An Approach Combining SysML and Modelica for Modelling and Validate Wireless Sensor Networks

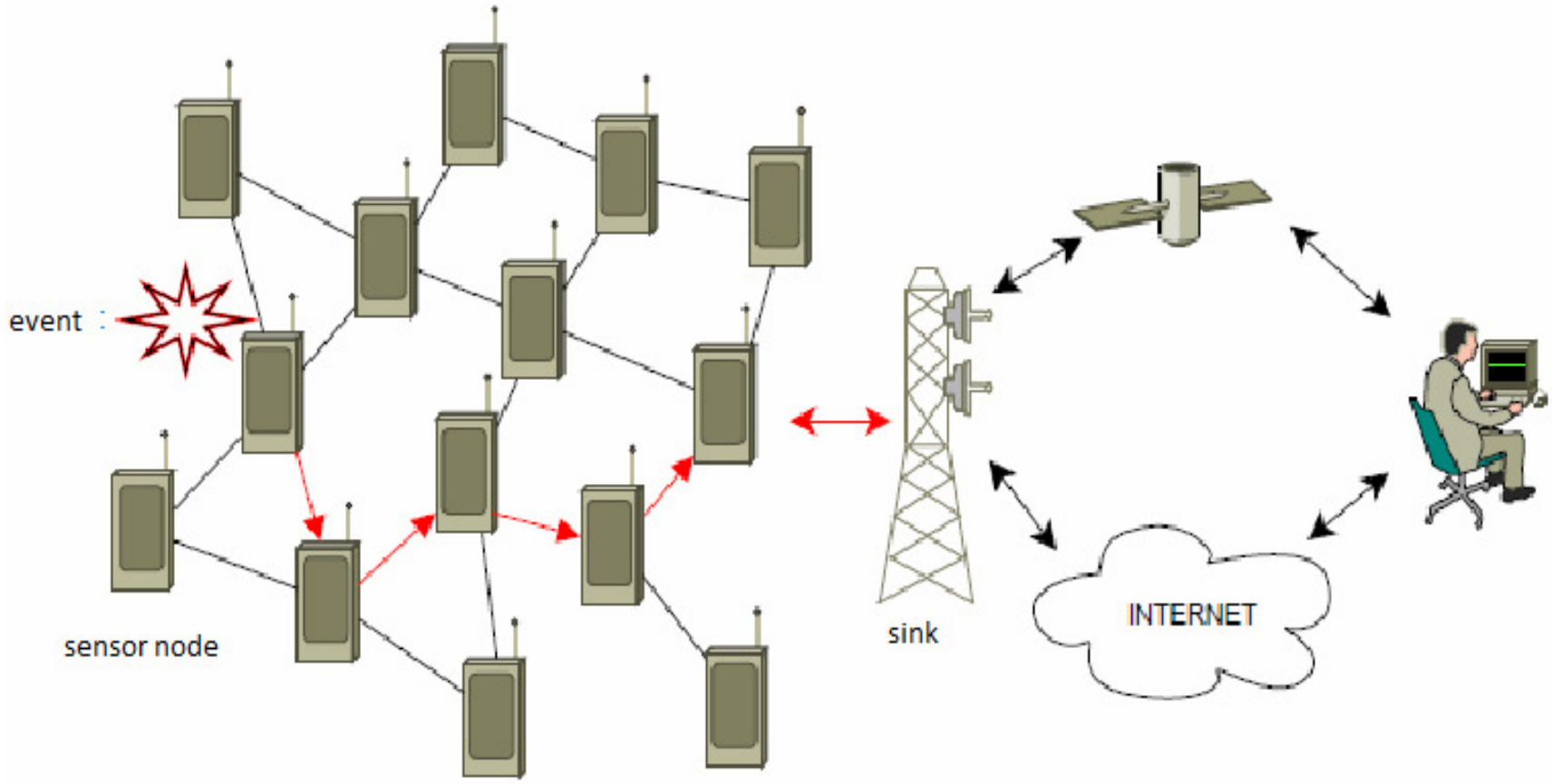
Hammad, H. Mountassir and S. Chouali

Femto-ST Institute, University of Franche-Comté
hmountas@femto-st.fr

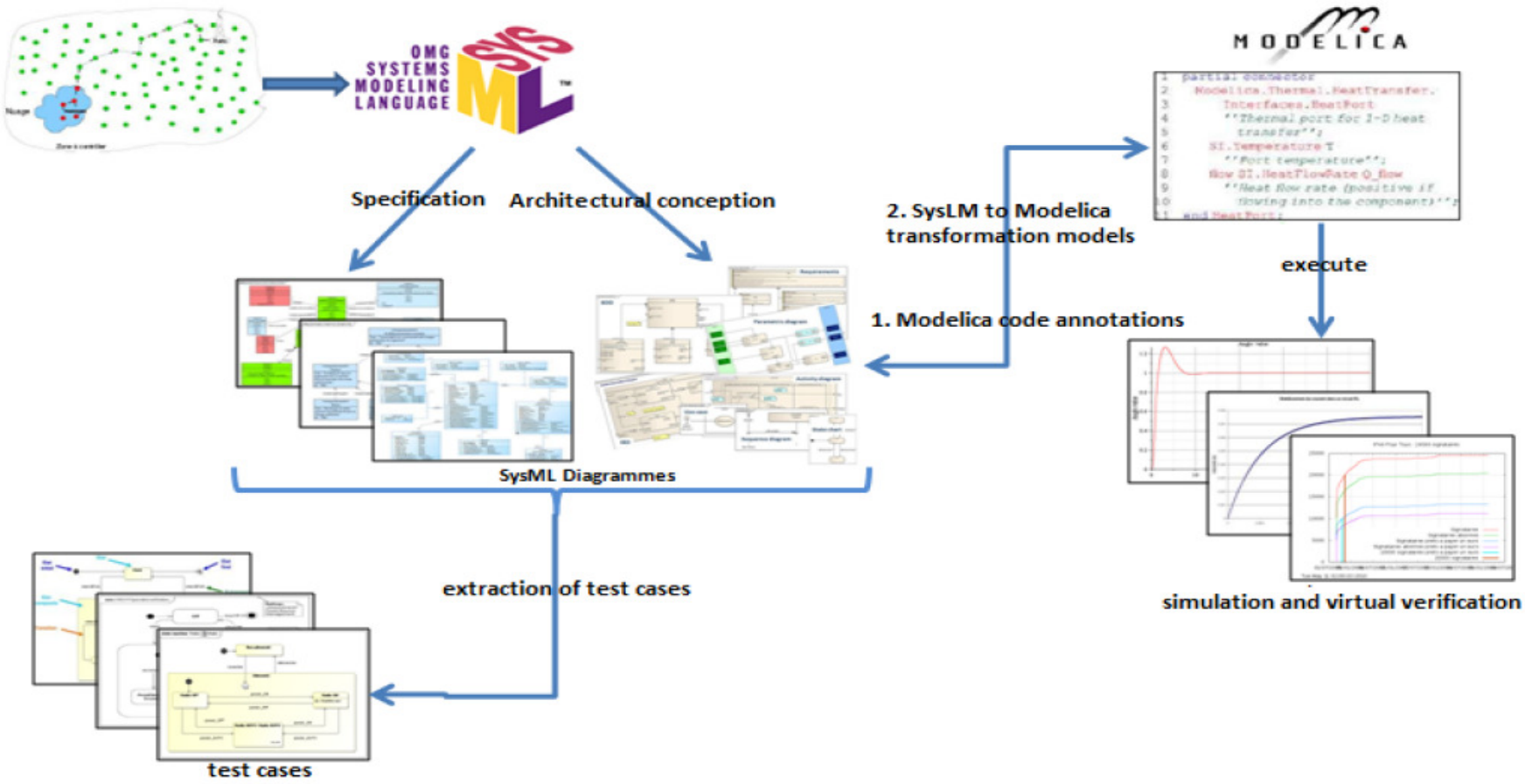
Outline

- Introduction
- Proposed Approach
- From SysML to Modelica
- Illustration
- Conclusion and Future work

