Discrete Event Modeling and Simulation of Wireless Sensor Network performance

T. Antoine-Santoni[1], J.F. Santucci[1], E. De Gentili[1], B. Costa[1]
[1]University of Corsica - UMR CNRS 6134 Quartier Grossetti, BP 52 - 20250 Corte - FRANCE
{antoine-santoni, santucci, gentili, bcosta}@univ-corse.fr

***Abstract* : The wireless distributed microsensor networks profit of recent technological advances and it seems essential to understand precisely these systems. Modeling and simulation appear like an essential aspect to predict the Wireless Sensor Network specific behavior under different conditions. We want to provide a new approach of modeling, simulation and visualization of Wireless Sensor Network using a discrete event approach. Described by Zeigler in the 70 's, the Discrete Event system Specification is ideal to describe the asynchronous nature of the events occuring in WSN. We try to provide a basis model to analyze WSN performance, as routing management, energy cunsumption or relative CPU activity. Our approach use a detailled definition of node oriented components and it wants to bring some ways to visualize the network at different level of abstraction.**

***Index Terms*— Modeling, Simulation, Visualization, DEVS, WSN, performance.**

I. Introduction

Advances in hardware technology and engineering design have led to reductions in size, power consumption, and cost. This has enabled compact, autonomous nodes, each containing one or more sensors, computation and communication capabilities, and a power supply. Networks of wireless sensors are the result of rapid convergence of three key technologies [1]:

- Computing/Internet : computing power is becoming small and inexpensive enough to add to almost any object. networks of computers facilate collaboration trough information and resoure sharing
- Sensor : miniaturization, micromachining and low cost leads to smaller sizes, low power, lower costs. Alows to monitor with higher granularity. many types of sensors and more on the way
- Wireless/ Antennas : Spans a host ot technologies including Bluetooth and WiFi networks, cellular and satellite communications.

These recent tehcnologial advances have led to the definition and use of Wireless Sensor Network (WSN). The sensor nodes are usually scattered in a sensor field [2] as shown in Figure 1. Each of these scattered sensor nodes has the capabilities to collect data and route data back to the sink. Data are routed back to the sink by a multihop infrastructureless architecture through the sink. The sink may communicate with the task manager node via Internet or satellite. The design of the sensor network as described by Figure 1 is influenced by many factors, including fault tolerance, scalability, production costs, operating environment, sensor network topology, hardware constraints, transmission media, and power consumption.

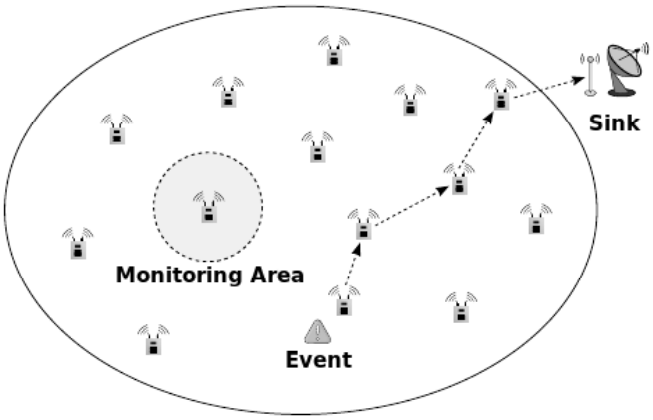


Figure 1. Sensor nodes in a sensor field

A sensor node combines the abilities to compute, to communicate and to sense [3]. In a sensor network, different functionalities can be associated with the sensor nodes [4]. In earlier works, all sensor nodes are assumed to be homogenous, having equal capacity in terms of computation, communication and power. However, depending on the application a node can be dedicated to a particular special function such as relaying, aggregation.

The goal of sensor is to send collected data, usually via radio transmitter, to a command center (sink or Base Station) either directly or through a data concentration center (a gateway)

Based of node description in [5] [2], the main components of sensor consist of a sensing unit, a processing unit, a transceiver, and a power unit as shown in Figure 2.

In order the user to understand the behaviour of Wireless Sensor Network we have to point out the following five information :

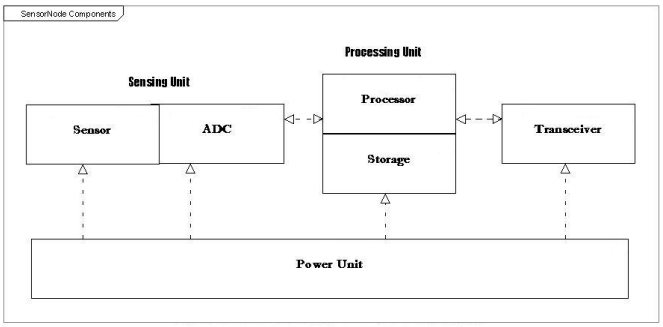


Figure 2. Components of a sensor node

1) The communication tool is represented by the antenna and its role is to send some information on the channel;
2) Memory is the unit for storage action of information evolving in the node. Information have two sources : information coming from an other node and information coming from sensoboard in environmental monitoring case. In these both conditions, information are treated by processor ;
3) Processor treats all information in the node. CPU manages activity in the mote and reacts according to instruction type.
4) Battery defines lifetime of a node. Each component according to realized action consumes some energy. Energy consumption exists also in sleep state.
5) Sensor board regroups monitoring activities. It can transmit some information collected by sensor but also transmits message alert if some critics thresholds are reached.

Modeling and simulation appear to be an essential aspect to understand the behavior of Wireless Sensor Network under specific conditions. The network simulation for sensors is a challenging problem as it has faithfully to model the constraints hardware and energy, which is typical with sensor nodes and also have to model various aspects exclusive to sensor networks. The hierarchical nature of DEVS makes it perfect for describing a system like sensor mote. The discrete-event nature improves the execution performance of a model like this due to the asynchronous nature of the events occurring in WSN. Some works exist for the modelling of Wireless ad-hoc networks using DEVS. In [6], the authors describe how to use the Cell-DEVS formalism in order to model routing protocol Ad hoc On Demand Distance Vector (AODV). In this paper, DEVS is used to formally specify discrete events systems using modular description. This strategy allows the reuse of tested models, improving the safety of the simulations and allowing reducing of development time. As it is discrete event formalism, it uses a continuous time base, which allows accurate timing representation, and reduces CPU time requirements. This very interesting work leans on the DEVS formalism in order to study the routing in wireless adhoc networks. In [7], a coupling between the NS-2 simulator (Ns [8], also popularly called ns-2, in reference to its current generation, is a discrete event network simulator) and the DEVS formalism is clearly presented. This paper describes how the behavior of a sensor node's application and its environmental behaviors such as battle fields have been defined using DEVS modeling. Furthemore the authors point out the roles of networking protocol behaviors which are assigned to NS-2 since NS-2 has well-designed network protocol libraries. However there is no modular aspects concerning the components involved in the sensor's behavior and thus its seems difficult to implement specific environmental scenario. According to these previous remarks we choose to define all components of Wireless Sensor Network using DEVS formalism.

The rest of the paper is organized as follows : Section 2 introduces briefly the Wireless Sensor Network area. In Section 3 we present the DEVS formalism. Section 4 present the DEVS formalism based approach we defined in order to describe the behaviour of Wireless Sensor nodes. The implementation and the validation of the proposed approach through results of simulation examples are detailed in Section 4. Finally, in Section 5 we give some conclusions and directions of future research works.

II. Overview of WSN Simulation

Nowadays, Wireless Sensor Network research has different focus and several fields of application: channel access control, routing protocol definition, network management, QoS, energy consumption or CPU activity. We can find different simulators to represent activity and performance of WSN.

