Jul 1st, 12:00 AM

Cell-DEVS Modeling of Environmental Applications

Gabriel Wainer

Cell-DEVS Modeling of Environmental Applications

Gabriel Wainer

Dept. of Systems and Computer Engineering, Carleton University, Ottawa, ON. Canada.
gwainer@sce.carleton.ca

Abstract: Recently, different research teams used Cellular models for the analysis of environmental systems using Cellular models. Some of these techniques are based on Cellular Automata, which have some problems constraining its power, usability and feasibility for studying large complex systems. In other cases, the Cellular Automata have been combined with the DEVS (Discrete-Event Systems Specifications) formalism, which requires expertise in advanced programming techniques, visualization, distributed computing, etc. Instead, the Cell-DEVS formalism and the CD++ toolkit Cell-DEVS simplify the construction of complex cellular models by allowing simple and more intuitive model specification. We present the definition of different models, focusing on how to define such applications using Cell-DEVS methodology.

Keywords: DEVS, Cell-DEVS, Cellular models.

1. INTRODUCTION

In recent years, a number of research efforts have focused on presenting solutions for modeling and simulation of environmental systems using Cellular models. The most popular technique to study Cell spaces is Cellular Automata (CA), which have been widely used to describe a variety of complex systems (Toffoli [1987], Wolfram [2002]). A Cellular Automaton is organized as an n-dimensional infinite lattice of elements, each holding a state value and a computing apparatus. The state of each cell is changed by this device (a local computing function), which uses the present value for the cell and a finite set of neighboring cells to compute the new state.

CA popularity grew in the last few years, and they have been widely used in environmental sciences. Nevertheless, CA have problems that constrain its power, usability and feasibility to analyze complex systems. CA are restricted by the simplicity of their formal description (neighborhood, uniformity of the cells, one state per cell, closure of the system to external events, infinite lattices). CA use a discrete time base for cell updates, constraining the precision and efficiency of the simulated models. Thus, CA often need to be modified for simulation purposes.

The Cell-DEVS formalism defined (Wainer and Giambiasi [2002]) and the CD++ toolkit (Wainer [2002]) permit to overcome these problems. Cell-DEVS allows defining asynchronous cell spaces with explicit constructions for timing definition. This approach permits describing cell spaces as discrete event models, based on the formal specifications of the DEVS formalism (Zeigler et al. [2000]). Each cell in a cellular model is seen as a DEVS atomic model, and a procedure for coupling cells is defined based on the neighborhood relationship. Explicit delays can be used to define precise timing in each cell, which is defined by a local computing function combined with a delay construction (a sketch of Cell-DEVS models is presented in Figure 1). Only the active cells in the cell space are triggered, independently from any activation period, reducing compute time.

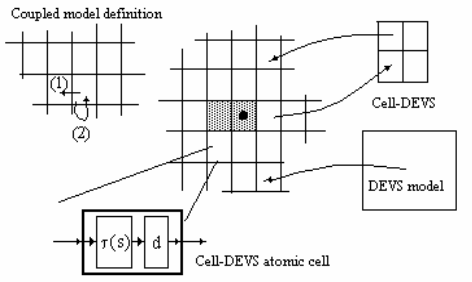

Figure 1. Informal definition of Cell-DEVS.

We will show how these facilities can help a modeler of environmental systems to build advanced models based on cell spaces. We will present different facilities provided by the formalism and the tools, and will discuss different examples of application in this field of application. We have put into consideration two important issues: how to keep the ability of CA to describe very complex systems using very simple rules (its main advantage), while using CD++ facilities to create a cell space.

