Activity-based Modeling and Simulation Life Cycle

Alexandre Muzy

November 27, 2009

LISA lab, UMR CNRS 6240
Università di Corsica – Pasquale Paoli
Campus Grossetti, BP 52
20250 Corti, France

Abstract

The activity-oriented strategy is one possible world-view. Other usual strategies (process-oriented and event-oriented) can be used. One of these strategies is expected to be more easy to implement and more efficient according to the problem considered. Integrating usual world-views for system specification should provide a coherent framework. Activity is used as a shared aspect for the abstraction of other strategies. As a new domain, a shared understanding of activity should be proposed. A description of activity details for modeling and simulation is proposed here.

Keywords: Activity, theory of modeling and simulation.

1 Introduction

The three usual world views (process-oriented, activity-oriented, and event-oriented) syntethized by Balci in 1988 [3] constitute a major contribution in trying to integrate and describe usual approaches emerging in the modeling and simulation field. These views can also be considered as "conceptual frameworks", "simulation strategies", and "formalisms" to guide scientists for the development of their simulation model. We contribute here to the analysis and the integration of these usual views. The whole framework is unified in an activity-based modeling and simulation life cycle.

The whole approach first considers the basic elements of discrete-event system specifications introduced initially by Bernard P. Zeigler as a theory of modelling and simuation [7]. The discrete-event system specification of general system structure (*cf.* the mathematical structure presented in [1]) consists of basic elements: Discrete-events and models as components composing networks. Models are piloted by abstract simulators. The efficiency of abstract simulators received significant attention [6, 4][5] The efficiency of simulators executions depends on their structure (and the basic advantages/disadvantages which can be obtained in distributed and sequential implementation structures) as well as on the structure of the models they are connected to. A choice has to be made between the level of specification of models and the deepening of the corresponding simulator hierarchy. We believe that activity can be used as a measure of the computational cost of both model and simulator structures. Then, using this measure, a trade-off can be done between the advantages (reduced execution times and antinomic increased reusability) and the opposite disadvantages (increased execution times and antinomic reduced reusability.)

Let's consider first the basic definitions of words *activity*, *event*, and *process*. An activity "is what transforms the state of a system over time." It begins with an event and ends with another. An event is what causes a change in the state of a component. A process "is a sequence of activities or events ordered in time." We simply consider activity here as a measure of the event occurrences. If no event occur, the system is considered as qualitatively inactive. If event occur in the system, the system is considered as qualitatively active. Conversely, a quantitative measure of the number of event occurrences in the system provides information on resource usages and their localization.

Activity is expected to be worth for: (i) optimizing the performances of during simulation phase, (ii) optimizing the modeling phase, (iii) find an optimal trade-off for the choice of both model and simulator structure, (iv) guiding the modeler through the modeling and simulation life cycle.

Section 2 of this paper reconsiders usual model and simulator structures through usual world views. Section 3 presents the basic operations on model and simulators and how they can be combined and chosen through activity. All operations are considered at high level in an activity-based modeling and simulation life cycle. Section 4 first presents an abstract framework for decisions, engineering and modeling choices, then a didactical application of the activity-based modeling and simulation life cycle is provided. Finally, perspectives to this work are provided.

2 Integrative Activity-based Modeling and Simulation

Usual models and simulators structures can be reconsidered and abstracted. This defines the main tools, which can be combined then through an activity based modeling and simulation methodology.

2.1 Model

Figure 1 presents usual world views through packages and accesses between the packages. Both process and Activity are represented as central views, both using event scheduling. In the introduction, a broader view of activity has been presented. The whole conception constitutes an integration of the concepts of activity, events, and processes.

Figure 2 depicts this integrative view at a component level. Components can be "active" or "inactive." However, the activity of a system can also be