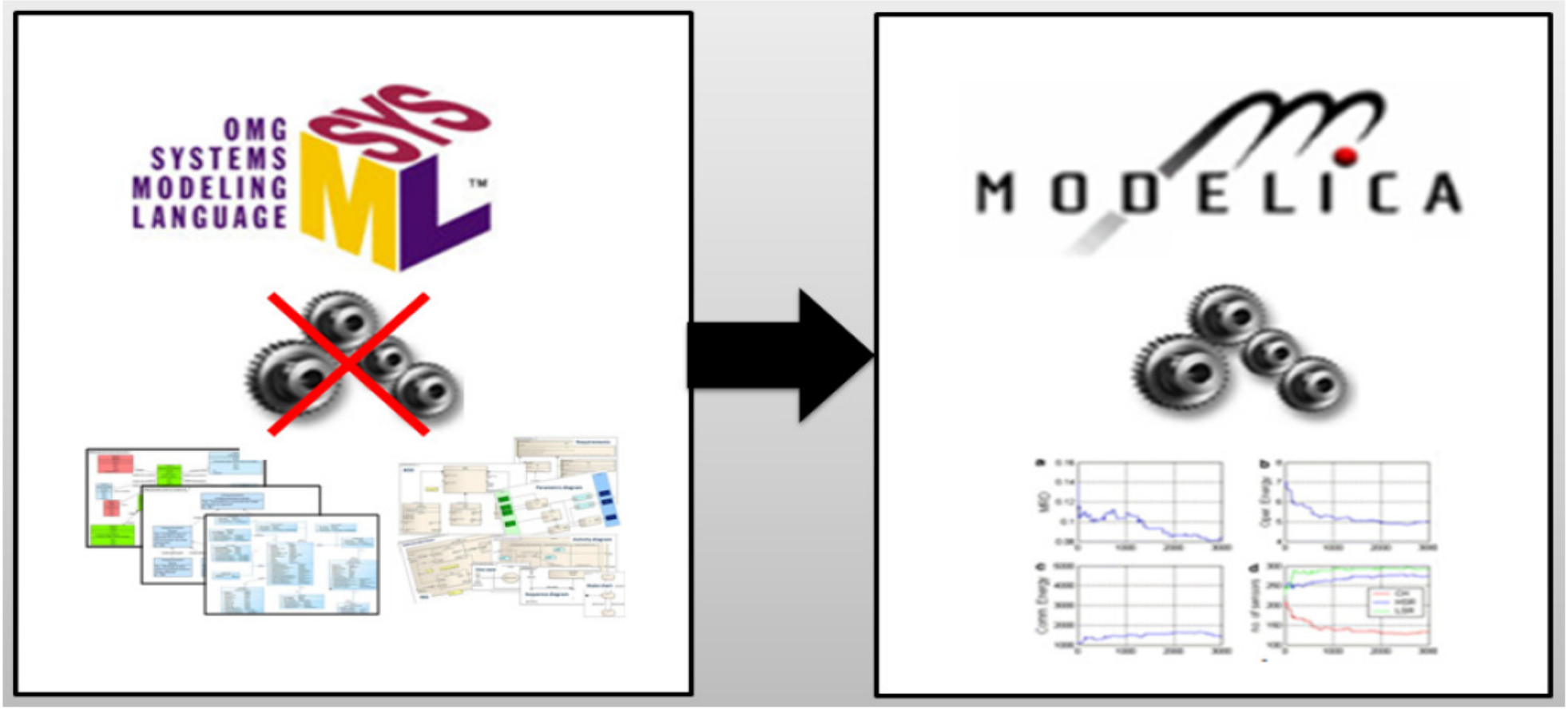
WSN Introduction



Proposed Approach



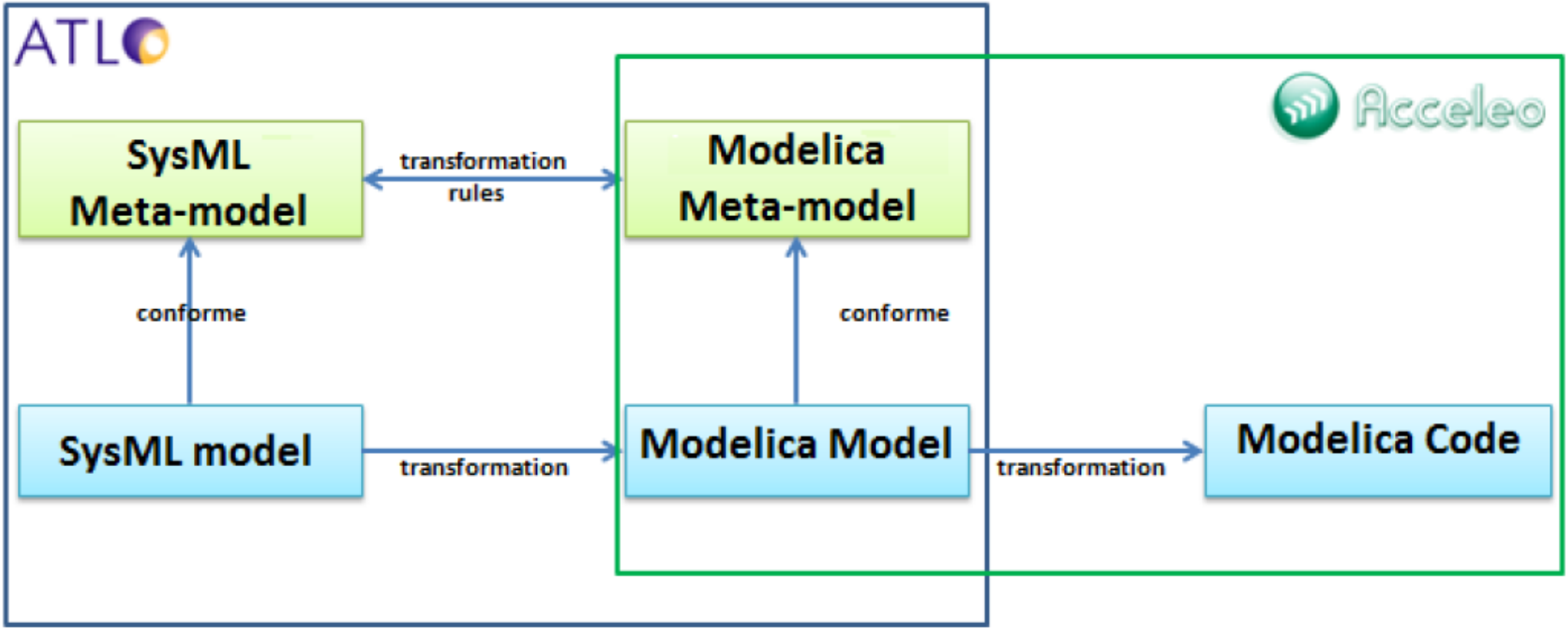
SysML/Modelica



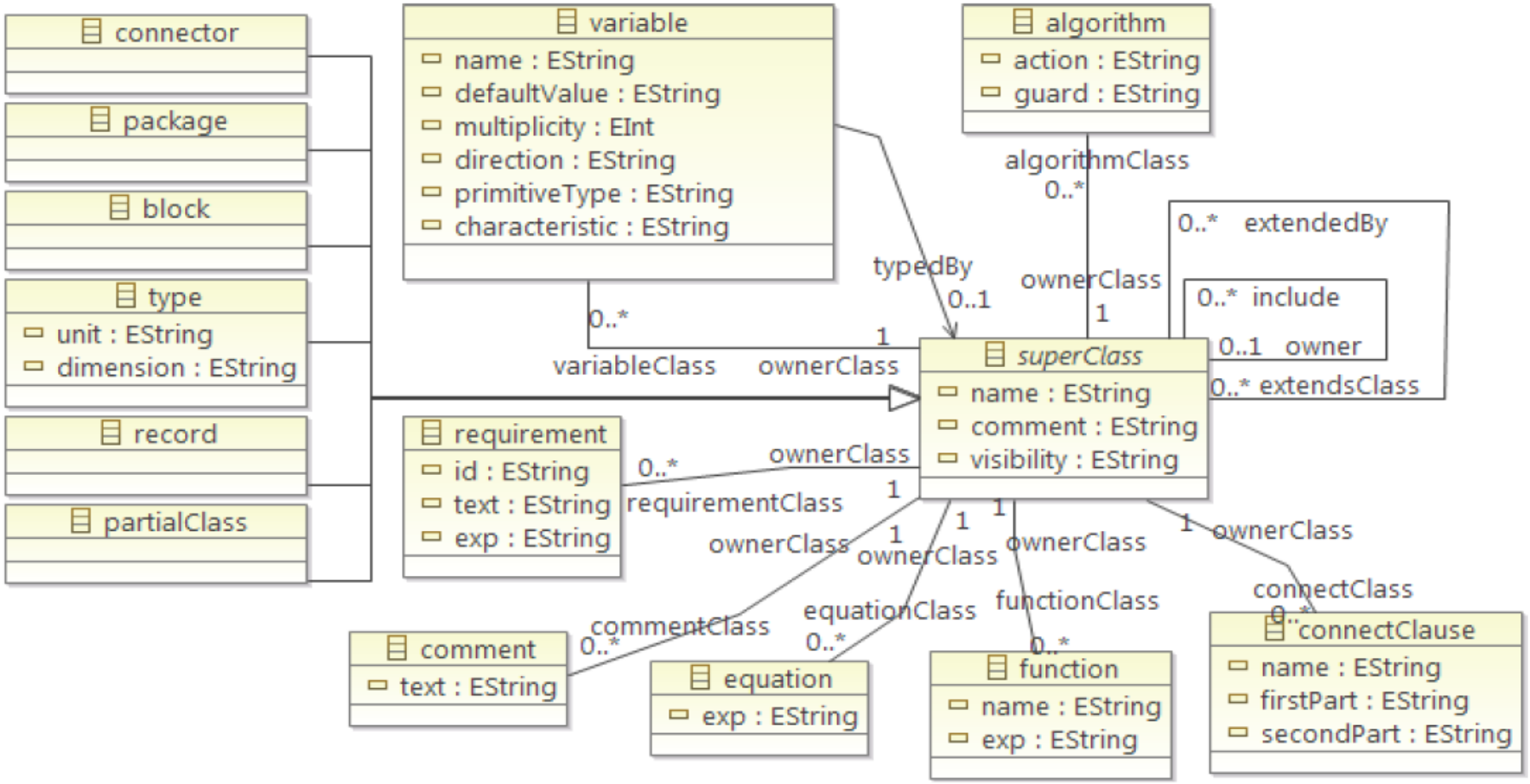
Modelica

- Open source language
- Allows an object-oriented modeling approach, a graphical representation and simulation;
- Considers at any time the equation set which describe the model as single system equations;
- Modelica compiler has single simulator that allows automatic synchronization between discrete and continuous parts.

