Multi-Paradigm Modeling for early Analysis of ROS-based Robotic Applications using a Library of AADL Models

Eric Senn
Lab-STICC
Université de Bretagne Sud
Lorient, France
eric.senn@univ-ubs.fr

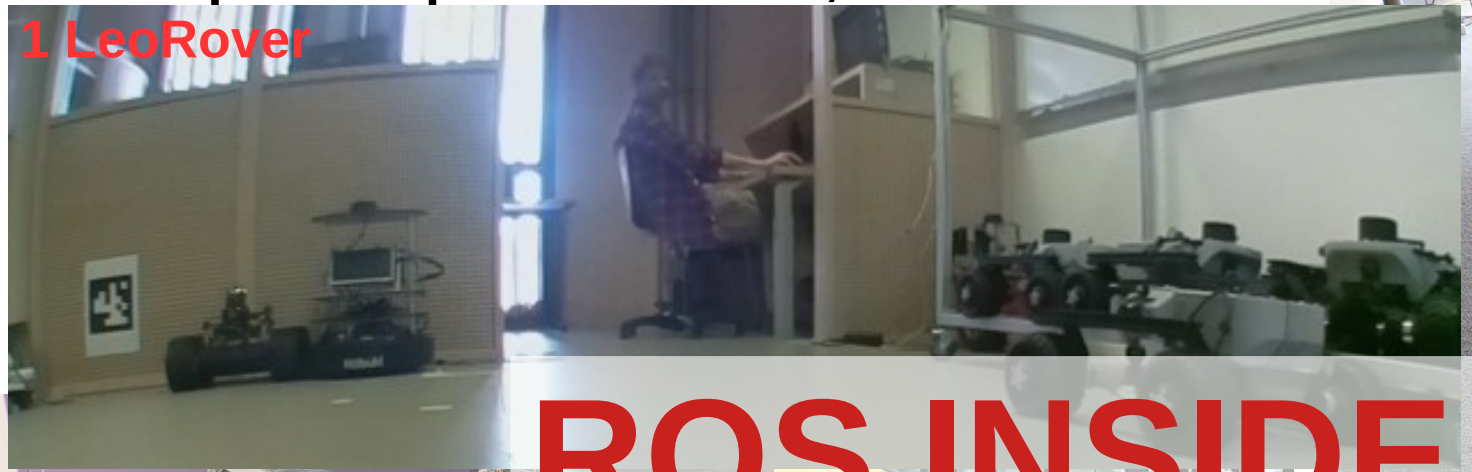
Lucie W. J. Bourdon
Lab-STICC
Université de Bretagne Sud
Lorient, France
lucie.bourdon@univ-ubs.fr

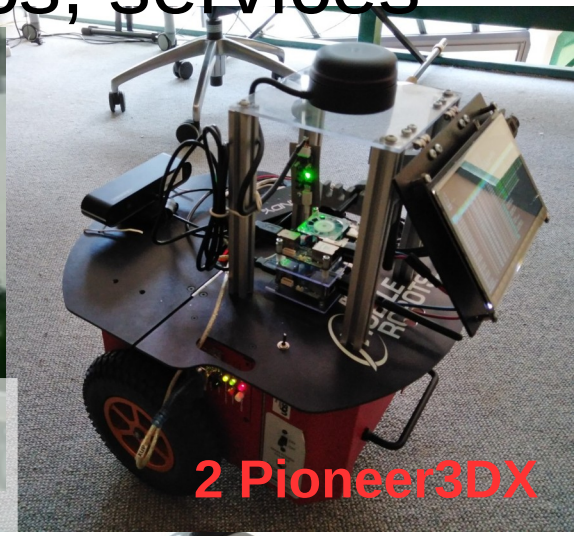
Dominique Blouin
LTCI, Telecom Paris,
Institut Polytechnique de Paris
Palaiseau, France
dominique.blouin@telecom-paris.fr

MODELS / MPM4CPS'22
Montreal – 23/10/2022
Séminaire FARO 23/11/2022

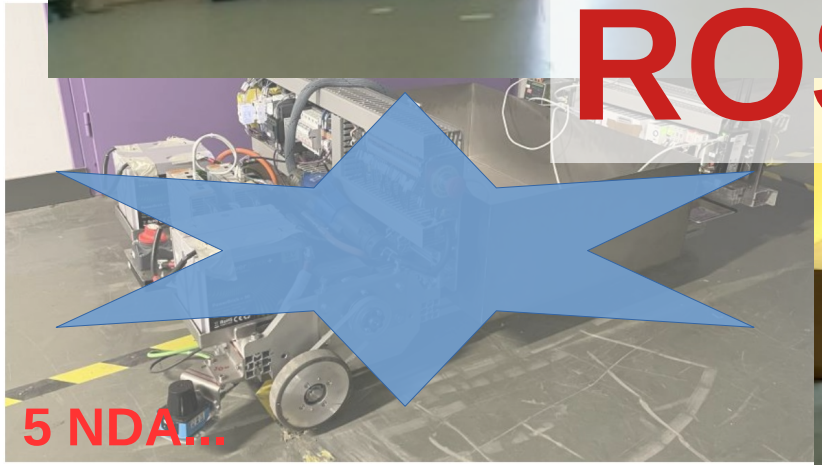
Our Robotics Systems

- Mobile robots in different domains : industry / transport / production, assistive robotics, services





ROS INSIDE



… Marine and Sub-marine Drones

- … Autonomous robots

8 E-Cobot

7 IMS

6 RTSys

ROS INSIDE

WHY ROS …?

ROS = Robot Operating System

- Middleware to ease the programming of robots
  - Standard synchronization and communication mechanisms to hide low-level OS services
  - Multi-platform / multi-OS

