MODEL-BASED DESIGN AND ANALYSIS OF AUTOMOTIVE APPLICATIONS ON MULTICORE PLATFORMS: AN EFFECTIVE APPROACH

Abdoulaye Gamatié
LIRMM / CNRS-UM, Montpellier

Journée thématique du GDR SoC-SiP (15 mars 2016)
Systèmes Embarqués pour les Transports de Demain

DreamCloud European project (2013 -- 2016)

DreamCloud

DYNAMIC RESOURCE ALLOCATION IN EMBEDDED AND HIGH-PERFORMANCE COMPUTING

http://www.dreamcloud-project.org

Main Overview News Results Partners Photos Contact

SEVENTH FRAMEWORK PROGRAMME

DreamCloud is a Specific Targeted Research Project (STREP) of the Seventh Framework Programme for research and technological development (FP7) - the European Union's chief instrument for funding research projects that commence over the period 2007 to 2013.

9
J'aime 2

FP7 ICT NEWS

THE DREAMCLOUD PROJECT

The main objective addressed by DreamCloud is to enable dynamic resource allocation in many- core embedded and high performance systems while providing appropriate guarantees on performance and energy efficiency.

DreamCloud will address techniques to allocate computation and communication workloads onto computing platforms during run-time whilst respecting performance and energy budget requirements. To achieve this, DreamCloud brings together teams from embedded computing and high-performance computing (HPC) to address these common ambitious yet achievable goals:

- Provide complex embedded systems with cloud-like capabilities such as those available in today's high-performance computing, allowing them to dynamically tune resource usage but without sacrificing critical application-specific constraints in performance and energy.
- Enable HPC and cloud computing systems to balance workload and manage resources so that they can offer more meaningful guarantees in terms of performance and energy, focussing not only on improving average behaviour but also on reducing variability and upper bounds of timing and energy metrics.

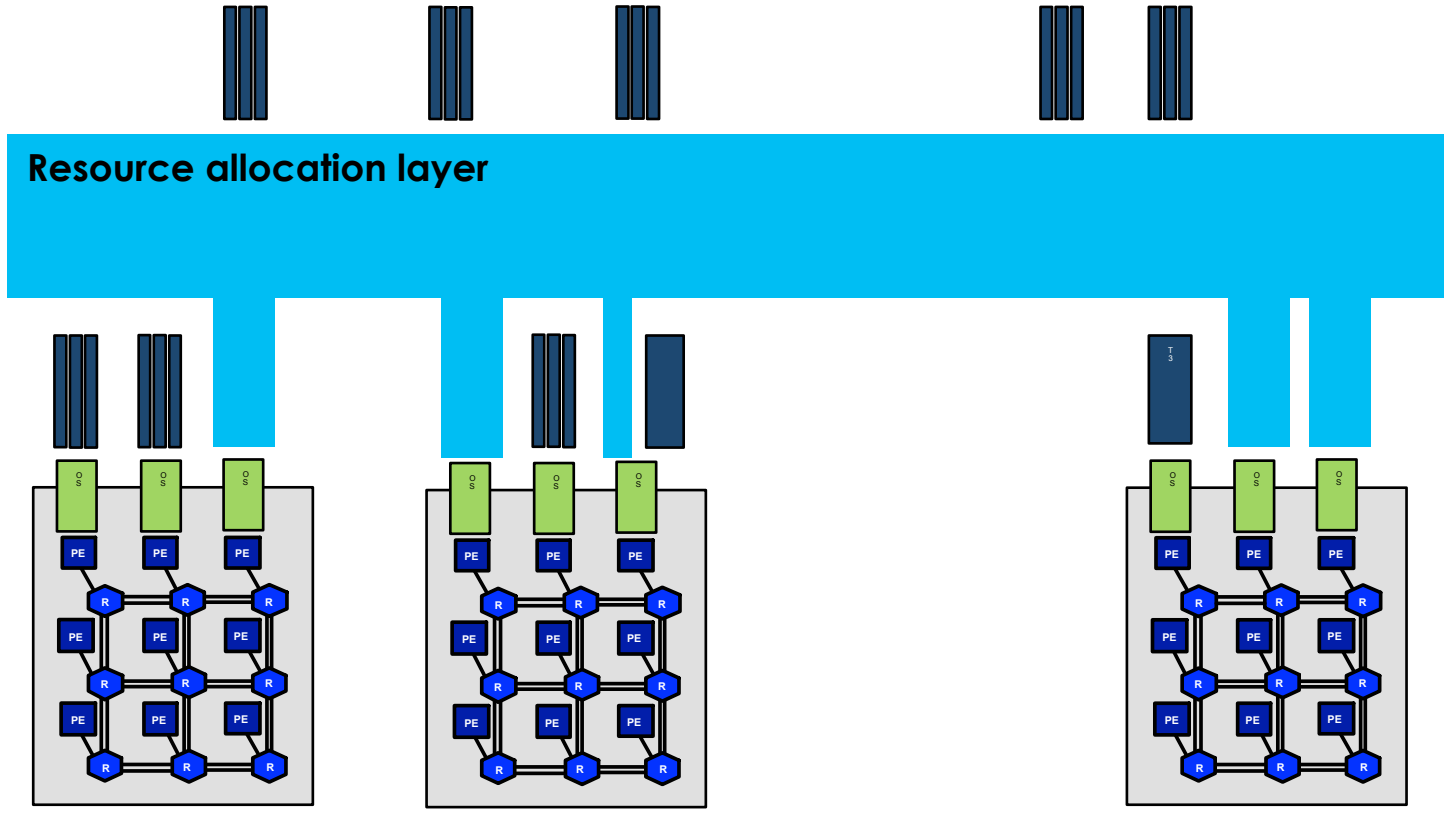
DreamCloud brings together industrial partners from deeply embedded systems (e.g. automotive), consumer embedded systems (e.g. household media), and high performance computing (e.g. HPC platforms); academic partners from embedded systems, real-time systems and HPC. This will enable DreamCloud to develop better approaches by cross-fertilising expertise and experience from multiple industrial domains and academic communities.