SensorSim [9] extends the ns-2 network simulator with models of sensor channels, accurate battery and power consumption. Each node has a sensor stacks that acts as a sink to the signals in the sensor channels, accurate battery and power consumption.

Atemu [10] is a software emulator for AVR processor based systems. Along with support for the AVR processor, it also includes support for other peripheral devices on the MICA2 sensor node platform such as the radio. Atemu can be used to perform high fidelity large scale sensor network emulation studies in a controlled environment. Though the current release only includes support for MICA2 hardware, it can be easily extended to include other sensor node platforms. It allows for the use of heterogeneous sensor nodes in the same sensor network. Atemu can't represent different activities of hardware components because it uses an high abstraction level.

TOSSIM [11] and PowerTOSSIM [12] are two important simulators which can describe correctly routing protocol, node applications or energy consumption but they are strongly dependents of TinyOS and can't represent generic framework for heterogeneous platforms.

In [13], Glonemo can be considered like a close approach of our work. Indeed, Glonemo bring some solutions as the MAC layer for description of Wireless sensor node however certain parameters appear uncertain as CPU activity, general energy consumption, sensing activity.

SENS [14] is an application-oriented wireless sensor network which models ad-hoc static nodes. It provides models for a limited set of sensors, actuators, a model for the environment and a framework for testing applications. SENS appears like

	Components	Customizable	Modular	Generic	Environment	Topology	Energy
TOSSIM/PowerTossim	+ +	-	-	- -	-	- -	+
SensorSim	-	+	+	+	-	+	+
SENS	-	+	+ +	+ +	+	Fuzzy	-
Glonemo	+	-	-	-	++	-	- -
ATEMU	-	-	+ +	- -	-	-	- -
Avrora	+	-	-	-	-	-	+ +

Table I
COMPARAISON OF EXISTING APPROACHS

a suitable WSN simulator however some characteristics like addition of new models of sensor and modeling arbitrary ubiquitous computing environments are missing.

It is particularly difficult to find a generic, customizable in easy way, modular simulator or a model able to represent behavior of a node and able to generate particular environmental scenario. We resume these different aspects for each existing approach illustarted by the Table I.

The need is to have a simulator able to represent sensor node at different abstraction level, which be able to describe components behavior in particular conditions. Modular aspects of components in sensor model don't exist clearly. In this paper we focus on representation capacity of DEVS formalism. This possibility to distinguish different abstraction level is clearly essential to allow in the one hand the definition of components activity of node and on the other hand general behavior in the network

III. DEVS FORMALISM

Based on systems theory, DEVS formalism was introduced by Professor B.P. Zeigler in the late 70s. It allows a hierarchical and modular way to model the discrete event systems. A system (or model) is called modular if it possesses the input and output ports permitting interaction with its outside environment. In DEVS, a model is seen as a "black box" S which receives and broadcasts messages on its input and output ports. This section deals with the basic notions of the DEVS formalism. The Discrete Event System Specification (DEVS) formalism introduced by Zeigler [15] provides a means of specifying a mathematical object called a system. Basically, a system has a time base, inputs, states, outputs, and functions for determining next states and outputs given current states and inputs [15]. The DEVS formalism is the simple way in order to characterizes how discrete event simulation languages may specify discrete event system parameters. It is more than just a means of constructing simulation models. It provides a formal representation of discrete event systems capable of mathematical manipulation just as differential equations serve this role. Furthermore by allowing an explicit separation between the modeling phase and simulation phase, the DEVS formalism is the best way to perform an efficient simulation of complex systems using a computer.

In the DEVS formalism, one must specify:

1) basic models from which larger ones are built, and
2) how these models are connected together in hierarchical fashion.

Basic models (called atomic models) are defined by the following structure:

$$AM = < X, S, Y, \delta int, \delta ext, \lambda, t_a >$$

Where,
X is the set of input values,
S: is the set of sequential states,
Y: is the set of output values,
δint is the internal transition function dictating state transitions due to internal events,
δext the external transition function dictating state transitions due to external input events,
λ is the output function generating external events at the output,
and t_a is the time-advance function which allows to associate a life time to a given state.

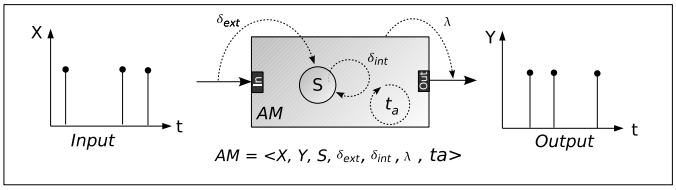


Figure 3. DEVS atomic model

Figure 3 represents an AM atomic model with its output data Y calculated according to input data X. The AM atomic model has a state variable S that can be reached during the simulation. The functions δ_{ext}, λ, δ_{int} and t_a respectively allow the model's change of state when an external event occurs on one of those outputs (external transition function), the disposal of the output Y (output function), the model's change of state after having given an output (internal transition function) and finally the determination of the duration of the model's state (time advance function).

The behavior of an atomic model is illustrated as follows: the external transition function describes how the system changes state in response to an input. When an input is applied to the system, it is said that an external even has occurred. The next state s' is then calculated according to the current state s. The internal transition function describes the autonomous (or internal) behavior of the system. When the system changes

state autonomously, an internal event is said to have occurred. The next state *s'* is calculated only according to the current state s. The output function generates the outputs of the system when an internal transition occurs. The time advance function determines the amount of time that must elapse before the next internal event will occur, assuming that no input arrives in the interim.

An atomic model allows specifying the behavior of a basic element of a given system. Connections between different atomic models can be performed by a coupled model (CM) [15]:

$$CM = < X, Y, D, \{M_i\}, \{I_i\}, \{Z_{i,j}\} >$$

Where,
X is the set of input values,
Y is the set of output values, D is the set of model references, For each $i \in D$, M_i is an atomic model, I_i is the set of influences of model $Z_{i,j}$ is the i to j translation function (output function).

A coupled model, tells how to couple (connect) several component models together to form a new model. This latter model can itself be employed as a component in a larger coupled model, thus giving rise to hierarchical construction.

The coupled models are defined by a set of sub-models (atomic and/or coupled) and express the internal structure of the system's sub-parts thanks to the coupling definition between the sub-models.

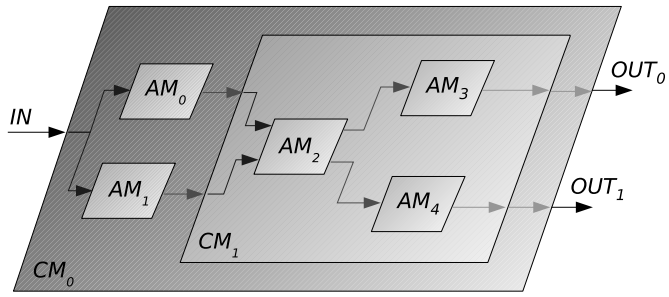


Figure 4. DEVS coupled model

Figure 4 shows an example of the hierarchical structure of coupled model CM_0 which has an input port IN and two output ports OUT_0 and OUT_1. It contains the atomic sub-models AM_0, AM_1 and also the coupled model CM_1. The latter can encapsulate other models such as atomic models AM_2, AM_3 and AM_4. A coupled model is specified through the list of its components (AM_0, AM_1, AM_2, AM_3, AM_4 and CM_1), the list of its internal couplings ($AM_0 \rightarrow CM_1$ and $AM_1 \rightarrow CM_1$), the list of the external input couplings ($IN \rightarrow AM_0$ and $IN \rightarrow AM_1$), the list of the external output couplings ($CM_1 \rightarrow OUT_0$ and $CM_1 \rightarrow OUT_1$) and the list of the sub-model's influence ($CM_1 = \{AM_0, AM_1\}$ or CM_1 and influenced by AM_0 and AM_1).