lidar node

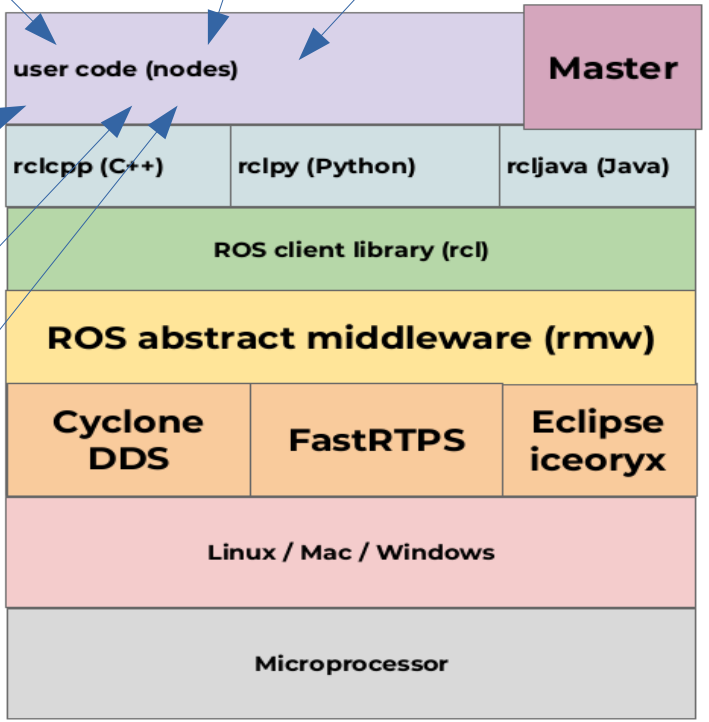
depth cam node

teleop node

obstacle detect node

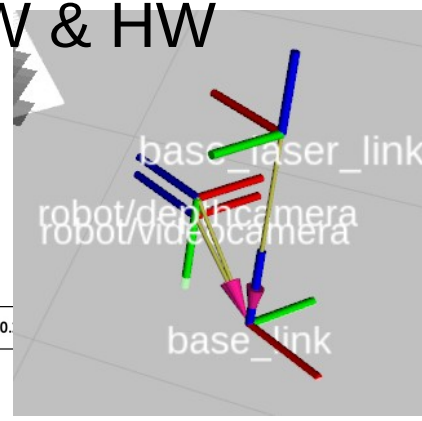
robot base node

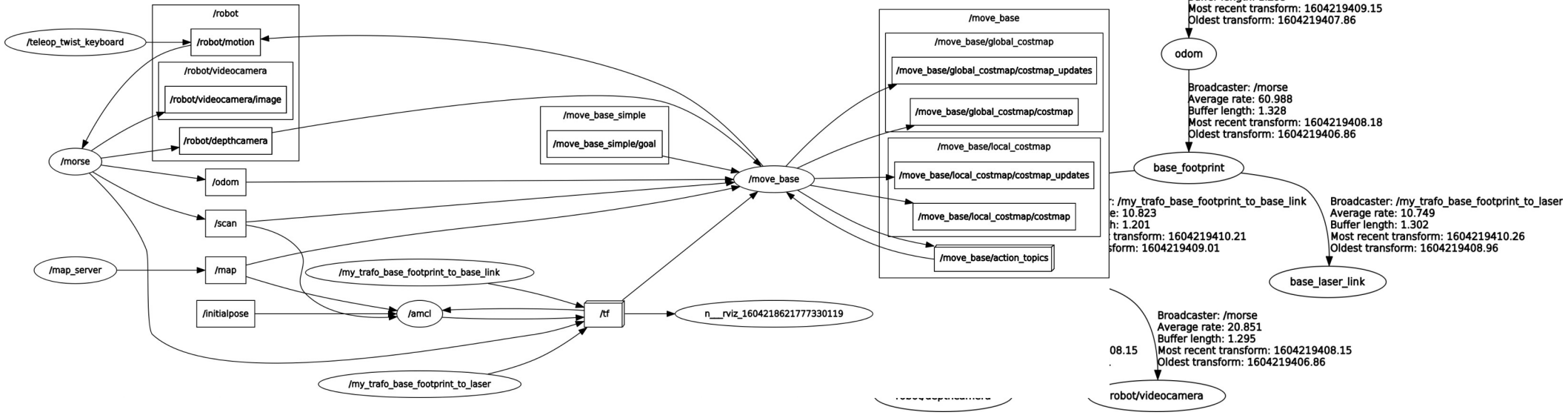
mapping node



ROS because ...

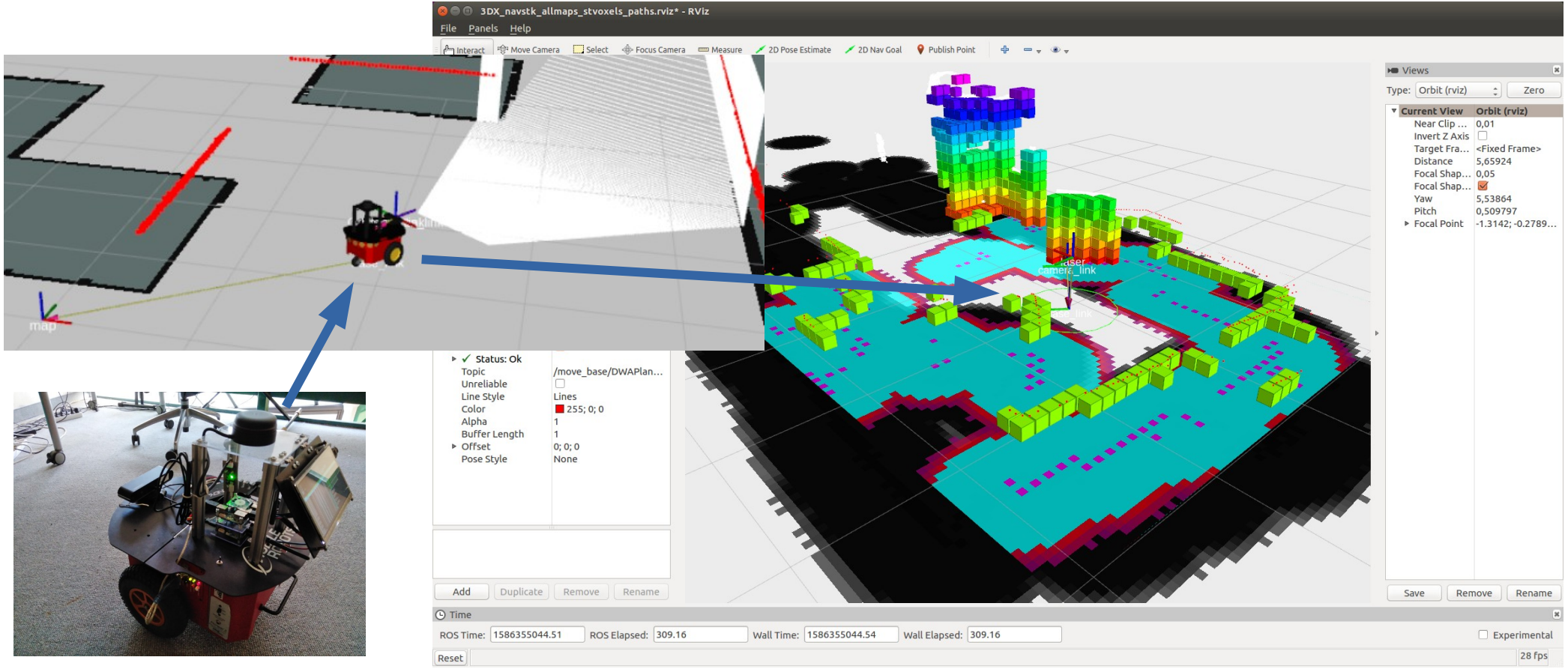
- Simplifies and accelerates the development of robots SW & HW
- Standard with many sensors and robots hardware
- Used in industry : demand from our clients / partners
- Set of tools for developing, monitoring, debugging
  - Nodes connections, frame transformations ...