Model Transformation Tools



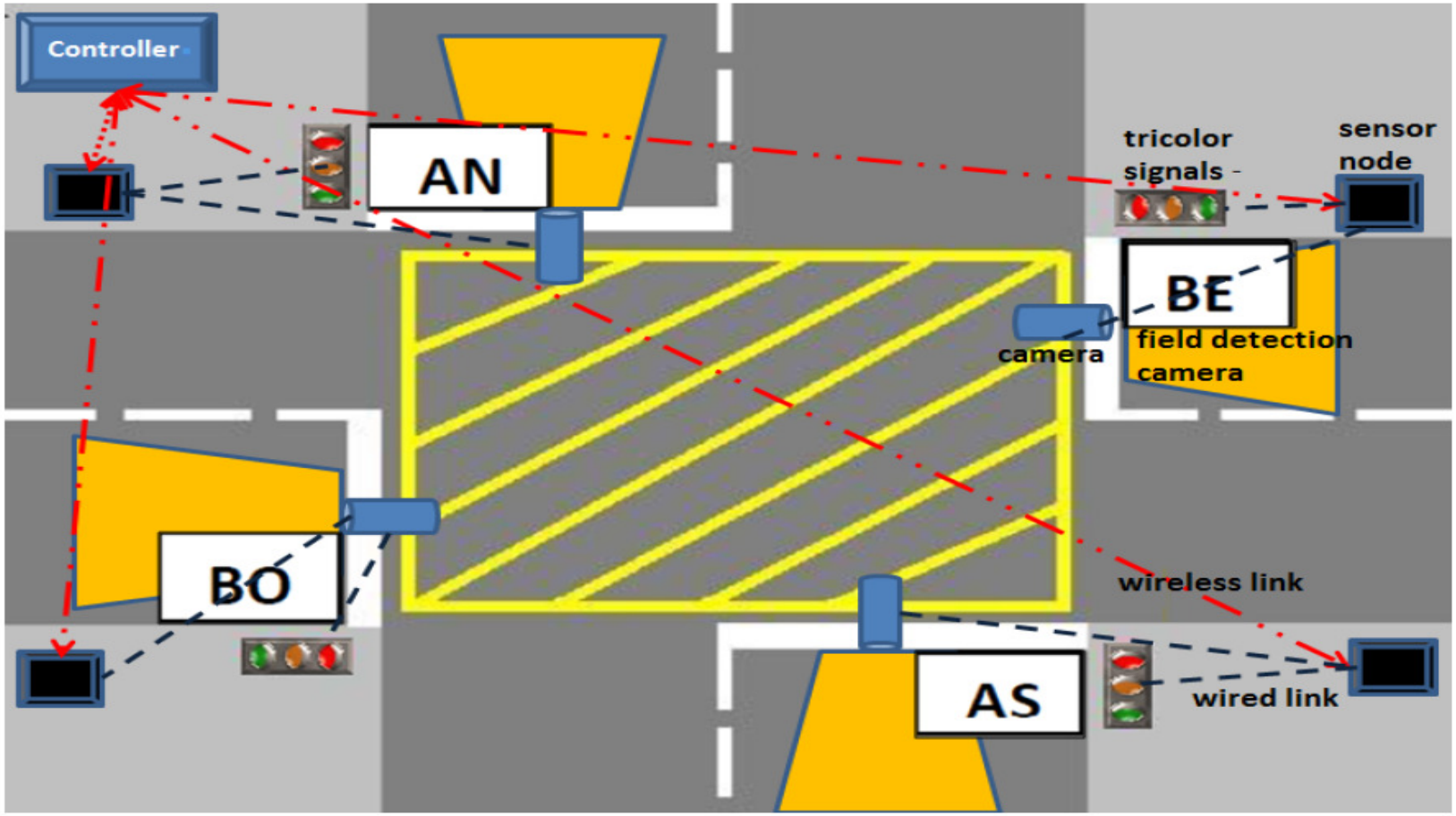
Modelica Meta-model



SysML to Modelica Transformation rules

SysML Model	Modelica Model
package, block, abstract-block	package, block, partial-class
flow-specification, value-type	connector, type
flow-property, flow-port	property, connector
connector flux(x,y)	equation connect(x,y)
constraint property	equation
state machine guard	'when' statement guard
operation without a value to return	instruction block of 'when' statement
operation with a value to return	function
requirement	boolean expression (invariant or safety constraint)

Case study: Crossroads system



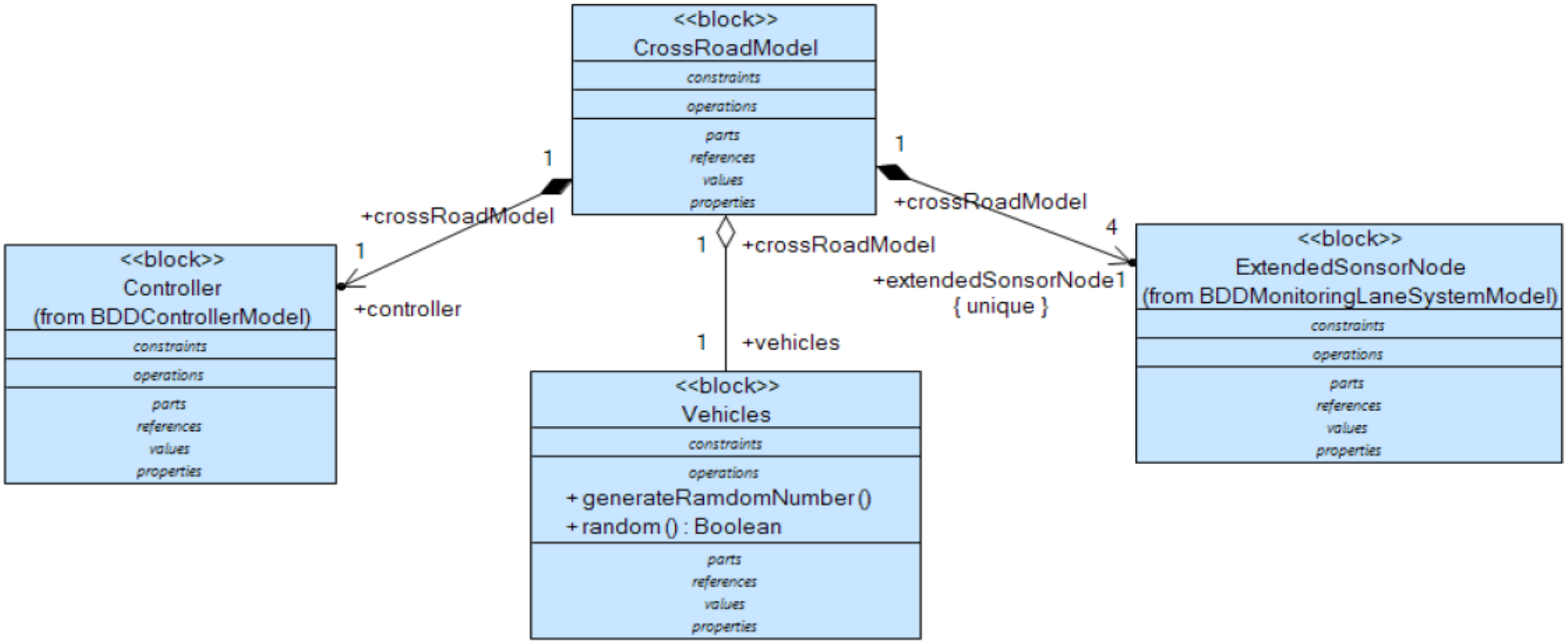
Adopted Energy Model

$$E_{SensorNode} = (E_{Sensing} + E_{Processing} + E_{Transmitting} + E_{Receiving} + E_{Transient} + E_{Logging})^{1}$$

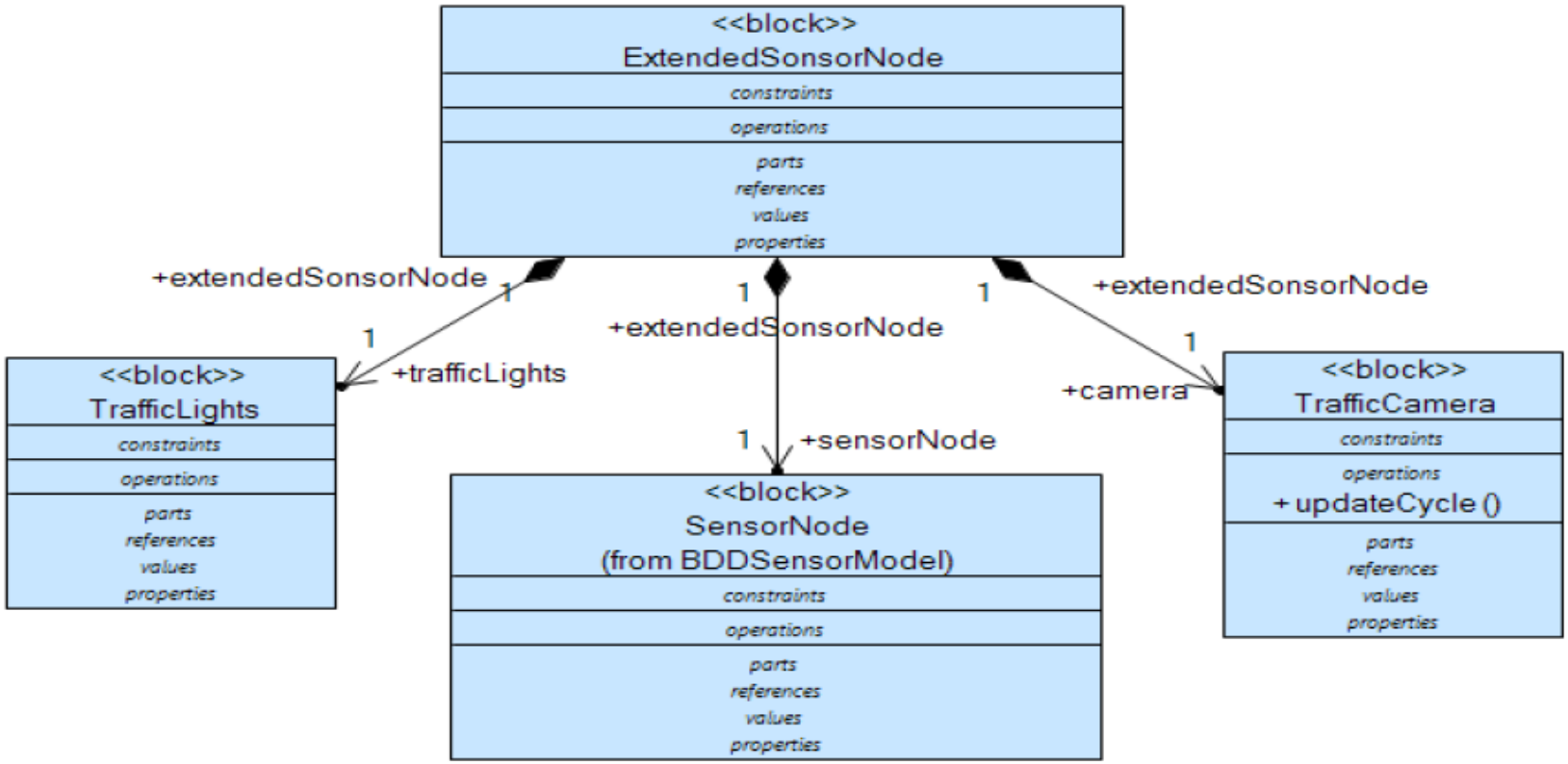
- $E_{Sensing} = bV_{sup}I_{sens}T_{sens}$
- $E_{Processing} = E_{switch} + E_{leak}$;
- $E_{Transmitting} = bE_{elec} + bd_{ij}^{n}E_{amp}$
- $E_{Receiving} = bE_{elec}$
- $E_{Transient} = T_A V_{sup}[c_N I_A + (1 - c_N)I_S]$
- $E_{Logging} = E_{write} + E_{read}$

[1] : M. N. Halgamuge, M. Zukerman, K. Ramamohanarao, H. L. Vu. An estimation of sensor energy consumption. Progress In Electromagnetics Research B (PIERB).

