GLONEMO : Global and Accurate Formal Models for the Analysis of Sensor Networks.

http ://www-verimag.imag.fr/∼samper/Glonemo/

Ludovic Samper, Florence Maraninchi, Laurent Mounier and Louis Mandel

Context and objectives

Sensor networks :

- Hundreds or thousands of nodes
- No infrastructure
- Low rate
- Low battery
- Applications : detection or monitoring of an event in a distributed manner

Objectives :

Executable and analyzable models of sensor networks

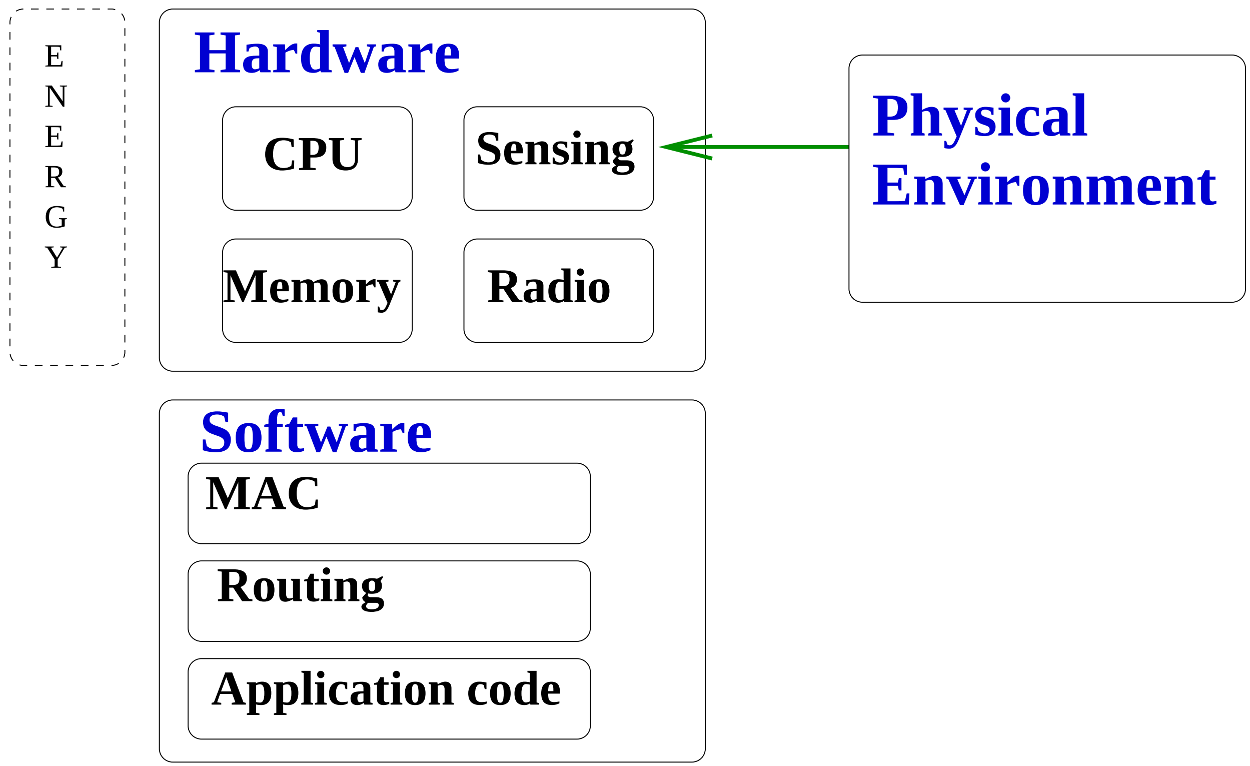
with special emphasis on energy consumption.