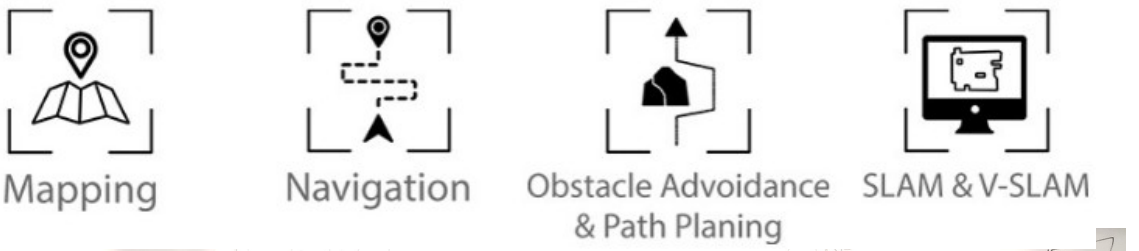
Visualization and Simulation

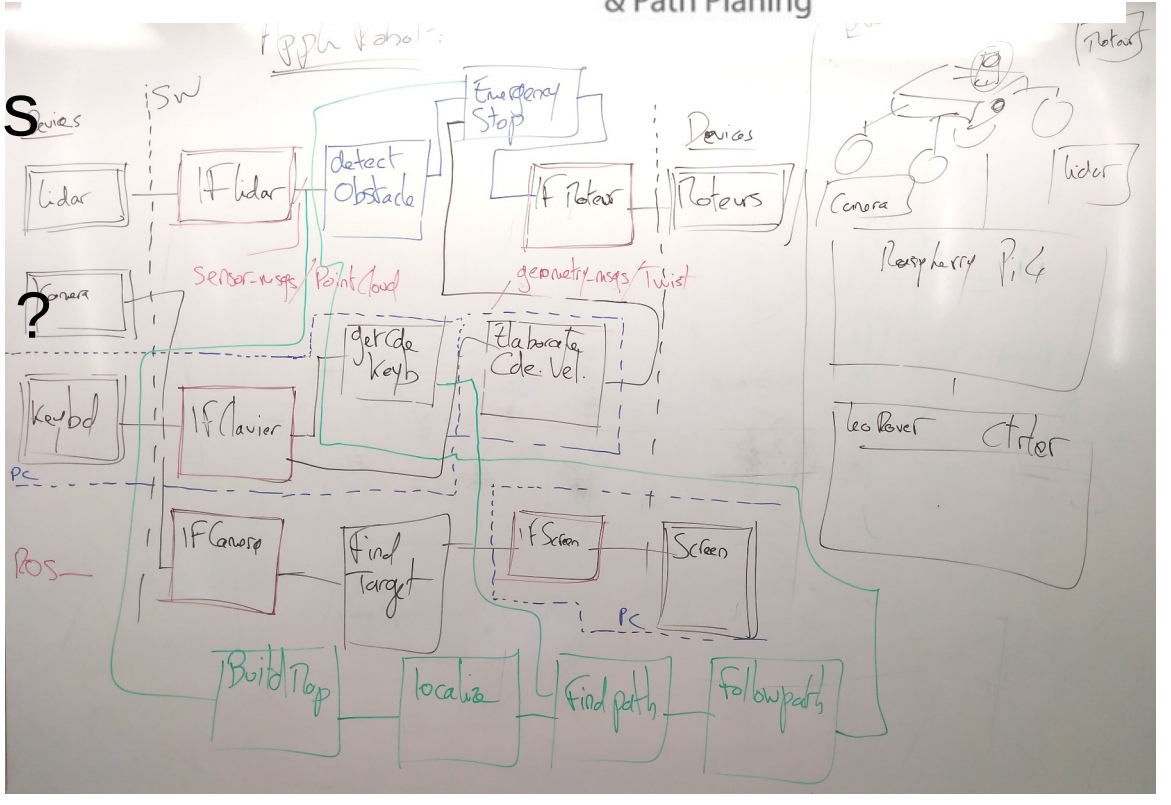
- More tools for multi-physical simulation (Gazebo, Morse …)
- Digital twins in the development process



BUT … Help is Needed ...

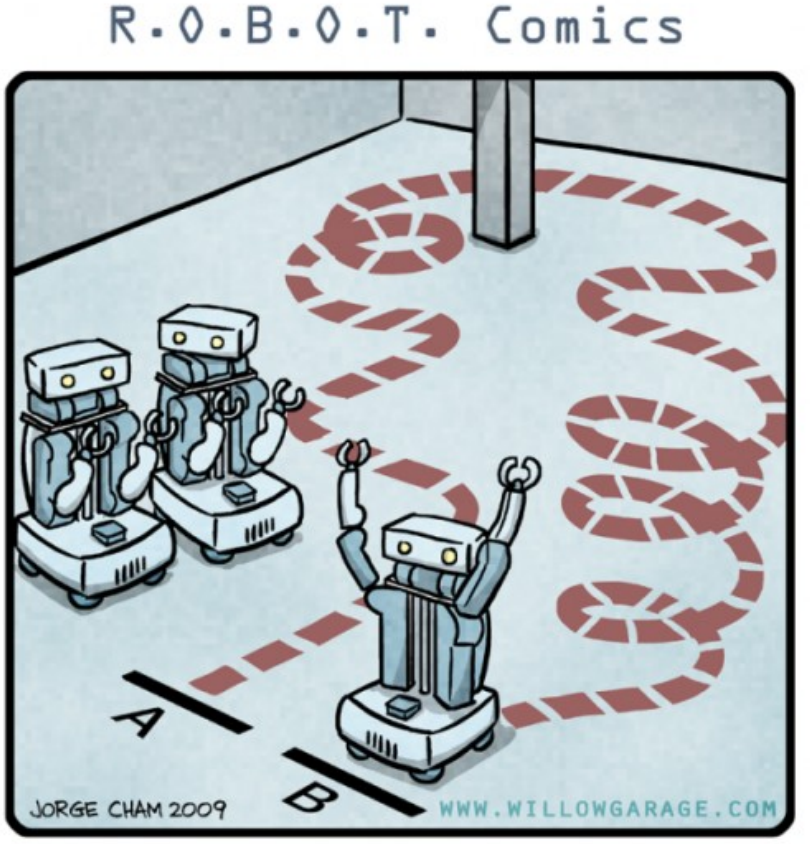
- Many services and relation
  - With many parameters
  - Difficult to represent
- Many deployment solutions
  - Which computer boards ?
  - Which services + parameters ?
  - Large design space
    - Difficulty to explore
    - Especially for non Real Time Embedded Systems experts





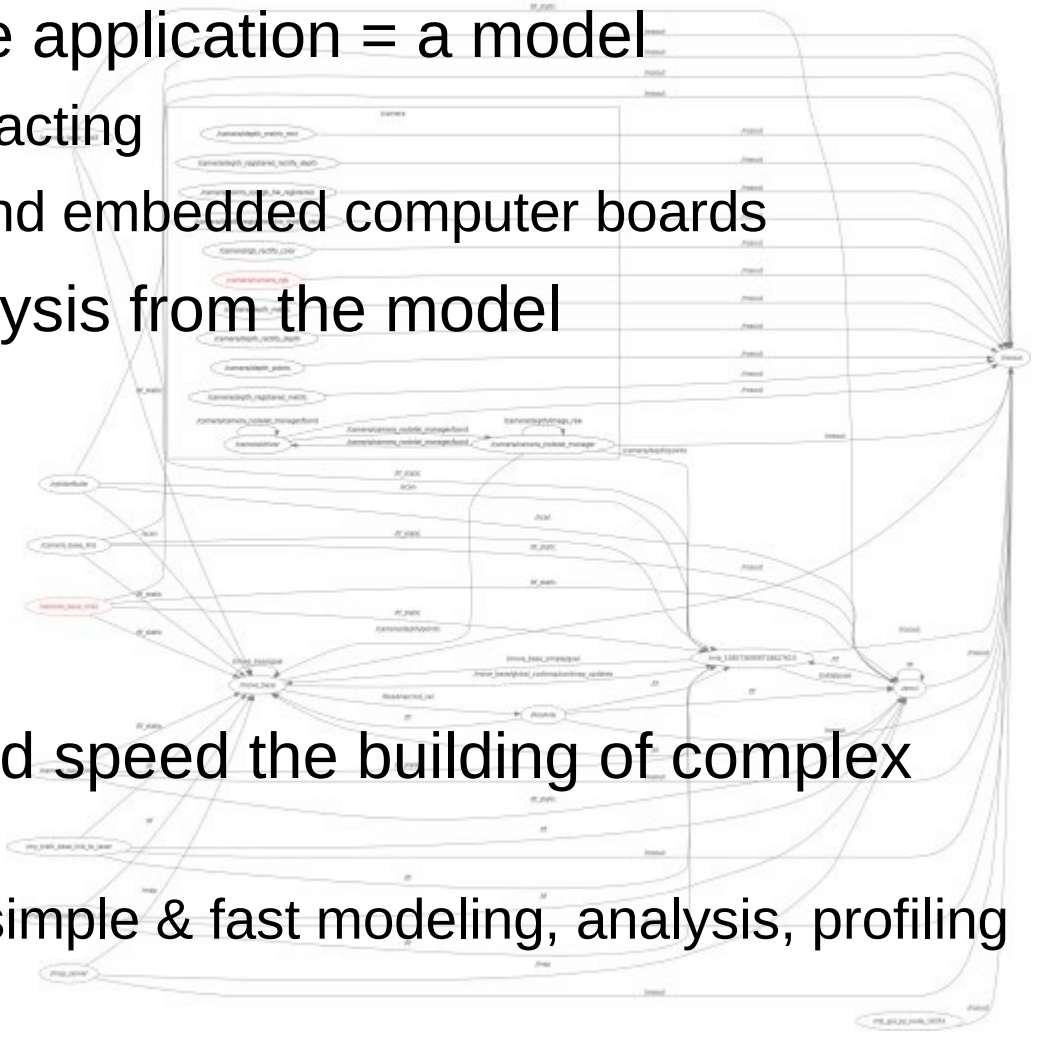
And we face … Performance Issues

- Non functioning or malfunctioning robots
  - The robot is too slow, or loses its way, or its target
  - We observe :
    - High CPU load
    - Slow communications
    - Missed deadlines



