Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds

APPLICATION OF THE DEVS AND CELL-DEVS FORMALISMS FOR MODELING NETWORKING APPLICATIONS

Gabriel A. Wainer [1]
Misagh Tavanpour [1]
Emilie Broutin [1 2]

Department of Systems and Computer Engineering [1]
Carleton University Centre on Visualization and Simulation (V-Sim). Carleton University.
Ottawa, ON K1S 5B6, CANADA

Université de Corse [2]
UMR CNRS 6134 SPE
Campus Grimaldi
Corte, 20250, France

ABSTRACT

We present the use of DEVS and Cell-DEVS formalisms to model different approaches in networking applications. We discuss various applications of discrete event system specifications for modeling and simulation of Wireless networks and Wireless Sensor Networks (WSN). We discuss how to use the Cell-DEVS formalism to model a WSN for investigating on stochastic properties of malware propagation and the intrinsic characteristic of WSN. We also discuss the use of DEVS to model a cellular network including a wide geographical area, various Cells and varied User Equipment. Finally, we discuss how to use the cell DEVS formalism to model mobile networks, and how the Cell-DEVS formalism can be used to track mobile user movement in a covered area. The latter model tries to find out the number of Base Stations which cover a mobile user in different location of an area and how to improve QoS based on different configurations (in particular for the UEs near the cell borders).

1 INTRODUCTION

Existing forecasts reveal that the usage of computer networks has increased rapidly. For instance, the number of Internet world total users increased over 500% from 2000 to 2012 (Internet World Stats 2013). Due to this increased use of networks, different approaches have been introduced to improve user experience. At the same time, the need for analysis tools is also rising. These tools have been extended to support the Verification and Validation of new proposed approaches in networking and telecommunications.

An efficient way to analyze related issues in networks is through Modeling and Simulation. This approach allows studying and investigating protocols, standards, methods, and using the simulation results to provide precise information for decision-making. Different methods have been used to model and simulate networks. In this work, we explore the use of the DEVS formalism (Discrete EVent System specification) and Cell-DEVS with this purpose. DEVS (Zeigler et al. 2000) is a well-known formalism for modeling and simulation of complex systems. Cell-DEVS (Wainer 2009) is an extension of DEVS that includes the definition of cellular models with explicit timing delays.

We show the application of these methodologies for modeling and simulation of networking protocols and issues. We first introduce a model of malware propagation in wireless sensor network (WSN). These kinds of networks are deployed in open environments, and they use wireless communication for data transmission. These networks have resources constraints, such as limited power and limited memory. In WSN, security issues are very important, as they are more vulnerable to malicious activity due to their characteristics, including malware propagation. The consequences of virus or worm attacks can be very expensive. For example, the Blaster worm launch in 2003 has rapidly infected a large number of systems (Robiah et al 2010). The malware propagation model presented here describes the spatial-temporal pro-

cess of propagation. This can be helpful for setting up effective countermeasures. For example, it can show how to slow down the infection or which node should be fixed first to achieve faster recovery.

Wireless communication is ubiquitous these days, and one of the needs for further extension and development of wireless communication is user mobility, flexibility and reduced cost. We show how to model mobile networks and follow User Equipments (UE) movements. One of the main goals in cellular networks is to provide a better quality of service for subscribers, and different mobile generation standards have been introduced recently. The method used to allocate resources to UEs, has considerable effect on the user's experience. The UEs movement, and consequently, the location management have important impact on resource management; therefore, tracking the mobile terminals location in the covered area can be considered as a key issue in the resource allocation policies. Besides this, new standards for cellular networks such as LTE-Advanced, consider that the users location in each cell is important, and there are different techniques to provide high data rate for all the customers, even those close to the cells' borders. We show how the Cell-DEVS formalism has been used to model such networks, allowing tracking the users at each time to determine the number of base station's signal a user can receive.

2 RELATED WORK

Network modeling is a widely investigated area; we will show how to use the DEVS formalism as the underlying modeling mechanism. DEVS (Zeigler et al. 2000) provides a framework for the construction of hierarchical models in a modular fashion, allowing model reuse, reducing development and testing time. The hierarchical and discrete event nature of DEVS makes it a good choice to modeling and simulating networking algorithms. DEVS models are timed, which also enables us to define timing properties for the models under development. Each DEVS model can be built as a behavioral (atomic) or a structural (coupled) model. A DEVS atomic model is described as:

$$\mathbf{M = <X, S, Y, \delta_{int}, \delta_{ext}, \lambda, D >}$$

In the absence of external events, the model will remain in state $s \in S$ during $ta(s)$. Transitions that occur due to the expiration of $ta(s)$ are called internal transitions. When an internal transition takes place, the system outputs the value $\lambda(s) \in Y$, and changes to the state defined by $\delta_{int}(s)$. Upon reception of an external event, $\delta_{ext}(s, e, x)$ is activated using the input value $x \in X$, the current state s and the time elapsed since the last transition e. Coupled models are defined as:

$$\mathbf{CM = < X, Y, D, \{M_i\}, \{I_i\}, \{Z_{ij}\} , select >}$$

They consist of a set of basic models (*Mi*, atomic or coupled) connected through their interfaces. Component identifications are stored into an index (*D*). A translation function (*Zij*) is defined by using an index of influencees created for each model (*Ii*). The function defines which outputs of model *Mi* are connected to inputs in model *Mj*. The *select* function serves as tiebreaker for two simultaneous models.