Click the video below for an introduction to the project and technologies being developed.

PROJECT PARTNERS

aicas
BOSCH
cnrs
Rheon Media
THE Open GROUP
Universität Stuttgart
THE UNIVERSITY of York

Resource mapping/allocation heuristics



Examples

- Control automation-inspired: feedback loop
- Bio-inspired: bee pheromone signalling
- Market-oriented heuristics

This talk: effective model-based design framework

- **Motivation**: assessment of mapping heuristics on multicore systems
  - Real-time properties
  - Compute and communication performances
  - Energy consumption
- **Proposal**: modular and seamless simulation framework
  - Transaction-level modeling
  - Cycle-accurate
  - **McSim**: Manycore platform Simulation tool-suite

Outline

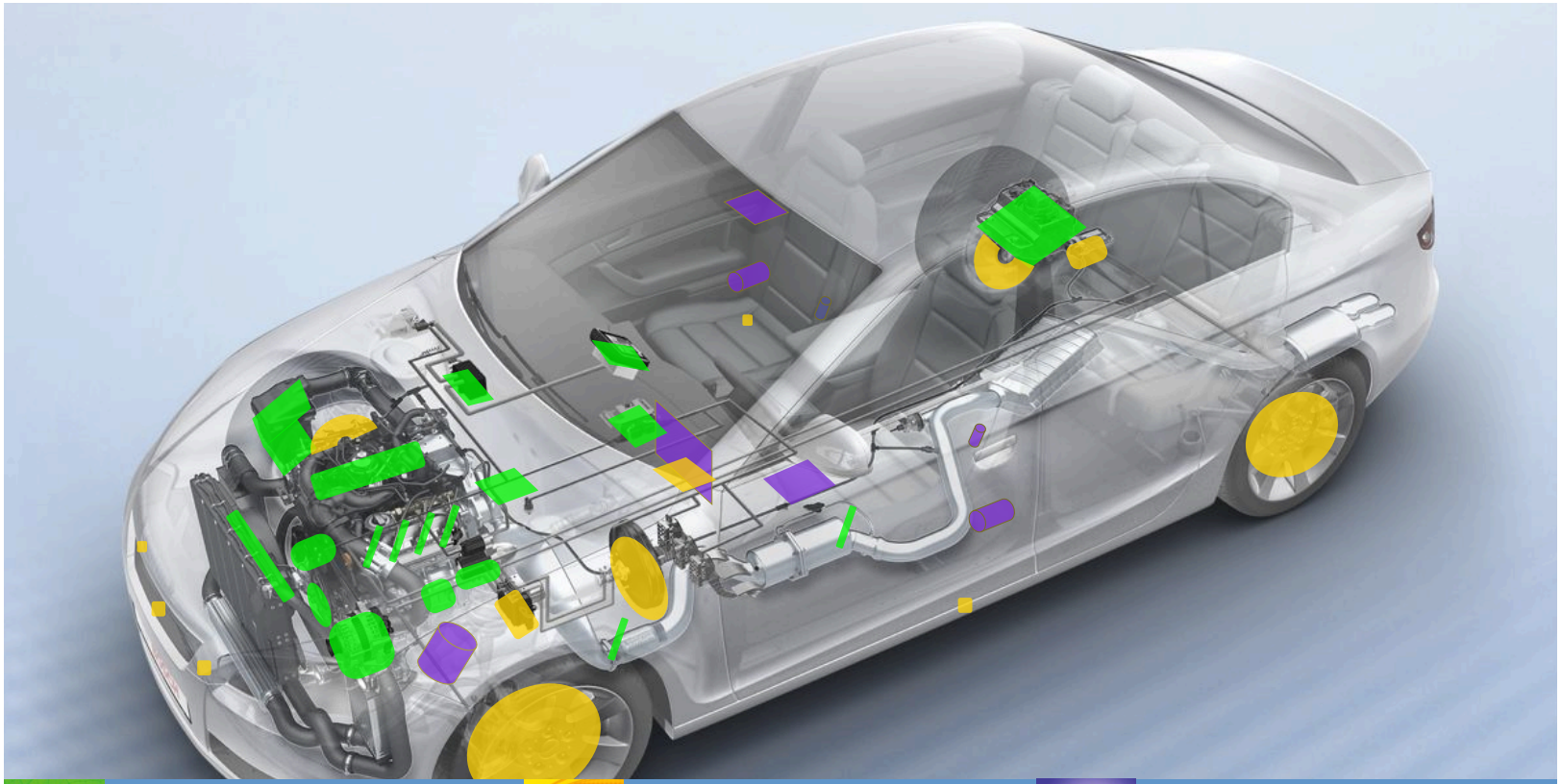
- Model-based design framework
- Sample case-study: automotive application
- Improving mapping decision: analysis & prediction
- Summary