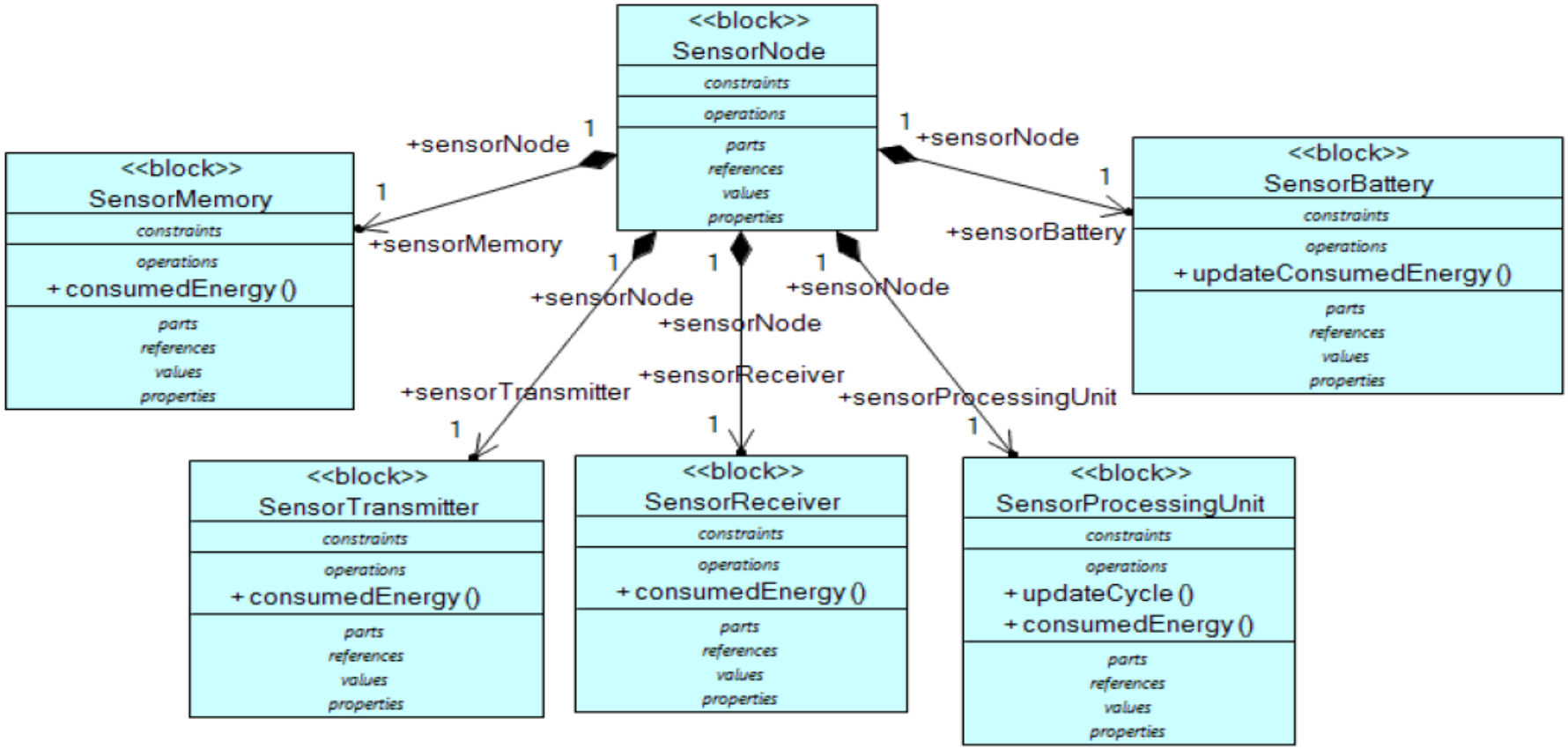
BDD of the Global System



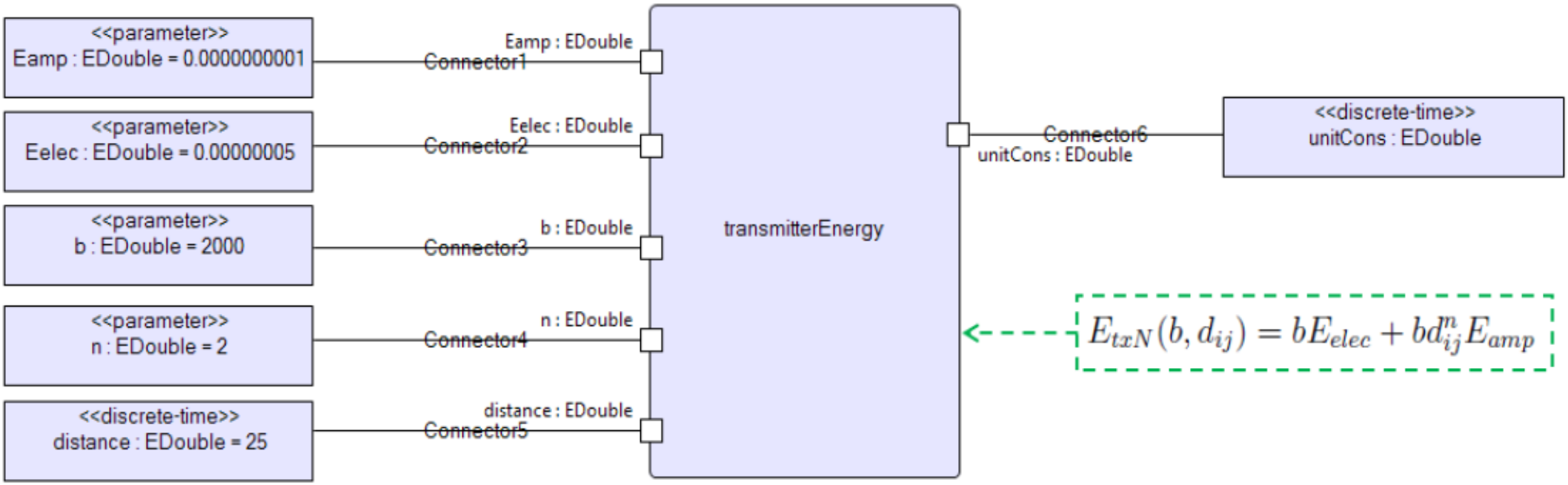
BDD of the extended Sensor



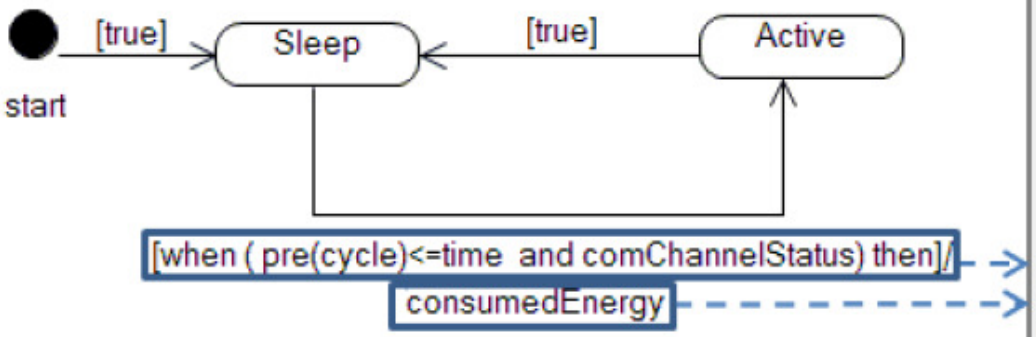
BDD of the simple Sensor



Transmitter Parametric Diagram

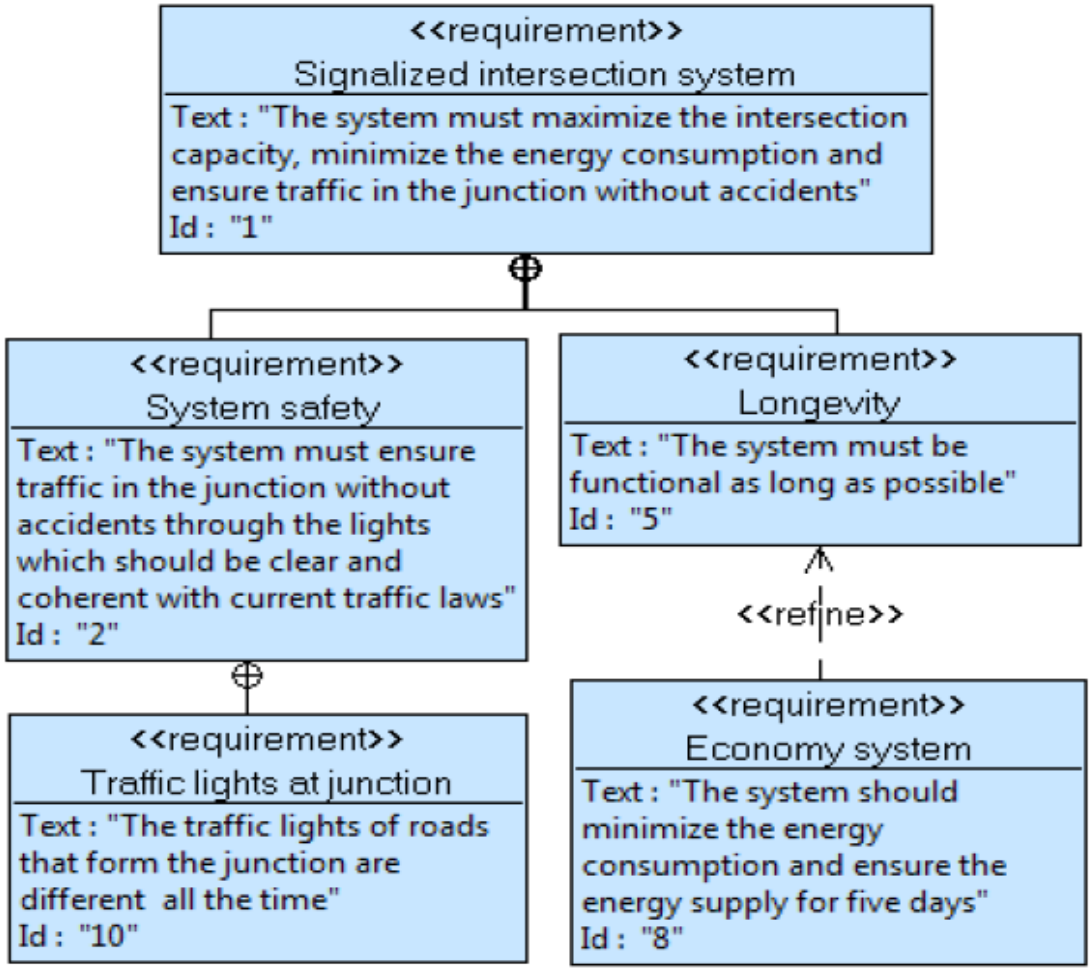


Transmitter State Machine Diagram

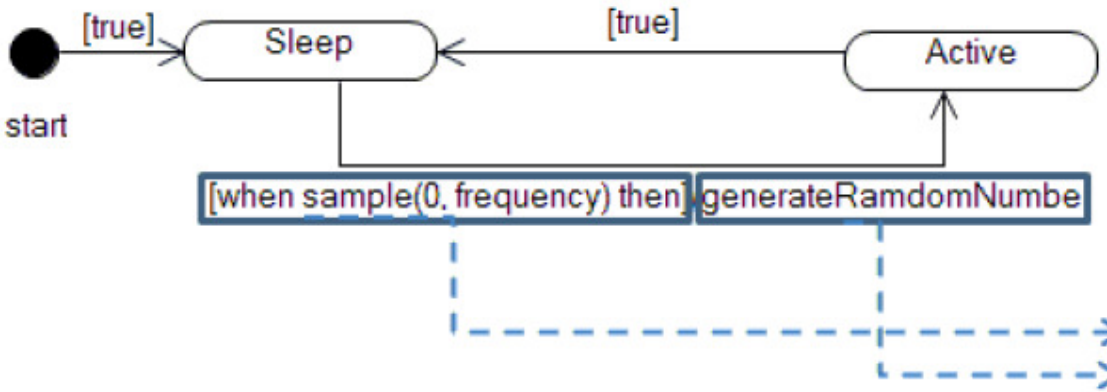



```
1  block SensorTransmitter
2    ...
3    Real distance(start = 25);
4    Real unitCons(start = 0);
5    Real cycle(start = 0.01);
6    ...
7  equation
8     ...
9  algorithm
10   when pre(cycle) <= time and comChannelStatus then
11      consumedEnergy:=pre(consumedEnergy) + unitCons;
12      ...
13   end when;
14 end SensorTransmitter;
```