2. CELL-DEVS AND CD++

Cell-DEVS was defined as an extension to CA combined with DEVS. The DEVS formalism provides a framework for the construction of hierarchical modular models. Basic models (called **atomic**) are specified as black boxes, and several DEVS models can be integrated together forming a hierarchical structural model (called **coupled**). Cell-DEVS defines a cell as a DEVS atomic model and a cell space as a coupled model. A Cell-DEVS atomic model is defined as:

$$TDC=< X, Y, S, N, delay, \delta_{ext}, \delta_{int}, \tau, \lambda, D > \quad (1)$$

A cell uses a set of *N* input values to compute its future state, which is obtained by applying the local function *t*. A *delay* function is associated with each cell, after which, the new state value is sent out. The user only needs to focus on the behavior of the *t* and *delay* functions (the rest of the functions are defined by the formal specifications of Cell-DEVS). After the basic behavior for a cell is defined, a complete cell space can be built as a coupled Cell-DEVS:

$$GCC = <Xlist,Ylist,X,Y,n,\{t_1,...,t_n\},N,C,B,Z > \quad (2)$$

A coupled Cell-DEVS is composed of an array of atomic cells (*C*), each of which is connected to the cells in the neighborhood (*N*). The border cells (*B*) can be provided with a different behavior than the rest of the space. The *Z* function defines the internal and external coupling of cells in the model. The *Xlist* and *Ylist* are used for defining the coupling with external models.

The CD++ tool implements both DEVS and Cell-DEVS theories. The tool is built as a hierarchy of models, each of them related with a simulation entity. Atomic models can be described using a graphical notation. A specification language allows defining the model's coupling, including the initial values and external events. The tool includes an interpreter for a specification language that allows describing Cell-DEVS models. The behavior specification of a Cell-DEVS atomic model is defined using a set of rules with the form: `POSTCONDITION ASSIGNMENTS DELAY PRECONDITION`. These indicate that when the `PRECONDITION` is satisfied, the state of the cell changes to the designated `POSTCONDITION`, and its output is `DELAY`ed for the specified time. If model's state variables need to be modified, the `ASSIGNMENTS` section can be used (optional). The main operators available to define rules and delays include: Boolean, comparison, arithmetic, neighborhood values, time, conditionals, angle conversion, pseudo-random numbers, error rounding and constants (i.e., gravitation, acceleration, light, Planck, etc.). At present, CD++ is able to execute models in single processor, parallel or real-time mode. The execution engine uses model's specifications, and it builds one object to control each component in the model hierarchy. The tool includes facilities to run quantized Cell-DEVS, including models with dynamic quantization. CD++ also permits defining *zones* with differentiated behavior in the cell space. Each zone is defined by a set of cells determined by the cell range $\{(x_1,..., x_n)...(y_1,..., y_n)\}$. Each zone is associated to a different set of rules. Using this capability, different zones into the same cellular model can present different behavior. The following example represents a surface tension model defined in Toffoli [1994].

```
[Tension]
type : cell          dim : (40,40)  delay
: transport        border : wrapped
neighbors : (-1,-1) (-1,0) (-1,1) (0,-1)
(1,-1) (1,0) (1,1) (0,0) (0,1)
localtransition : Ten-rules

[Ten-rules]
rule : 0 100 { statecount(0) >= 5 }
rule : 1 100 { t }
```

Figure 2. A Cell-DEVS specification in CD++

Figure 2 follows Cell-DEVS coupled model's formal definitions. In this case, *Xlist* = *Ylist* = { ∅ }. The set {*t1*, *t2*} is defined by *dim*, which specifies the size of the cell space (in this example, *t1*=*t2*=40). The *N* set is defined by the *neighbors* keyword. The border *B* is wrapped (cells in one border communicate with the ones in the opposite border). Using this information, the tool builds a cell space, and the *Z* translation function following Cell-DEVS specifications.

3. POLLUTION ON A BASIN: THE CASE OF THE VENETIAN LAGOON

On Bianchini et al. [1999] the authors present a model on the contamination of the Venetian lagoon, produced by substances like nitrogen and phosphorus. The goal is to learn about these properties in order to be able to control these substances in an ecosystem formed by various lakes, produced in the ecosystem because of industry influence. This permits to study how fauna and flora is affected by these substances, as several species do not resist pollution, and they tend to disappear, producing a change in the ecosystem. We reproduced this model with ease as a Cell-DEVS model, adding extra complexity with ease, as we will show in this section.