DEVS formalism is mainly used for the description of discrete event systems. It constitutes a powerful modeling and simulation tool permitting a system modeling on several levels of description as well as the definition of the models' behaviors. One of DEVS formalism's important properties is that it automatically provides a simulator for each model. DEVS establishes a distinction between a system modeling and a system simulation so as any model can be simulated without the need for a specific simulator to be implemented. Each atomic model is associated with a simulator in charge of managing the component's behavior and each coupled model is associated with a coordinator in charge of the time synchronization of underlying components. A simulator is associated with the DEVS formalism in order to exercise coupled model's instructions to actually generate its behavior. The architecture of a DEVS simulation system is derived from the abstract simulator concepts associated with the hierarchical and modular DEVS formalism.

One of the main interests in DESV formalism is the fact that it allows an explicit separation between the modeling and simulation part. This means that we can define the model representing the behaviour of a given system without having to consider the simulation phase.

IV. WSN MODELING APPROACH

It seems important to represent the different basics hardware components of the node. We have developed a generic approach allowing to define out-of-context behaviors of components in order to reuse these behaviour in an in-context manner [16]. A context-out model is an abstraction of a model. It represents a behaviour allowing it to be store in a model library. A context-in model is a context-out model extracted from a library and formatted in order to be directly reusable in its environment. This generic approach leads us to define the behaviour of different components. We try to delimit the different reaction of the node units to move towards the description of a general behaviour of a sensor using a discrete event formalism DEVS. The advantages of this formalism for description of complex system in discrete-event scale appear clearly in number field of research however definition of sensor network and in particular sensor node don't exist. As sensor networks gain more importance in the research communities, it's very crucial to show the advantages of DEVS formalism and to have a simulator with a modular structure. The use of this formalism in accordance with its definition implies for this research area two essential points: a modelling specification step and consequently a clear interpretation of simulations results in the real world and a non ambiguous operational semantic step allowing the introduction of a formal specification of mechanics of simulation using an abstract simulator.

In this section we will first deal with the kinds of messages which are going to be exchanged between models of a WSN in the sub section IV-A. We will then introduce the different atomic and coupled models required in order to model the behaviour of a WSN. These models are derived from the components of a given node of the WSN as shown in Figure 2. Sub-section IV-B is dedicated to the description of an atomic model called AM COM which has been defined in order to represent the communication involve in a node of the WSN. The Coupled model, called CM Processor, which is a key component in order to deal with routing information is explained in sub-section IV-C. Then we describes the following atomic

models : atomic model AM Battery in sub-section IV-D, the atomic model AM Memory in sub-section IV-E, the atomic model AM SensorBoard and the atomic model AM Env in sub-section IV-F, allowing to deal with the behaviour of the battery, the memory and the interaction with the environment involved in a sensor. Finally the reader will find in sub-section IV-G the description of a coupled model called SENSOR which describes the overall behaviour of a sensor own to the couplings involved in coupled models.

A. The messages involved in WSN modeling

The messages which are exchanged between the components of a WSN involve different kinds of information in order the simulation to be able to efficiently describe the behaviour of such a network. A message is going to involve the following ten fields :

<Origin, Sender, Destination, Ndid, Type, Hop, Link, Data, Port> where :

- **Origin** defines the node which is the source of this message. Parent, this field is one of characteristic of reliable route protocol. It determines the node nearest to the basic station, the highest in routing table.
- **Sender** defines the node which sent this message.
- **Destination** defines the destination of message which has been treated by a node ;the destination can be the sink or an other kind of node.
- **Ndid** : defines the node which is going to be identified by a nodeID which correspond at an identifier of a node group.
- **Type** defines the action of the different components of the system.
- **Hop** defines a parameter of a reliable route protocol. However it can be used for other routing protocol.
- **Link** defines an attribute of reliableRoute protocol which indicates the quality of connectivity between two nodes and is very important for the definition of the routing table.
- **Data :** is composed by the different information carried by the node as *(i)Temp* which indicates temperature parameter coming from Sensor board,*(ii)Humidity, (iii) Pressure, (iv) GPS, (v) Conso* defines the last energy value of Origine node and (*vi*) *Acitivity* determines the last CPU activity of Origine node.
- **Port :** it defines for each messages the output port for a message according to the DEVS rules.

Data contains several informations : environnemental data (Temperature, humidity rate, GPS) , processor activity, energy cunsumption of the sender node.

In order to model the behaviour of a WSN we have defined a set of atomic models and coupled models which will be described in detail in the following sub-sections : the atomic model AM COM in sub-section IV-B, coupled model CM Processor involving the atomic model AM Net, the atomic model AM Flash and the atomic model AM Processor in sub-section IV-C, the atomic model AM Battery in sub-section IV-D, the atomic model AM Memory in sub-section IV-E,the atomic model AM SensorBoard in sub-section IV-F and the coupled model CM Sensor in sub-section IV-G.

We have also defined different types of message in order to describe the action which should be performed by the receiving atomic model. We give the main types of message we have defined. Even if the following list is not exhaustive the reader will be able to discover the main defined types of message and the associated action to be performed by the receiving atomic model :

- ***Router*** Message for AM Net : the associated action concerns the routing information ;
- ***BSCollect*** Message for AM Sensorboard : the associated action will deal with collect environmental data ;
- ***MemCollect*** Message for AM Memory : the associated action will consists in storing information ;
- ***ACK*** Message for AM Net : the associated action will performed of a received information ;
- ***WhiteFlag*** Message for Net :the associated action will concern the architecture discovery signal and updateof routing table ;
- ***DEAD*** Message for all Models : the associated action will point out the fact that no energy is present ;

B. Description of the atomic model AM COM

The atomic model AM COM is used an atomic model for representation of communication in a node. The goal of this atomic model is to address message towards good nodes according to the routing table in the coupled model CM Processor.

Figure 5. Atomic Model COM

In Figure 5, we have only represented a link on input port or on output port with another sensor node ; however it's possible to have more links depending of connected nodes. We have defined different states in order to describe the behaviour of such an device :

- Receipt state describing the arrival of a message coming on Inport1 from a node or Base Station (BS),
- Transmit state describing the arrival of a message coming out of a sensor node towards an other node or the base station BS,
- Busy state correspondding to the state of transition when a message is treated by MC processor,
- Free state describing the fact taht when there is no activity in node (when node is listening the channel),
- DEAD state describing the fact that there is no battery in sensor.