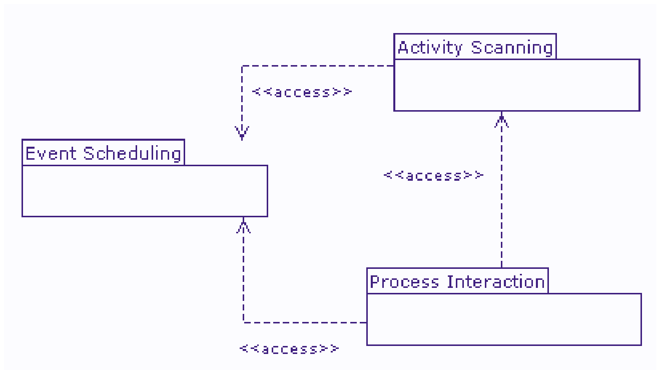

Figure 1: Integration of usual world views

determined by the number of discrete-events occurring in the system. Only describing active components does not fully specify a model. Components can be selected according to the quality of the of their *achievement*. Achievement can be computed as the *score* of the simulation model (how well the model satisfies objectives) divided by the effort invested (or resource consumption): activity. Components can compose networks. They interact then with other networks and components through external events. Finally, components autonomously shedule internal events.

2.2 Simulator

Figure 3 presents a flat simulator corresponding to the activity concepts previously introduced. At a simulation level, all the previous concepts can also be integrated[1]. First, an active set is used to compute only active components. Second, a routing function is used to update the active set with the influenced components. Third, structural changes are achieved. More information about the structure of this kind of simulator can be found in [2].

[1]Vocabulary related to traditionnal world views is underlined.

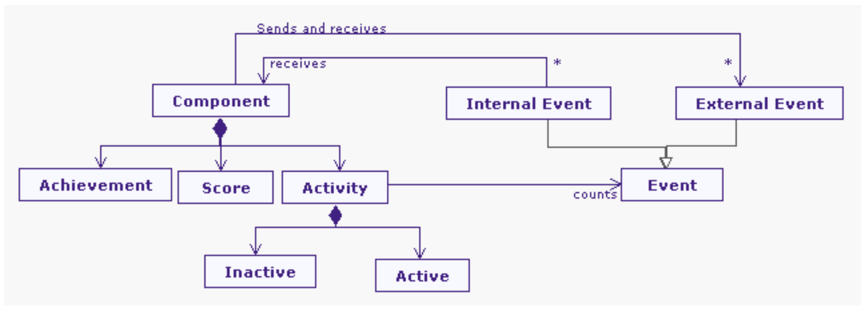

Figure 2: Active component

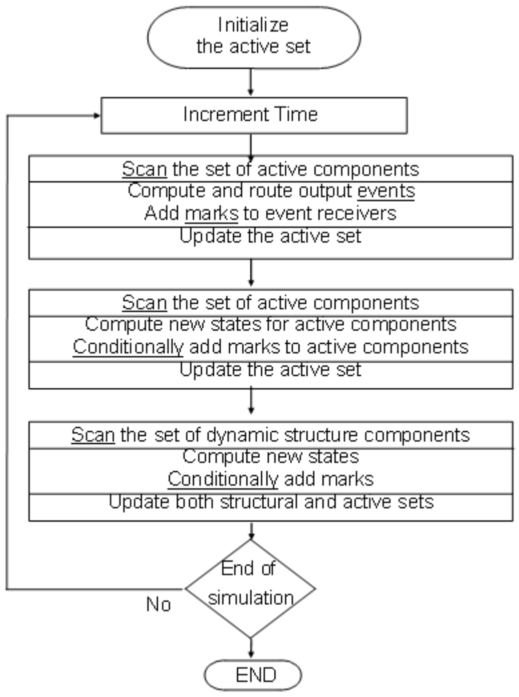
Initialize the active set
Increment Time
Scan the set of active components
Compute and route output events
Add marks to event receivers
Update the active set
Scan the set of active components
Compute new states for active components
Conditionally add marks to active components
Update the active set
Scan the set of dynamic structure components
Compute new states
Conditionally add marks
Update both structural and active sets
End of simulation
No
END

Figure 3: Simulation process

3 Operational Modeling and simulation

Operations on both model and simulator structures can be piloted through activity. Activity can also be used for guiding modeling and simulation choices achieved for manipulating model and simulator structures.

3.1 Operations on models and simulators

Decoupling models and simulators is currently a well accepted conceptual technique in the modeling and simulation community. This allows enhancing reusability and separates modeling and simulation concerns thus improving clarity.

The degree of autonomy of models corresponds to the degree of structure specification of a system and its parts:

- First, a fully integrated single model of a system can be considered (this corresponds to a single atomic component.) The system is not described by sub-models but just by one model, which consists of a transition function operating on a set of states.

- Second, partially autonomous, interdependent, sub-models of the system can be described (this corresponds to a multicomponent model.) Transitions of components can be directly influenced by other influencing components.