Requirement Diagrams

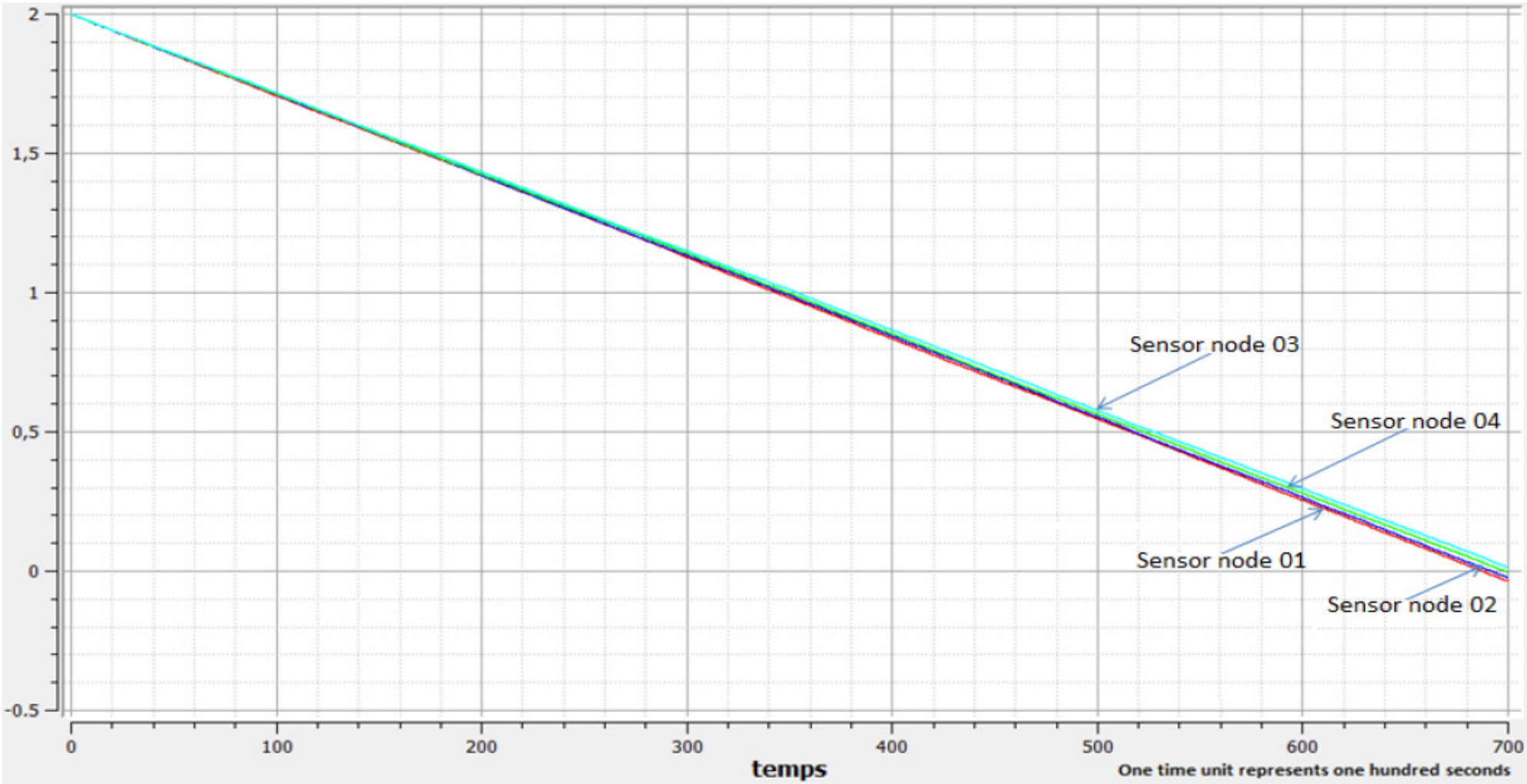


Random Scenario

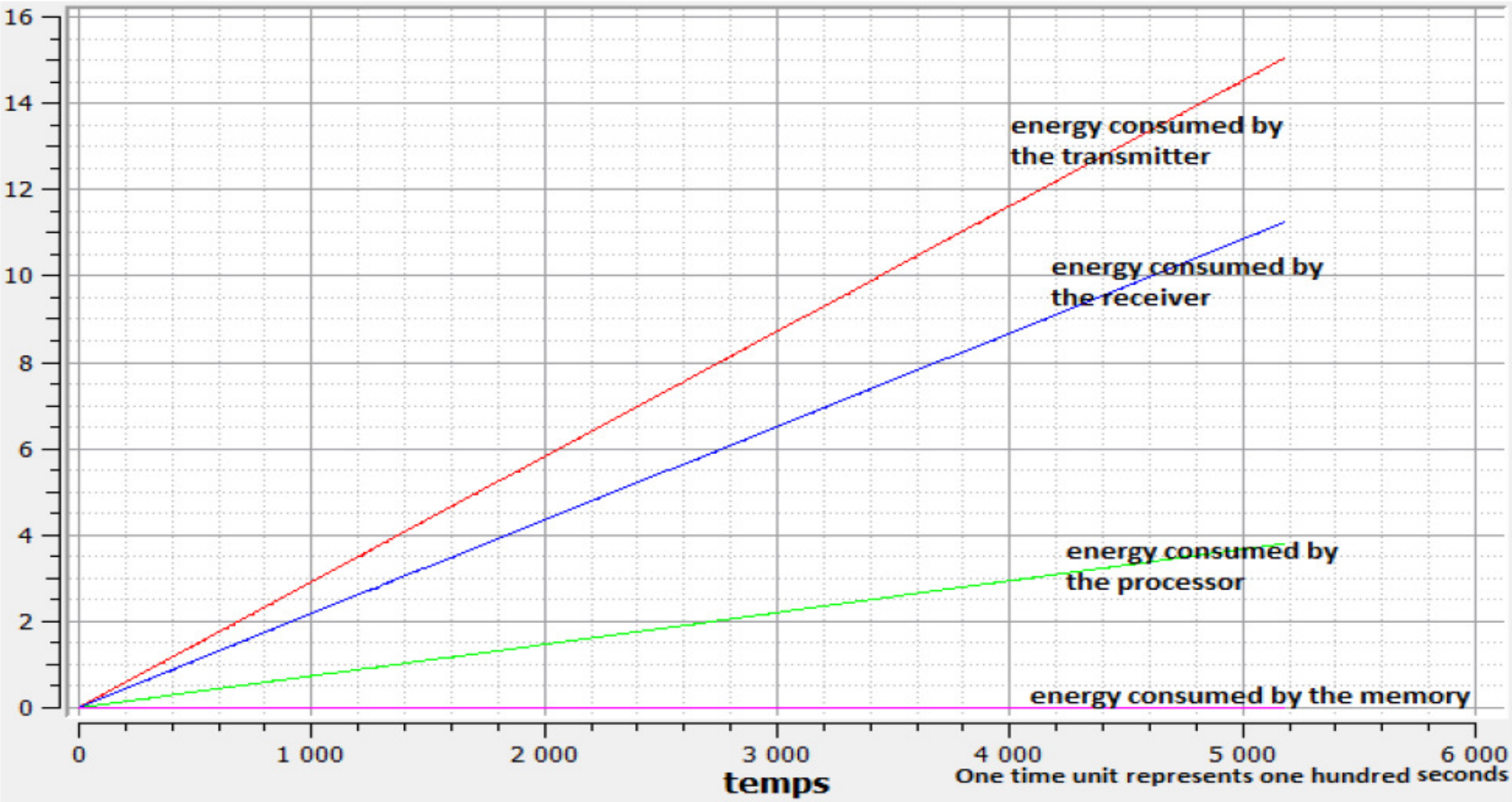



```
1  block Vehicles
2    ...
3    parameter Real frequency = 0.05;
4    output Data nbrLane12;
5    output Data nbrLane21;
6    output Data nbrLane22;
7    output Data nbrLane11;
8  algorithm
9    when sample(0, frequency) then
10     (nbrLane11.value,seed1):=random(pre(seed1));
11     (nbrLane12.value,seed2):=random(pre(seed2));
12     (nbrLane21.value,seed3):=random(pre(seed3));
13     (nbrLane22.value,seed4):=random(pre(seed4));
14   end when;
15 end Vehicles;
```