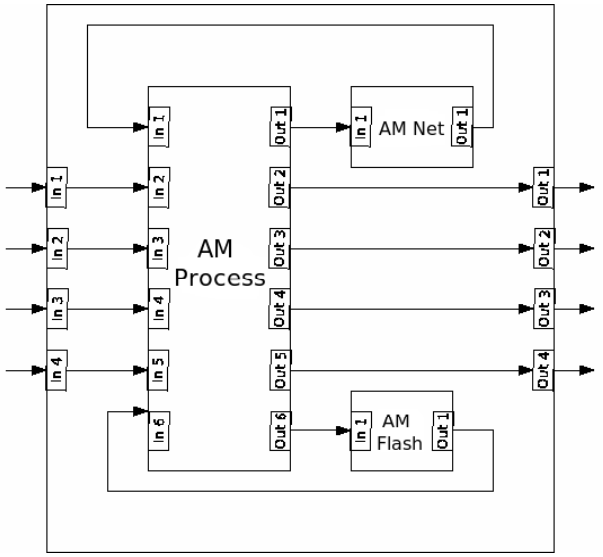


Figure 6. Coupled Model Processor

C. Description of the coupled model CM Processor

The coupled model CM Processor is shown on Figure 6. It is one of the key of our approach : all messages coming from an atomic model AM COM or an atomic model AM Sensorboard are necessarily taken into account by this model in order to deal with routing information routing information. The atomic model AM Processor allows to manage all messages and all components. It is difficult to represent all action of processor but we have been able to propose a solution using a generic approach. The defined coupled model CM Processor is able to represent the simplest representation of a generic Operating System. Indeed we make the choice to decompose the behaviour of the coupled model CM Processor into three Atomic models: (i) the atomic model AM Process which allows to represent action management of Operating System (OS) and Processor, (ii) AM Net which allows to manage the Network aspect and (iii) the atomic AM Flash which is a space to store information but also is able to adapt the system at new parameters, by example a new type of message. These three atomic models have only three states :

(i) the Busy state allowing to point out that a model is in action,

(ii) the Free state allowing to point out that a model is in sleep mode,

(iii) the Dead state when there is not enough energy in the node.

When a message comes, the atomic model AM Process sends it to the atomic model AM Net that is a model allowing to describe routing management. AM Flash is a very simple atomic model. It can stock some informations like a new node ID.It can also allows the definition of new types of messages, unknowed by the system. The management of new types of messages with the AM Flash allows the user to redefine new types of applications.

The atomic AM Process's behaviour allows to deal all messages in the model ; we can express relative activity of a node by counting of each action performed by the atomic model AM Process.

D. Description of atomic model AM Battery

The atomic model AM Battery as illustrated on Figure 7 is an atomic model connected to all models representing the components of the sensor. Each time there is an action using some energy, the atomic model AM COM, the coupled model CM Processor and the atomic model AM Sensorboard send a message to the atomic model AM Battery.

For representation of enrgy consumption, we use for these first experiments a linear mode based on [17]. In linear model, the battery is treated as linear storage of current. The maximum capacity of the battery is achieved regardless of what the discharge rate is. The simple battery models allow user to see the efficiency of the user's application by providing how much capacity is consumed by the user. The remaining capacity C after operation duration of time t_d can be expressed by the following equation :

$$C = C' - \int_{t=t_0}^{t_0+t_d} I(t)dt\,, Eq.(1)$$

where C' is the previous capacity and I(t) is the instantaneous current consumed by the circuit at time t. The linear model assumes that $I(t)$ will stay the same for the duration t_d, if the operation node of the circuit does not change for the duration t_d.

When a special value called size reached 0, a Dead message is sent to all components and therefore all models enter the DEAD phase. All input ports are blocked and it is impossible for all models to change their state. Let us precise that all models have a common important state called DEAD phase. When a sensor model enters in this special phase, it cannot act any more in the network. This particularity is essential for networking management.

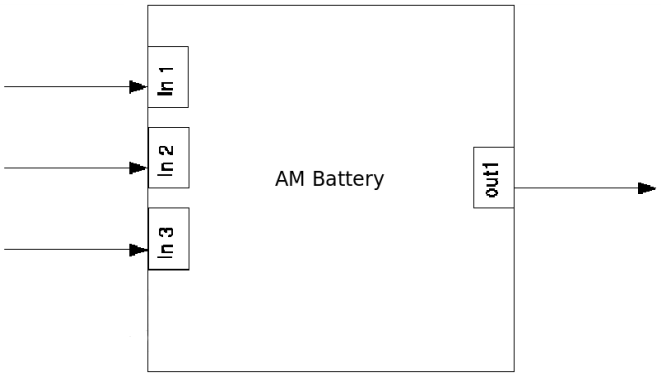


Figure 7. Atomic Model Battery

For the energy cunsumption of each component, we use the Table II.

SYSTEM SPECIFICATIONS		
Currents		**Example Duty Cycle**
Processor		
Current (full operation)	8 mA	1
Current sleep	8 µA	99
Radio		
Current in receive	8 mA	0.75
Current transmit	12 mA	0.25
Current sleep	2 µA	99
Logger Memory		
Write	15 mA	0
Read	4 mA	0
Sleep	2 µA	100
Sensor Board		
Current (full operation)	5 mA	1
Current sleep	5 µA	99
Computed mA-hr used each hour		
Processor		0.0879
Radio		0.0920
Logger Memory		0.0020
Sensor Board		0.0550
Total current (mA-hr) used		**0.2369**
Computed battery life vs. battery size		
Battery Capacity (mA-hr)		Battery Life (months)
250		1.45
1000		5.78
3000		17.35

Table II
ENERGY CONSUMPTION FOR EACH COMPONENTS [18]

This energy conusption depends of the duty cycle. The Duty cycle is the proportion of time during which a component, device, or system is operated. Suppose a node processor operates for 1 second, then is shut off for 99 seconds, then is run for 1 second again, and so on. It runs for one out of 100 seconds, or 1/100 of the time, and its duty cycle is therefore 1/100, or 1 percent. In a periodic phenomenon, the ratio of the duration of the phenomenon in a given period to the period.

$$D = \frac{\tau}{T}$$

where :

D is the so-called duty cycle;

τ is the duration that the function is non-zero;

T is the period of the function.

In this considiration, the defined system uses the following Duty Cyle to calculate energy consumption, illsutrated by the Table III.

Components	τ (s) for $T = 4$ (s)	Duty cycle
Processor	0,04	1%
COM (receipt/trasnmit)	1/1,33	25%/75%
Sensor Board	0,04	2,5%
Memory	0,01	0,25%

Table III
DUTY CYCLE FOR EACH COMPONENTS

E. Description of atomic model AM Memory

The atomic model AM Memory is a simple atomic model as shown on Figure 9, that allows storage of environmental data by CM Processor. However, a coupled model CM Processor can have different kind of actions on atomic model AM Memory according to the type of message.We may point two messges (MemCollect and StoreData) in sub-section IV.1 which are enough explicit and which are used in order to represent actions of the processor.

F. Description of the atomic model AM SensorBoard

The goal of the atomic model AM Sensorboard is to represent interactions between environment and sensors. It is an important point of our approach. The atomic model AM Sensorboard is connected with a special atomic model AM Env where the atomic model AM Sensorboard can collect data of environment . The atomic model AM Env is an external model as shown on Figure 8 using environmental message to communicate with sensorboard. This interconnection between these two models represents sensing action of nodes in an environment or a specific phenomenon (wildfire). The atomic model AM Env contains different values for each environemental parameters. The atomic model AM Env allows to define an environemental scenarion for the global network, or for several groups of sensors or for each nodes of the WSN. To represent variation of environmental parameters, we have implemented a simple scenario with an increasing of the temperature on each sensor. The atomic model AM Sensor board has five states:

(i) the Busy state allowing to point out that a model is in action ;

(ii) the Free state allowing to point out that a model is in sleep mode ;

(iii) the Wait state when the model collect informations in the atomic model AM Env ;

(iv) the GO state when the model send a message to the coupled model Process if the message doesn't contain a value superior at a fixed threshold ;

(v) the Dead state when the model send a Dead message to the coupled model Process if the message contains a value superior at a fixed threshold.

