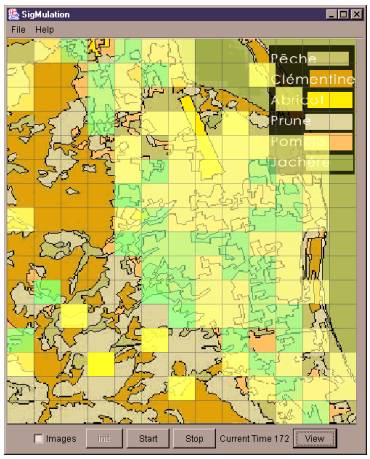

Figure 4 : Bugs colony propagation during a simulation in JDEVS

5 VISUALISATION OF A 3D CELLULAR MODEL: HYDROLOGICAL MODELING AND NATURAL RESSOURCE MANAGEMENT.

To manage natural resources such as water, it is necessary to model the phenomena that will alter those natural resources in order to quantify and qualify them. Some models, like pollution dispersion models or water flow models (Figure 5) require 3d visualization since they represent 3d data. The 3d simulation panel serves for the visualization of the simulation of these phenomena, it uses Java3d in order to paint the outputs of 2d or 3d cellular models. It is using the height matrix exported from the GIS to reconstruct a 3d world that the user can move.

(Figure 5) shows a model that is currently tested to quantify the pollution by phosphates that goes into the sea. The model is using the pollution data collected into the GIS, then perform the simulation using the rain data. The 3d panel enables here the visualizations of the untouched fresh water reserves.

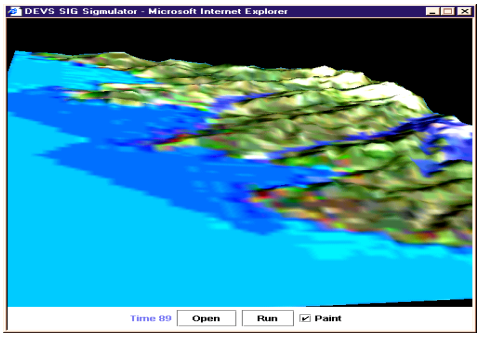

Figure 5 : 3D cellular simulation of a pollution model in south west of Corsica

6 IMPLEMENTATION OF A HIERARCHICAL BLOCK DIAGRAM MODEL WITH NEURAL NETWORK: A PHOTOVOLTAIC SYSTEM.

The solar energetic farm also known as PV system, is a solar power plant (Figure 6). It is composed of solar power cells, a battery and a switch (Jungst et al. 2000). Solar power cell behavior is well known, thus it is modeled using the physical description provided. The battery is modeled like an electric power tank, with charge and throughput efficiency. The neural network damage model is decreasing the battery maximum capacity runtime.

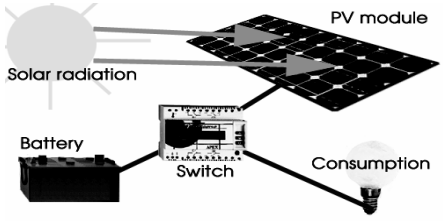

Figure 6 : PV System

The benefits of JDEVS for the modeling of this model is clear, the structure of the model is quickly created without writing a line of code. The neural network is directly included in the atomic damage model and data (for the solar radiation) is imported from the GIS.

(Figure 7) shows the experiment results by JDEVS. For every experiment with any initial states, an HTML page is generated showing graphs and analysis of the

simulation results. This model can be used as it is to perform cost viability studies for PV system in any place in the world, with any battery. Simulation time for 3 years of hourly solar data and consumption is about 30 seconds on a Pentium 4 1.5 Ghz.

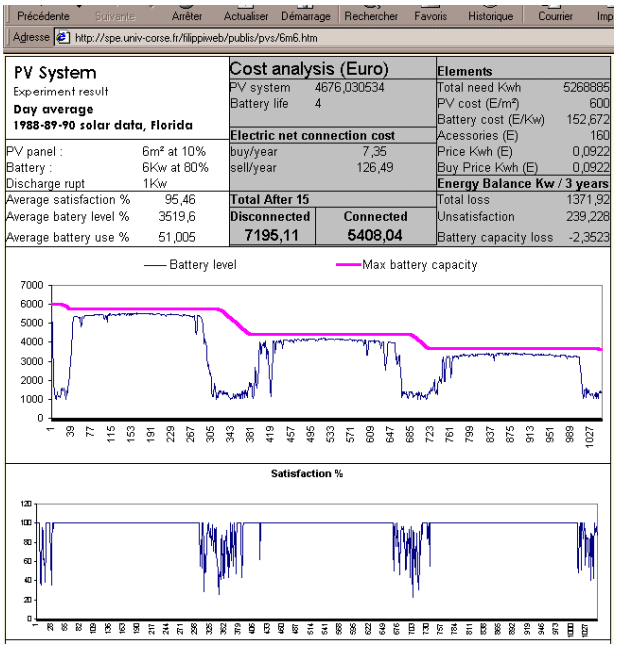

Figure 7 : JDEVS generated experiment results (HTML)