- Third, fully autonomous modular sub-models of the system can be considered (this correponds to a coupled model of atomic models.) Atomic models are modular. They interact through input/output ports. Their state is fully encapsulated through port access. Their interface is designed to *re-act* to input events and to *act* through output events. Events are sent by influencing components.

Simulators correspond to the computing layer of models. Simulators are in charge of activating transition functions of atomic models, as well as sending events to other models. Coordinators are in charge of exchanging messages according through components couplings. In a network of computers, coordinator nodes can be compared to hubs, containing the references (addresses) of the machines to be connected through message exchanges. Couplings between simulators can be metaphorically compared to Ethernet cables. Both bandwith and distance of these cables determine the communication cost between simulators. These costs have to be minimized, minimizing communications, increasing autonomy of models (machines), as well as the state encapsulation of these models. On a single computer, using no hierarchy will be the more efficient simulation solution. In a distributed simulation environment, the degree of hierarchy will depend on the physical characteristics of the network (bandwiths and computational resources.)

Figure 4 depicts the possible structural operations on model and simulator structures. On the top left is represented a hierarchy of models (with coupled models *CM* and atomic models *AM*.) On the right is represented the corresponding fully distributed simulator hierarchy (with coordinators *Coo* and simulators *S*.) As we will see the degree of model decomposition and simulator hierarchy deepening[2] relates to the degree of specification and autonomy of models and simulators. This distributed hierarchical simulation tree corresponds to the definition presented in [Bernie 2000]. Here the degree of parallelization is maximum.

[2]The "deepening" concept has been presented and introduced in KIM

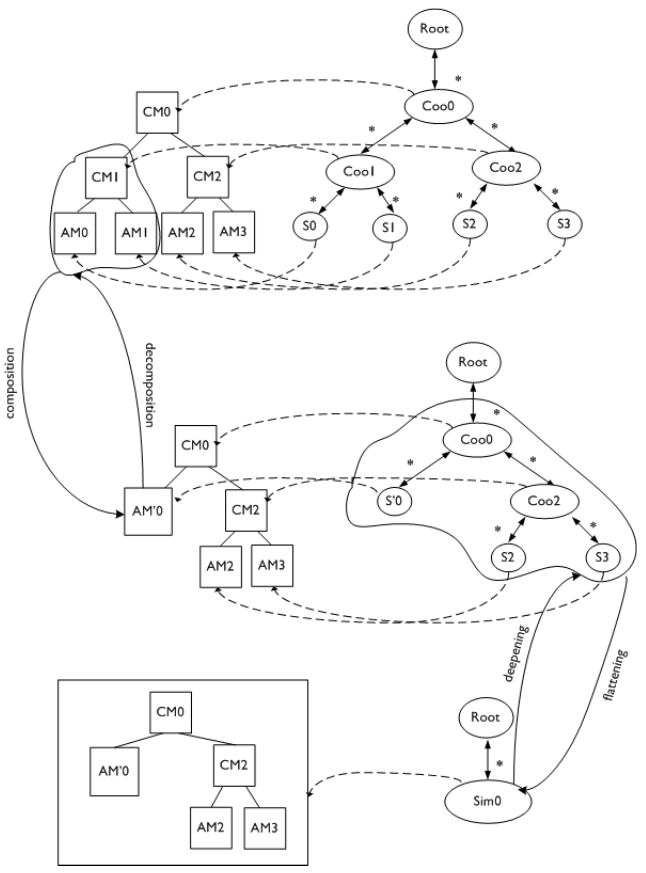

Figure 4: Model and simulator structure operations

The possible simetric operations are:

- At the modeling level: 1. Decomposition: to describe a model with more details, using coupled models; and 2. Composition: to reduce the level of specification of a model, composing its submodels into single models.

- At the simulation level: 1. Deepening: To add hierarchical levels, and 2. Flattening: To flat hierarchies.

Choosing an operation has both advantages and disadvantages. The more the models and simulators are integrated, the faster and the more efficient they are. However, if one tries to isolate parts of the simulation model, both interface specification (subpart models) and computational layer (the simulator) will not be interoperable. Conversely, the more the models decomposed and the simulators are deep, the slower and the less efficient they are. However, they are highly interoperable and reusable. To sum up, composition and flattening increase efficiency and decreases interoperability while decomposition and deepening decrease efficiency and increase interoperability.