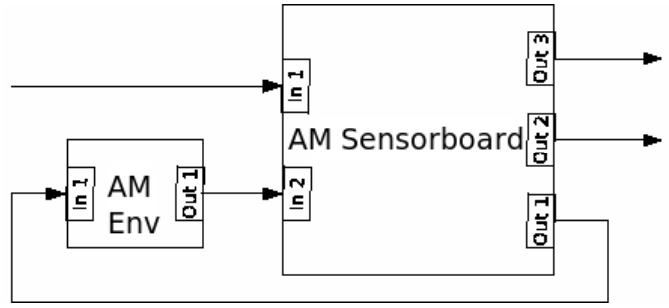


Figure 8. Atomic Model SensorBoard

G. Description of the coupled model CM Sensor : definition of coupling

The model illustrated by Figure 9 represents fcoupling of the DEVS Coupled model of sensor we have defined. This definition is essential because it determines the connectivity between the model but also architecture characteristics of the future Wireless Sensor Network. This model requires two input ports In1 and In2 and two output ports Out1 and Out2. In1 and Ou1 represent connectivity with a node. Let us note that the coupled model CM Sensor allows to have several connections with several nodes and consequently the outports and the inports increase. In2 and Out2 represent the connectivity with the environment. We have pointed out in Figure 9 appears the central role of the coupled model MC Process.

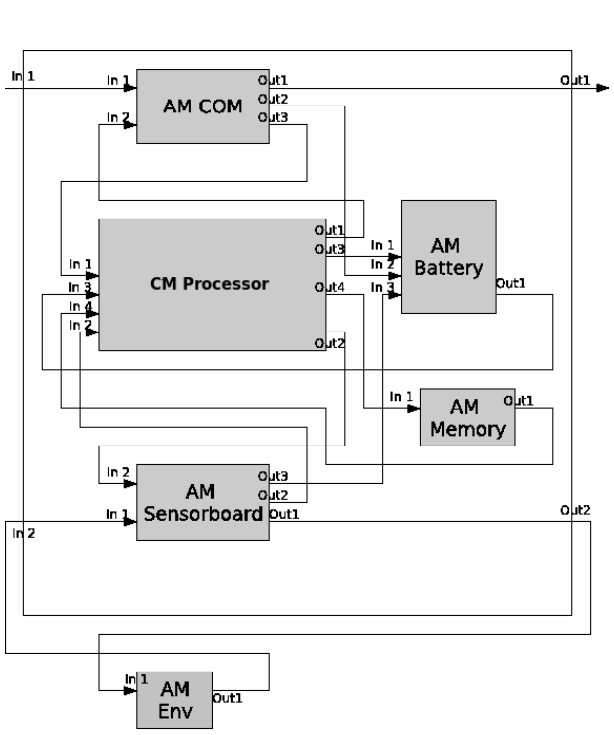


Figure 9. Coupled Model Sensor

When dealing with the coupled model CM Processor, and more precisely Atomic Model Net, we work with two specific protocol, called Gradient Based Routing protocol [19] and reliable route protocol or Xmesh protocol [20].

Gradient-Based Routing (GBR) is a variant of directed diffusion. The key idea in GBR is to memorize the number of hops when amessage is diffused through the whole network. As such, each node can calculate a parameter called the height of the node, which is the minimum number of hops to reach the BS. When multiple paths pass through a node, which acts as a relay node, that relay node may combine data according to a certain function. GBR uses mainly a stochastic scheme, where a node picks one gradient at random when there are two or more next hops that have the same gradient.

Xmesh protocol is more elaborated that GBR and it allows the node to estimate the quality of the link from the other nodes passively by collecting statistics on packets it happens to hear, or by actively probing. Link quality is measured as the percent as of packets that arrived undamaged on a link. Link status and routing information are maintained in a neighborhood table. The goal is to have a neighborhood management algorithm that will keep a sufficient number of good neighbors in the table regardless of cell density. To maintain this routing table, this protocol uses an algorithm based on a frequency count for each entry in the table. On insertion , a node is reinforced by incrementing its count. A new node will be inserted if there is an entry with a count of zero; otherwise the count of all entries is decremented by one and the new candidate is dropped. The neighbour table contains many fields : Group Ids, Parent node Ids, Chil Ids, reception link quality, link estimator data structures. To estimate link quality, Shortest Path protocol is used. For Shortest Path protocol, a node is a neighbor if its link quality exeeds threshold *t*. An another parameter is the selection of parent node. The cost metric is used to guide routing. The cost of a node is an abstract measure of distance ; it may be number of hops, expected number of transmissions, or estimate energy required to reach the sink. A neihgbor is selected as a potential parent only if its cost is less than the currrent cost of a node. This protocol is able to detect and avoid cycles, detect failures of transmision and eliminate node in the tree if link quality worsens.

V. Implementation and results

In order to validate the theoretical approach presented in [16] we choose to implement the described atomic and coupled models using the PythonDEVS simulator [21]. The PythonDEVS Modeling and Simulation package provides an implementation of the standard classic DEVS formalism described in section III. The package consists of two files, DEVS.py and simulator.py. The first one provides class architecture that allows hierarchical classic DEVS models to be easily defined by subclassing the AtomicDEVS and CoupledDEVS classes. The Simulator engine (SE) is implemented in the second file. Based on the principles of simulation describe in section III, it allows to perform discrete event simulation. Even if the PythonDEVS software involves a simulation engine which

offers limited means to terminate a simulation and provides no easy model-reinitialisation possibilities we have been able to use it in order to efficiently test our approach. We have been specially able to introduce the concepts of out-of-context and in-context models because of the open source quality of the PythonDEVS package. Furthermore by subclassing both the atomic model Class and the coupled model class inherent to the PythonDEVS package we have been able to perform the simulation algorithm of PythonDEVS by calling the methods extransition, intransition, etc... of an atomic model class and to use the model's composition and connectivity including ports and sub-models offered by the Coupled Model Class.

Furthermore we implement a Generator atomic model dedicated to generate the required events in order to validate our approach. This generator atomic model has been defined according to the semantics defined by B.P. Zeigler in [15].

We describe in this section how the proposed approach has been validated. We choose to implement a benchmark network composed by eight nodes and a Base Station.

On Figure 10, we represent the eight nodes and the BaseStation and we can see different relations between the nodes. Theses relations represent the connectivity and exprim capacity of communication between two nodes. If there is no connection between two nodes, it means that the nodes are too much distant to exchange some information by example. During the simulation, all nodes send periodically messages towards the BaseStation. Figure 10 shows only a predefinition of relations between nodes. We make the choice of this representation to work on routing protocol and representation capacity of our model. During simulation, we want the nodes make a choice according to routing protocol rules to reach the base station.

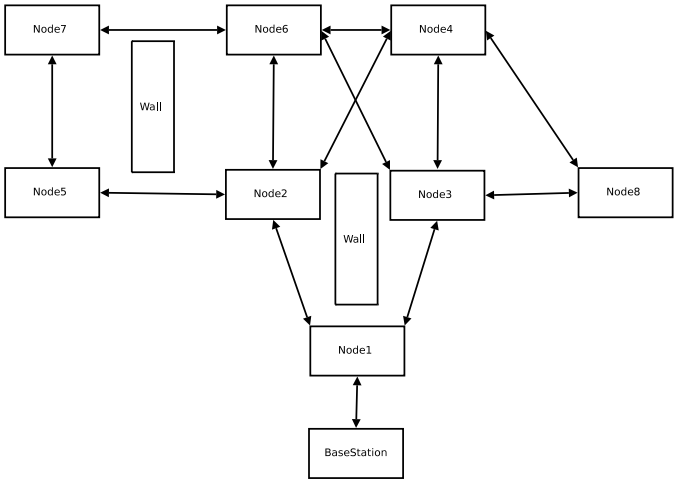


Figure 10. Network architecture for simulation

A. Identical parameters betwenn GBR and Xmesh.

The sensing activity and the time of message arrival on the Base station are the same for the two protocols. This fact is clearly understandable for the sensing activity because it is only dependant of the atomic model AM Env however for the time of latency we must provide some precisions. Based on the number of hops, GBR and Xmesh have the same time of arrival of the messages on the base station.

