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

<table>
<thead>
<tr>
<th>SysML Model</th>
<th>Modelica Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>package, block, abstract-block</td>
<td>package, block, partial-class</td>
</tr>
<tr>
<td>flow-specification, value-type</td>
<td>connector, type</td>
</tr>
<tr>
<td>flow-property, flow-port</td>
<td>property, connector</td>
</tr>
<tr>
<td>connector flux(x,y)</td>
<td>equation connect(x,y)</td>
</tr>
<tr>
<td>constraint property</td>
<td>equation</td>
</tr>
<tr>
<td>state machine guard</td>
<td>’when’ statement guard</td>
</tr>
<tr>
<td>operation without a value to return</td>
<td>instruction block of ’when’ statement</td>
</tr>
<tr>
<td>operation with a value to return</td>
<td>function</td>
</tr>
<tr>
<td>requirement</td>
<td>boolean expression (invariant or safety constraint)</td>
</tr>
</tbody>
</table>
Case study: Crossroads system
Adopted Energy Model

\[ E_{\text{Sensor Node}} = (E_{\text{Sensing}} + E_{\text{Processing}} + E_{\text{Transmitting}} + E_{\text{Receiving}} + E_{\text{Transient}} + E_{\text{Logging}})^1 \]

- \( E_{\text{Sensing}} = b V_{\text{sup}} I_{\text{sens}} T_{\text{sens}} \)
- \( E_{\text{Processing}} = E_{\text{switch}} + E_{\text{leak}} \)
- \( E_{\text{Transmitting}} = b E_{\text{elec}} + b d_{ij}^n E_{\text{amp}} \)
- \( E_{\text{Receiving}} = b E_{\text{elec}} \)
- \( E_{\text{Transient}} = T_A V_{\text{sup}} [c_N I_A + (1 - c_N) I_S] \)
- \( E_{\text{Logging}} = E_{\text{write}} + E_{\text{read}} \)


Progress In Electromagnetics Research B (PIERB).
BDD of the Global System
BDD of the extended Sensor
BDD of the simple Sensor
Transmitter Parametric Diagram

\[ E_{TN}(b, d_{ij}) = bE_{elec} + b d_{ij}^{n} E_{amp} \]
Transmitter State Machine Diagram

```
block SensorTransmitter
...
Real distance(start = 25);
Real unitCons(start = 0);
Real cycle(start = 0.01);
...
equation
...
algorithm
when pre(cycle) <= time and comChannelStatus then
    consumedEnergy := pre(consumedEnergy) + unitCons;
...
end when;
end SensorTransmitter;
```
Requirement Diagrams

<<requirement>>
Signalized intersection system
Text: "The system must maximize the intersection capacity, minimize the energy consumption and ensure traffic in the junction without accidents"
Id: "1"

<<requirement>>
System safety
Text: "The system must ensure traffic in the junction without accidents through the lights which should be clear and coherent with current traffic laws"
Id: "2"

<<requirement>>
Longevity
Text: "The system must be functional as long as possible"
Id: "5"

<<requirement>>
Traffic lights at junction
Text: "The traffic lights of roads that form the junction are different all the time"
Id: "10"

<<requirement>>
Economy system
Text: "The system should minimize the energy consumption and ensure the energy supply for five days"
Id: "8"
Random Scenario

18
System Energy Consumption
Sensor node Energy Consumption

- Energy consumed by the transmitter
- Energy consumed by the receiver
- Energy consumed by the processor
- Energy consumed by the memory

One time unit represents one hundred seconds.
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
Non-functional property

<<requirement>>
Economy system
(from RequirementCrossRoadsModel)
Text: "The system should minimize the energy consumption and ensure the energy supply for five days"
Id: "8"

<<satisfy>>
<<block>>
CrossRoadModel
(from BDDGlobalCrossRoadsModel)

{reqId8 : reqId8Flag = if time < 4320 and
 (extendedSensorNode1.sensorNode.sensorBattery.storedEnergy < 0 or
 extendedSensorNode2.sensorNode.sensorBattery.storedEnergy < 0 or
 extendedSensorNode3.sensorNode.sensorBattery.storedEnergy < 0 or
 extendedSensorNode4.sensorNode.sensorBattery.storedEnergy < 0)
 then true else reqId8Flag}

1 block CrossRoadModel
2 ...
3 Boolean reqId8Flag(start = false) "The system should minimize the energy consumption and ensure the energy supply for five days"
4 equation
5 reqId8Flag = if time < 4320 and
6 (extendedSensorNode1.sensorNode.sensorBattery.storedEnergy < 0 or
7 extendedSensorNode2.sensorNode.sensorBattery.storedEnergy < 0 or
8 extendedSensorNode3.sensorNode.sensorBattery.storedEnergy < 0 or
9 extendedSensorNode4.sensorNode.sensorBattery.storedEnergy < 0)
10 then true
11 else reqId8Flag;
12 ...
15 end CrossRoadModel;
Approach

Vérification of functional properties

Simulation

Specification

Transformation

Requirements (Properties)

Simulation of non-functional properties

Vérification of functional properties
Case study

Verification

Assert(□ Energy>0)
Architecture of a node
block ControllerReceiver
{
  Real cycle (start = 0.1);
  input WaveSignal inletRadio1;
  input WaveSignal inletRadio2;
  input WaveSignal inletRadio3;
  input WaveSignal inletRadio4;
  input Data inCycle;
  output WaveSignal outletSignalPacket1;
  output WaveSignal outletSignalPacket2;
  output WaveSignal outletSignalPacket3;
  output WaveSignal outletSignalPacket4;
  equation
  outletSignalPacket1.value = inletRadio1.value;
  outletSignalPacket1.addvalue = inletRadio1.addvalue;
  outletSignalPacket2.value = inletRadio2.value;
  outletSignalPacket2.addvalue = inletRadio2.addvalue;
  outletSignalPacket3.value = inletRadio3.value;
  outletSignalPacket3.addvalue = inletRadio3.addvalue;
  outletSignalPacket4.value = inletRadio4.value;
  outletSignalPacket4.addvalue = inletRadio4.addvalue;

  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
    updateReceivingCycle()
  end when;
  cycle := pre(cycle) + 1;
}

Consumption Energy with a sensor

01 unité de temps : représente 100 seconds.