Related Work

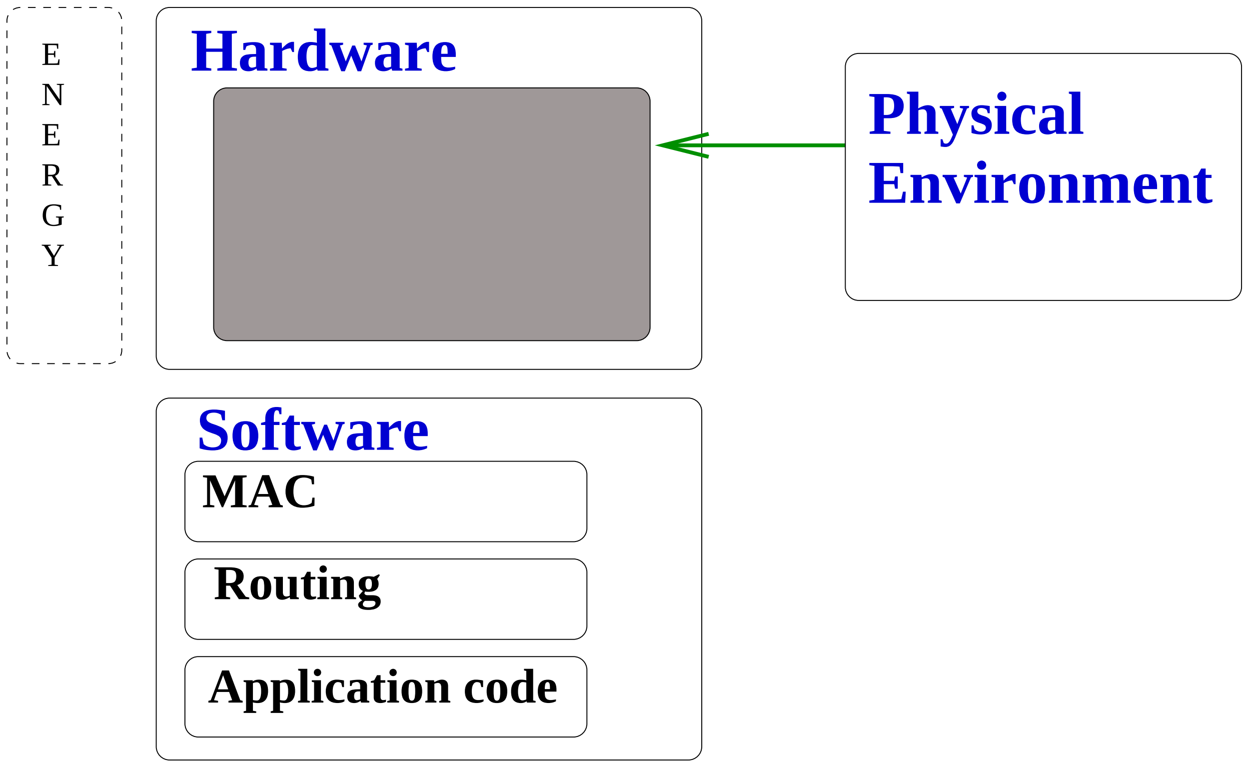
- Classical network simulators, not dedicated to sensor networks :
  - NS2 (The Network Simulator), Opnet, Glomosim, ...
  - NAB (Network in A Box)
- Simulators dedicated to sensor networks :
  - PowerTOSSIM, the consumption is calculated from the number of packets transmitted and the number of instructions executed.
  - Avrora, written in Java and cycle-accurate
  - Atemu, executes binary codes
  - ...
- Formal models applied to sensor networks :
  - Real-Time Maude (Peter C. Ölveczky)
  - ...
- **Our work :**
  - A model dedicated to sensor networks
  - Accurate model of the energy consumption
  - Model and program using a formally defined language
  - The aim is to use formal methods

Approach

- **A global model**
  - Detailed Hardware
  - Software : the protocol layers and the application code.
  - Physical Environment