The model is represented as a cell space, in which each cell contains the speed of the water flowing in the cell (and its direction), and the level of contamination of the cell. Pollution is produced when the lake receives nitrogen from the exterior. The rules in the model represent how the substance contaminates. The conceptual idea is to define the diffusion of the polluting substance because of the water flow, which is determined by a velocity field. The model's rules are:

1) The value of the current cell is computed as (0,0)+∑ [((20 –(0,0))/20)*Si*Wi], in which the addition is carried out in all of the cells influencing the cell (0,0). In this case, *Si* represents the concentration of the pollutant in the cell i, *Wi* the speed on the cell, and *Si * Wi* is the contribution of a neighboring cell in the direction to cell (0,0). Finally, *(20 – (0,0)) / 20* represents the reception capacity of the current cell.

2) After evaluating rule l, and after consuming a delay representing the pollution rate, the new value of contamination of the cell is computed as (0,0) - [(20 - Si / 20) * (0,0)*WC], where *WC* is the velocity on the central cell, *Si* represents the concentration of the pollutant in cell i (the cell receiving pollution), and (20 - SC(t)) / 20 is the capacity of reception of the receiving cell.

3) After evaluating rules 1 and 2, the cell waits for the delay time. Then, we consider the cases in which one or more velocity directions in the surrounding cells enter within the cell, and water in the origin cell is zero. This is computed as (0,0) + ∑ [((20–(0,0))/ 20)*Si*Wi] - [(20-Si/20) * (0,0) * WC].

4) If 1)-3) are not executed (i.e., there is no neighboring cell pointing to the origin, and the speed of water in the cell is not affecting the origin cell), the current pollution value is maintained.

We built this model as a Cell-DEVS application, and run different test scenarios, discussed following. In the following figures, in which we distinguish water (blue cells) or pollution (represented in lighter yellow; darker cells are contaminated). On the first example presented in Figure 3, the model is fed with pollution from two different sources (built as standard DEVS models representing pollutant generators).

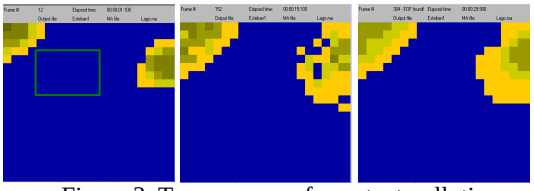

Figure 3. Two sources of constant pollution

In this case, we show the simulation results on a continuous focus of pollution during several hours (the factories discharge 560 l/h of pollutant). We modified the original model, and included a Cell-DEVS zone with a different behavior: in this case, we included subsurface vegetation, which makes the pollution to spread slower. This is implemented as a zone (marked by a square in the figure) in which the model introduces a different behavior than the rest of the cell space. We can see how the pollutant concentrates in the places where the pollutant is being discharged. As the velocity field and the presence of vegetation allow stationary water, diffusion is slow. As we can see, the differences between the second and third graphic (which represent 24 hours of simulated time) is not large. In the following figure, we show the results obtained when vegetation in the model is eliminated.

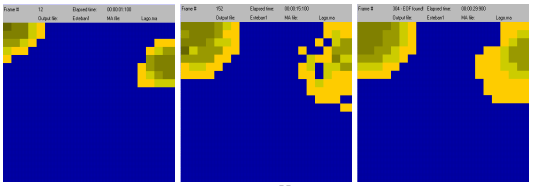

Figure 4. Constant pollution, no vegetation.

As we can see, the leftmost part of the model, as it does not contain any vegetation (which was the case in Figure 3), the pollution concentrates and expands more than in the previous case (in which the presence of plants favored the reduction of pollution in the area). The NE side of the figure is the same than in the previous case. In the following figure, we show the influence of the velocity field. Here, all the cells have a single direction of movement, using random equiprobable values. In this case, we do not include vegetation in the model.

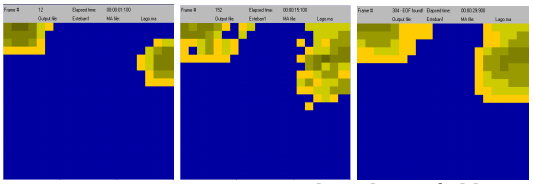

Figure 5. Variation in the velocity field

In this case, pollution expands more homogeneously, as water flows in every direction. Likewise, we can see that there are higher levels of contamination getting to farther places than in the previous case, polluting places distant from the source of pollution. The following figure presents the original model (with vegetation and slow probability of absorption of the pollutant by the plants), but only producing a unique input of pollutant at the beginning of the simulation (representing an accident in the plant, showing how the toxic elements will spread in the case of accidents).