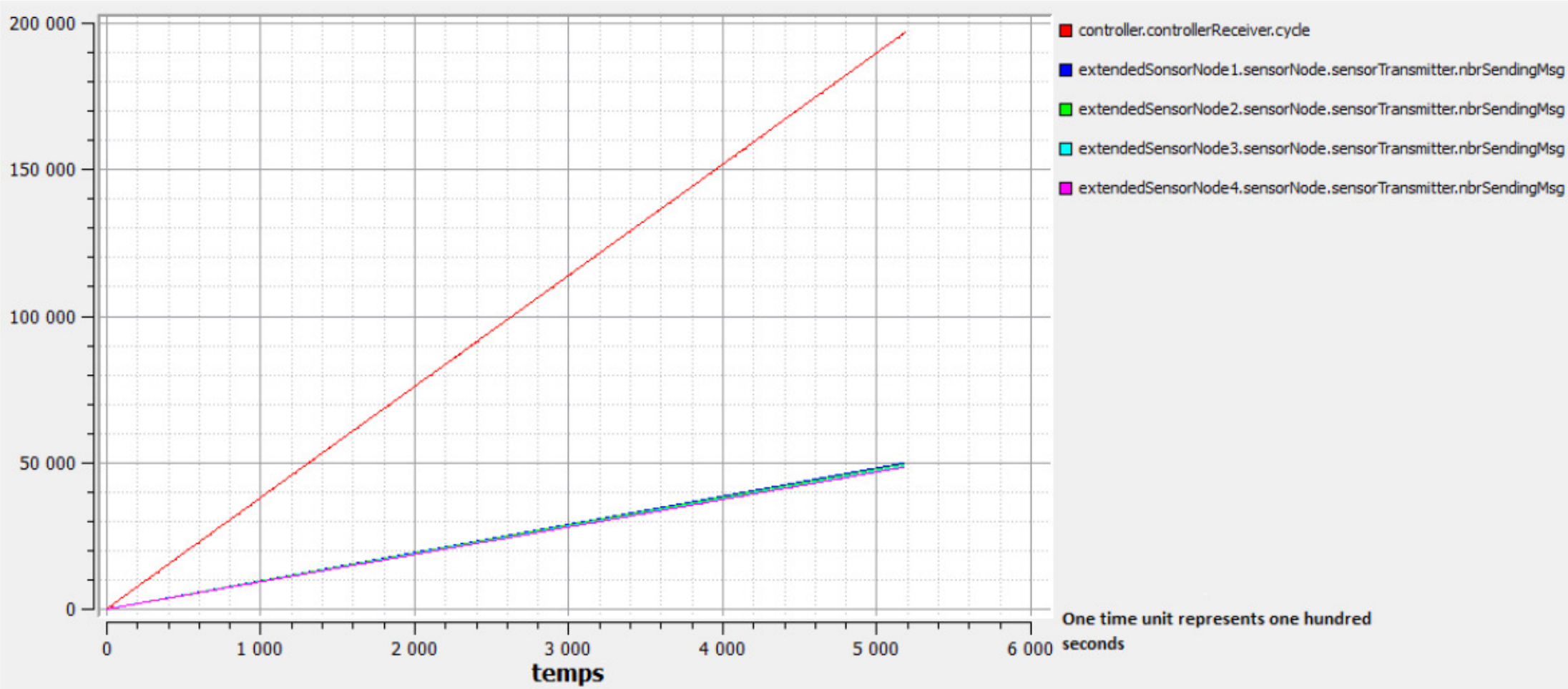
System Energy Consumption



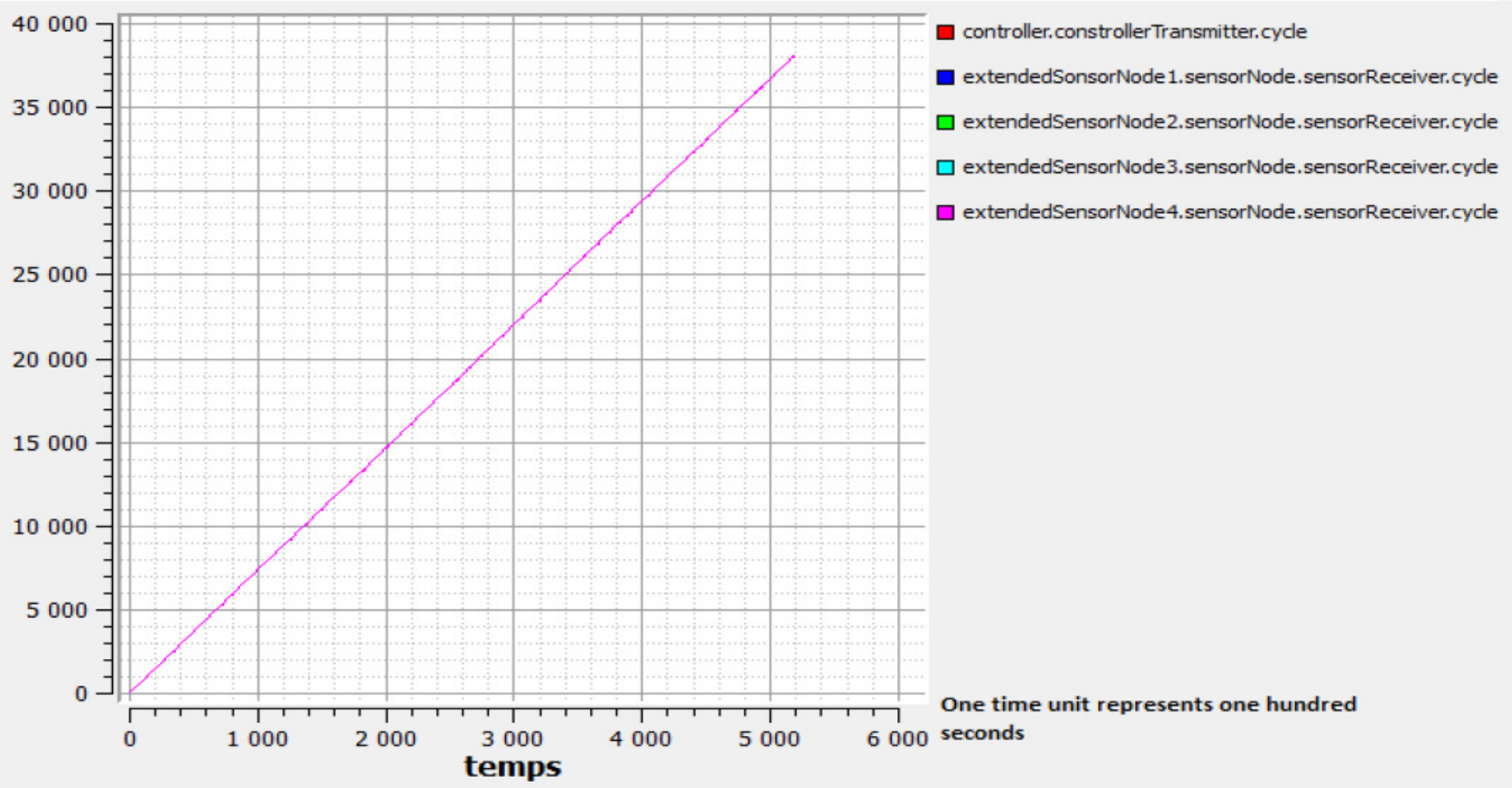
Sensor node Energy Consumption



Messages received by the Controller



Messages sended by the Controller



Conclusion

- Approach to design WSN using the MDA standard;
- Set of rules to transform the SysML model to the Modelica model taking into account static, dynamic and requirement diagrams of SysML;
- A virtual verification/simulation and a requirement tracing of WSN.

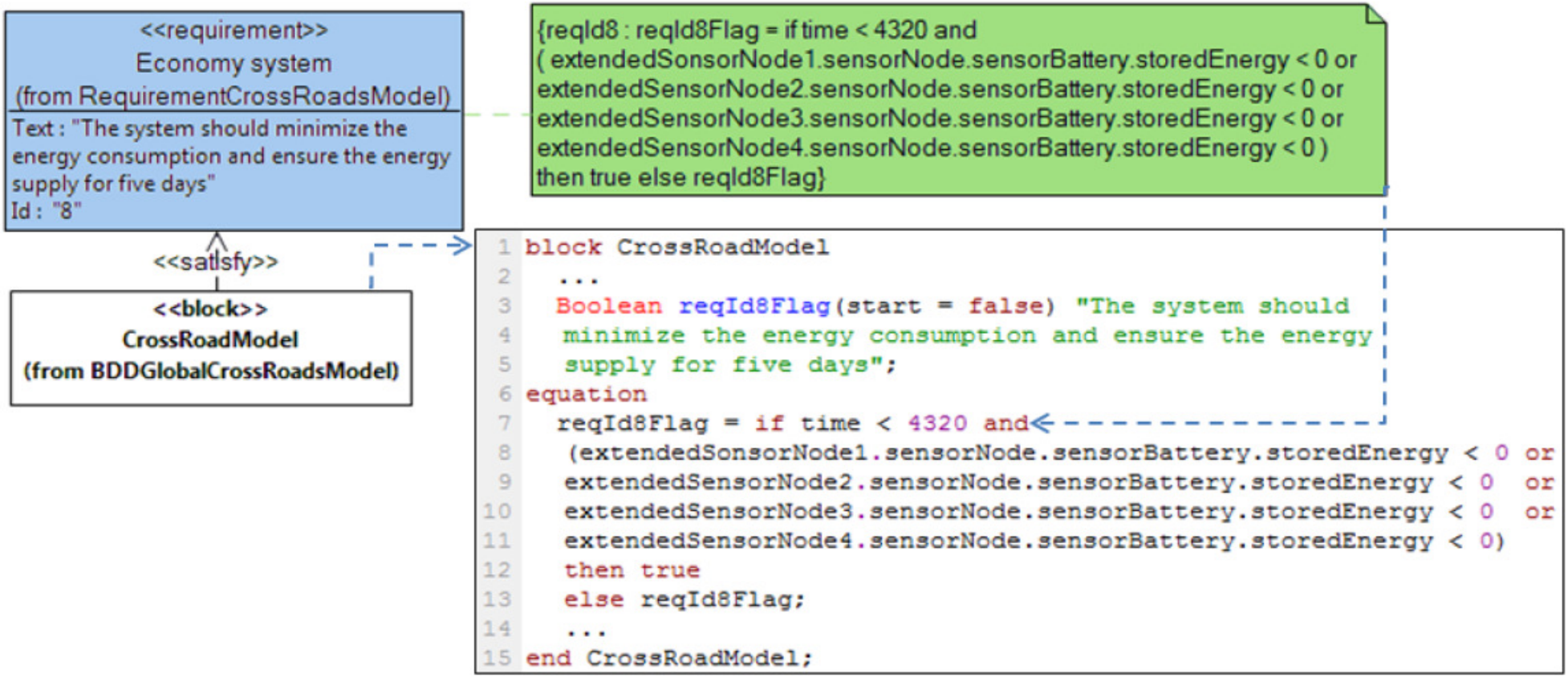
Future work

- Use Sequence and Activity Diagrams to describe the actions of State Machine Diagrams;
- Validate the correctness of transformation rules;
- Verification of functional properties;
- Investigate the automatic reconfiguration property of WSN.