MODEL-BASED DESIGN

Multi abstraction level simulation framework

Automotive technology

- Multicore µCs (<6) already present in automotive embedded systems
- Higher performance only via Manycore µCs (>6) architectures

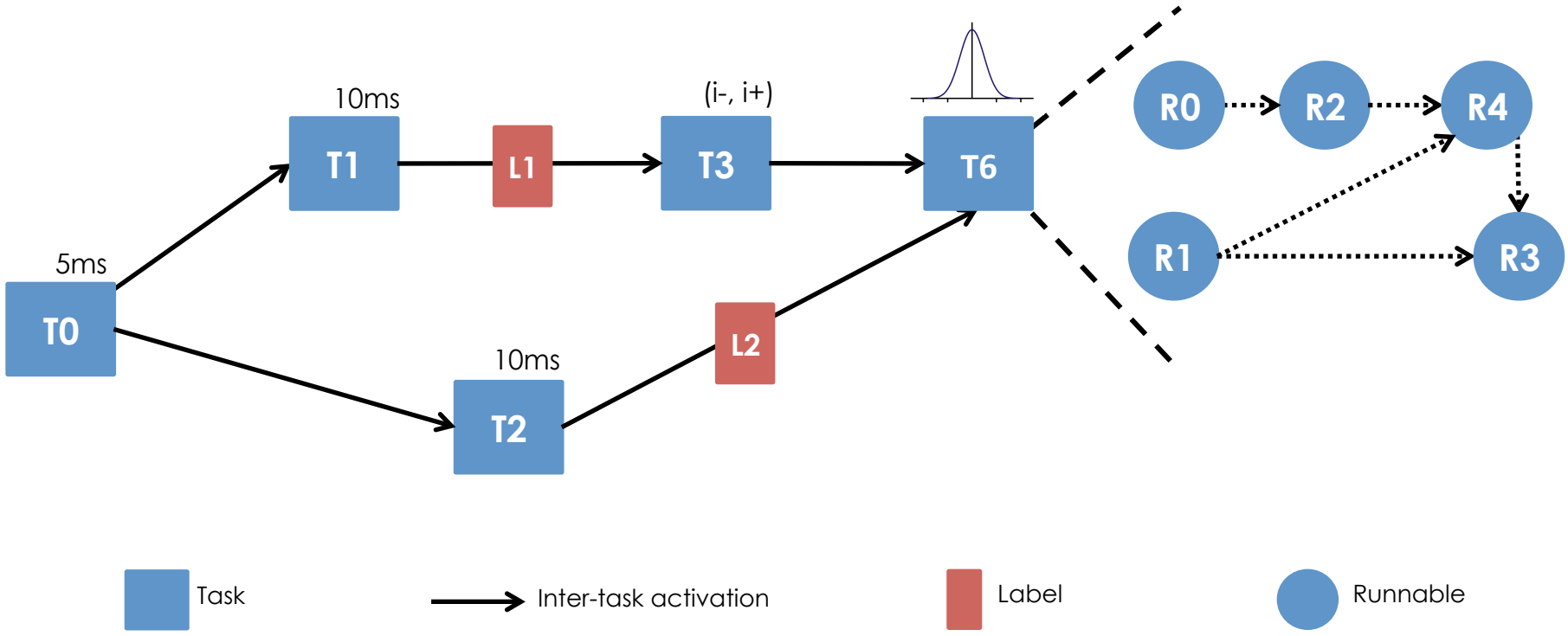


Source: http://www.mpsoc-forum.org/previous/2014/slides/jochen_haerdtlein.pdf

Model-based design for automotive applications

- **UML-Marte** (Modeling and Analysis of Real-Time and Embedded Systems) - http://www.omgmarte.org
  - general modeling concepts
- **AUTOSAR** (AUTomotive Open System ARchitecture) - http://www.autosar.org
  - single uniform address memory
- **Amalthea** - http://amalthea-project.org
  - AUTOSAR-compatible
  - rich support for multicore systems
  - support for product-line engineering (variability of modern motor vehicles)