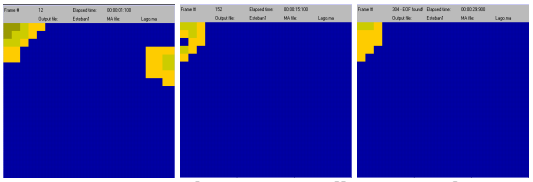

Figure 6. Behavior in a pollution accident.

In the first part of the figure, the pollution spreads following the rules in the model. In this version, there is vegetation in all the lake. When the source of pollutant stops, the contamination is slowly absorbed by natural factors, which degrades the contamination up to disappearing in most of the lake. We can see that, even the vegetation collaborates in eliminating the pollution, due to the hydrological

characteristics of the lake; there is a contamination region on the NW area that does not disappear or is absorbed by vegetation. The second part of the figure repeats the tests, but in this case, the vegetation has a higher rate of absorption. The final example we present here includes changes in the velocity field. In this case, there is displacement of water in the direction NE-SW, using the same flow rate.

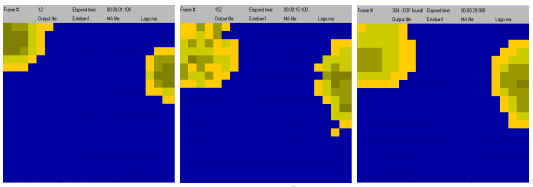

Figure 7. Water in direction NE-SW.

The main difference between the two examples introduced in Figure 7 is that in the second one we have eliminated plants. As we can see, even a small number of vegetation with low absorption rate influences the results obtained. In both cases, the pollution expands homogenously on the left (and not so evenly on the NE). This is due to the velocity field, which, on the left, influences the spread rate, reversing the diffusion effect. The only difference in the second example is a higher concentration of contaminant.

4. SIMULATING VEGETATION DYNAMICS

We now discuss the definition of dynamics of vegetation population, based on the work defined by Bandini an Pavese [2002]. In this case, sunlight, water, fertilizers are example factors that influence the growth of a vegetation population. The competitive nature of the population for resources during growth for survival is depicted in the model. Each cell represents a given portion of the yard, and contains multiple resources. If conditions are favorable, the cell can host a tree; otherwise, it will be empty. The tree may grow, survive, reproduce or die depending on the conditions. Trees growing closer to each other may have the disadvantage of losing nutrients to its neighbor. Environmental factors that may affect the growth such rain and fauna are also accounted for.

Each cell represents a square portion with specified dimensions, and it includes information about the seeds scattered in the cell by tree in the cell or the neighbors. At each update, the tree in the cell takes the nutrients needed, and it uses them to grow and produce seeds. If more nutrients are available, the tree stores them. Conversely, if not enough nutrients are present, then the tree uses the stored nutrients. A tree dies if it has no stored nutrients and there is none available in the neighborhood. There can only be one tree in a cell, which could produce seeds. Seeds can be scattered in the cell or the in neighbor cells. Each cell produces nutrients but cannot exceed the maximum value. The flow of the nutrient goes from the richer cells to poorer cells, so that bigger trees in a particular cell may make use of the nutrient in neighboring cells.

The model uses 3D cells to store different information, and each plane uses a hydrological model to determine the diffusion of water and nutrients. Each 3D cell is subdivided into six different planes to represent the attributes of the cell (water, nitrogen, potassium and other relevant resources). There are three types of trees: Locust, Pine, and Oak. Thus, a cell having the value (0, 0, {2, 100, 2.4, 1.1, 17, 2.3}) implies that the origin cell (0, 0, 0) has a tree of type Pine, with 100 ml water, 2.4 grams Nitrogen, 1.1 grams Potassium and 17 Joules sunlight. The tree height is 2.3 m.