Our Needs

- A comprehensive view of the whole application = a model
  - The software: a set of ROS nodes interacting
  - The hardware: the robot, its sensors, and embedded computer boards
- A tool to perform performance analysis from the model
  - Timing : schedulability & latency
  - CPU load analysis
  - BUS load analysis
  - ASAP in the development cycle
- A library of components to ease and speed the building of complex models
  - Something simple, fast and accurate : simple & fast modeling, analysis, profiling



Our Choice

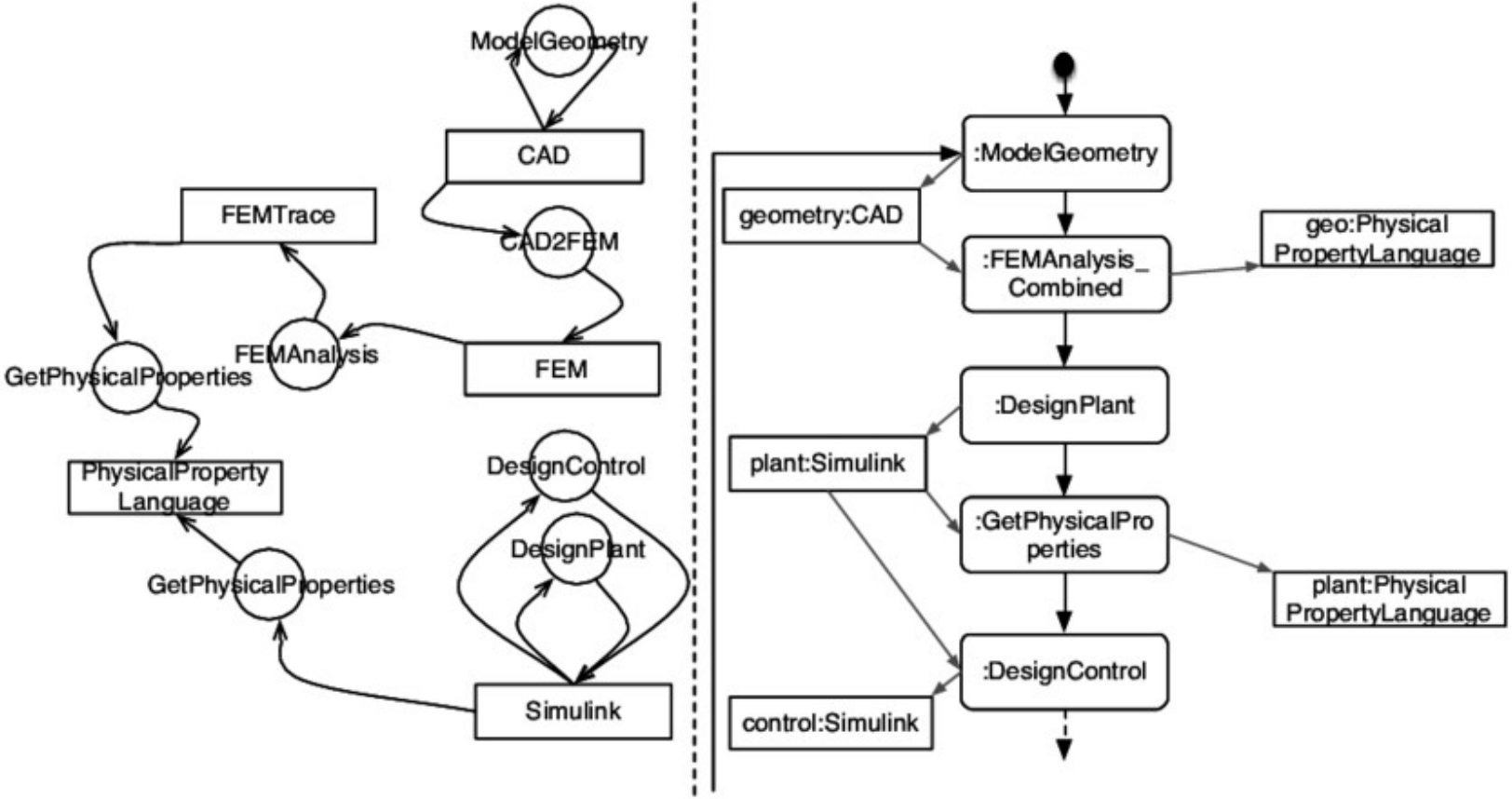
- AADL (Architecture Analysis and Design Language)
  - Covers the domain of Cyber-Physical Systems (CPS, including robotics) with a focus on real-time embedded systems including:
    - Software components (process, thread, data, port …)
    - Hardware components (processor, bus, memory, devices …)
    - Deployment specification with bindings : specify to which HW component(s) a SW component is bound to
  - Embeds in its heart several paradigms, making it a **multi-paradigm** modeling language allowing to cover several parts / aspects of CPSs
    - It is Object-Oriented (OO), which is very helpful in building component libraries
    - Synchronous Data Flow (SDF) through its data port construct
    - Discrete Event Dynamic Systems (DEv) through its event data port construct and its DEVs Annex (DA, among others)

Multi-Paradigm Modeling for CPSs

- Modeling paradigms as generalization of programming paradigms
- Principles:
  - Model every part and aspect of a system explicitly
  - At the most appropriate level(s) of abstraction, with the most appropriate modeling formalism(s) for the activity
  - Do not try build a single modeling language that captures everything
  - *Combine* the most appropriate formalisms and *workflows*

Modeling Modeling and Workflows

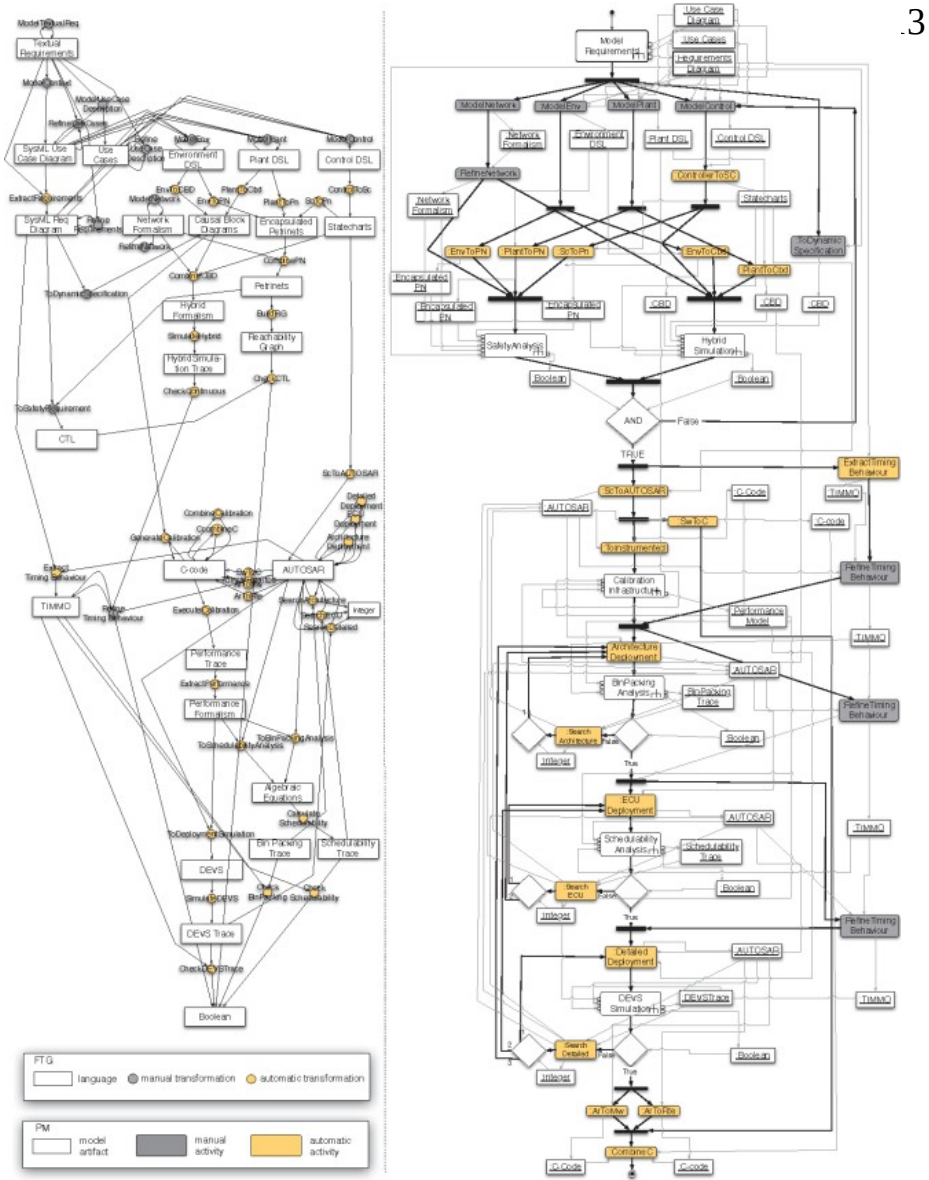
- FTG+PM (Formalisms Transformation Graph + Process Model)