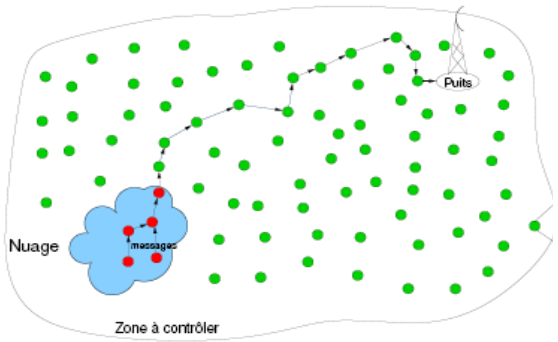
Tank you for your Attention

Context/problematic

Non-functional property



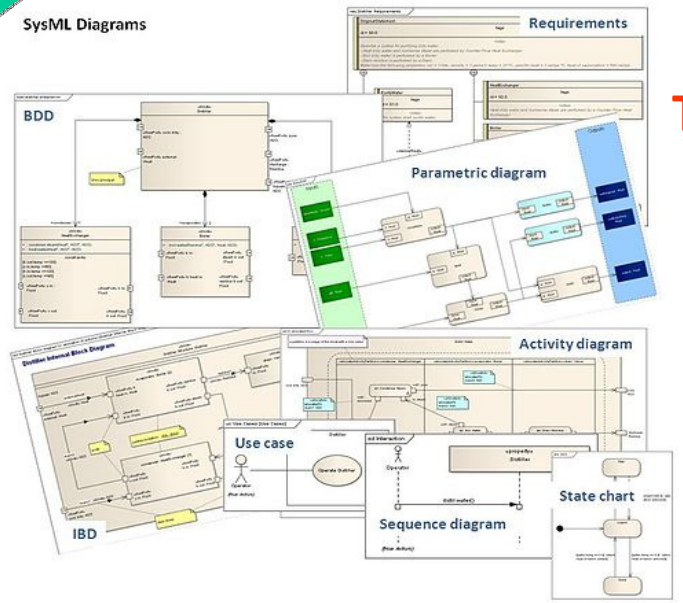
Approach



Specification

```
partial connector
  Modelica.Thermal.HeatTransfer.
      Interfaces.HeatPort
      ''Thermal port for 1-D heat
        transfer'';
    SI.Temperature T
      ''Port temperature'';
    flow SI.HeatFlowRate Q_flow
      ''Heat flow rate (positive if
        flowing into the component)'';
end HeatPort;
```

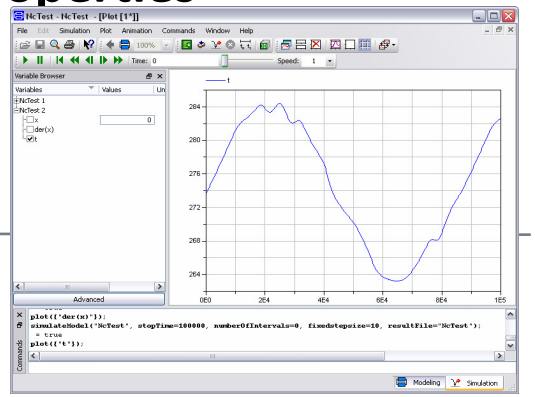
Transformation



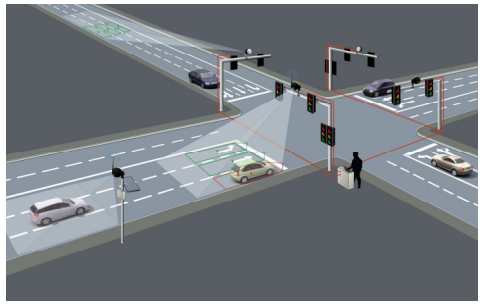
Requirements (Propertiess)

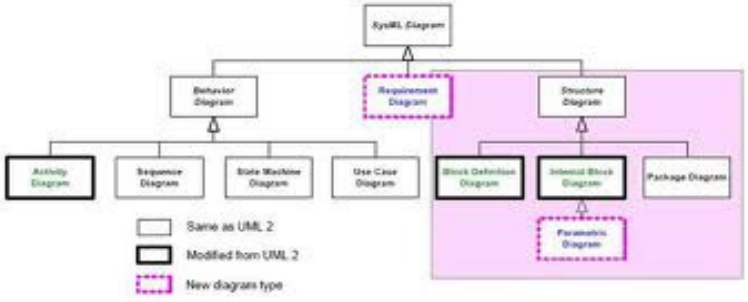
Vérification of functional properties

Simulation of non-functiona properties



Case study

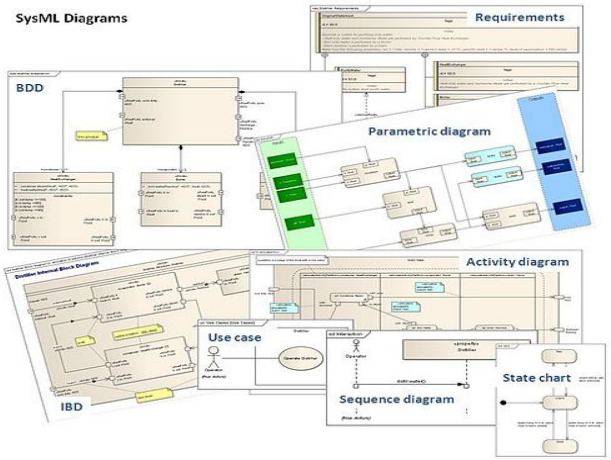




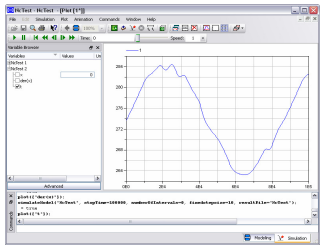
MODELICA

```
partial connector
  Modelica.Thermal.HeatTransfer.
      Interfaces.HeatPort
      ''Thermal port for 1-D heat
        transfer'';
    SI.Temperature T
      ''Port temperature'';
    flow SI.HeatFlowRate Q_flow
      ''Heat flow rate (positive if
        flowing into the component)'';
end HeatPort;
```

□ (⌉ )

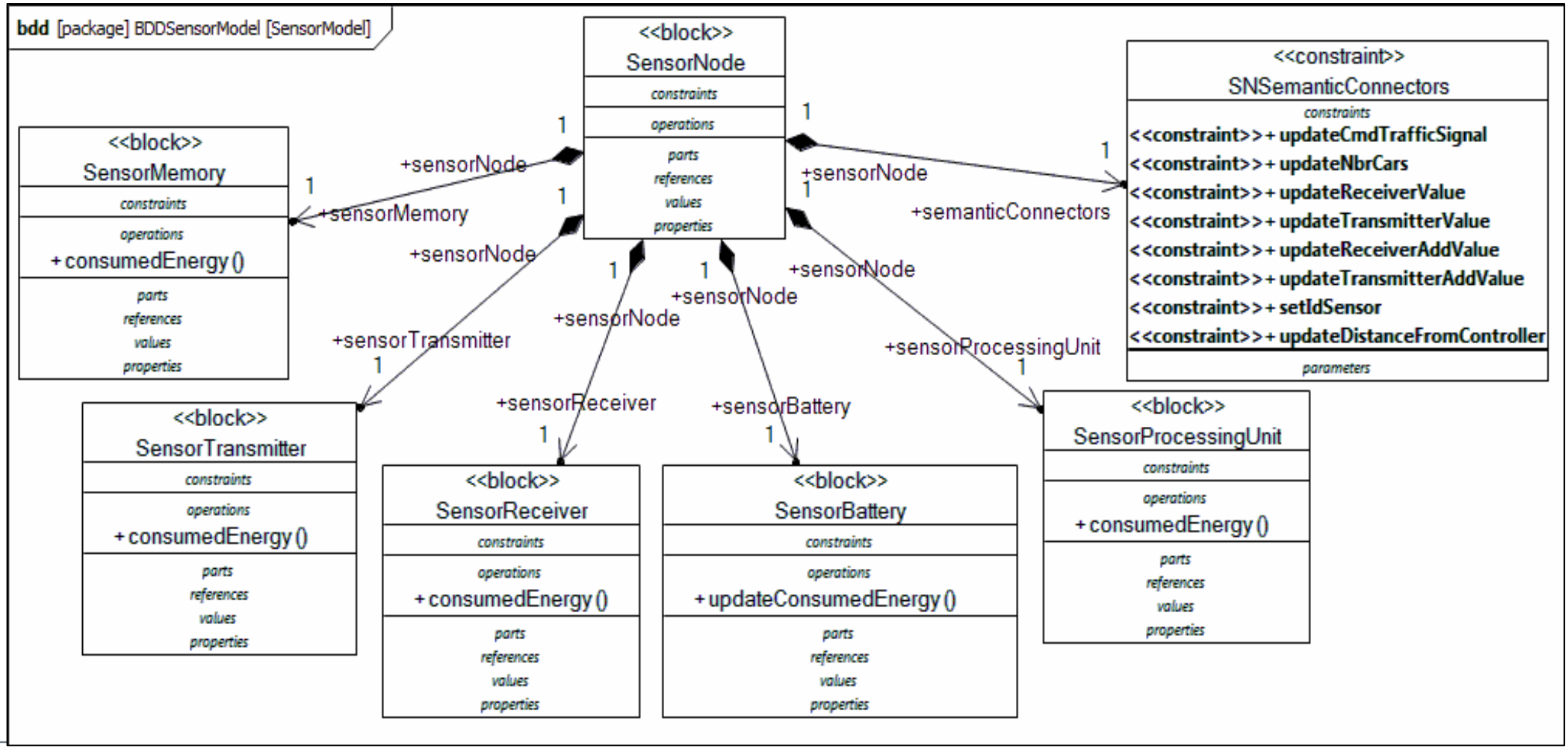


Assert(□ Energy>0)