A Cell-DEVS model is represented as a cell space, where each cell is represented as an atomic DEVS model. Each cell is connected to the neighboring cells. A delay mechanism in each cell (transport delay or inertial delay) is used to delay the propagation of state change events through the cell space, providing the means for defining complex temporal behavior. An Atomic Cell-DEVS can be defined as follows:

$$\mathbf{TDC = < X, Y, I, S, \theta, N, d, \delta int, \delta ext, \tau, \lambda, D >}$$

Where *X* is the set of external input events; *Y* is the set of external output events; *I* represents the definition of the model's modular interface; *S* is the set of possible states for a given cell; θ is the definition of the cell's state variables. *N* is the set of values for the input events; *d* is the delay of the cell; δint is the internal transition function; δext is the external transition function; τ is the local computing function; λ is the output function, and *D* is the duration function.

A Coupled Cell-DEVS model is built by connecting a number of Atomic Cell-DEVS models together into a cell space (including 2D and 3D cell spaces). The borders of the cell space can be either wrapped, in which case the cells at the border from one side of the cell space are considered neighbors to the cells at the border on the opposite side of the cell-space, or non-wrapped, in which case the border cells must have special rules defined by the modeler. A formal definition of Coupled Cell-DEVS is:

$$\mathbf{GCC = < Xlist, Ylist, I, X, Y, n, \{t1,….,tn\}, N, C, B, Z, select >}$$

Where *Xlist* is the input coupling list; *Ylist* is the output coupling list; *I* represents the definition of the model's modular interface; *X* is the set of external input events; *Y* is the set of external output events; *n* is the dimension of the cell space; *{t1,....,tn}* is the number of cells in each of the dimensions. *N* is the neighborhood set; *C* is the cell space; *B* is the set of border cells; *Z* is the translation function; and *select* is the tie-breaking function.

DEVS has been used for modeling and simulation of networking algorithms. For instance, (Hojun, Zeigler, and Kim 2008) introduced a DEVS framework for solving parameter optimization for a link-11 gateway. They used the framework to find the most optimized parameter to adjust the sampling rate, which leads to improved performance. In (Chuan et al 2007) the authors present an event-driven intrusion detection architecture using the DEVS formalism and DEVS-Java for the implementation and the validation of the model. In (Davoust et al. 2012), the authors introduced a DEVS framework for the simulation of peer-to-peer (P2P) file sharing that provides extensible and reusable models for the file sharing protocol and for the behavior of the peers. In (Ahmed et al. 2005), a tool for the modeling and simulation of networks was presented. It provides a library of models to create network topologies. Using these models, networks devices such as routers and hubs can be modeled.

Different efforts have focused in modeling wireless networks. In (Balya et al. 2004), the authors presented a modeling method for analyzing the AODV protocol, including an algorithm for solving deadlock problem between multiple pairs of senders/receivers. In (Qela et al. 2009), a WSN was simulated using Cell-DEVS. In (Antoine-Santoni et al. 2008), the authors provide a model for the analysis of WSN performance in terms of routing management, energy consumption and CPU activity.

Different models for malware propagation in WSN have been proposed in recent years (see, for instance, (Shengjun and Junhua 2009), (Khouzani et al. 2010) and (Badakhshan and Arifler 2007)).

The work introduced by (Ali et al. 2007) focuses on movement in mobile networks. In this case, the authors showed that keeping track of the mobile terminals in 3G cellular networks has an important role in resource management. The authors also proposed a dynamic movement-base location management scheme for 3G networks, and a simulator based on MATLAB. Likewise, (Xiao et al. 2004) discussed how location management provides a key factor to provide consistence service for user equipments while they are traveling in the covered area. In this work, an analytical method has been used to evaluate location management schema.

In our case, we are interested in studying new protocols using DEVS and Cell-DEVS, and we have used the CD++ toolkit as workbench (Wainer 2009). CD++ implements DEVS and Cell-DEVS theories. The defined models are built as a class hierarchy, and each of them is related with a simulation entity that is activated whenever the model needs to be executed. New models can be incorporated into this class hierarchy by writing DEVS models in C++, overloading the basic methods representing DEVS specifications: external transitions, internal transitions and output functions. In the following sections, we show the design and implementation of varied algorithms using these concepts and tools. The models and simulations are available at http://www.cell-devs.sce.carleton.ca/.

3 CELLULAR NETWORK

Mobile networks or cellular networks are kind of radio networks distributed over land areas. Each of these land areas is known as a Cell. Each Cell has at least one fixed transceiver called Base Station (BS). These BSs communicate with each other through X2 interfaces. The Cells support radio coverage over a geographic area by cooperating with each other. This enables portable transceivers (called user equipments - UE) to communicate with each other via fixed transceivers. The left part of Figure 1 presents a simplified mobile network. In this section, show how to use DEVS to design such a model for mobile networks. The right part of Figure 1 illustrates a DEVS coupled model representation. The model includes an area with numerous cells. Each cell has one BS and it can have various UEs.