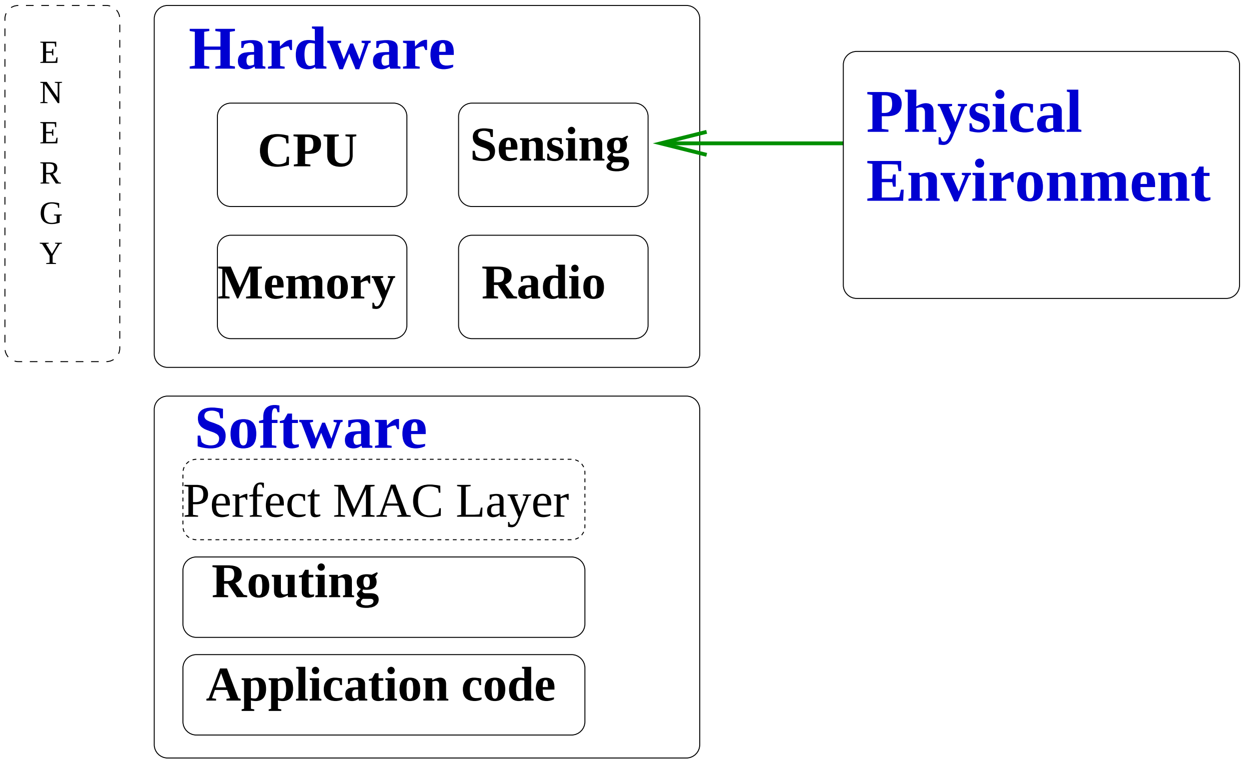
Modular abstractions :



We want much more than an executable tool :

The aim is to use verification tools, runtime-verification tools or formal test.

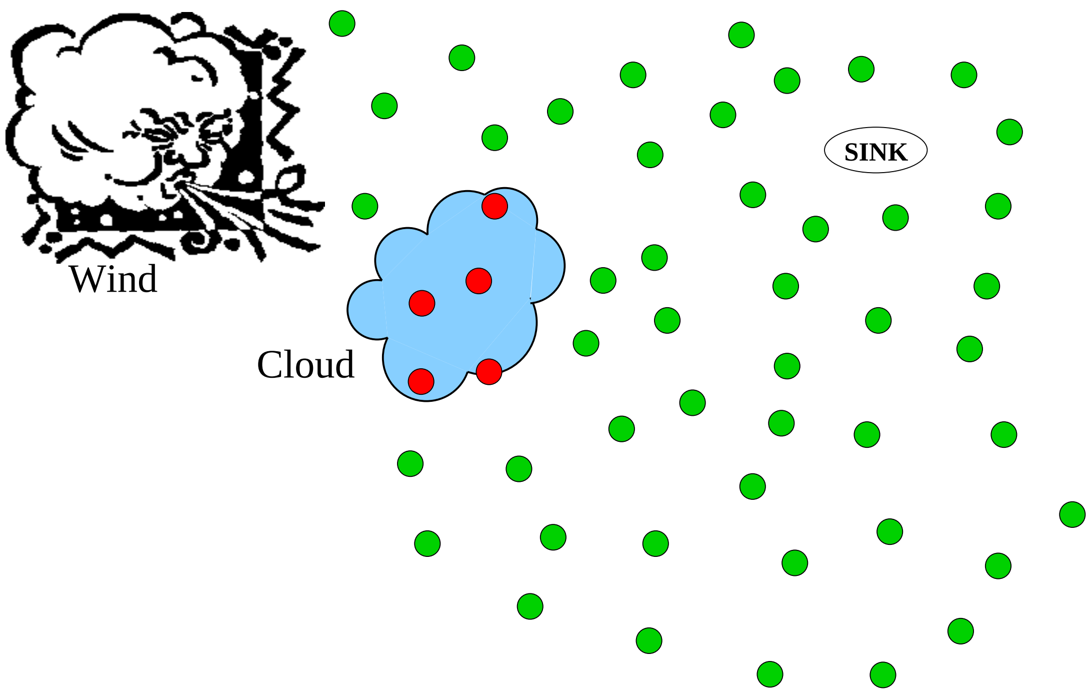
Modular abstractions :



We want much more than an executable tool :

The aim is to use verification tools, runtime-verification tools or formal test.