1) Sensing activity: According to our approach, we implemented the atomic model AM Env with a simple temperature model with a rapid increase of temperature as it can be observed in wildfire case for instance . On Figure 11, we analyse environmental data sent periodically by the nodes and we observe that the simple temperature model is clearly represented. We observe for each node a rapid increase of temperature .

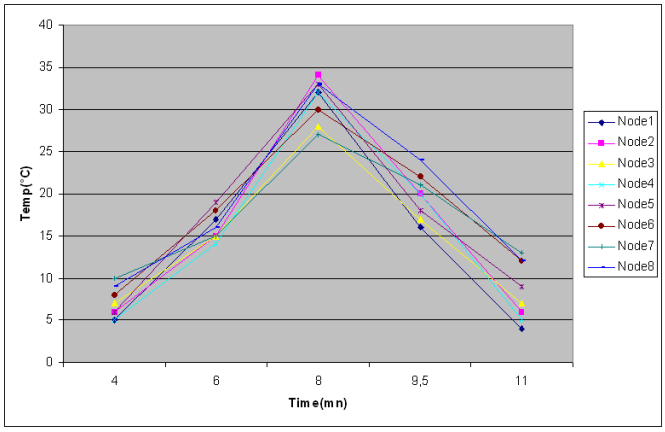


Figure 11. Environmental Temperature parameter during simulation

2) Latency Time: On Figure 12, we have highlighted the time of apparition of each node in the sink table that shows the difference between a node near the sink and a node more distant. However this time is not very important because we can see on Figure 12 that Node7 appears after 80 seconds. Figure 12 shows a real important shift due to the fact taht there is no direct relation between nodes and the BaseStation.

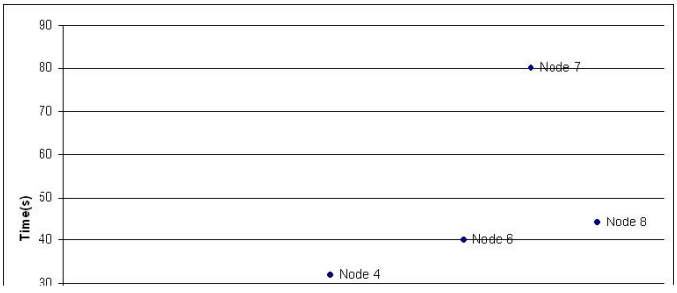


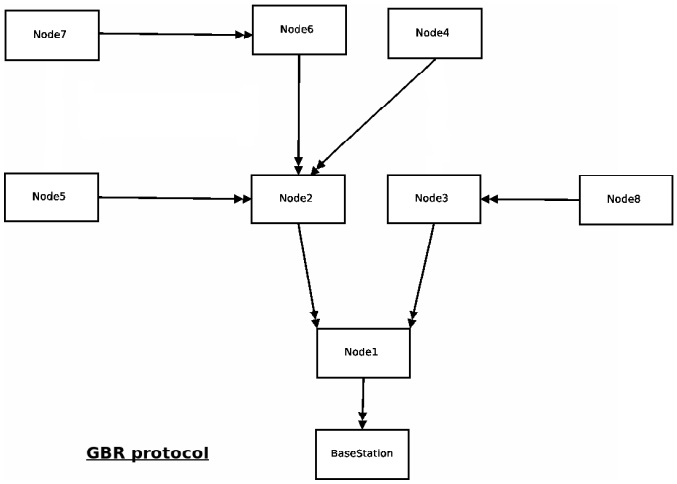
Figure 12. First apparition of node message on the sink

The main explanation once again is the routing protocol. As we already mentioned above a message issued from Node7 needs to go through Node5, Node2, Node1 in order to reach theBaseStation.

B. Comparison of Network Management

In this part, we compare GBR and Xmesh in using different parameters.

On Figure 13, we can see architecture of Wireless Sensor Network. This figure shows priviligied relations ,i.e. the first neighbor in routing table of each node according to the routing protocols GBR and Xmesh. On Figure 14, we can see the evolution of architecture of WSN for Xmesh protocol. We can observe that the relations between sensors are differents. Indeed, this evolution means that routing table of Node 6 has for first neighbor Node2 instead of Node3. This selection of the first neighbor is made by routing rules of Xmesh according to a good link quality. GBR uses a stochastic scheme based on a random selection between the nieghbors with the same hops number. This selection technique doesn't allow, in this case of simulation, a new relation definition between the node.



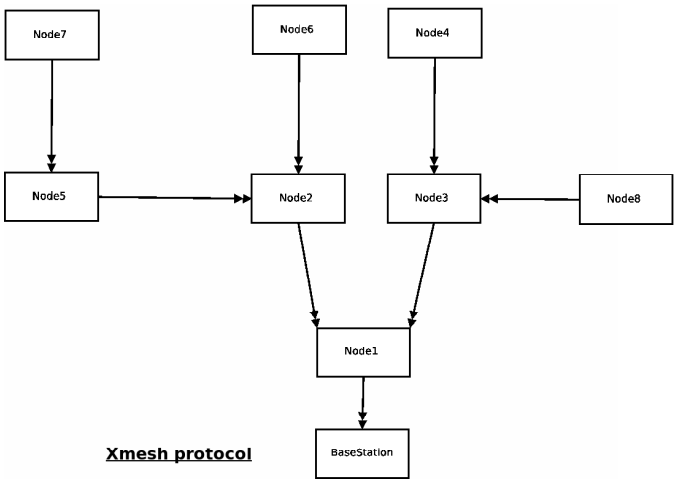
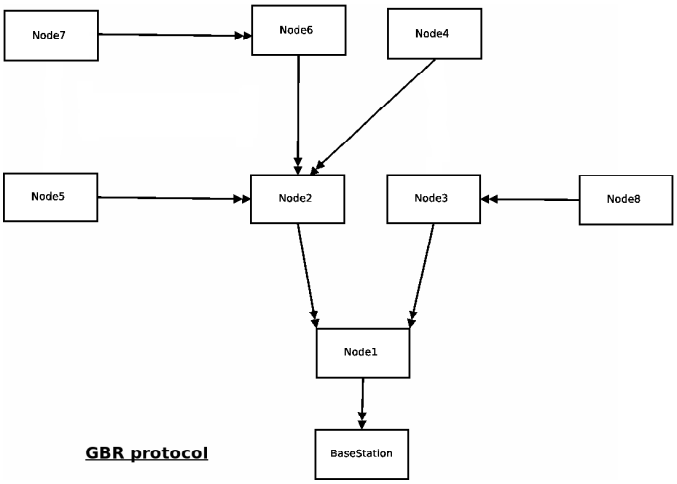


Figure 13. WSN priviligied communications after 10 mn of simulation



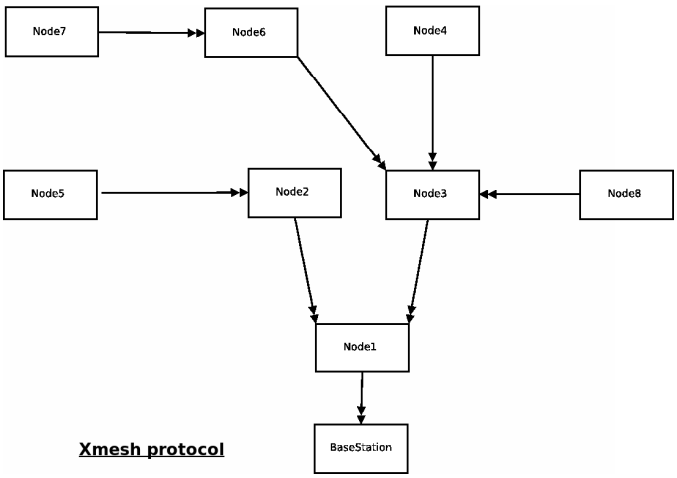


Figure 14. WSN priviligied communications after 15 mn of simulation

C. Comparison of Energy consumption and CPU activity

Our approach allows to distinguish energy cunsumption and processor activity. To illustrate this fact, we propose to compare the Xmesh protocol [16] at the GBR protocol according these two previous parameters, illustrated by the Figure 15 and Figure 16 on the next page.