From D. Istvan, J. Denl and H. Vangheluwe, “Towards Inconsistency Management by Process-Oriented Dependency Modeling”

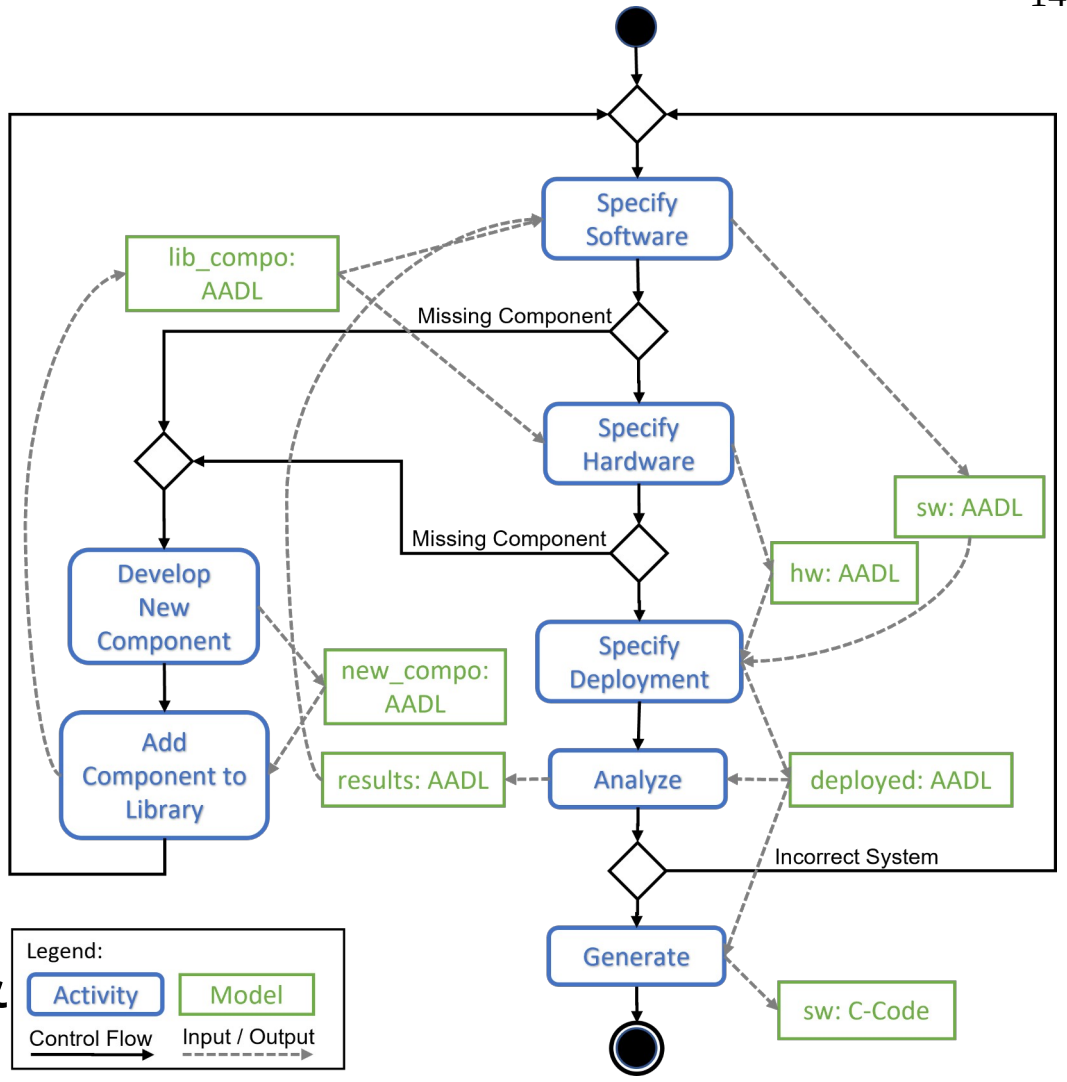
Realistic FTG+PM

- Automotive power window



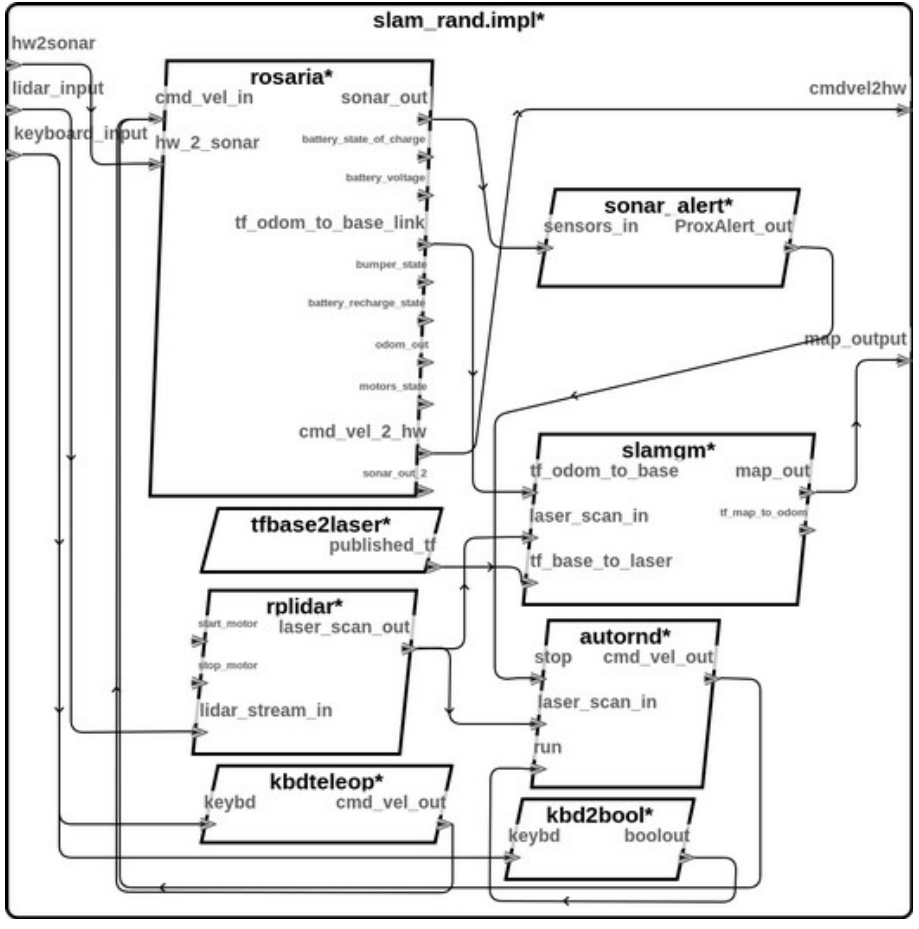
Our Workflow

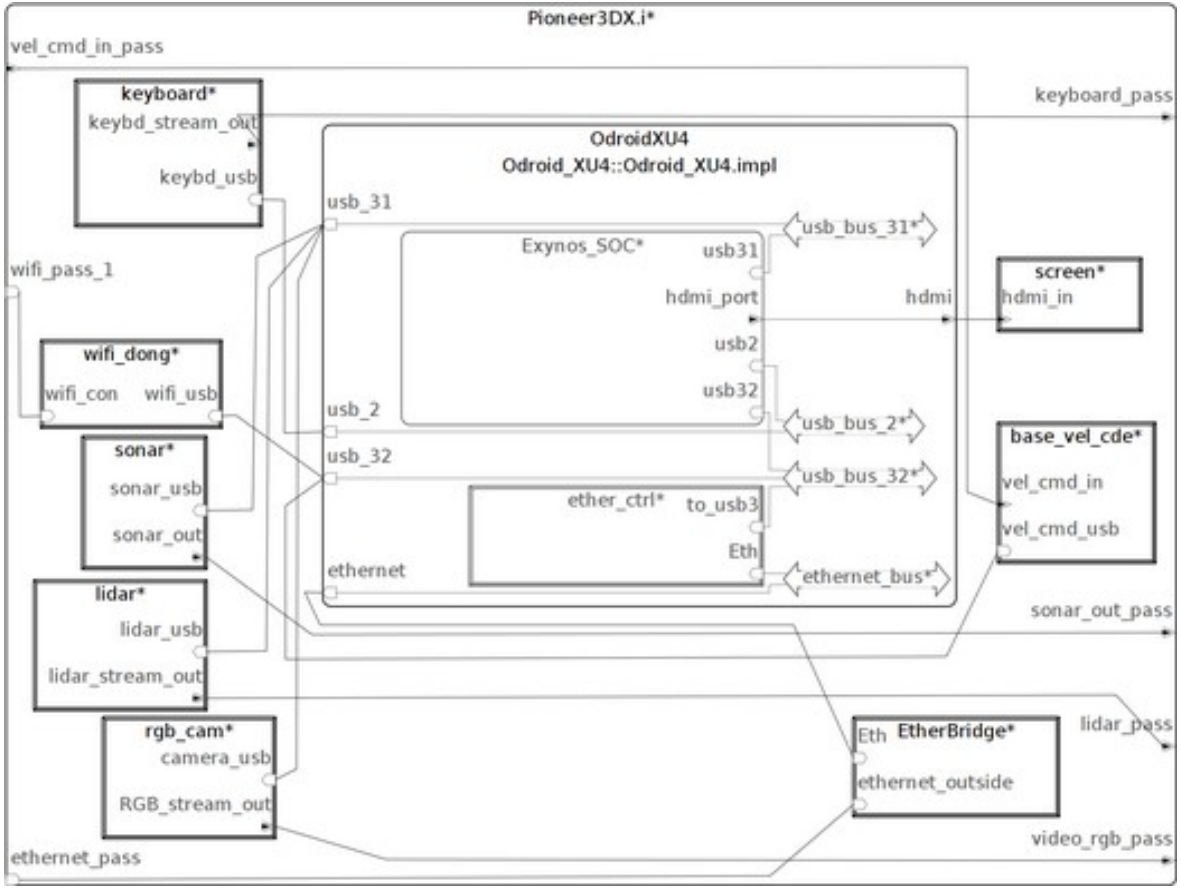
- From a deployed system model
  - Run different analyses depending on the properties in the components models
    - E.g. CPU vs BUS load
  - Iterative exploration process
- If component not in lib.
  - dev. new model ⇒ profiling & benchmarking



SLAM Robot AADL Models

- Graphical concrete syntax





Graphical Textual Syntax

- Blended modeling

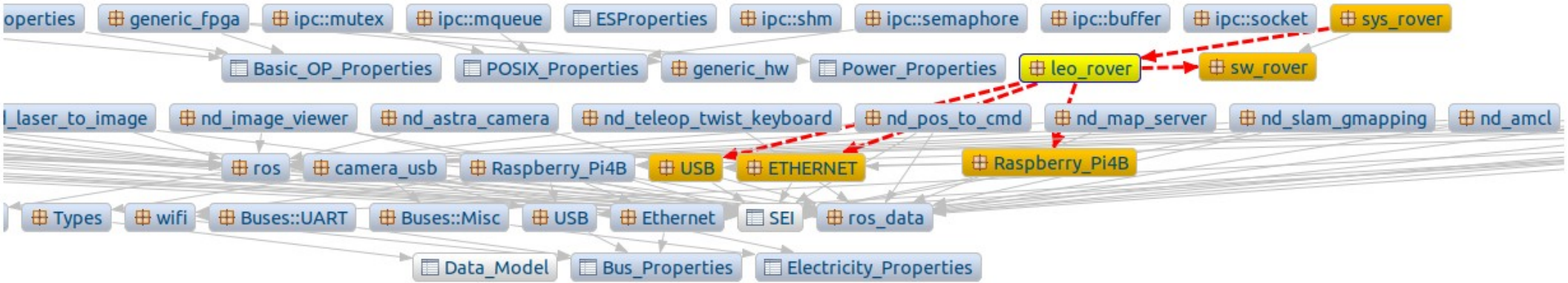
```
system Exynos_5422
    features
        usb2: requires bus access USB::USB.usb2;
        usb31: requires bus access USB::USB.usb3;
        usb32: requires bus access USB::USB.usb3;
        hdmi_port: out event data port;
    end Exynos_5422;

system implementation Exynos_5422.impl
    subcomponents
        big_procs_cluster: system big_procs.i;
        little_procs_cluster: system little_procs.i;
        gpu: processor arm_mali_T628.impl;
        dram_2GB: memory DRAM.impl;
        AMBA_bus: bus AMBA.impl;
        USB2_bridge: device usb2_bridge.impl;
        USB31_bridge: device usb3_bridge.impl;
        USB32_bridge: device usb3_bridge.impl;
        hdmi_dev: device hdmi.impl;
    end Exynos_5422.impl;

system implementation big_procs.i
    subcomponents
        big_proc1: processor Cortex_A15.Processor1;
        big_proc2: processor Cortex_A15.Processor2;
        big_proc3: processor Cortex_A15.Processor3;
        big_proc4: processor Cortex_A15.Processor4;
        big_cache: memory cache.big;
    properties
        SEI::MIPSCapacity => 8000.0 MIPS;
end big_procs.i;
```