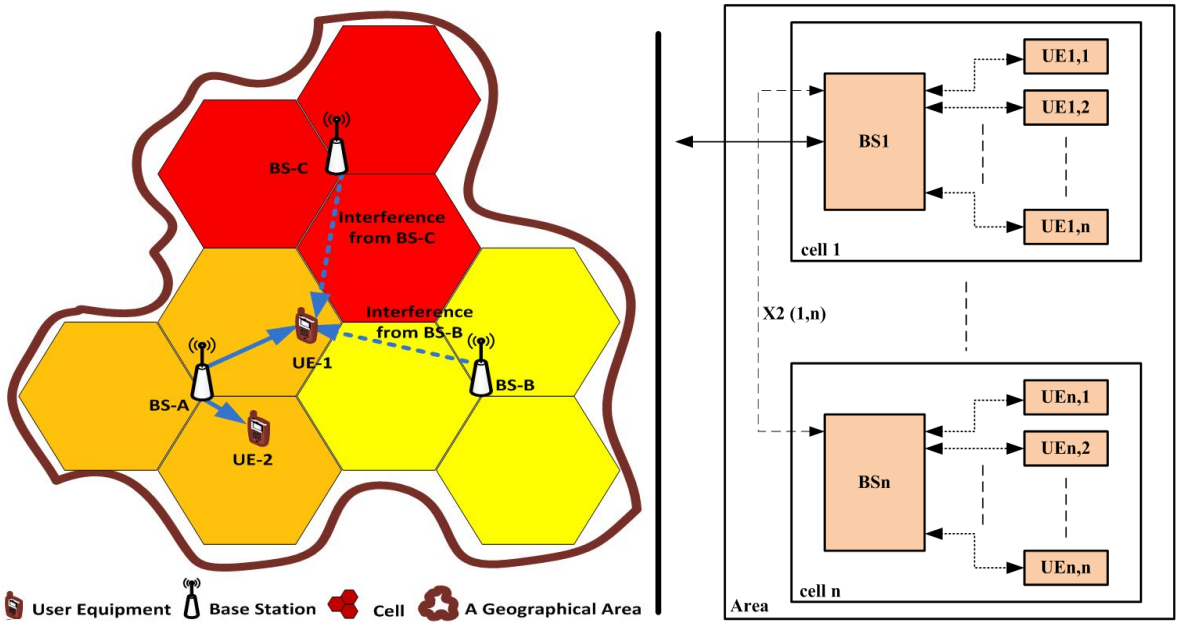

Figure 1: Sample mobile network and its simplified DEVS model.

In this model, the geographical area and cells have been considered as coupled models, whereas the BSs and UEs have been defined as atomic models. The BS atomic model can be in one of these two states: *Idle* or Receive Packet (*RecPack*). While *Idle* state, a BS waits for an input. When an input event occurs, the model executes its external transition function, processes the received packet and it changes to *RecPack*. Based on the packet information, the BS can forward the received packets to their destination. After that, the internal transition function is called and it resets the model to *Idle*.

An UE is defined using four states: *Idle, SendPack, RecPack* and *RecAck*. When an incoming event arrives, the external transition function performs a transition into *RecAck* or *ReckPack* (depending on the type of the packet received). When a packet is received by an UE, an acknowledgment message is sent back to the UE that has generated the packet (*RecPack*). Figure 2 shows a DEVS graph of this model.

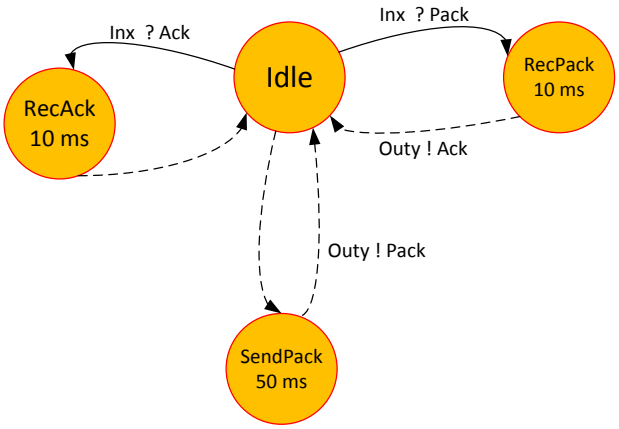

Figure 2: DEVS graph of the UE atomic model.

We mentioned that area and cells are coupled models. These coupled models are defined according to the structure that was described earlier in Figure 1. Both atomic and coupled components functionality has been tested separately. Figure 3 shows that how in the final step a model file has been produced in order to define all of the components and their interconnections. A top down approach has been employed

to define coupled components of the target model. Figure 3 demonstrates that in this mobile network model hierarchy, the area as the top model consists of number of cells. The connections among the top model components have been considered by defining links between them. In the next step, each of area's coupled components has been defined.

```
components: Cell1 Cell2 ...
in : In               out : Out

Link : Out4@BS5Port1 In2@BS3Port2
...
Link : Out3@BS3Port1 In3@BS3Port2
...

[Cell1]
components : BS3Port1@BS3Port1 UE1@UE1 ...
in : In1

Link : Out1@BS3Port1 In@UE1
...
Link : Out3@BS3Port1 Out1

[Cell2] ...
```

Figure 3: Simplified Model file of an Area.

An UE can only communicate with its serving BS, but a BS can communicate with other BSs plus the UEs in its range. The BSs communicate with each other in two general cases: when they want to exchange control information, and when they try to forward the UEs data packets. Likewise, the BSs may send/receive packets to/from the UEs in their covered area.

```
00:00:00:25 In 910000  → Data packet from a UE in another area
00:00:00:40 In 910002  → ...
...
00:00:02:15 In 910010
00:00:02:35 In 910012
00:00:02:55 In 910014

00:00:00:035 out 910001 → ACK for the source UE that is in another area
00:00:00:050 out 910003 → ACK for the source UE that is in another area

00:00:00:100 out 120002
00:00:00:150 out 120004 → Data packet for a UE that is in same area as the source UE
00:00:00:200 out 120006
00:00:00:250 out 120008
...
00:00:00:500 out 120018
00:00:00:550 out 120020
00:00:00:600 out 120022
...
00:00:01:000 out 120038
00:00:01:025 out 910005
...
00:00:01:165 out 120042
00:00:01:215 out 120044
00:00:01:265 out 120046
```

Figure 4: Test results of a sample UE.