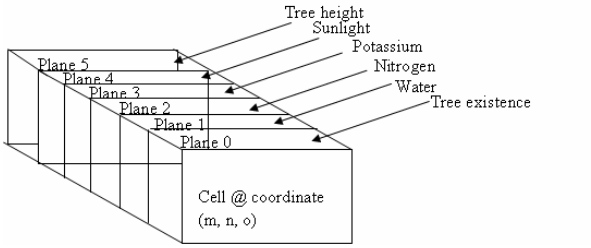

Figure 8. Cell representation with six planes

We use the eight neighboring cells in the same layer for each plane. In addition, five other neighbors are added to the *Height of the tree* plane, because the resources available at the cell (water, nitrogen, potassium and sunlight) influence the tree's height. Hence in total the neighbors' list have fourteen cells: the immediate neighbors on same plane, which exchange nutrients among them in the same plane, the height plane, accessing the nutrients below in order to decide the amount of growth, planes having to access the planes above and below in order to decide the amount consumed by the tree, and information about existence of the tree must be confirmed.

The vegetation population was simulated for a yard size of (10x10x6). The scaling factor can be decided as preferred during the simulation. Different zones are declared to restrict the space in the rules applied; hence, the other cells in the yard will not be affected. For example, updating the nitrogen should not affect the potassium or water layers. These zones depict a hydrological behavior: the water would be inserted along the cells indicated by {(9, 0, 1)..(9, 9, 1)}. Once inserted, they will diffuse and penetrate into the neighboring cells. Finally, zones for reproduction and update height are defined, where the height only involves the cells in the sixth plane and the reproduction involves all the cells in the zero planes.

The Cell-DEVS atomic model is defined as:

Vegetation = < X, Y, S, delay, δint, δext, τ, λ, D>

$X \subseteq \boldsymbol{R}$ is the set of external input events,

$Y \subseteq \boldsymbol{R}$ is the set of external output events,

S = {R, M, P, T, Z_T, U^G_T, U^S_T, R_T, M_T, G_T, S}, with R={Water, Sunlight, Nitrogen, Potassium}; M = maximum amount of each resource in the cell; P = amount of each resource produced by the cell at each update; T={0–No,1–Locust, 2–Pine, 3–OAK}; Z_T = size of the trunk in consideration; U^G_T = vector defining the amount of each resource needed at each update step by the tree to *grow;* U^S_T =vector defining the minimum amount of each resource the tree needs at each update step to *survive*; R_T =amount of resources stored by the tree; $M_{T=}$ maximum amount of each resource that contained by the cell and S is a vector defining the number of seeds present in the cell for each of the *l* species growing in the territory.

Delay = {transport; 1000 milliseconds }

τ = the rules of this model change the vegetation population, using these rules in each zone.

- *Update-Nitrogen*: initially the available nitrogen in the cell and the presence of a tree is evaluated. Then the amount required for the growth is subtracted from the available amount, and the nitrogen available in adjacent and current cell are added and equally distributed between neighbors. The concentration of nutrient changes with the number of dying vegetation and other external factors. The maximum amount of nitrogen for each cell is kept as the boundary, and checked during each update.
- *Update-Potassium*: equivalent to nitrogen rule.
- *Update-Water*: equivalent to nitrogen rule.
- *Update-Sun:* equivalent to nitrogen rule.
- *Reproduction*: a seed would be dropped in a cell with no tree present. Conditionally, trees in adjacent cells must have reached reproduction age, different for different types of trees. The seed survivability is higher for the seeds dropped later in time. The survivable seed of this cell would be the one dropped later in time, regardless of the number of seeds.
- *Update Height*: in a cell with a tree/seed, with enough resources to grow or sprout the growth would be enabled, depending on the tree type. If there are not enough resources then the tree dies immediately. It has been considered that the amount of resources requirement for the growth and death of each tree is different for different types of trees.

