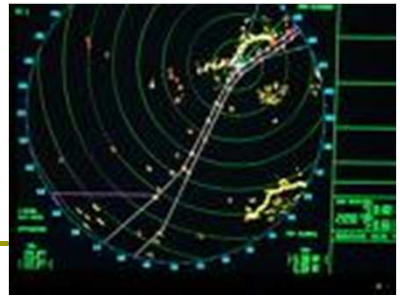
# AADL: a radar case study

# Back to radar case study



- Goal: to model a simple radar system
- Let us suppose we have the following requirements

1. System implementation is composed by physical devices (Hardware entity): antenna + processor + memory + bus
2. and software entity : running processes and threads + operating system functionalities (scheduling) implemented in the processor that represent a part of execution platform and physical devices in the same time.
3. The main process is responsible for signals processing : general pattern: transmitter -> antenna -> receiver -> analyzer -> display
4. Analyzer is a periodic thread that compares transmitted and received signals to perform detection, localization and identification.
5. [..]

# Tools used for modeling

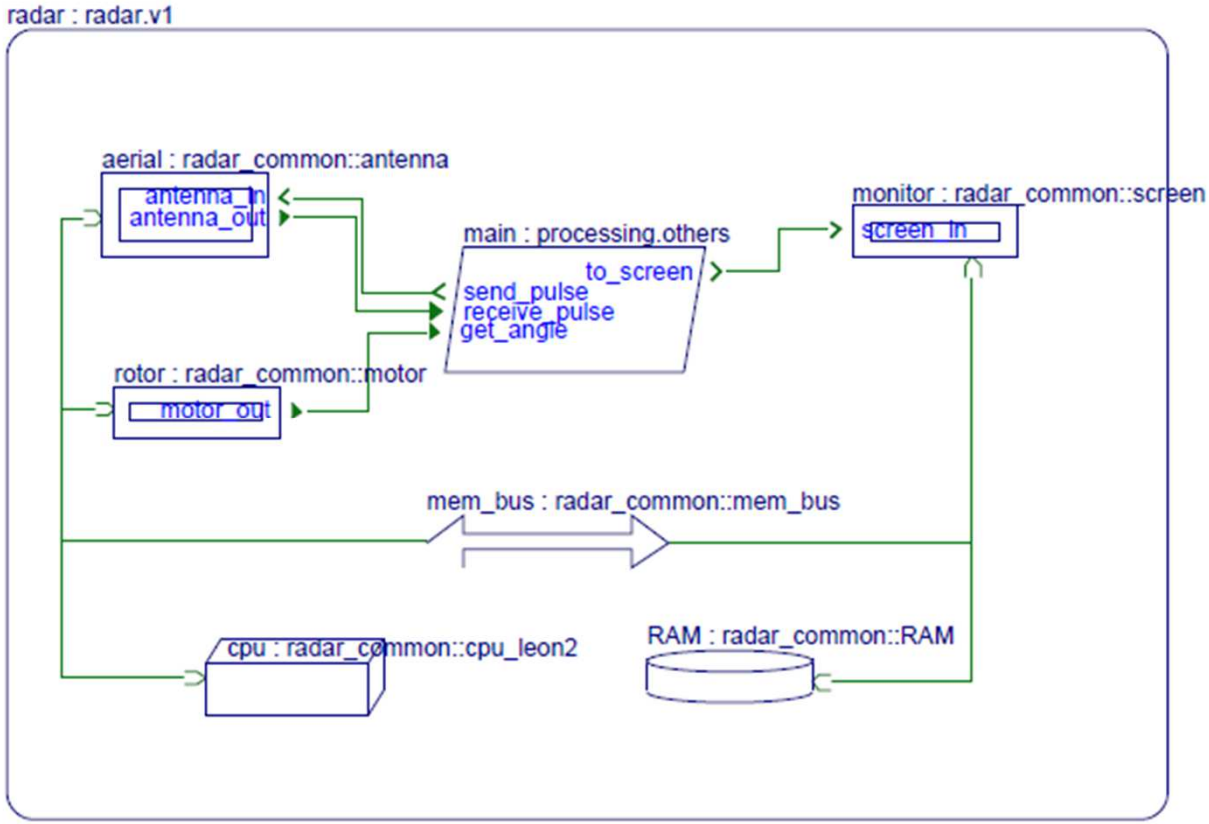
- AADL syntax is both textual and graphical, with several editors available
  - Modes exist for emacs, vi
  - OSATE provides a comprehensive textual IDE on top of Eclipse, and additional plug-ins
    - IMV : Instance Model Viewer
    - Consistency checkers, statistics, various analysis.
  - Stood for AADL:
    - Top-down modeling approach
    - Instance Model graphical editor
    - Generation of textual AADL for tool interoperability
  - …
- In the following, we will use Stood