On Figure 15, we represent the events treated by each node processor model during the time of simulation. In our approach, the atomic model AM Processor manages all the components and treated all actions involved in a node.

First, independently of the routing protocol, we can observe an important activity of Node 1 because it has a central role in the network and it is the bridge between the other nodes and the sink. This activity of Node 1 is the direct effect of the nodes deployment and network architecture chosen for the test represented on Figure 10.

The Figure 15 a) illustrates the processor activity of "GBR nodes" and the Figure 15 b) represents the processor activity of "Xmesh nodes".

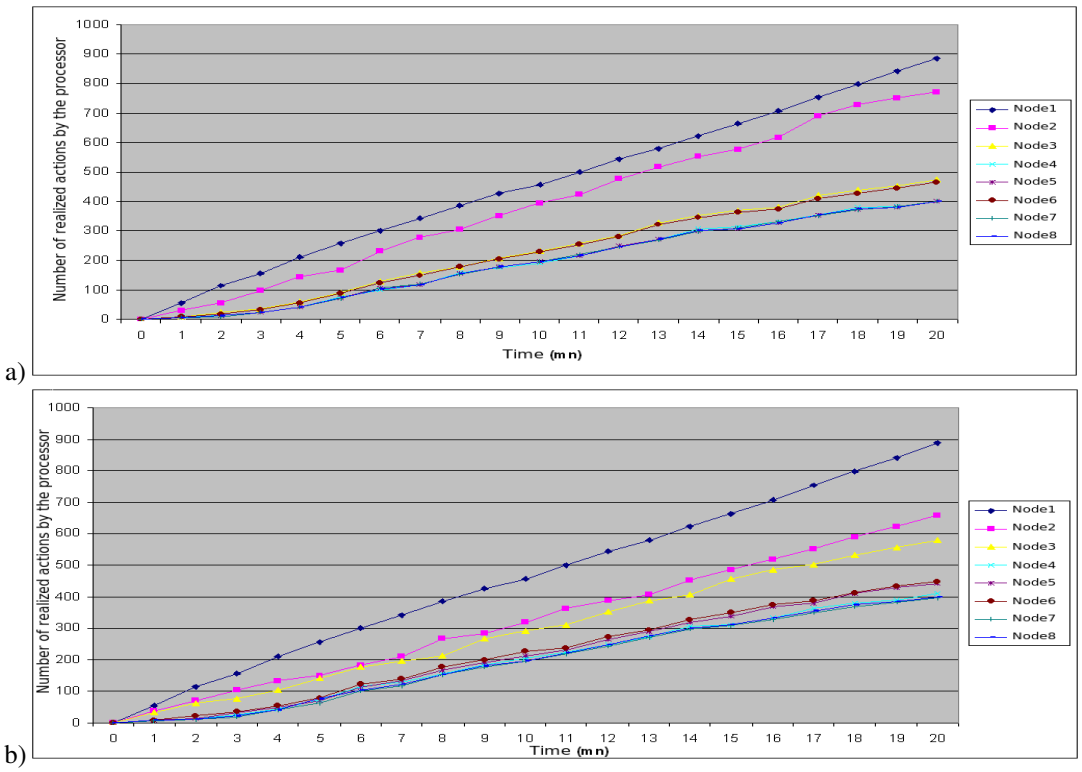


Figure 15. Relative CPU activity in the WSN

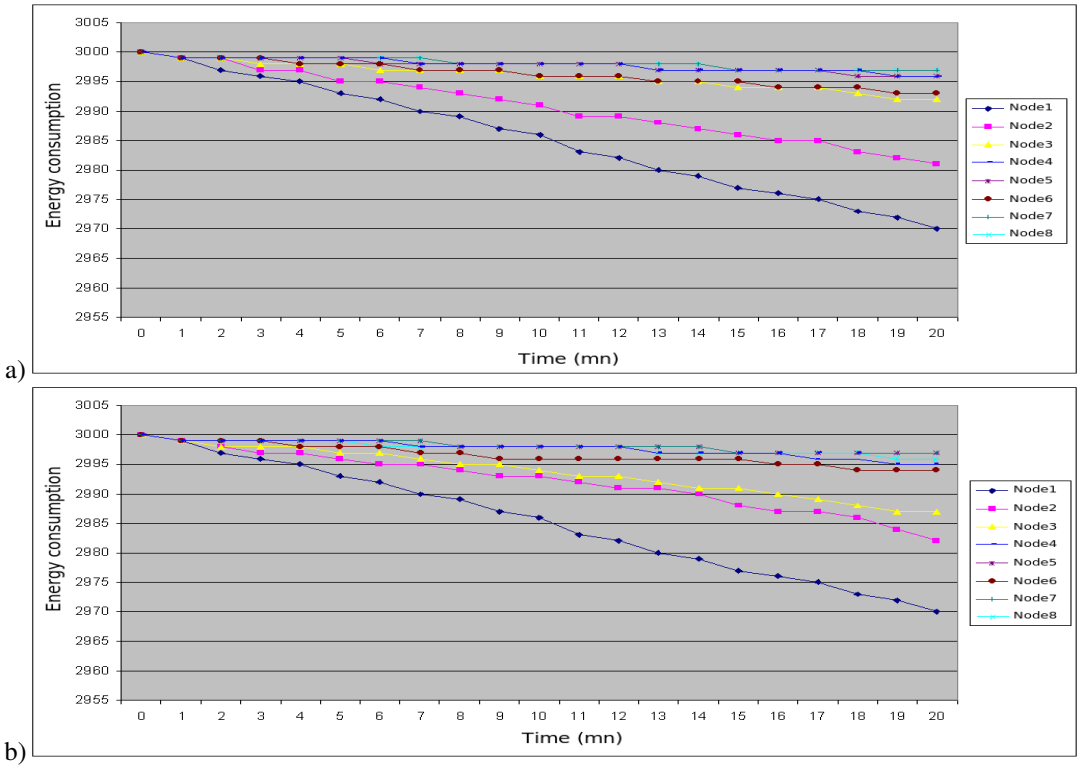


Figure 16. Relative energy cunsumption in the WSN

We can observe, in this case of simulation, that Xmesh protocol allows a more important sharing of the routing stacks than GBR protocol. Indeed, an important processor activity is the sign of the important role of the node in the routing. We can observe with GBR that Node 2 is more often sought than the Node 3. We can point out that Xmesh reduces this problem encouraging a better routing tasks sharing.