As we mentioned, an UE may generates traffic packets and sends them for other UEs as the destination of those packets. Figure 4 shows the output packets of a UE during simulation time. In response to that, the destination UE sends ACK packets for the source UE of the received packet. The following out-

puts in Figure 4 show the reaction of one of the UEs to the input packets. In these simulations the packet's sequence is according to even numbers and the acknowledge message for a received packet has been defined as *PackNum+1*. As it can be seen in Figure 4, the UE received packets from other UEs in outside and inside the area.

```
00:00:01:35 In1 120006  →  00:00:01:045 out3 120006
00:00:01:55 In3 120009  →  00:00:01:065 out1 120009
00:00:02:15 In1 120016  →  00:00:02:025 out3 120016
00:00:02:35 In1 910013  →  00:00:02:045 out2 910013
00:00:03:15 In1 120018  →  00:00:02:045 out2 910013
00:00:03:35 In3 120019  →  00:00:03:045 out1 120019  ...
```

Figure 5: Test results of a sample BS.

Figure 5 demonstrates the reaction a BS to the received packets. This BS forwards packets to their final destination based on the packet information.

4 MALWARE PROPAGATION IN WSN

Wireless Sensor Networks (WSN) are ad-hoc networks consisting of spatially distributed sensing and processing devices (Cunha et al. 2005; Healy et al. 2008). WSN are used in many different applications, such as medicine, transportation and urban monitoring, traffic control, military, environment and habitat monitoring, energy management, industrial applications, etc. (Quela et al. 2009). WSN differs from classical network by the fact that they have limited power, they are highly distributed and they are self-organized. Therefore, new techniques that take into account these specifications have to be developed.

Like any other network, WSNs can be attacked by malware such as viruses or worms. Malware attacks on classical network have been widely investigated; nevertheless, techniques from these studies cannot be applied to WSN.

```
[malware]
 type : cell            dim : (20,20,2)       delay : transport
 border : nowrapped     neighbors : ...

 localtransition : malware-rule
 zone : energy-reduction-rule { (0,0,1)..(19,19,1) }

[malware-rule]
 rule : 3 100 { (0,0,0) = 0 and stateCount(3) >= 1 and ( random <= 0.3 ) }
 rule : 4 100 { (0,0,0) = 0 and stateCount(4) >= 1 and ( random <= 0.3 ) }
 rule : 2 100 { (0,0,0) = 3 and ( random <= 0.01 ) and (0,0,1) > 1 }
 rule : 2 100 { (0,0,0) = 4 and ( random <= 0.01 ) and (0,0,1) > 1 }
 rule : 3 100 { (0,0,0) = 4 and (0,0,1) > 1 }
 rule : 4 100 { (0,0,0) = 3 and (0,0,1) > 1 }
 rule : 2 100 { (0,0,0) = 2 and (0,0,1) > 1 }
  ...
 rule : 0 100 { t }

[energy-reduction-rule]
 rule : {(0,0,0)*.8 }  100   {(0,0,0)>1 and (0,0,-1)=3}
 rule : {(0,0,0)*.8 }  100   {(0,0,0)>1 and (0,0,-1)=4}
 rule : {(0,0,0)*.95}  100   {(0,0,0)>1 and (0,0,-1)=2}
 rule : {(0,0,0)*.99}  100   {(0,0,0)>1 and (0,0,-1)=0}
 rule : 0              100   {(0,0,0)<=1}
```

Figure 5: Cell-DEVS malware coupled model.

In (Song and Ping Jiang 2008), the authors analyzed the process of malware propagation in WSN using cellular automata. Based upon this work, we show a Cell-DEVS model for malware propagation in WSN. This model is composed of maximum N stationary and identical sensors. The sensors are randomly

placed on a rectangular two-dimension grid composed of L*L cells. Each cell is occupied by at most one sensor node. A sensor node can establish wireless links with other nodes within a circle of radius r.

A sensor node can be into one of these states:

- Susceptible (represented by cell value 0): The sensor node can become infected. This can happen with probability β when a packet containing the malware is received from an infected neighbor.
- Dead (Cell value 1): Sensors have restrained power that decreased during wireless communication, so a node can run out of energy. This can occur at a rate γ.
- Recovered (Cell value 2): sensors can recover from an infected state with probability δ.
- Infected (Cell value 3 or 4): The node tries to spread the malware to its neighbors.

A cell in the grid without any node sensor is considered *dead*. A node consumes energy at the highest rate in the *infected* state because it broadcasts the malware to other nodes. In the *susceptible* state, the node consumes energy at the lowest rate.

The Cell-DEVS model uses a 3D lattice (20*20*2 cells). The first plane describes the sensor nodes. A second plane represents the energy status of corresponding node in the first plane.

Figure 5 presents the coupled model for this application. Each plane uses its own rules. For example, in the first plane, the rule in the first line defines a state change for sensor nodes. This rule means that a node goes from susceptible state to infected state with 30 % probability (β=0.3). In the rules for the same plane, lines three and four illustrate the state change of a cell from *infected* to *recovered* with 1% probability (δ =0.01) as long as its energy level is greater than 1. In the second plane, the first rule shows that if the state of the cell (sensor node) is *infected*, then its energy is reduced to 80% of the original.

Initially, all the sensors start the simulation with full power and one node is infected with malware, this infected node is represented by a gray cell in simulation results shown on the left part of Figure 6. In the next simulation step, the energy of the *infected* sensor decreases to 80. Likewise, the energy of the other sensors in the *susceptible* state decreases to 99. The malware spreads and infects two more sensors as it can be seen in Figure 6b. In each simulation step, the energy of the infected sensors is reduced to 80% of the previous value and the energy of the susceptible sensors is reduced to 99% of the previous value. The malware spreads further and infects more sensors. Later, one of the infected sensors turns to the *recovered* state, as it can be seen in the figure 6d (light gray cell). Note that the energy of the recovered sensors is reduced by 95% in each simulation step.