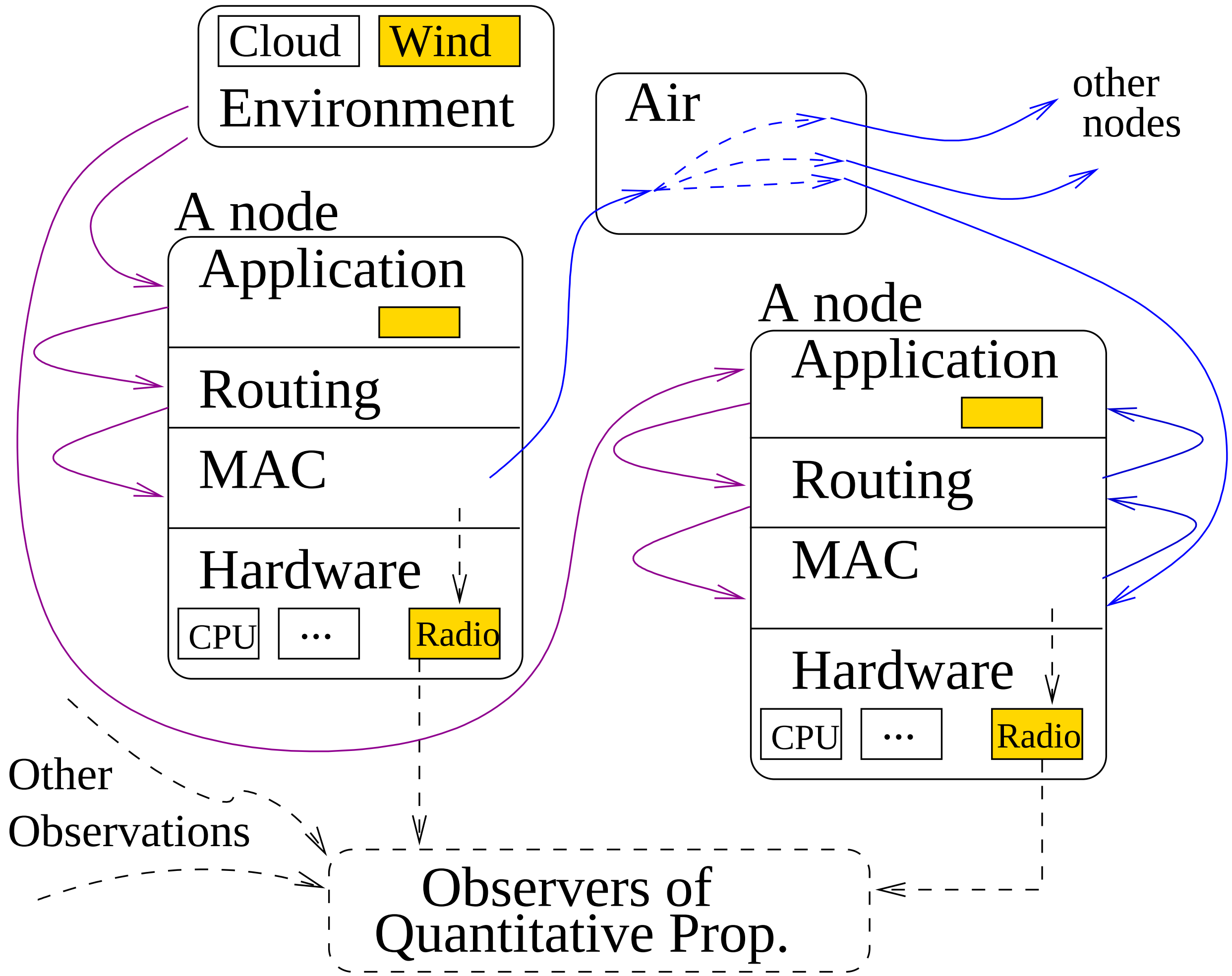
Typical Example



- **Application :** Detection of a radioactive cloud
- **Routing :** Directed diffusion

  (C. Intanagowiwat, R. Govindan, D. Estrin, J. Heidemann, F. Silva)
- **Medium Access Control :** A preamble sampling MAC protocol
- **Environment :** A cloud moving under the influence of the wind.