Application modeling in Amalthea

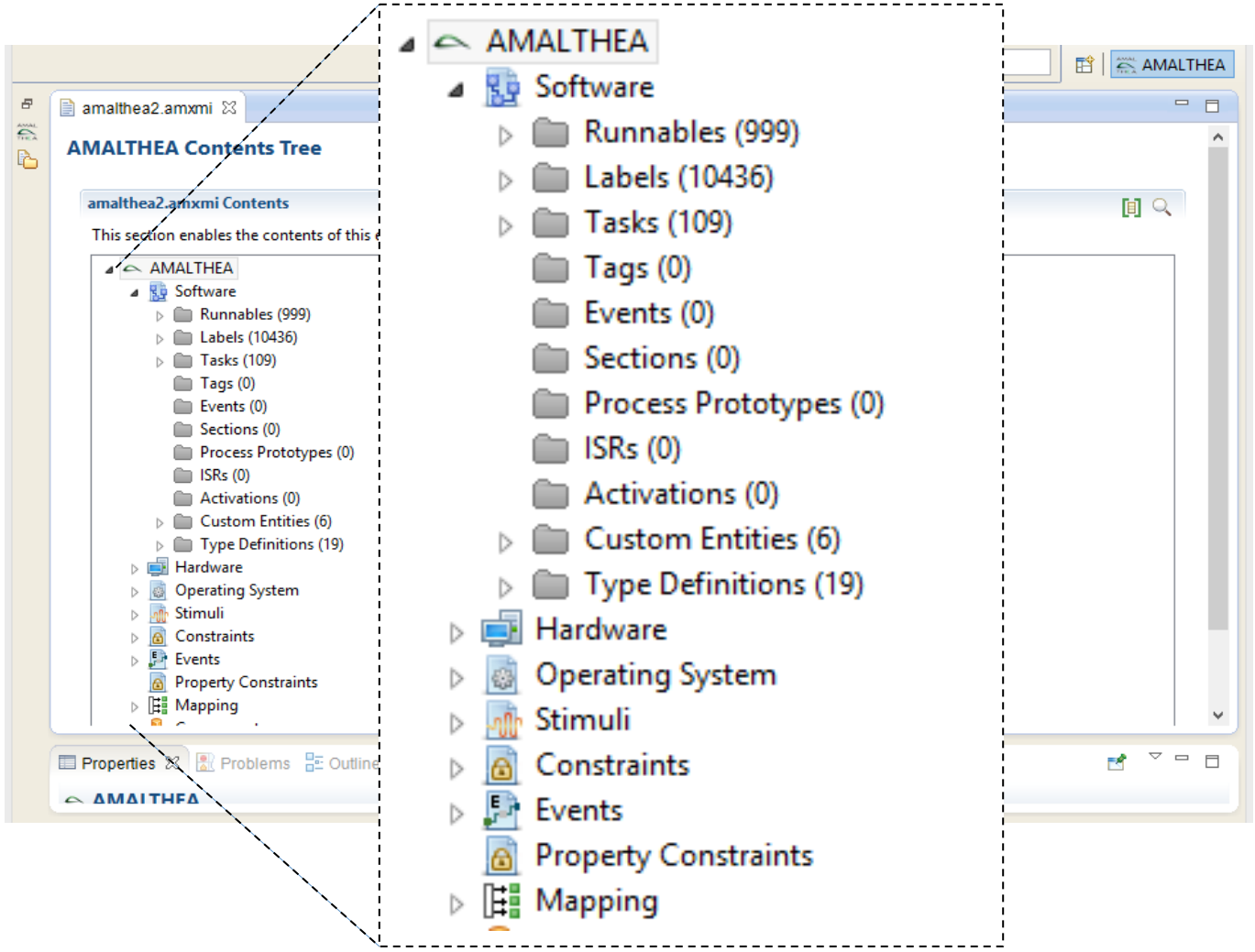


Instructions in a runnable

- computation: constant delay, delay distribution law
- communication: read (data size) or write (data size)

Application modeling in Amalthea (cont'd)

An Eclipse plugin as a user-friendly environment



Application modeling in Amalthea (cont'd)