# Radar case study

- Hardware/Software breakdown: components

```
PACKAGE radar_v1
PUBLIC
-- …
SYSTEM radar
END radar;
-- …
PROCESS processing
-- …
END processing;
-- …
END radar_v1;

PACKAGE radar_common
PUBLIC
-- …
DEVICE screen
-- …
END screen;
-- …
END radar_common;
```

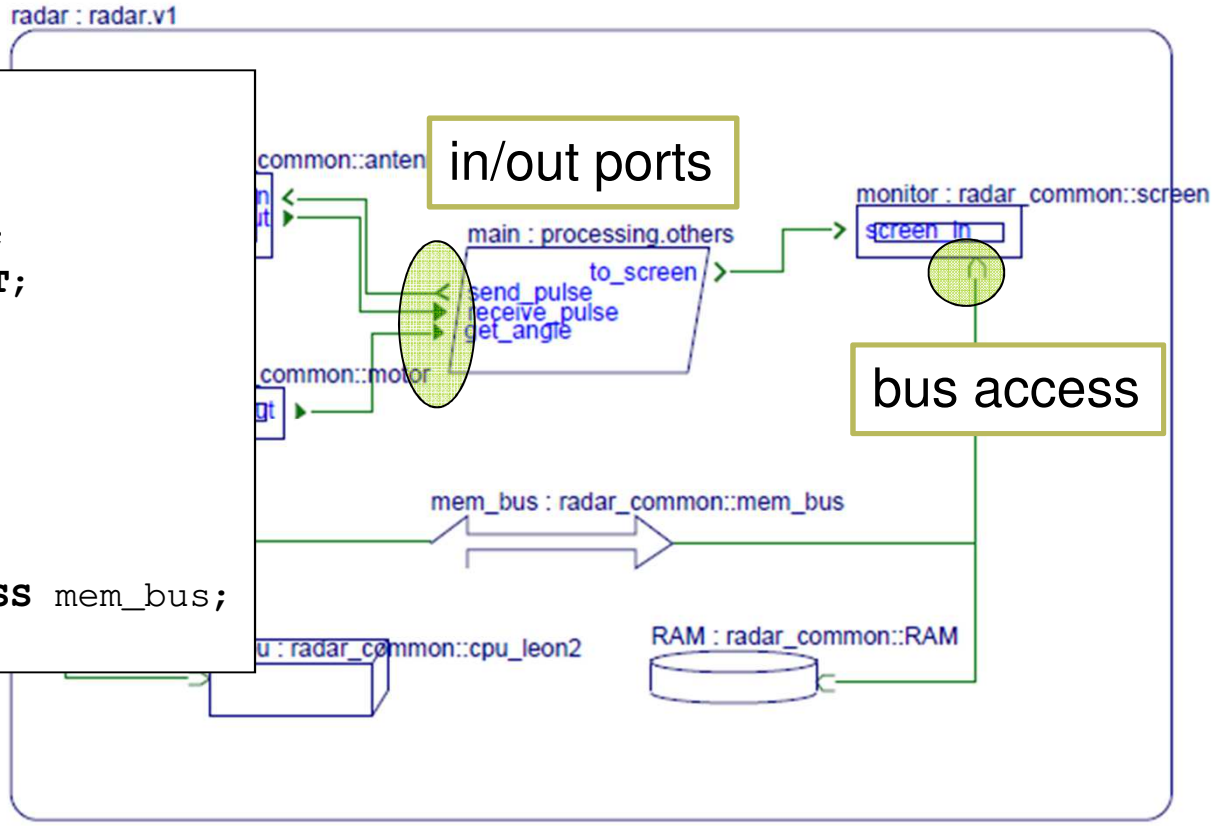


# Radar case study

- Hardware/Software breakdown: features