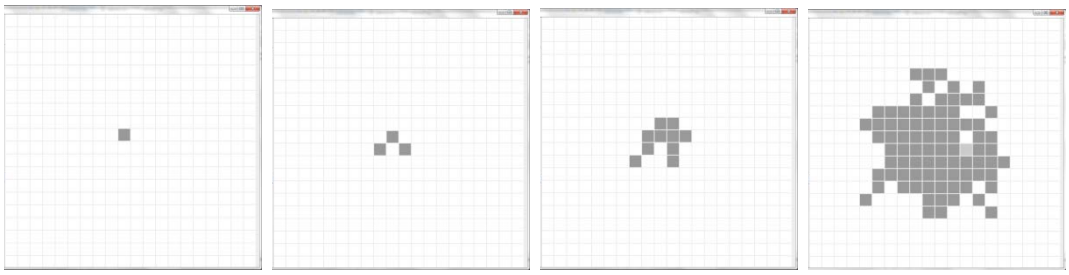

Figure 6: The effect of infected cell on its neighbor cells.

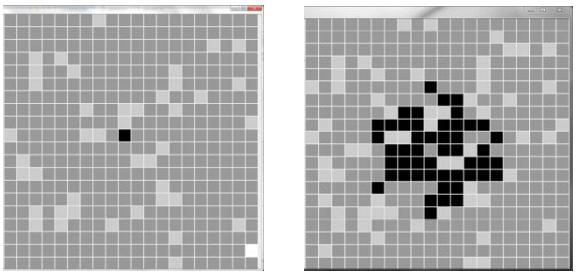

Figure 7: Dead cells progression.

As we can see on Figure 7a, after malware advances, the first infected sensor becomes *dead* after running out of energy (black cell). Figure 7b shows the progression of the dead cells.

5 NETWORK COVERAGE

In Section 3, we introduced a model of a mobile network. As discussed earlier, in a mobile network, a high data rate is relatively easy to maintain when the UE is close to its serving BS. In this situation, the received signal at the UE side has a high signal strength (i.e., the signal to noise ratio is high). On the other hand, as the distances between an UE and its serving BS increases, it is more difficult to maintain the data rate. The most challenging situation is when an UE is close to the Cell's edge (UE1 in Figure 1). In this case, when an UE travels around the Cell borders, it can sense the neighboring Cell's BSs signal in addition to its serving BS signal. In this situation, the connection between the UE and its serving BS suffers of two main problems:

- Lower signal strength, because of the distance between UE and its serving BS.
- Higher interference level from the neighboring BSs, as the UE is closer to them.

We have used Cell-DEVS to model a network like the one presented in Figure 1. This model tries to find out how many BSs can provide network coverage for each mobile and fixed UE. In other words, this model determines how many BSs signals can be received by each of UEs in different locations (*Note*: from now, *Cell* refers to the previously defined concept in section 3and *cell* refers to Cell-DEVS cell).

Consider a 2D area in which the network coverage is provided by a number of Cells. The UEs travel within the area base on a random path. After each movement, their new position, number of BSs that cover them at the new position, and the signal strength (according to their distance from the serving BS) are recalculated. This model uses five variables stored in different planes: *Presence, Movement, UE signal strength, Number of BSs* and *signal strength*. This model is not wrapped and there are no external inputs or outputs (except initialization information). Figure 8 shows how these are defined using CD++.

```
[coverage]
type : cell        dim : (13,13,5)   delay : transport
border : nowrapped  neighbors : . . .   % which neighbors are accessible?
initialCellsValue : Coverage.val
localtransition : Presence %{(0,0,0) .. (12,12,0)}
zone : Movement {(0,0,1)..(12,12,1)}   zone : SigStrUE {(0,0,2)..(12,12,2)}
zone : BSNum    {(0,0,3)..(12,12,3)}   zone : SigStr   {(0,0,4)..(12,12,4)}
```

Figure 8: Initial setting of Cell-DEVS model of mobile network.

In the first plane, we keep track of the UE's position. At the beginning of the simulation, the number of the UEs is chosen randomly the values can be one or zero representing the presence or absence of an UE in that cell. At the beginning of the simulation, the model considers a random number of UEs, and it assigns a random location to each of them. Figure 9 shows the rules to control the Presence plane's cell. The first rule shows that in absence of network coverage in a cell, the model assigns the value zero. The second rule illustrates that if an UE wants to stay in same cell for the next step, the model assigns the value one to the cell. The second groups of rules implement movement priority among UEs. The first rule in this part shows that an UE wants to go to the E if the destination cell is empty. If all the conditions are satisfied, then the UE in the cell will leave this cell and the model will assign a zero to this cell for the next step. The second rule describes a movement to a cell in the S. Here, the model should check if there is an UE to the W of destination cell, which wants to move to the destination cell. In this case, the movement to the E has higher priority than a movement to the S. Again, if all the conditions are satisfied, it means that the UE will leave the cell and the model assigns zero to this cell. The second group of rules to check the situations in which a cell will be empty. The third groups of rules check the conditions in which a cell will be occupied by UEs. In this case, a cell should be empty and it should have network coverage. In addition, there should be at least one request from UEs in the neighboring cells.

```
[Presence]
...
rule : 0 100 {(0,0,0)=1 and (0,0,3)=10}
rule : 1 100 {(0,0,0)=1 and (0,0,1)=6}
...
rule : 0 100 {(0,0,0)=1 and (0,1,0)=0 and (0,0,1)=2}
rule:0 100 {(0,0,0)=1 and (1,0,0)=0 and (0,0,1)=3 and ((1,-1,0)=0 or (1,-1,1)!= 2)}
...
rule: 1 100 {(0,0,0)=0 and (0,0,3) != 10 and (((0,-1,0)=1 and (0,-1,1)=2) or
  ((-1,0,0)=1 and (-1,0,1)=3) or ((0,1,0)=1 and (0,1,1)=4) or ((1,0,0)=1 and
      (1,0,1)=5))} ...
```

Figure 9: The Presence plane rules.