Structure of the model



Parallel processes with synchronization

Tools used to program the model

– ReactiveML (Louis Mandel, LIP6) :

  – The ML-language with parallelism
  – As expressive as the Caml language
  – Parallelism is a top-level primitive
  – Belongs to the family of synchronous languages

  The hardware model, the software and the simulation engine are implemented with ReactiveML

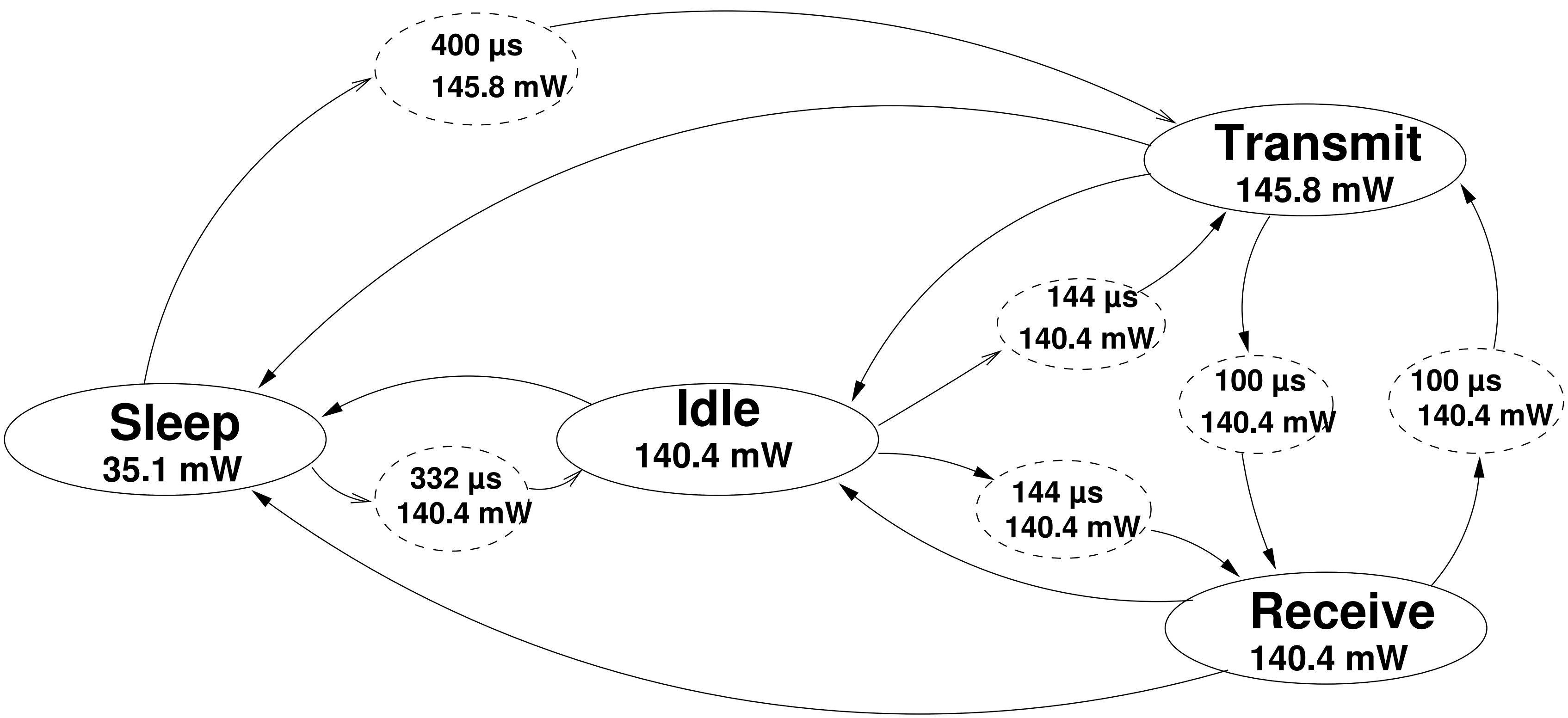
– Lucky (E. Jahier, P. Raymond, VERIMAG) :

  – A constraint-based language
  – A language for describing and simulating stochastic reactive systems
  – Lucky is connected to ReactiveML

  The cloud and the wind are implemented with Lucky

The consumption model of the radio

The MAC layer drives this automaton.
An "observer" checks the current state to calculate the consumption of the node.