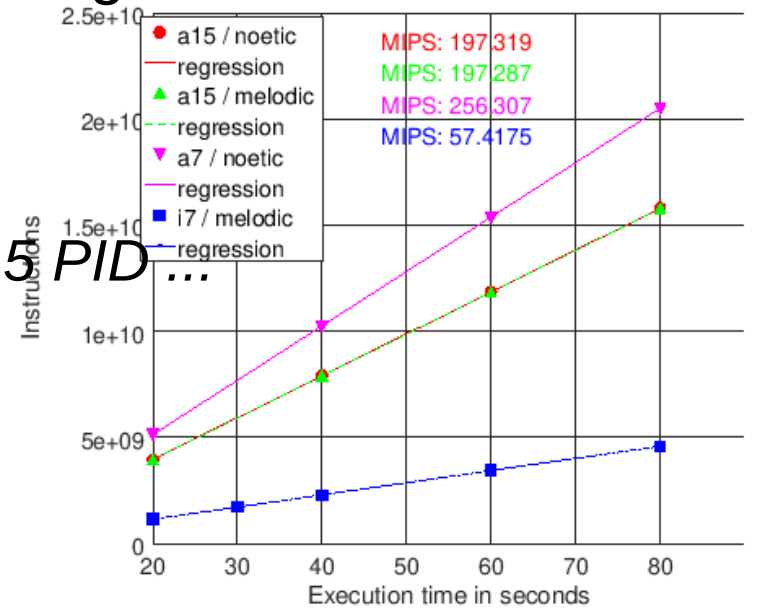
Our AADL Library

- Models for software components organized in packages according to ROS based applications
  - ROS nodes and complex services from mainstream ROS distributions / ROS data types and messages / ROS synchronization and communication mechanisms
- Models for hardware components
  - SBC : Jetson Xavier, Nano, Odroid XU4, Raspberry Pi4, Pi3 … / SoC : Exynos 5422, Broadcom BCM2711 … / SoPC : Xilinx, Altera with hardcores/softcores (PowerPC, µBlaze, NIOS …) / Robots : (Pioneer3DX, LeoRover, TurtleBot …)
  - Example of lib. package tree for a complete robotic app.



Profiling a ROS Node

- Launch the node, in realistic situation and setting
  - CPU frequency : *cpufreq-set* ...
  - CPU affinity : *launch-prefix="taskset -c 5,6,7" …*
  - Process scheduling policy & priority : *chrt -f -p 15 PID …*
- Record performance for different durations
  - *Perf stat -p PID -- sleep duration*
- Using scripts (shell, awk …)



number of instr	number of cycles	freq (GHz)	duration	MIPS	MIPS/frame	ipc	cycles/frame	run_time/fra	proc load	core
3947426403	2751780463	1.992	20.00375	197334288	6577810	1.43	4599867	0.00231	6.92751	A15
7899407914	5736961174	1.992	40.00616	197454813	6581827	1.38	4769440	0.00239	7.18289	A15
5121547760	15567239635	1.4	20	256077388	8535913	0.329	25945399	0.01853	55.59728	A7
10244464108	31191572553	1.4	40	256111603	8537053	0.32844	25992977	0.01857	55.69924	A7

Analysis with OSATE (Main AADL Tool)

- Deployment 1

A15.1: usbcam	A7.1: pos_to_cmd
A15.2: ocv_color_tracking	A7.2: sonar_alert
A15.3:	A7.3: rosaria
A15.4:	A7.3: roscore

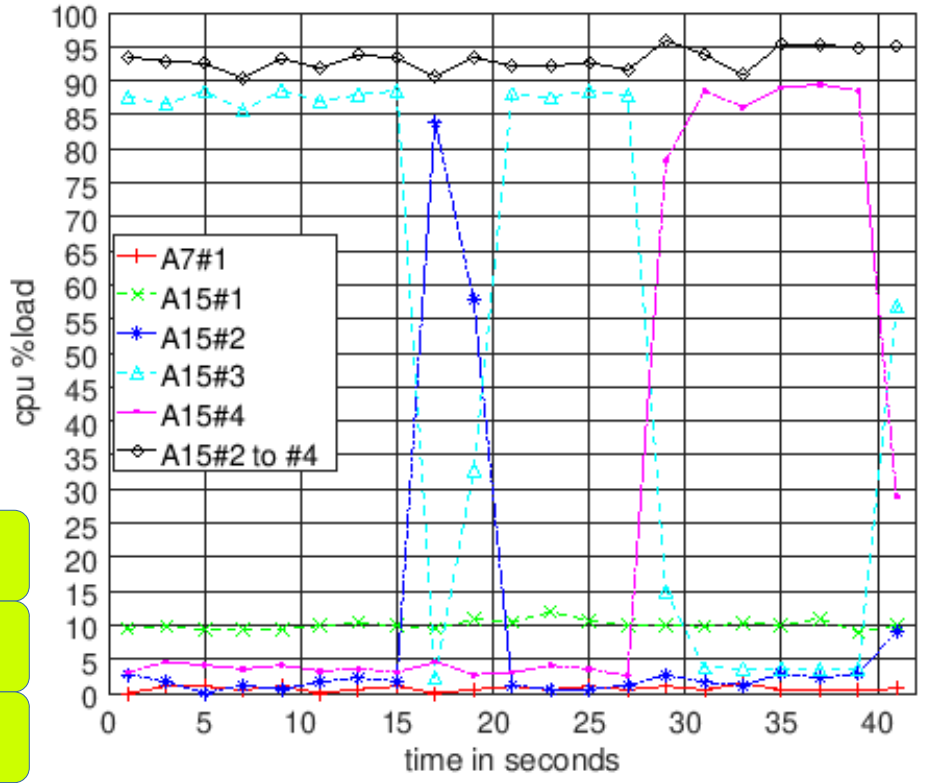
- Results
  - Processor p3dx.(...).big proc1: Total MIPS 140.545 MIPS of bound tasks within MIPS capacity 2.000 GIPS of p3dx.(...).big proc1 : CPU load is 7.03%
  - Processor p3dx.(...).big proc2: Total MIPS 2.242 GIPS of bound tasks exceeds MIPS capacity 2.000 GIPS : CPU load is 112.10%
  - Processor p3dx.(...).little proc1: Total MIPS 7.380 MIPS of bound tasks within MIPS capacity 1.100 GIPS of p3dx.(...).little proc1 : CPU load is 0.67%
- ⇒ Change deployment !
  - ocv_color_tracking now bound to A15.2,3,4

Analysis vs Measurements

- mpstat …

… and a few scripts

CPU	Measured load %	Analysis %	Error %
A7 #1	0.65	0.7	0.05
A15 #1	9.5	7	2.5
A15 #2,3,4	93	112	19

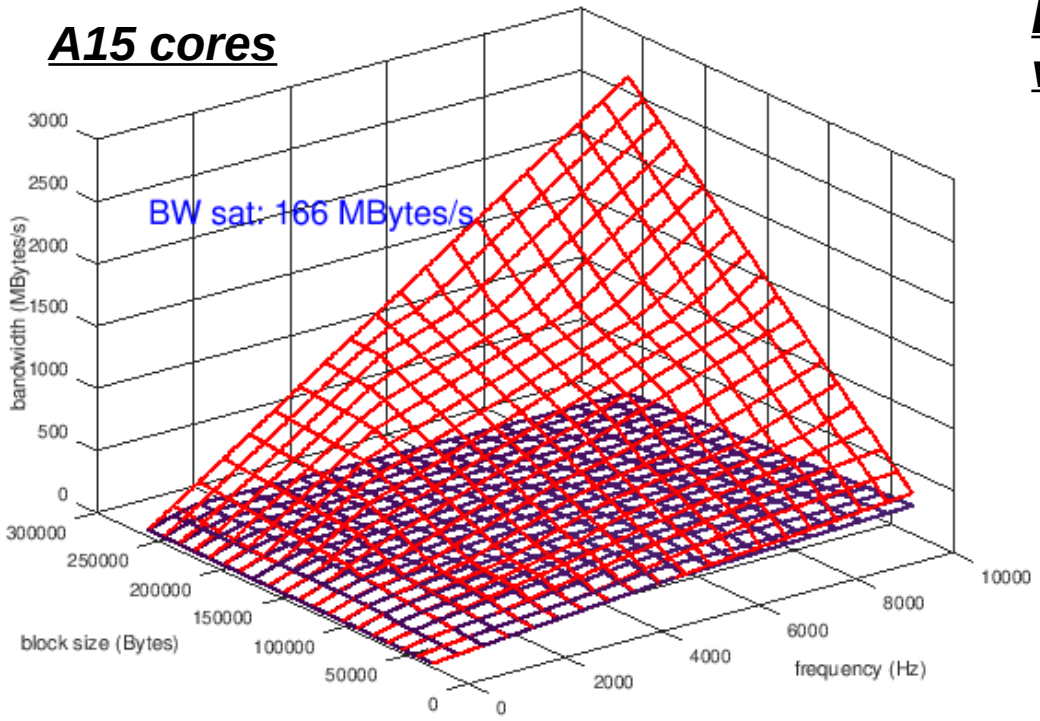


A15.1: usbcam	A7.1: pos_to_cmd
A15.2: ocv_color_tracking	A7.2: sonar_alert
A15.3: ocv_color_tracking	A7.3: rosaria
A15.4: ocv_color_tracking	A7.3: roscore