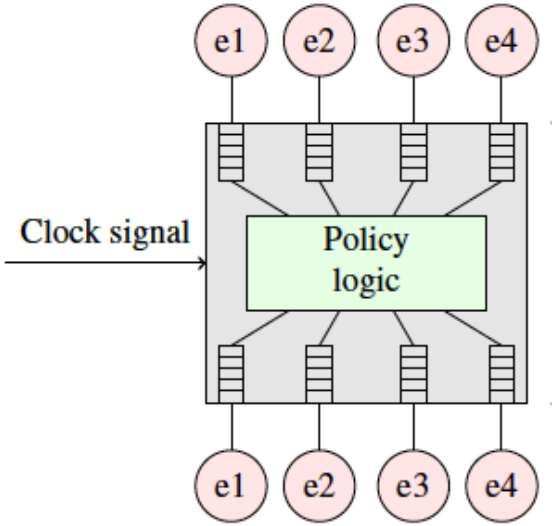
XML intermediate representation

amaltheademo.amxmi

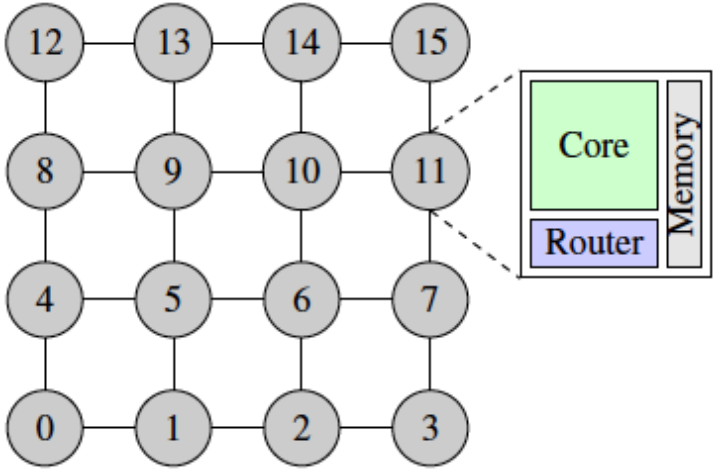
```
<tasks xmi:id="_Uf6MEO5IEeSv-euQbOE0dw" name="Task_T2" priority="8" stimuli="
  <size xmi:id="_cFdoMO5IEeSv-euQbOE0dw" numberBits="800"/>
  <deadline xmi:id="_Zvbk8O5IEeSv-euQbOE0dw" value="400"/>
      <callGraph xmi:id="_A3qksO5JEeSv-euQbOE0dw">
        <graphEntries xsi:type="sw:CallSequence" xmi:id="_CO9HwO5JEeSv-euQbOE0dw">
          <calls xsi:type="sw:TaskRunnableCall" xmi:id="_DgqnQO5JEeSv-euQbOE0dw" runnable="_rP5
        </graphEntries>
      </callGraph>
    </tasks>
<runnables xmi:id="_iPHuEO5IEeSv-euQbOE0dw" name="Runnable_R1">
  <size xmi:id="_oPG6kO5IEeSv-euQbOE0dw" numberBits="50"/>
  <deadline xmi:id="_mUV3AO5IEeSv-euQbOE0dw" value="90"/>
    <labels xmi:id="_VKQ7MO5JEeSv-euQbOE0dw" name="Label_L1">
      <size xmi:id="_YKtNYO5JEeSv-euQbOE0dw" numberBits="600"/>
    </labels>
  </swModel>
  <stimuliModel xmi:id="_G1tbAO5KEeSv-euQbOE0dw">
    <stimuli xsi:type="stimuli:InterProcess" xmi:id="_qqdQwO5KEeSv-euQbOE0dw" name="Stimuli_Tas
  </stimuliModel>
</central:AMALTHEA>
```

Multicore architecture modeling

- **Core**: Amalthea MoC-specific model
- Inter-core **communication**: bus, crossbar, Network-on-Chip (NoC)