The Figure 16 a) illustrates the energy consumption of "GBR nodes" and the Figure 16 b) represents the energy consumption of "Xmesh nodes". We can observe that the Node 2 consumption is more important in GBR case than in Xmesh case. These results confirm that the previous conclusions and Xmesh seems to provide a more important tasks balance in the network than GBR. Xmesh is a protocol more complex based on the number of hop to reach the base station but also a neighborhood selection more elaborated than GBR, thus fostering a better routing tasks sharing.

VI. Conclusion and Future Work

This article provides a modeling and simulation of performance of a Wireless Sensor Network. This work is based on the DEVS formalism for the modelling and the simulation of complex discrete event system. We have demonstrated the capacity of our approach to analyze the evolution of Wireless Sensor Network architecture and analyse certain performance of a WSN according to two routing protocols, GBR and Xmesh. We have been able to implement the concepts presented in the paper using the PythonDEVS package and to validate this approach by providing results of simulation of a Wireless Sensor Network with eight nodes. We have highlighted the routing parameters, the relative energy cunsumption and the CPU activity . These results allows to show that this first approach for modelling WSN using the DEVS formalism is promising and provides a new level of visualization of node. Indeed, a DEVS description of components allows us to visualize the characteristics of each component. These results confirm both that the proposed approach is fine when dealing with WSN characteristics and that the DEVS formalism is efficent in order to model the behaviour of a WSN. The Modular aspect of DEVS allows to easily change the component model of a sensor into another one, by example in our example the atomic model AM Net for the routing protocol test. After the completion of the main components of the Sensor network an application to test the model can be created. This application is based on a DEVS simulator written in Pyhton developped by [21]. It is divided in four packages : DEVS package, ComponentsNodes package, SimulationTools package, and Wireless Sensor Network specification package. A simulation tool called DEVS-WSN is currently in development phase. This simulation tool will allow to us to work on a particular phenomenon : the wildfire. Indeed, the atomic model AM Env and the capacity to study the routing protocol and the network topology in easy way will allow to analyze the WSN behavior in forest fire conditions.

References

[1] A. Hac, *Wireless Sensor Designs*. John Wiley and Sons Ltd, 2003.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[3] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: scalable coordination in sensor networks," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking(MobiCom '99)*, (Seattle, Washington, United States), pp. 263–270, ACM Press, 1999.

[4] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "A taxonomy of wireless micro-sensor network models," *SIGMOBILE Mobile Computer Communication Revue*, vol. 6, no. 2, pp. 28–36, 2002.

[5] I. Khemapech, I. Duncan, and A. Miller, "A survey of wireless sensor networks technology," in *Proceedings of the 6th Annual PostGraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting(PGNET'05)*, (Liverpool, UK), 2005.

[6] U. Farooq, B. Balya, and G. Wainer, "Modelling routing in wireless ad-hoc networks using cell-devs," in *Proceedings of 2004 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'04)*, (San Jose, CA, USA), pp. 285–292, 2004.

[7] T. Kim, "Devs-ns2 environment : An integrated tool for efficent network modeling and simulation," Master's thesis, 2006.

[8] NS2, "Available: http://www.isi.edu/nsnam/ns/," 1995.

[9] S. Park, A. Savvides, and M. B. Srivastava, "Sensorsim: a simulation framework for sensor networks," in *Proceedings of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems (MSWIM'00)*, (Boston, Massachusetts, United States), pp. 104–111, ACM Press, 2000.

[10] J. Polley, D. Blazakis, J. Mcgee, D. Rusk, and J. S. Baras, "Atemu: a fine-grained sensor network simulator," in *Proceedings of IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON'04)*, (Santa Clara, CA, USA), pp. 145–152, 2004.

[11] P. Levis, N. Lee, M. Welsh, and D. Culler, "Tossim: accurate and scalable simulation of entire tinyos applications," in *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys'03)*, (Los Angeles, CA, USA), pp. 126–137, ACM Press, 2003.

[12] V. Shnayder, M. Hempstead, B. R. Chen, G. W. Allen, and M. Welsh, "Simulating the power consumption of large-scale sensor network applications," in *Proceedings of the 2nd international conference on Embedded networked sensor systems (SenSys'04)*, (Baltimore, MD, USA), pp. 188–200, ACM Press, 2004.

[13] L. Samper, F. Maraninchi, L. Mounier, and L. Mandel, "Glonemo: global and accurate formal models for the analysis of ad-hoc sensor networks," in *Proceedings of the first international conference on Integrated internet ad hoc and sensor networks(InterSense'06)*, (Nice, France), p. 3, ACM Press, 2006.

[14] S. Sundresh, W. Kim, and G. Agha, "Sens: A sensor, environment and network simulator," in *Proceedings of the 37th annual symposium on Simulation (ANSS'04)*, (Washington, DC, USA), p. 221, IEEE Computer Society, 2004.

[15] B. P. Zeigler, *Theory of Modeling and Simulation*. Academic Press, 1976.

[16] T. Antoine-Santoni, J. F. Santucci, E. D. Gentili, and B. Costa, "Simulation and visualization method of wireless sensor network performances," in *Proceedings of International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'07)*, pp. 476–483, SCS, IEEE, 2007.

[17] S. Park, A. Savvides, and M. B. Srivastava, "Battery capacity measurement and analysis using lithium coin cell battery," in *Proceedings of the 2001 international symposium on Low power electronics and design(ISLPED'01)*, (New York, NY, USA), pp. 382–387, ACM Press, 2001.

[18] Crossbow-Technology, *Wireless Sensor Network Seminar*. 2006.

[19] C. Schurgers and M. Srivastava, "Energy efficient routing in wireless sensor networks," in *MILCOM Proceedings on Communication for Network-Centric Operations : Creating the Information Force*, (McLean, VA, USA), 2001.

[20] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems (SenSys03)*, (Los Angeles, California, USA), pp. 14–27, ACM Press, 2003.

[21] J. S. Bolduc and H. Vangheluwe, "pythonDEVS : A modeling and simulation package for classical hierarchal DEVS," tech. rep., 2001.

BIOGRAPHY :

Thierry Antoine-Santoni maintained his doctoral thesis in 2007 at the University of Corsica. His work focuses on modelling and simulation of complex systems based on the DEVS formalism developed by BP Zeigler, Wireless Sensor Network and Bioinformatic. He has been author and co-author of many papers published in international journals or conference proceedings.

Jean-François Santucci is Professor in Computer Sciences at the University of Corsica since 1996. His main research interests are modeling and simulation of complex systems. He has been author or co-author of more than 100 papers published in international journals or conference proceedings. He has been the scientific manager of several research projects corresponding to European or industrial contracts. Furthermore he has been the advisor or co-advisor of more than 20 PhD students and since 1998 he has been involved in the organization of more than 10 international conferences. He is conducting newly interdisciplinary researches involving computer sciences, archaeology and anthropology: in the one hand he is performing researches in the archeaoastronomy field (investigating various aspects of cultural astronomy throughout Corsica and Algeria using tools issued from Computer Sciences) and on the other hand he is applying computer sciences approaches such as GIS (Geographic Information Systems) or DEVS (Discrete EVent System specification) to anthropology.

Emmanuelle de Gentili maintained her doctoral thesis in 2002, Assistant Professor at the University of Corsica since 2004. His work focuses on modelling and simulation of complex systems based on the DEVS formalism, fuzzy logic and dynamic systems.

Bernadette COSTA is Professor in Electronics at the University of Corsica since 1989. His main research interests are electronics, signal processing and acoustics. She has been author and co-author of many papers published in international journals or conference proceedings.