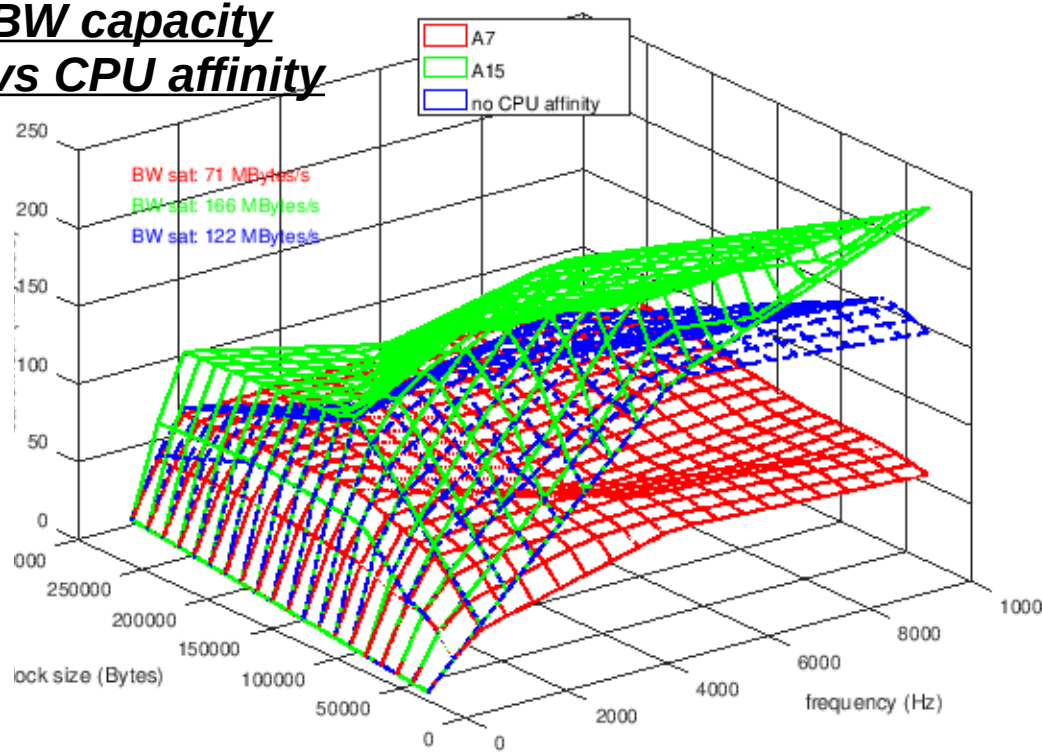
Profiling Bus Capacities

- Experimental setup : producer → listener with growing messages sizes and rates

A15 cores



BW capacity vs CPU affinity



To Conclude

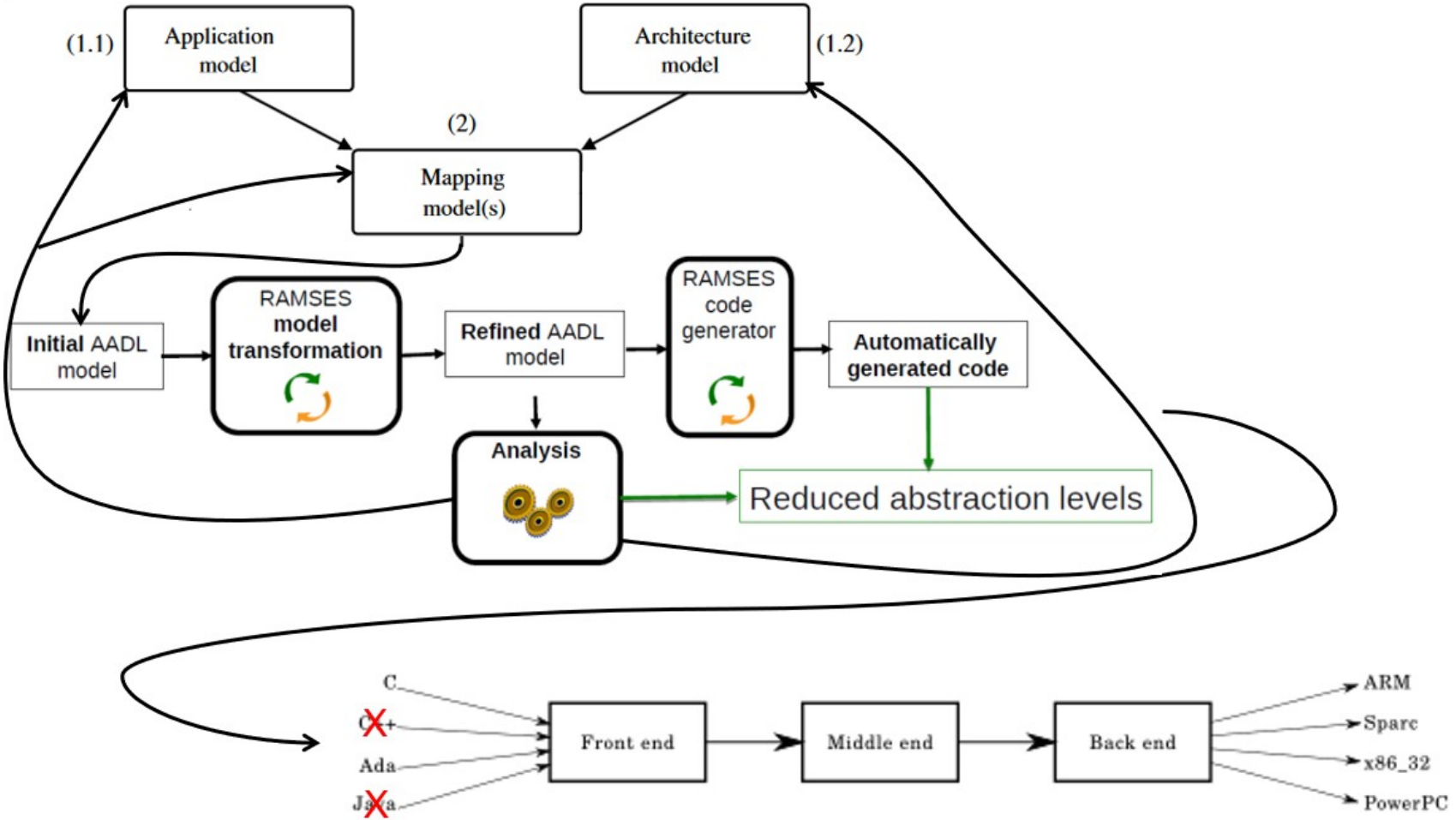
- An AADL library of ROS components with dedicated properties to allow for multiple analysis
  - Using OSATE
    - Resource allocation analysis
      - CPU load / Bus load / Memory capacities / Power consumption / Weights
    - Timing (flow latency) and scheduling analysis

  Very useful for industrial robotics applications

- Workflow
  - Choosing hardware targets & software architectures
  - Exploring binding solutions / balancing between CPU vs Bus load
  - To guarantee reaction time for robotic applications
  - **Generate code**

RAMSES

- Refinement of AADL Models for the Synthesis of Embedded Systems



Future Work

- Automatic code generation for ROS
- Based on AADL library

- Modular architecture of RAMSES
  - Open source core
  - Develop ROS extension (Eclipse plugins)