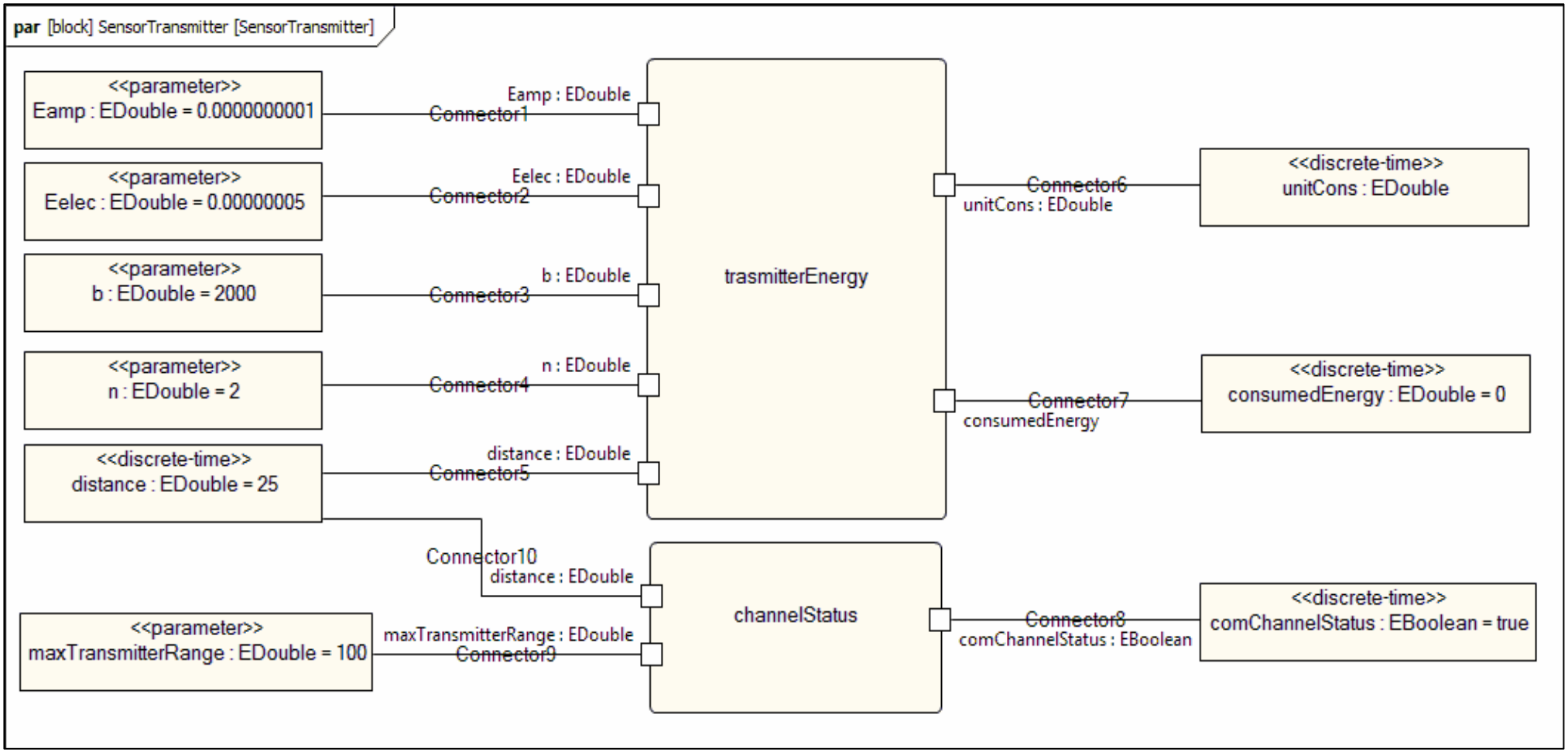


Verification

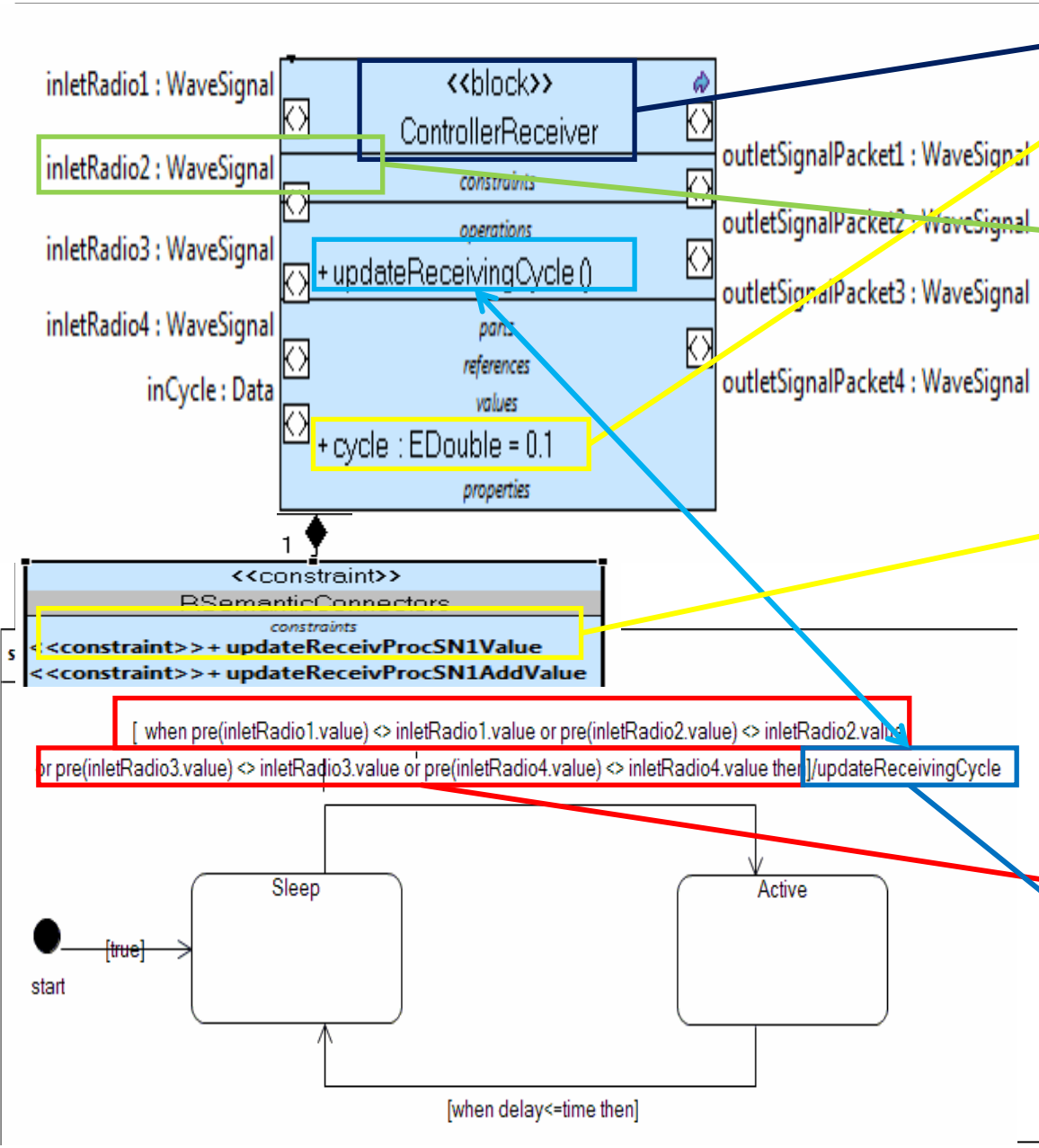
Architecture of a node



Parametric Diagram

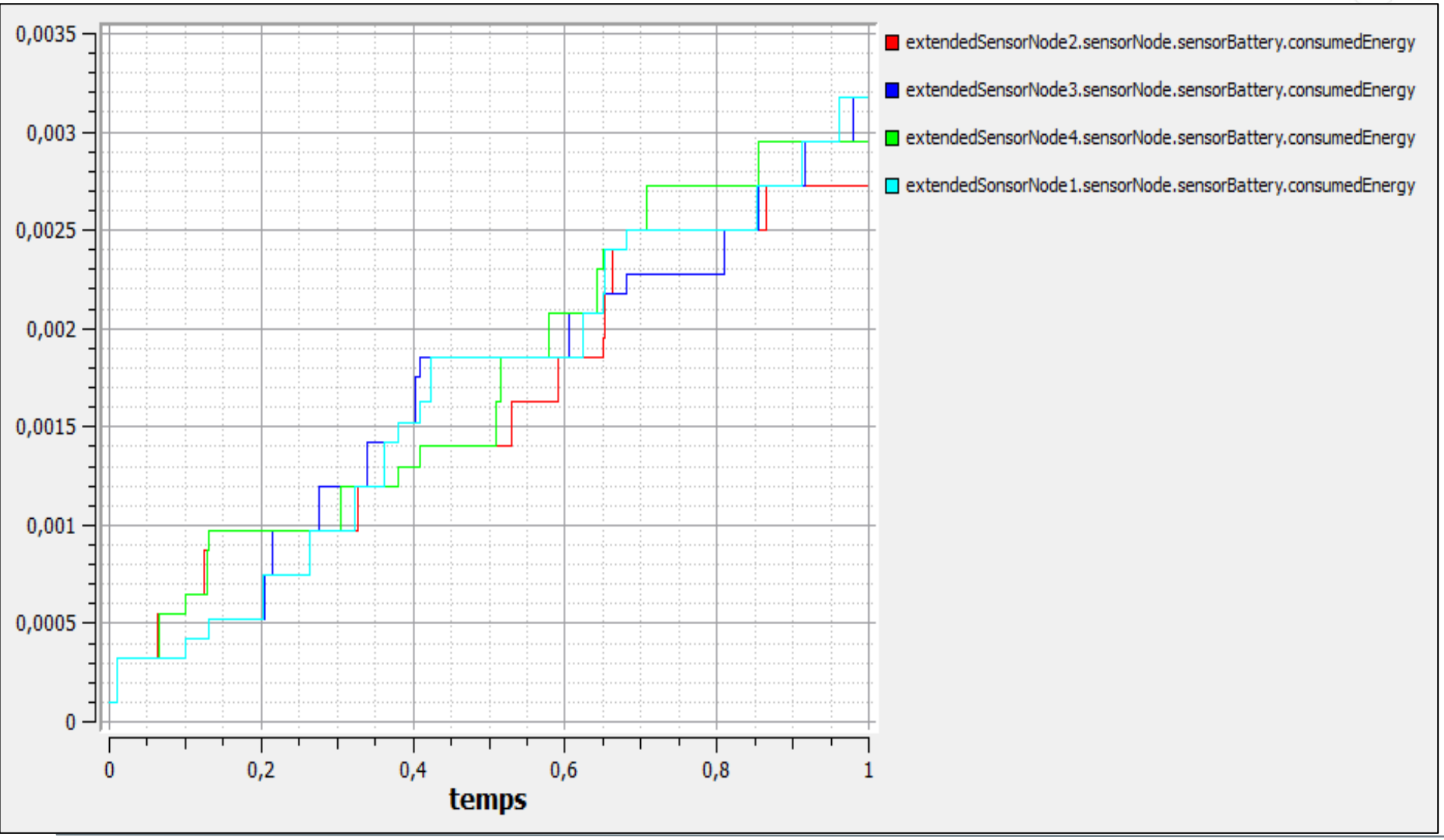


SysML2Modelica




```
block ControllerReceiver
  Real cycle(start = 0.1);
  input WaveSignal inletRadio1;
  output WaveSignal outletSignalPacket1;
  output WaveSignal outletSignalPacket2;
  input WaveSignal inletRadio2;
  output WaveSignal outletSignalPacket3;
  input WaveSignal inletRadio3;
  output WaveSignal outletSignalPacket4;
  input WaveSignal inletRadio4;
  input Data inCycle;
equation
  outletSignalPacket1.value = inletRadio1.value;
  outletSignalPacket1.addvalue = inletRadio1.addvalue;
  outletSignalPacket2.addvalue = inletRadio2.addvalue;
  outletSignalPacket2.value = inletRadio2.value;
  outletSignalPacket3.addvalue = inletRadio3.addvalue;
  outletSignalPacket3.value = inletRadio3.value;
  outletSignalPacket4.addvalue = inletRadio4.addvalue;
  outletSignalPacket4.value = inletRadio4.value;
algorithm
  when pre(inletRadio1.value) <> inletRadio1.value or
       pre(inletRadio2.value) <> inletRadio2.value or
       pre(inletRadio3.value) <> inletRadio3.value or
       pre(inletRadio4.value) <> inletRadio4.value then
         cycle:=pre(cycle) + 1;
  end when;
end ControllerReceiver;
```

Consumption Energy with a sensor



01 unité de temps : représente 100 seconds.