Activity corresponds to resource usage. When a lot of computational resources are disponible, judiciously increasing resource usage (and activity) can improve performances. Everything depends on "how well" the resources are used (i.e., to the degree of achievement of the components chosen for increasing their resource capabilities) and "how much" resources are disponible.

3.2 Optimization of modeling and simulation

Activity can be used as a measure balance between reusability and autonomy/encapsulation to drive model and simulator operations (i.e., dynamic structure changes.) Dynamic structure changes can be used at two levels: 1. At the performance optimization level: For one simulation model, an *optimal satisfying level* of modeling and simulation hierarchy exists, for the same precision. The satisfaction amount depends on a balance between resource usage and resource disponibility.[3]2. At the modeling optimization level: During the building phase of the simulation model, an *optimal satisfying level* of detail (obtained by aggregation and elaboration operations) exists. This level depends on: the satisfaction of modeling objectives, and (ii) on the resource availability optimized at level 1.

The modeling process consists of a balance between

[3]If a lot of resource is disponible, the modeler would not care about resource usage optimization. For example, if the resource is the execution time, if the final maximum execution time (the resource used) is very small compared to the maximum execution time tolerated by the modeler (the disponible resource), the modeler would not care about resource usage optimization

Operation	Reuse	Autonomy	Activity
Model Aggregation	no impact	-	-
Model Elaboration	no impact	+	+
Model Composition	-	-	-
Model Decomposition	+	+	+
Simulator Flattening	no impact	no impact	-
Simulator Deepening	no impact	no impact	+

Table 1: Activity-based Operation Balance for engineering M&S. Sign '+' means increase, and sign '-' means decreases.

modeling and performance optimization levels. Table 1 describes all the possible dynamic structure operations through an activity. Activity can be used to pilot dynamic structure changes (on models and simulators) to select and operate (on) models. Then, at a lower simulation level, activity tracking automatically selects active models. Table 1 can thus be used to drive the modeling and simulation process through an optimal trajectory. Notice that simulator flattening increases with composition while simulator deepening increases with decomposition.

3.3 Activity-based modeling and simulation life cycle

We discussed before the possibility for activity to be used as a parameter to guide and drive both modeling and simulation processes. Figure 5 represents the activity awareness to be used within a modeling and simulation life cycle. This cycle is iterative and incremental. First, according to the activity level within the simulation model, an *activity decision* is taken. The feedback of data *activity analysis* necessary to compute achievement can be computed: (i) Ad-hoc: Activity is analyzed from data directly obtained within the simulation loop, or (ii) A-priori: Activity has been determined from previous analysis and experiments. An *activatability* level is then determined before the simulation [*e.g.*, a probability function of activability of a process (of decision, of biological behavior, etc.)] *Activity analysis* [Akerkar] of timed data streams consists of determining spatial and temporal activity patterns. The detection of activity in data can be based on the causality of events in time and space (if a particular position in space is active, there is a higher probability that the neighboring positions will turn active; if that position is active at some time, there is a higher probability for the cell to be active at the next time step.)

Activity selection uses achievement levels to select sub-models composing the whole modeling and simulation structure.

The usual activity world view consists of: (i) A *condition* to be fulfilled for the activity to take place, and then (ii) An *action* related to the activity. *Activity condition* consists here too to the condition for the activity to take place. Action is called here *re-action* to pinpoint re-activatability aspect of processes, objects, or components, as well as to be a metaphor of both physical and biological reactions. Notice that activity awareness modeling and simulation life cycle consists of two parts: 1. The Model: Beyond models, in the reality this part corresponds to a mind layer (and can be refined to decision and modeling processes), and 2. Simulator: Beyond simulators, in the reality this part corresponds to a physical, physiological, biological, chemical... layer. Both parts are consistant with the model/simulator decoupling at a lower level.

4 Decision-based and Activity-based Modeling and simulation life cycles

Simulation can be used as a decision guide for modelers and decision makers. At different detail levels, both modelers and decision makers deal with the same objects: Decisions, parameters, and resources. Again activity can be used to explicitly describes the different steps of modeling and simulation for both decision makers and modelers.