7 CONCLUSION

All the features already implemented in the JDEVS toolkit have been introduced. The tool saves modeling and simulation time in the field of ecosystem and natural system modeling and simulation. It has been used in different applications showing its versatility, and the flexibility of the toolkit. One of the concerns was also to perform the data fusion, with the simulation results into the GIS, and it is possible using the tool. The current work being made on this tool is the development of polygonal models propagation methodology, that will describe models using dynamically generated points and vectors rather than with cells. We hope that this kind of models will be a better alternative than cellular DEVS for some application since they would avoid the rasterization of the topological vector GIS data.

REFERENCES

Bernardi, F.De Gentili, E. Santucci, JF. “Reusable models integration in a devs-based modeling and simulation environment.” In Proceedings of the SCS ESS 2002 conference on simulation in industry, vol 1, p 644.

Balci, Osman, Anders I. Bertelrud, Charles M. Esterbrook, and Richard E. Nance, "Visual Simulation Environment," Proceedings of the 1998 Winter Simulation Conference, Washington, DC, 13-16 December, pp. 279-287.

Clark J.J. and Yuille A.L., “Data Fusion for Sensory Information Processing Systems”. Kluwer Academic Publishers, 1990

Delhom, M. 1996. “Modélisation et Simulation Orientées Objets” Ph.D. thesis.

D'Alch-Buc, F. Patillon, J-N. 1999. "Discharge prediction of rechargeable batteries with neural networks" Journal of Systems Architecture, no. 6, pp. 41-53.

Euzénat, J. 1994. "Granularité dans les représentations spatio-temporelles" Tech. Rep. 2242, INRIA Rhône-Alpes.

Faure, X. 2001. Modélisation et simulation de la dispersion de la mouche méditerranéenne des fruits (Ceratis capitata) en Corse . Technical report UMR CNRS 6134 Lab.

Filippi, JB. Bisgambiglia, P. Delhom, M. 2001 “Neuro-DEVS, an hybrid methodology to describe complex systems” In Proceedings of the SCS ESS 2002 conference on simulation in industry, vol 1, p 646.

Hecht-Nielsen, R. 1989. "Neural network primer: part I" AI Expert.

Jungst, RG. Urbina, A. L.Paez, T. 2000. "Stochastic modeling of rechargeable battery life in a photovoltaic power system" Tech. Rep. 1541C, Sandia national laboratories.

Lee, Kangsun and P. A. Fishwick 1999. OOPM/RT: A Multi-modeling Methodology for Real-Time Simulation, ACM Transactions on Modeling and Computer Simulation, Volume 9, Number 2, April 1999, pp. 141-170.

OpenGIS 2001, Open Geographic information system project consortium, http://www.opengis.org/.

Oussalah, C. 1989. "A framework for modeling and linking the structure and the behavior of a system" Artificial Intelligence in Scientific Computation.

Sarjoughian, H.S. and Zeigler, B.P. 1998. "Devsjava: Basis for a devs-based collaborative M&S environment" in Proceedings of the SCS International Conference on Web-Based Modeling and Simulation, vol. 5, pp. 29-36, San Diego, CA.

SEDRIS, 2001, Synthetic Environment Data Representation and Interchange Specification project http://www.sedris.org/.

Wainer, G. A., Giambiasi , N. 2001, “Application of the Cell-DEVS paradigm for cell spaces modelling and simulation”. In Simulation, January 2001.

Zeigler, B.P. 1990. “Object-Oriented Simulation with Hierarchical, Modular Models”, Academic Press.

AUTHOR BIOGRAPHIES

Jean Baptiste Filippi. Obtained a Ba in Communication from the University of Coventry in England, graduated from Högskolan i Boras, Sweden, and obtained his Msc with “best honnours” in computer science in Corti, Corsica (FR) in 2000. He’s entering now his 2nd year of PhD in the UMR CNRS 6134 lab of the university of Corsica. He is also working as a researcher under contract with the French Ministry of Finance and the European Union. He is the developer of several software for data mining and computer simulation. His email and web addresses are : `<filippi@univ-corse.fr>` `<http://spe.univ-corse.fr/filippiweb>.`

Frederic Chiari. Obtained his Msc with "best honours" in computer science in Corti, Corsica (FR) in 1997. He's entering now his 4th year of PhD in the UMR CNRS 6134 lab of the university of Corsica. He worked also as System & Network Administrator for 4 years at the UC. His email address is : `<chiari@univ-corse.fr>.`