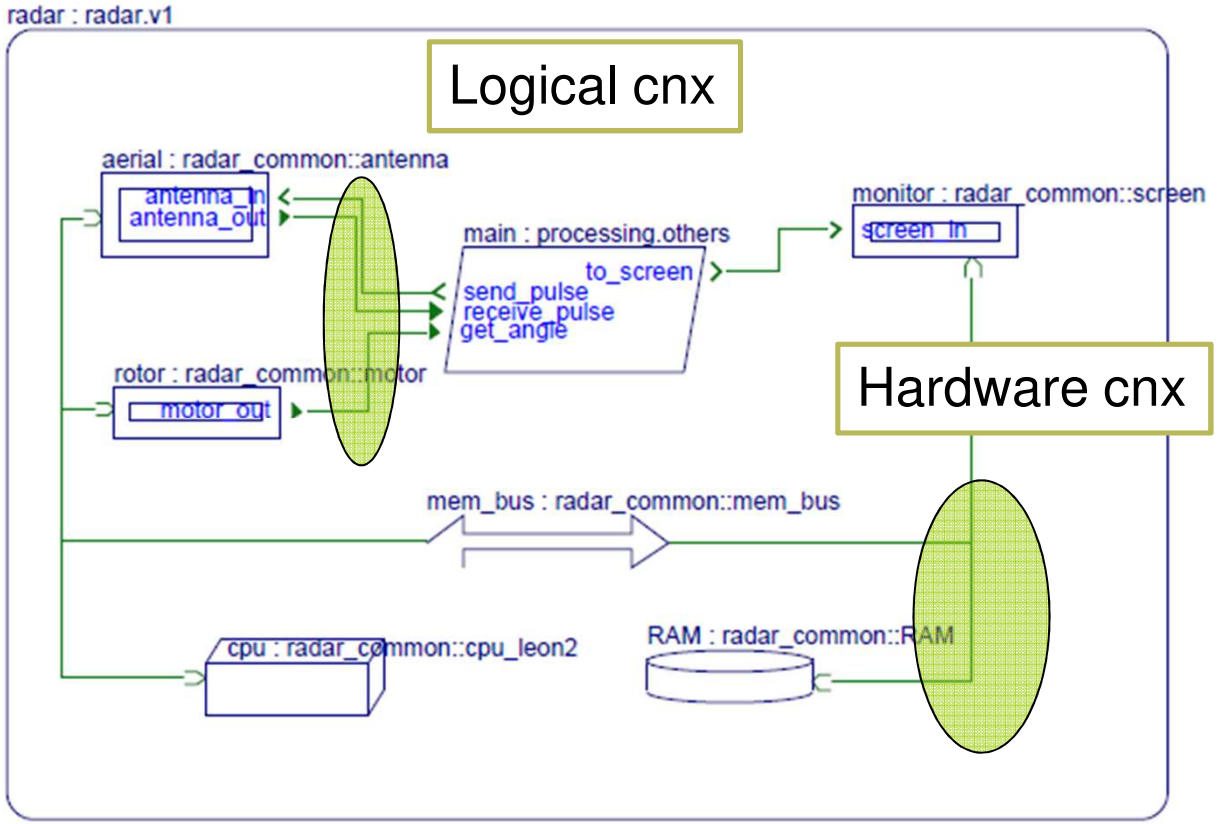
```
PROCESS processing
FEATURES
  to_screen : OUT EVENT PORT;
  send_pulse : OUT EVENT PORT;
  receive_pulse : IN DATA PORT;
  get_angle : IN DATA PORT;
END processing;

DEVICE screen
FEATURES
  screen_in : IN EVENT PORT;
  mem_bus : REQUIRES BUS ACCESS mem_bus;
END screen;
```