4.1 Decisions in the Think-Build-Act framework

The think-build-act framework is described in Figure . This framework consists of three interdependent levels:

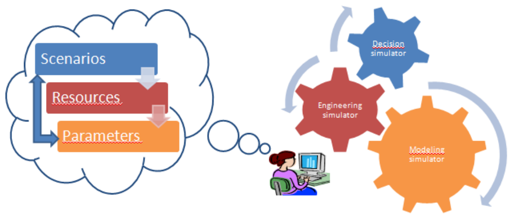

Figure 6: Simulation-support cycle

1. The Modeling level: Following the usual analysis-design-verification-validation cycle, modelers test, evaluate and compare the *parameters* of the model.

2. Engineering level: The output results obtained at the modelling level are compared to *resources*.

3. Decision-based level: According to the comparison result between usage and availability of resources, resources can be reallocated. *Scenarios* (conditions to restricte the initial conditions of the model) are used. These initial conditions have a direct impact on resources.

To detail framework application, we will consider a simple example. Let's consider the development of a land-use simulator. This simulator consists of two kinds of stakeholders: The government and landowners. Both can be represented by agent models. The following simplified simulation sequence can be used. First, the government takes constructability decisions for non built parcels. Second, landowners can choose to protest against this decision or not. Third, the government can be more concerned by ecology, the welfare of landowners or by the fiscal revenue they will obtain if the parcels are built. Conversely, landowners can be for the constructability of their parcels constructible or not.

The land-use simulator is composed of the three usual simulators of our framework:

1. The modeling simulator dealing with parameters: The decision making process of landowners

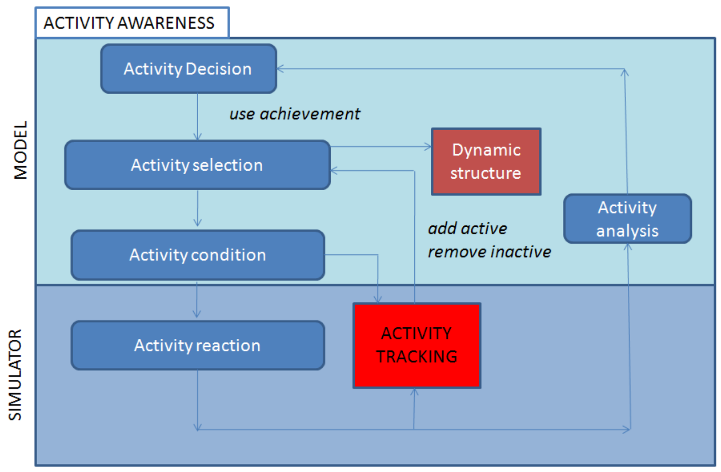

Figure 5: Activity awareness modeling and simulation life cycle

and of the government can be studied through various parameter values. One parameter can be attributed to each propensity of agents: Ecology, welfare and fiscal revenue (for the government), and constructability desire (for landowners.) The impact of every parameter level on outputs can be observed to experiment the model.

2. The engineering simulator dealing with resources: Resource in this example is the land availability for construction;

3. The decision simulator dealing with scenarios: The combination of the parameters of the model constitutes scenarios. A building rate is obtained for every scenario. If all agents are of the same kind, there are *5* propensity parameters and *6* possible combinations of the parameters. If the parameter values are continuous, and if there are many agents, the number of scenarios is exponential.

Notice on Figure 6 that there exists two kinds of configurations: One is bottom-up (from parameters to scenarios), the other one is top-down (from scenarios to parameters.) Both configurations usually need to be iteratively combined. The bottom-up approach allows improving knowledge. Changing the parameters, the modeler better understand the behavior of the simulator, delimitating the possible scenarios. Outputs of the agent-based simulation of the modeling simulator can be analyzed to calculate a building rate.The reverse top-down configuration allows calibrating the model. Through the top-down configuration, an a-priori building rate is provided as an input of the simulation. Comparing both emergent and input building rates allows calibrating the model.

All the think-build-act cycle can now be explicity refined and generalized through the activity-based modeling and simulation life cycle. Each step of the modeling and simulation life cycle (*cf.* Figure 5) is applied through its use by: A modeler, a programmer and a policy maker.

4.2 Modeler activity-based cycle

- Objective: Understand and validate the behavior of the model.
- Decision: Activity = Number of decisions of a particular kind (protestation, sales, etc.), Score = No scenario is better than another, No achievement.
- Activity selection: Consists only of automatic activity tracking adding and removing non built parcels (active) close to built ones (active), through dynamic structure.