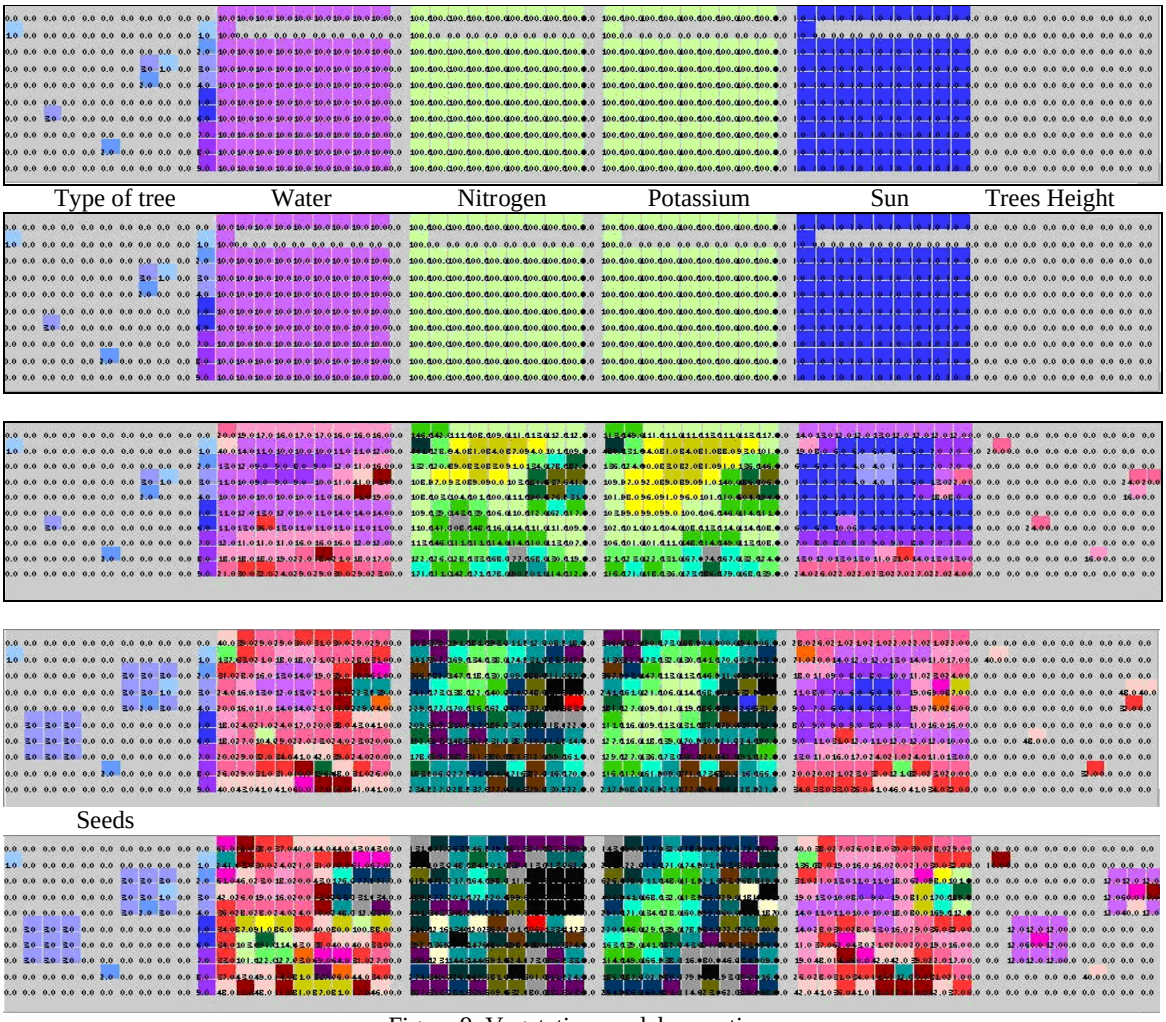

Figure 9. Vegetation model execution

The coupled model is defined as:

VegetationCM = <X,Y,X_{list},Y_{list},N,{t1,t2, t3},C,B,Z>

$X \subseteq$ T is the set of external input events; T = {water, nitrogen, potassium, sunlight}

$Y \subseteq$ T is the set of external output events; T = {growth}

Y_{list} = { (i, j, k) / i ∈ [0,t1], j ∈ [0,t2], k ∈ [0,t3]} is the list of output coupling. Where the i, j, k represent the index values of the cells (that couple with its neighbors) which are bound by t1, t2, t3 dimension.

X_{list} = { (i, j, k) / i ∈ [0,t1], j ∈ [0,t2], k ∈ [0,t3]} is the list of input coupling. Where the i, j, k represent the index values of the cells (that couple with its neighbors) which are bound by t1, t2, t3 dimension.