The second plane contains the direction of the next move for the UEs. Every time a new random value is assigned to the UEs. These values determine the direction of next move of the UEs. An UE can go to the E (white cells in the plane 2 in Figure 12), S (light gray cells), W (medium gray cells), N (dark gray cells) or stay in same position (black cells). If more than one UE want to go to the same target, the ones with higher priority complete their move, and the rest of the UEs with the same destination stay in their current location. The priority is chosen according to the order of direction mentioned (descending priority in order E, S, W and N for the movement of the UEs). Figure 10 shows the rules that we have used to control Movement plane's cell. The first rule in this plane says that when there is no coverage there will be no any active UE in this cell, so there is no need to calculate the next movement. When there are UEs in the neighboring cells, and at least one of them wants to move to that cell, then the model calculates the next movement for the UE. The second group of rules in plane 2 addresses this issue. The third group of rules is used when the current UE in a cell will stay in the same cell. The model calculates another movement for the current UE for the next step.

```
[Movement]
rule : 0 100 { (0,0,2) = 10 }
...
rule : {randInt(4) + 2} 100 { (0,0,-1) = 0 and (0,-1,-1) = 1 and (0,-1,0) = 2 }
rule : {randInt(4) + 2} 100 { (0,0,-1) = 0 and (-1,0,-1) = 1 and (-1,0,0) = 3 }
rule : {randInt(4) + 2} 100 { (0,0,-1) = 0 and (0,1,-1) = 1  and (0,1,0) = 4  }
rule : {randInt(4) + 2} 100 { (0,0,-1) = 0 and (1,0,-1) = 1  and (1,0,0) = 5  }
...
rule : {randInt(4) + 2} 100 { (0,0,-1) = 1 and (0,0,0) = 6 }
rule : {randInt(4) + 2} 100 { (0,0,-1) = 1 and (0,0,0) = 2 and (0,1,-1)  = 1 }
rule : {randInt(4) + 2} 100 { (0,0,-1) = 1 and (0,0,0) = 3 and (1,0,-1)  = 1 }
rule : {randInt(4) + 2} 100 { (0,0,-1) = 1 and (0,0,0) = 2 and ((0,1,-1)  != 1 and
(0,1,-1)  != 0) }
rule : {randInt(4) + 2} 100 { (0,0,-1) = 1 and (0,0,0) = 3 and ((1,0,-1)  != 1 and
(1,0,-1)  != 0)} ...
```

Figure 10: The Movement plane rules.

The third plane computes signal strength of the UE according to their host cell. Normally, when an UE is in the border of the Cell, it receives a weak signal from its serving BS. In the beginning of this section, we mentioned two major problems for Cell edge UEs. In these simulations, we consider that the mobile network has employed the Coordinate Multipoint (CoMP) technique to overcome those problems. Then some of Cell edge UEs (those ones that work in CoMP mode) can receive their serving BS signal with higher signal to noise ratio. Figure 11 shows the related rules that we have used in the *SigStr* plane. Based on the first group of rules in Figure 11, when there is no coverage in a cell or if no UE in a cell is in the next step, there is no need to calculate signal strength in that cell. The second group of rules is used to calculate signal strength in a cell that will be occupied by a UE.

The fourth and the fifth planes include information about the number of BS, which can support certain cell and signal strength on that particular cell. They just require refreshing previous values at each step.

```
[SigStrUE]
rule : 0 100 { (0,0,1) = 10 }
rule : 0 100 {(0,0,-2)=0 and (0,-1,-1)!=2 and (-1,0,-1)!=3 and (0,1,-1)!=4 and
          (1,0,-1)!= 5}
rule : 0 100 {(0,0,-2)=1 and (0,0,-1)=2 and (0,1,-2)=0}
rule : 0 100 {(0,0,-2)=1 and (0,0,-1)=3 and (1,0,-2)=0 and ((1,-1,-2)=0 or
           (1,-1,-1)!= 2) }
...
rule : 25 100 { (0,0,2) = 20 }
rule : 26 100 { (0,0,2) = 21 } ...
```

Figure 11: The SigStr plane rules.

In this model, when an UE moves to a location with no coverage (none of the BSs covered that location), it will be omitted from the network. Figure 12 shows an example of this kind of UEs. According to plane 0 information in step 5, there is a UE in the cell (8,0,0) that wants to move to the north ((8,0,1) = dark gray). According to the information from other planes, there is no network coverage on destination cell. Consequently, this UE will be omitted from the network in the next step.

When a UE wants to go outside of the network area, the model discards this movement; it keeps the UE in the same cell, and it plans for another move. As an example, according to Figure 12, this scenario has happened for the UE in the cell (0,8,0).

Considering the movement of the UEs with same destination, the UEs avoid collision according to the movement priority. Based on Figure 12 in the step 5, two UEs in the cells (10, 8, 0) and (11, 9, 0) have same direction. According to the information provided by plane 2, the latter UE wants to go to the W and the former UE wants to go to the S. In that situation base on the priority rules, the UE in the cell (10,8,0) will complete its movement and the other one does not move.