# Radar case study

- Hardware/Software breakdown: connections




*note:*
*bindings are not represented graphically with Stood*

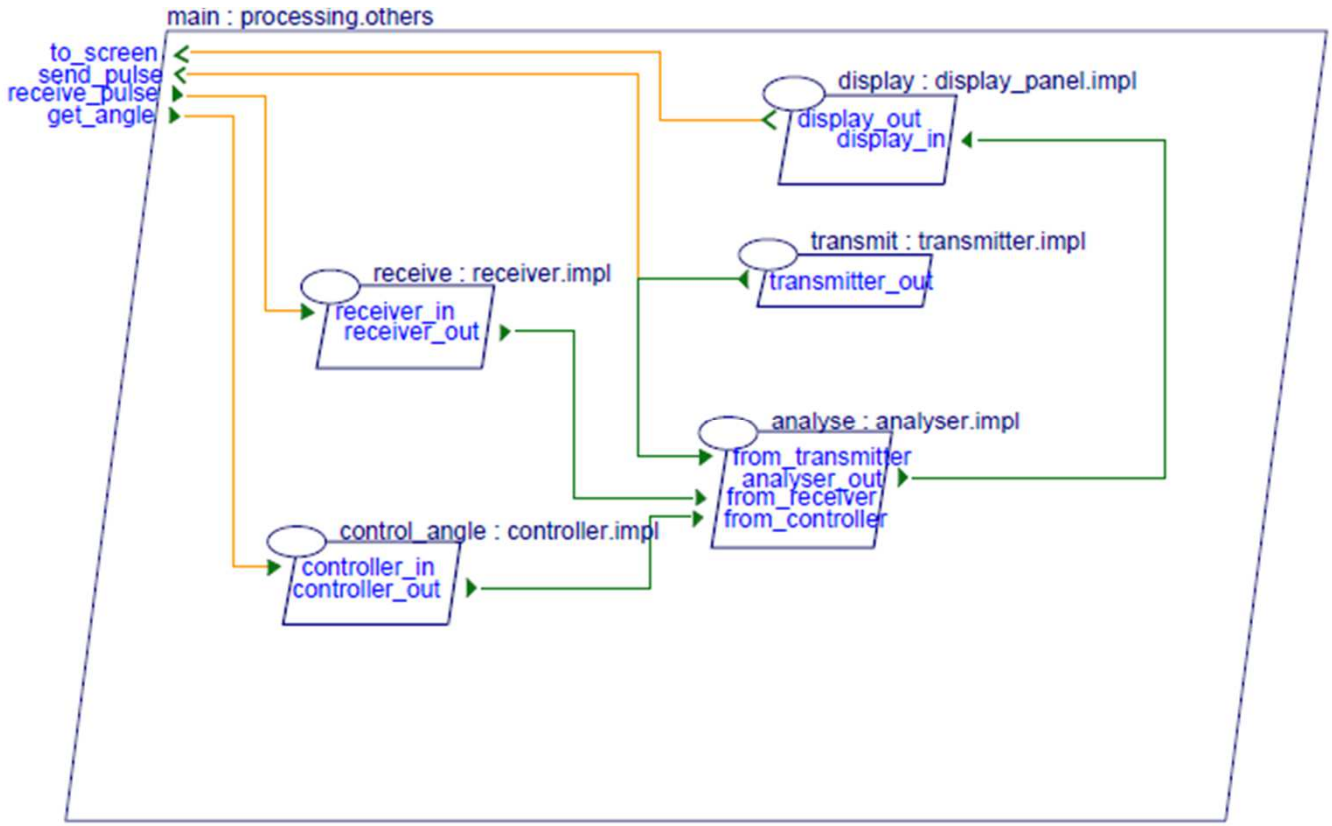
# Radar case study

- Hardware/Software breakdown: connections

```
SYSTEM IMPLEMENTATION radar.v1
SUBCOMPONENTS
  aerial : DEVICE radar_common::antenna;
  rotor : DEVICE radar_common::motor;
  monitor : DEVICE radar_common::screen;
  cpu : PROCESSOR radar_common::cpu_leon2;
  mem_bus : BUS radar_common::mem_bus;
  RAM : MEMORY radar_common::RAM;
  main : PROCESS processing.others;
CONNECTIONS
  cnx1 : PORT aerial.antenna_out -> main.receive_pulse;
  cnx2 : PORT rotor.motor_out -> main.get_angle;
  cnx3 : PORT main.send_pulse -> aerial.antenna_in;
  cnx4 : PORT main.to_screen -> monitor.screen_in;
  cnx5 : BUS ACCESS mem_bus -> aerial.mem_bus;
  cnx6 : BUS ACCESS mem_bus -> rotor.mem_bus;
  cnx7 : BUS ACCESS mem_bus -> monitor.mem_bus;
  cnx8 : BUS ACCESS mem_bus -> cpu.mem_bus;
  cnx9 : BUS ACCESS mem_bus -> RAM.mem_bus;
-- …
END radar.v1;
```

# Radar case study

- Software elements



# A few words on AADL usage

- AADL is for architectural description and early analysis
- Not to be compared with UML suites
  - Not a graphical representation of the source code
  - But can be associated with existing source code via Properties
- Keep in mind models support an objective
  - For now, it is just a high-level view of the design
- In the next sections, we will complete the models with properties to support schedulability analysis