Crossbar-based architecture

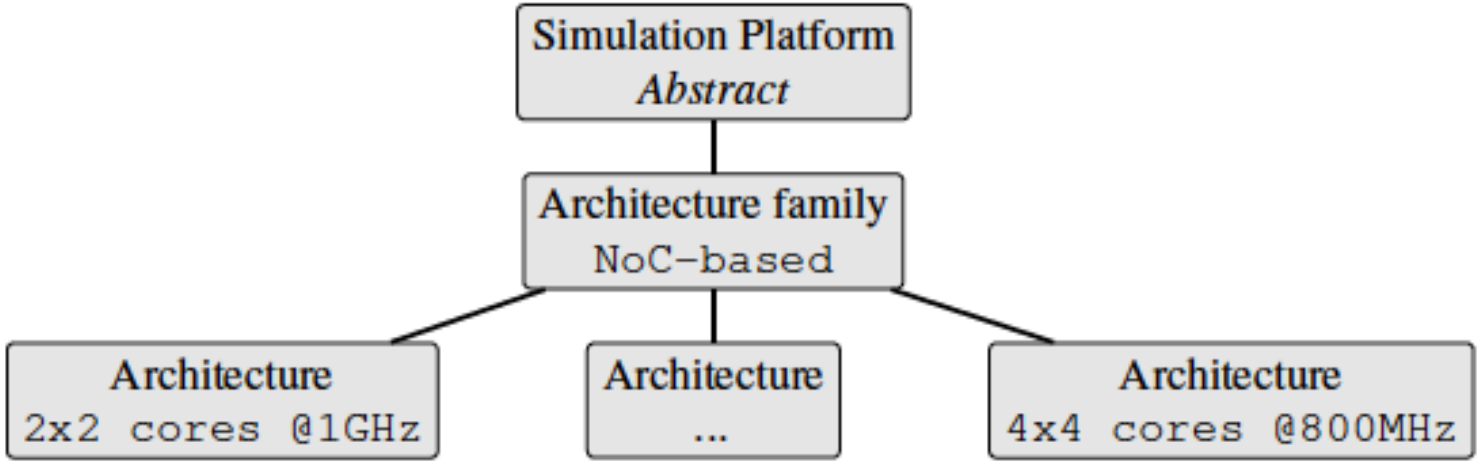


NoC-based architecture

Multicore architecture simulators: a quick survey

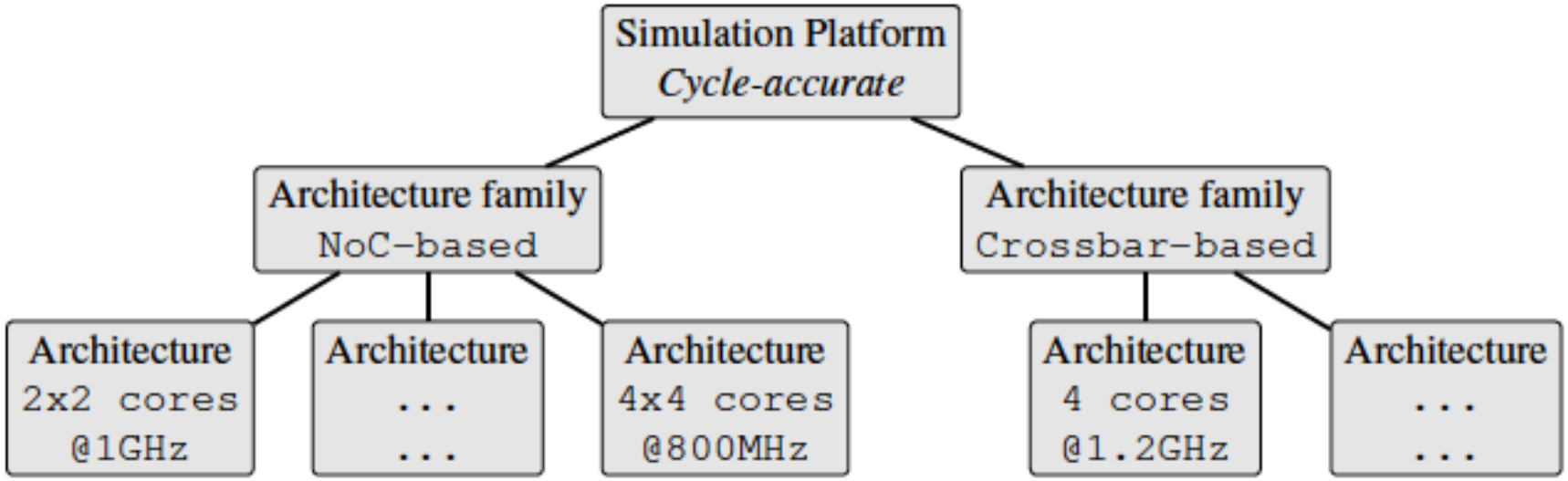
Simulator	Language	Comm. Infrastr.	Scalability	Accuracy	Real-Time comp. & comm.
gem5	Python/ C++	Various	No	Cycle	No
OVPsim	C++	Bus	No	Functional	No
MC-Sim	C	NoC	Yes	Cycle	No
PREESM	C	Bus	No	Instruction	No
Simics/GEMS	C++	Various	No	Functional	No
Flexus	C++	Various	Yes	Cycle	No
McSim	**SystemC**	**NoC, Crossbar**	**Yes**	**Transactional, cycle**	**Yes**