Values of the Motorola MC13192

Software : ReactiveML code for the application

```
let send_alarm self cloud_pos my_interest =
  if (present_cloud self cloud_pos) then
    (if (not self.node_pre_present_cloud) then
      (response self my_interest;
       self.node_pre_present_cloud <- true;
      )
    )
  else self.node_pre_present_cloud <- false;;
```

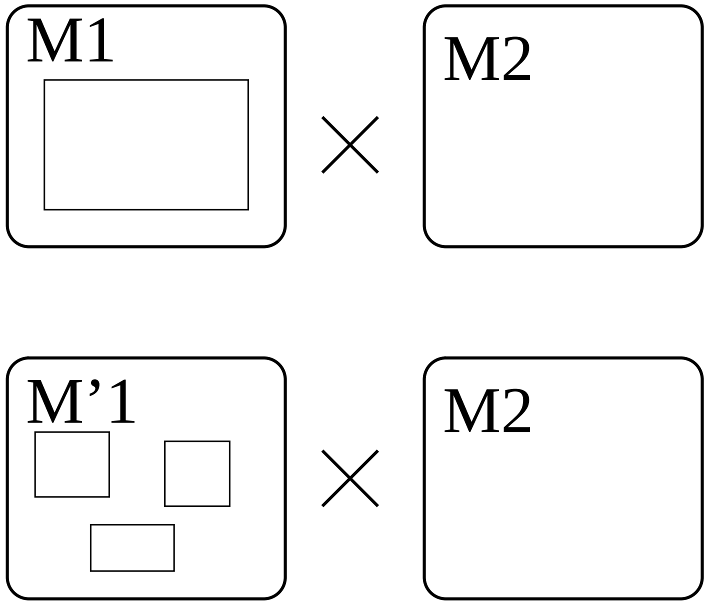
Environment : The Lucky code for the cloud

```
inputs { Wind_x : float ~init 0.0;
         Wind_y : float ~init 0.0;}
outputs {
  x_cloud: float ~init 400.0 ~max 1000.0 ~min -100.0;
  y_cloud: float ~init 300.0 ~max 1000.0 ~min -100.0;
}
transitions {
 init -> init ~cond
       (if Wind_y >= 0.0
          then (0.0 <= (y_cloud - pre y_cloud) <= Wind_y)
        else (Wind_y <= (y_cloud - pre y_cloud) <= 0.0))
  and (if Wind_x >= 0.0
          then (0.0 <= (x_cloud - pre x_cloud) <= Wind_x)
        else (Wind_x <= (x_cloud - pre x_cloud) <= 0.0))}
```

Environment : The Lucky code for the wind

```
inputs { }
outputs {
    Wind_x : float ~min -5.0 ~max 5.0 ~init 0.0;
    Wind_y : float ~min -5.0 ~max 5.0 ~init 0.0;
}
transitions {
 init -> init ~cond
    abs (Wind_y - pre Wind_y) < 5.0 and
    abs (Wind_x - pre Wind_x) < 5.0
}
```

Modular abstractions : Use of formal models



M_1' is more precise than M_1.

The consumption evaluated with the M_1' model must be smaller than the one evaluated with M_1 ; and this relation must stay true after composition.

$$M_1' \preccurlyeq M_1 \quad \Rightarrow \quad M_1' \parallel M_2 \preccurlyeq M_1 \parallel M_2$$

Conclusions and perspectives

– **The formalism** is more expressive than other formal models

– **The formalism** enables modular abstractions

– **Our Model**, GLONEMO, includes a realistic model of the environment

– GLONEMO can be used to perform simulations.
This is useful to have an intuition of the abstractions that could be done.

– Indeed, the simulator is quite scalable.

– This complete model can already be used to perform automatic testing ; the modular abstractions will allow the use of verification and runtime-verification tools.