- Conditions/reactions: At the mayors level, If a parcel is already constructed, then *re-act*: There is a probability for the neighboring parcels to be proposed for construction. At the citizens level, if a parcel is proposed for construction, then *re-act*: protest or not.

- Activity analysis: Ad-hoc: Decisions of agents (propositions to constructability, protestations, sales, etc.), A-priori: *activatability* corresponds to all the probabilities used in the decision making process: Protestation threshold, propensity of agent types (e.g., ecological) against other propensities.

4.3 Programmer activity-based cycle

- Objective: Minimize execution times and maximize reusability;

- Activity decision: Activity = Number of active components during the simulation, Score = Maximum possible decomposition level of the simulation model and faster detection algorithm of active components, Achievement: Minimize activity and maximize score.

- Activity selection: Automatically select active components, automatically unselect inactive components, through activity tracking and dynamic structure.

- Activity conditions and re-actions: At the cellular level, if a cell is constructed, then *re-act*: scan the neighoring cells and possibly add them to the set of active components. If a cell is proposed for constructability, then *re-act*: scan the agents concerned by this new activity. At the agent level, for residents: If a cell is proposed for constructability, then *re-act*: scan the agents concerned by this new activity; for the mayor: If a cell proposed for constructability recfeives to many protestations, then *re-act*: do not make it constructable.

- Activity analysis: Ad-hoc: Number of active components during the simulation, A-priori: *activatability* corresponds to all the cells close to already constructed areas.

4.4 Policy maker activity-based cycle

- Objective : Maximize welfare or/and fiscal revenues in scenarios.

- Activity decision : Activity = Protestations of a scenario, Score = Increase with adhesions and fiscal revenues; Decrease with protestations; Achievement = Minimize protestations and maximize adhesions and fiscal revenues.

- Activity selection: Choose scenarios with maximum achievement;

- Activity conditions and re-actions: At the scenario level: If a scenario provokes too many protestations or/and too few fiscal revenues, then *re-act*: remove it. If a scenario provokes few protestations or/and high fiscal revenues, then *re-act*: keep it. Use activity tracking and dynamic structure.

- Activity analysis: Ad-hoc: Number of protestations during the simulation. A-priori: *activatability* corresponds to all the cells susceptible to do not decrease welfare and increase fiscal revenue.

5 Conclusions

The usual elements of discrete-event specification and usual world views have been investigated through activity. From a low level of specification to a high level, activity allows modeling concisely, and simulating efficiently systems. Besides, activity can be used to drive the whole modeling and simulation cycle. However, a lot of work needs now to be achieved to ground more the approach. Once, many applications will have lead to a knowledge increase of activity mechanisms and elements, an automatization of an activity-based modeling and simulation too should be possible to be designed.

Acknowledgements

I would like to thank deeply Bernie Zeigler for its continuous guidance, stimulating discussions, interactive implementations and for providing so many advices concerning the definition of the new activity paradigm, in the name of Science...

This research is supported by the CNRS.

References

[1] M. D. Mesarovic, Y. Takahara. *General Systems Theory: A Mathematical Foundation*. Academic Press, New York, NY., 1975.

[2] A. Muzy, J. J. Nutaro. Algorithms for efficient implementation of the devs and dsdevs abstract simulators. In *1st Open International Conference on Modeling and Simulation (OICMS) (OICMS'05)*, pages 273 – 279, 2005.

[3] Osman Balci. The implementation of four conceptual frameworks for simulation modeling in high-level languages. In *WSC '88: Proceedings of the 20th conference on Winter simulation*, pages 287–295, New York, NY, USA, 1988. ACM.

[4] G. Wainer and N. Giambiasi. Application of the cell-DEVS paradigm for cell spaces modeling and simulation. *Simulation*, 76:22 – 39, 2001.

[5] W. B. Lee, T. G. Kim. Simulation speedup for devs models by composition-based compilation. In *Summer Computer Simulation Conference (SCSC'03)*, pages 395 – 400, 2003.

[6] X. Hu and B. P. Zeigler. A high performance simulation engine for large-scale cellular DEVS models. In *Proc. of High Performance Computing Symposium (HPC'04)*, 2004.

[7] Bernard P. Zeigler. *Theory of Modeling and Simulation*. John Wiley, 1976.