McSim TLM simulation



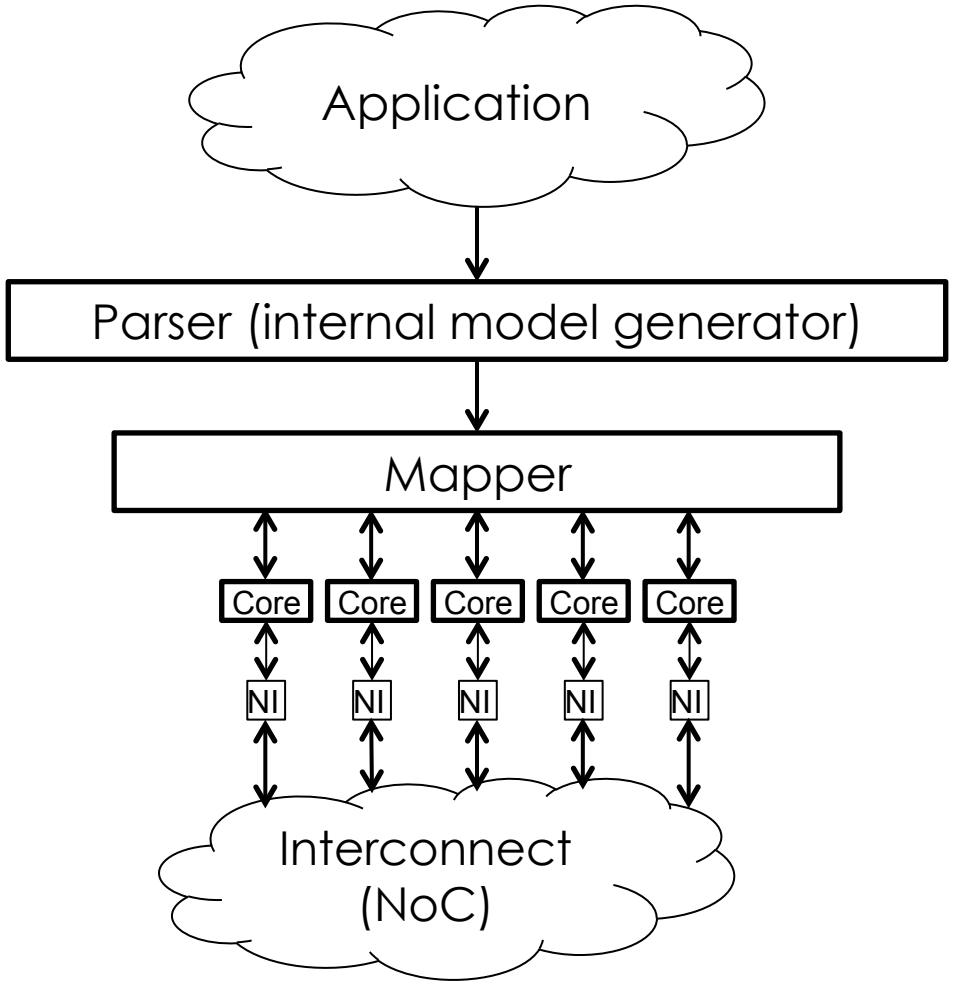
** L. Indrusiak and O. dos Santos « **Fast and accurate transaction-level model of a wormhole network-on-chip with priority preemptive virtual channel arbitration** » In Design, Automation Test in Europe Conference Exhibition (DATE), 2011.

McSim cycle-accurate simulation



** Tran, A. T. and Baas, B. « **Noctweak: a highly parameterizable simulator for early exploration of performance and energy of networks on-chip**. » Tech. Report ECE-VCL-2012-2, VLSI Comp Lab, ECE Dep. University of California, Davis, 2012.

McSim simulation workflow: example



** K. Latif, M. Selva, C. Effiong, R. Ursu, A. Gamatié, G. Sassatelli, L. Zordan, L. Ost, P. Dziurzanski and L. S. Indrusiak. **'Design Space Exploration for Complex Automotive Applications: An Engine Control System Case Study'**, RAPIDO'16 Workshop.

Typical output results - DemoCar app

OUTPUT_Execution_Report.log

```
**************************************************************
***** Summary of Platform (Core + NoC) Simulation Results*****
**************************************************************


 ##Configuration Information##

    Input Application(Amalthea Model): DEMO_CAR.amxmi
    Number  of   Core   in  System   : 16
    Number  of  Rows  in  Platform   : 4
    Number  of  Columns in  Platform : 4
    System  Operating  Frequency     : 1GHz
    Mapping Technique : HEURISTIC_LOCAL_MAXIMIZED


 ##Application Information Delivered by Parser##

Task Graph dcTaskGraph created
Time of Parse2: 0.002876 seconds.

    Number of Tasks: 3
    Number of Runnables: 43
    Number of Labels: 71


 ##Simulation Results##

    Completed Application Iterations: 1
    Execution Time: 910493 ns
    Number of Runnables, which Missed Deadlines: 0
    Packets Exchanged through NoC for Remote Label Accesses: 139
    Simulation Time: 1.53726 seconds.


Info: /OSCI/SystemC: Simulation stopped by user.
```