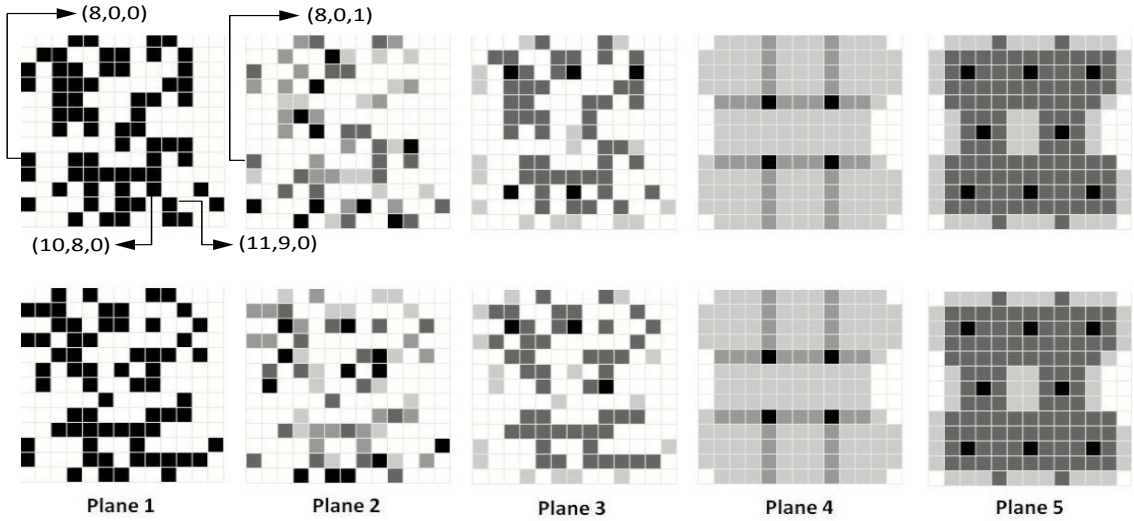

Figure 12: Steps 5 and 6 in simulation.

6 CONCLUSION

We provided an overview of the use of the DEVS and Cell-DEVS formalisms for modeling network applications. In this paper, we have presented three applications and used CD++ to develop them.

In the first application, we used DEVS to model a cellular network. We have represented all the components (UE, BS…) using atomic and coupled models and ran some tests to validate the modeling.

We also showed a Cell-DEVS model that investigates the behavior of malware in a WSN. This model shows how fast a malware can spread and disable a WSN due to the specific characteristics of the WSN (Limited power and memory for example). Some simulations were performed and results are presented graphically. Then, we used Cell-DEVS to track user's movements in a coverage area. We have modeled

the movement of UEs in a covered area, this model can determine how many BS signals a UE can receive in its current position.

The results obtained show how DEVS and Cell-DEVS can be used to model this kind of problems. The complexity of the problem can be simplified, and can be implemented effectively utilizing the CD++ toolkit and Cell-DEVS. This shows that mapping of traditional algorithms onto Cell-DEVS can lead into new ideas in theory and implementation of algorithms. The models can be easily improved by adding extra features. It is possible to insert physical objects, such as mountains, high-rise buildings etc. in the model. Addition of such phenomena is straightforward in Cell-DEVS because different models can be easily coupled.

ACKNOWLEDGMENTS

This work has been partially funded by NSERC and Ericsson Canada. The authors wish to thank Aizaz Chaudhry who authored some of the models discussed in the paper. We also want to acknowledge Gary Boudreau and Ron Casselman at Ericsson Canada for their support to the project.

REFERENCES

Ahmed, M. A. E., K. Yonis, A. Elshafei, G. Wainer. 2005. "Design and Implementation of a Library of Network Protocols in CD++". In *Proc. of 38th Annual Simulation Symposium*. San Diego, CA.

Ali, S., M . Ismail, K. Mat. 2007. "Development of a mobility management simulator for 3G cellular network". In *Proc. of Telecomm. and Malaysia Intl. Conf. on Communications*. Penang, Malaysia.

Antoine-Santoni T., J. Santucci, E. De Gentili, B. Costa. 2008. "Discrete Event Modeling and Simulation of Wireless Sensor Network Performance". SIMULATION 84 (2-3): 103–121.

Badakhshan, M., D. Arifler. 2007. "Simulation Based Analysis of Spreading Dynamics of Malware in Wireless Sensor Networks". *Procedings of Sensor Technologies and Applications*. Valencia, Spain.

Balya, B., U. Farooq, G. Wainer. "DEVS modeling of mobile wireless ad hoc networks". *Simulation Modeling, Practice and Theory*. Elsevier. Vol. 15, No. 3, pp. 285-314. 2007.

Castro, R., E. Kofman, and G. Wainer. 2010. "A Formal Framework for Stochastic DEVS Modeling and Simulation". SIMULATION 86: 587--611.

Chuan F., P. Jianfeng, Q. Haiyan, J.W. Rozenblit. 2007."Alert Fusion for a Computer Host Based Intrusion Detection System*". In proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS '07)*. 433,440. Tucson, AZ.

Cunha, R., A. Silva, and A. Loreiro. 2005. "Simulating Large Wireless Sensor Networks Using Cellular Automata". *Proceedings of the 38th Annual Simulation Symposium ANSS'05*. San Diego, CA.

Davoust, A., G. Wainer, and B. Esfandiari. 2012. "DEVS simulation of peer-to-peer file-sharing". *In High Performance Computing and Simulation (HPCS)*, 357–364. Madrid, Spain.