N= {(-1,-1,0), (0,-1,0), (1,-1,0), (-1,0,0), (0,0,0), (1,0,0), (-1,1,0), (0,1,0),(1,1,0),(0,0,-1),(0,0,-2),(0,0,-3),(0,0,-4),(0,0,-5),(0,0,1),(0,0,2),(0,0,3),(0,0,4), (0,0,5),(-1,-1,5),(0,-1,5),(1,-1,5),(-1,0,5),(0,0,5), (1,0,5),(-1, 1,5),(0,1,5),(1,1,5)}

These illustrated scenarios show the neighborhood list for the immediate neighbors in the same plane (nine of them) and their inverse list. The same applies for the other neighbors. Initially, we have a distribution of nutrients and some seeds on the ground. As we can see in figure 9, the concentration of resources change according to the rules defined for the model, and the trees grow in height as they consume the resources. As it can be seen, each of the resources changes in accordance with each type of the tree. The tree height keeps increasing and the concentration changes accordingly. Gradually, the cells are updated with the change in concentration of the resources, and tree grows with the consumption of resources. In the fourth slice in Figure 9, the trees have matured to the stage of reproduction. Two of the trees put seeds and in the next stage, they will grow if there are enough resources. The seeds contribute the growth of new trees in the last box.

5. CONCLUSION

We have showed how to apply the Cell-DEVS formalism for the construction of advanced models in the field of environmental science. Cell–DEVS allows describing physical and natural systems using an n-dimensional cell-based formalism. Input/output port definitions allow the definition of multiple interconnections between Cell-DEVS and DEVS models. Complex timing behavior for the cells in the space can be defined using very simple constructions. The CD++ tool implements the Cell-DEVS formalism and entitles the definition of complex cell-shaped models. We showed how to develop several Cell-DEVS models using the CD++ toolkit, which provides a general framework to define and simulate complex generic models.

CD++ simplifies the construction of complex cellular models by allowing simple and intuitive model specification. The CD++ logic rules facilitate the debugging phase and, consequently, reduce development time. Complex model modifications can now be easily and integrated to the models even by a non-computer specialist. It was shown that different kinds of applications to study the environment can be easily faced, allowing the study of complex environmental systems through simulation. The use of formal base improves the development, checking and maintaining phases, facilitating the testing and reuse of their components. The utilization of a discrete event formalism such as these ones can provide better precision and performance, while different visualization tools, simulators and an integrated development environment can enhance the modeling experience.

ACKNOWLEDGMENTS

Some of the models here presented were developed by different teams of students, including Sivaharan Thurairasa, Christy Gnanapragasam, Vijay Mahendren, Javier Ameghino and Esteban Fernandez Rojo.

REFERENCES

Bandini, S. and Pavesi, G. Simulation of Vegetable Population Dynamics Based on Cellular Automata. Proceedings of 5th International Conference on Cellular Automata for Research and Industry. Geneva, Switzerland. LNCS 2493. 2002.

Bianchini, A., Indovina, F. and Rinaldi, E. Cellular automata for the study of the diffusion of pollutants within the basins of the lagoon: the case of the Venetian lagoon. Proceedings of the 6th International Conference on Computers in urban planning and urban management. Venice, Italy. 1999.

Toffoli, T. Occam, Turing, von Neumann, Jaynes: How much can you get for how little? (A conceptual introduction to cellular automata). Proceedings of the International conference on Cellular Automata for Research and Industry. Rende, Italy. 1994.

Toffoli, T.; Margolus, N. *Cellular Automata Machines*. The MIT Press, Cambridge, MA. 1987.

Wainer, G. CD++: a toolkit to define discrete-event models. *Software, Practice and Experience.* 32(3), 1261-1306, 2002.

Wainer, G. and Giambiasi, N. N-dimensional Cell-DEVS. *Discrete Events Systems: Theory and Applications*. 12 (1), 135-157, 2002.

Wolfram, S. *A New Kind of Science*. Wolfram Media. 2002.

Zeigler, B., Kim, T. and Praehofer, H. *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press. 2000.