Healy, M., T. Newe, and E. Lewis. 2008. "Wireless Sensor Node Hardware: A Review". *In Proceedings of the 7th IEEE Sensors Conference*. 621-624. Lecce, Italy.

Hojun, L, B.P. Zeigler, K. Doohwan. 2008. "A DEVS-based framework for simulation optimization: Case study of Link-11 gateway parameter tuning*" In proceedings of Military Communications Conference (MILCOM 2008)*. 1-7. San Diego, CA.

Internet World Stats. http://www.internetworldstats.com/stats.htm. Last checked: April 2013.

Khouzani, M. H. R., S. Sarkar, E. Altman. 2010. "Maximum Damage Malware Attack in Mobile Wireless Networks".*In proceedings of INFOCOM 2010*, 1-9, San Diego, CA.

Nekovee, M. 2007. "Worm epidemics in wireless ad hoc networks". New Journal of Physics 9 (6): 189.

Nguyen H. , Y. Shinoda. 2009. "A novel analytical framework to model malware diffusion in heterogeneous wireless networks" *In Proceedings of 10th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WOWMOM 2009*, 1-10, Kos, Greece.

Qela, B., G. Wainer, and H. Mouftah. 2009. "Simulation of Large Wireless Sensor Networks using Cell-DEVS*", In Proceedings of the 2009 Winter Conference (WSC)* 3189 – 3200, Austin, TX.

Robiah, Y., S.S. Rahayu, S. Sahib, M.M. Zaki, M.A. Faizal, R. Marliza. 2010. "An improved traditional worm attack pattern". *In Proceeding of International Symposium in Information Technology (ITSim)*. Kuala Lumpur, Malasia.

Shengjun, W., C. Junhua. 2009. "Modeling the Spread of Worm Epidemics in Wireless Sensor Networks*". In Proceedings of 5th International Conference on Wireless Communications, Networking and Mobile Computing, WiCom '09*. Bejing, China.

Song, Y., and G. Ping Jiang. 2008. "Modeling malware propagation in wireless sensor networks using cellular automata*". Proc. of Intl. Conf. on Neural Networks and Signal Processing*, Nanjing, China.

Viani, F., G. Oliveri, M. Donelli, L. Lizzi, P. Rocca, A. Massa. 2010. "WSN-based solutions for security and surveillance" In *Proceedings of Microwave Conference (EuMC*). Paris, France.

Wainer, G. 2009. "Discrete-event modeling and simulation; a practitioner's approach". Taylor & Francis.

Xiao, Y., Y. Pan, J. Li. 2004. "Design and Analysis of Location Management for 3G Cellular Networks". IEEE Transactions On Parallel And Distributed Systems, vol. 15, no. 4.

Zeigler B P, H. Praehofer T.G Kim. 2000. Theory of modeling and simulation: Integrating discrete event and continuous complex dynamic systems. Second edition: Academic Press.

AUTHOR BIOGRAPHIES

GABRIEL A. WAINER (SMSCS, SMIEEE) received the M.Sc. (1993) and Ph.D. degrees (1998, *with highest honors*) at the University of Buenos Aires (UBA), Argentina, and Université d'Aix-Marseille III, France. In July 2000, he joined the Department of Systems and Computer Engineering at Carleton University, where he is now Full Professor. He has been a visiting scholar at ACIMS (The University of Arizona); LSIS (CNRS), and INRIA (Sophia-Antipolis), France. He has been invited professor at the UCM, UPC (Spain), Université Paul Cézanne, Université de Nice (France). He is the author of three books and over 270 research papers; he edited nine other books, and was a PC member/organizer of over 120 conferences, being one of the founders of SIMUTools, SimAUD and the Symposium of Theory of Modeling and Simulation. He has been appointed as Program Chair of the Winter Simulation Conference in 2017. Prof. Wainer is the VP Conferences, Was VP Publications, and a member of the Board of Directors of the SCS. He is Special Issues Editor of SIMULATION, member of the Editorial Board of IEEE Computing in Science and Engineering, Wireless Networks (Elsevier), and Journal of Defense Modeling and Simulation. He has been the recipient of various awards, including the IBM Eclipse Innovation Award, SCS Leadership Award, and various Best Paper awards. He has been awarded Carleton University's Research Achievement Award (2005-2006), the First Bernard P. Zeigler DEVS Modeling and Simulation Award, the SCS Outstanding Professional Award (2011), Carleton University's Mentorship Award (2013) and the SCS Distinguished Professional Award (2013). His e-mail and web addresses are <gwainer@sce.carleton.ca> and <www.sce.carleton.ca/faculty/wainer>.

MISAGH TAVANPOUR is a Ph.D. student (EE) at Carleton University. He obtained a B.S. in Computer Engineering (2006) at the Islamic Azad University, Iran, and a M.A. Sc (EE) at the Department of Computer Engineering, Science and Research University, Tehran, Iran (2009). His thesis focused on topologies and source mapping for QoS in NoC. He obtained varied experience in simulation during his Masters and undergraduate studies (where he focused on traffic engineering with Multi Protocol Label Switching, simulation of the MPLS and IP networks). His email address is <misagh@sce.carleton.ca>

EMILIE BROUTIN is a postdoctoral fellow at Carleton University. She received the Licence (2005), Master (2007) and Ph.D (2012) degrees in Computer Science from the Université de Corse.
Her thesis focused on Discrete Event Modeling and Simulation of natural complex systems using a multi-layered architecture. She is now working on mobile phone simulation applications using a RESTful server. Her email address is <emilie.broutin@univ-corse.fr>