Input app. and platform information

Application parsing output

Simulated timing information

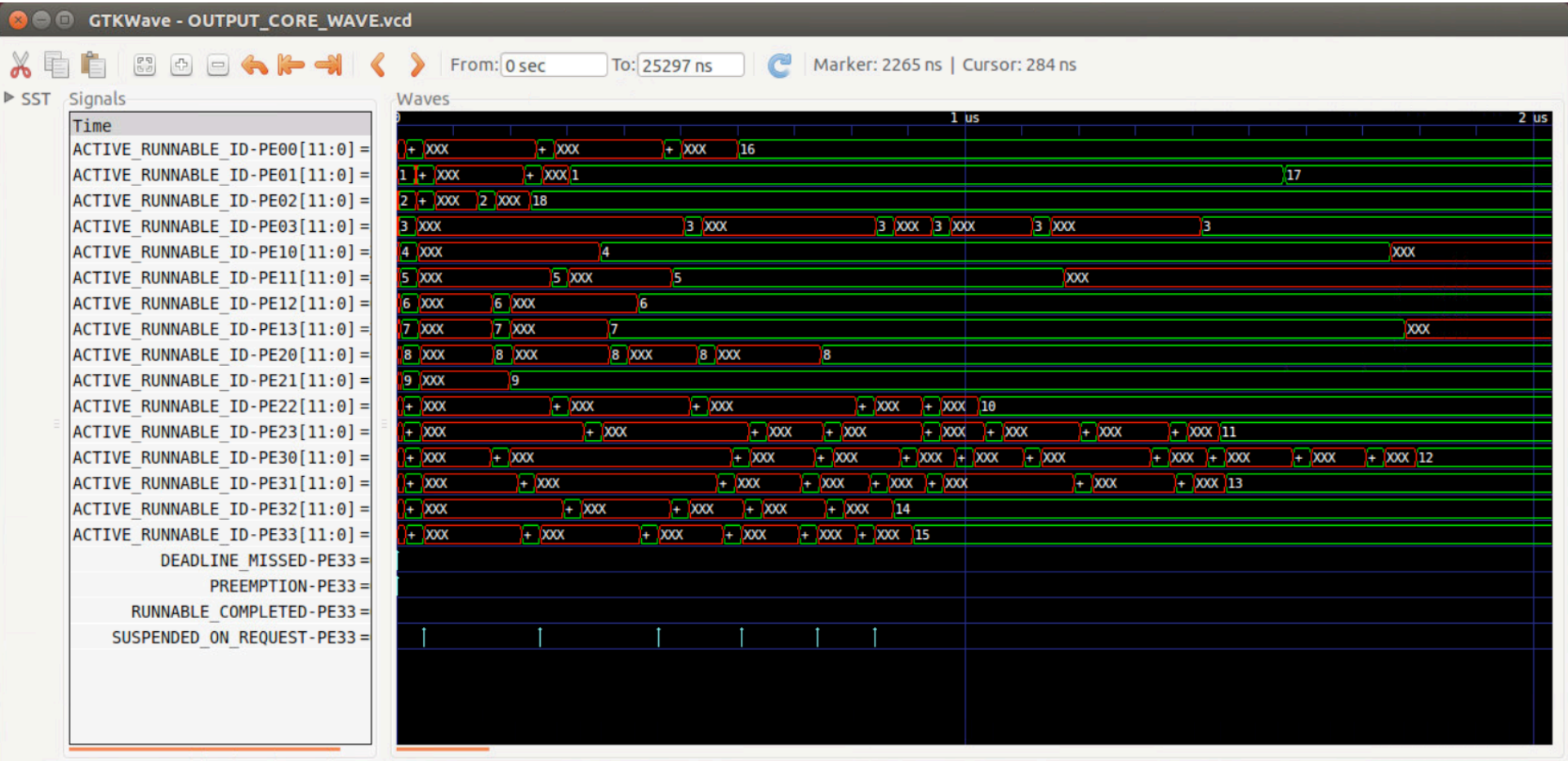
Typical output results - DemoCar app (cont'd)

*OUTPUT_Energy.log

```
************************************************************
*                                                          *
* Estimation of the total energy consumption of an         *
* application running on a multi-processor compute platform. *
* The estimated energy is mapping dependednt.              *
* The energy consumption computation, takes into account the *
* energy consumption of cores (instruction execution,      *
* label accesses (local and remote), context switch) and   *
* the energy consumption of NoC.                           *
*                                                          *
************************************************************

 #Input Configuration Information#

    Type of the core : ARM_A15
    System  Operating  Frequency : 1.00 (GHz)
    Total execution time : 910493 (ns)
    Number  of  Rows  in  Platform : 4
    Number  of  Columns in  Platform : 4
    Number  of  Cores   in  System : 16
    Packets Exchanged through NoC for Remote Label Accesses : 139


 #Summary of Results#

    Energy of NoC (dynamic and static) : 73.705154419 (µJ)
    Energy of cores (dynamic and static due to instruction execution,
    local label accesses and idle state) : 12865.7998 (µJ)
    Total Energy of the system is : 12939.5049 (µJ)
    Proportion of total energy consumption by cores : 99.43%
    Proportion of total energy consumption by NoC : 0.57%


 #Detailed Results#

    Average packet latency : 56.4173 (ns)
    Total execution time of computational instructions : 3635630 (cycles)
    Data size for label accesses (local and remote) : 219 (number of packets)
    Data size for local label accesses : 80 (number of packets)
    Data size for remote label accesses : 139 (number of packets)

    Power of an idle cpu : 0.3860 (W) static power obtained by McPAT
    Power of a busy cpu : 1.3510 (W) computed as 0.700 times the energy of idle core at 200 MHz

    #Breakdown of Energy Consumption by the platform#
    Static energy platform : 8025.911133 (µJ)
    Dynamic energy platform : 4913593.000000 (nJ)

    #Breakdown of Energy Consumption by Cores#
    Static energy cores : 7952.206055 (µJ)
    Dynamic energy cores : 4913593.000000 (nJ)
    Total energy consumed by labels at core levels : 2330.7549 (µJ)
```

Simulated energy estimation

Typical output results - DemoCar app (cont'd)



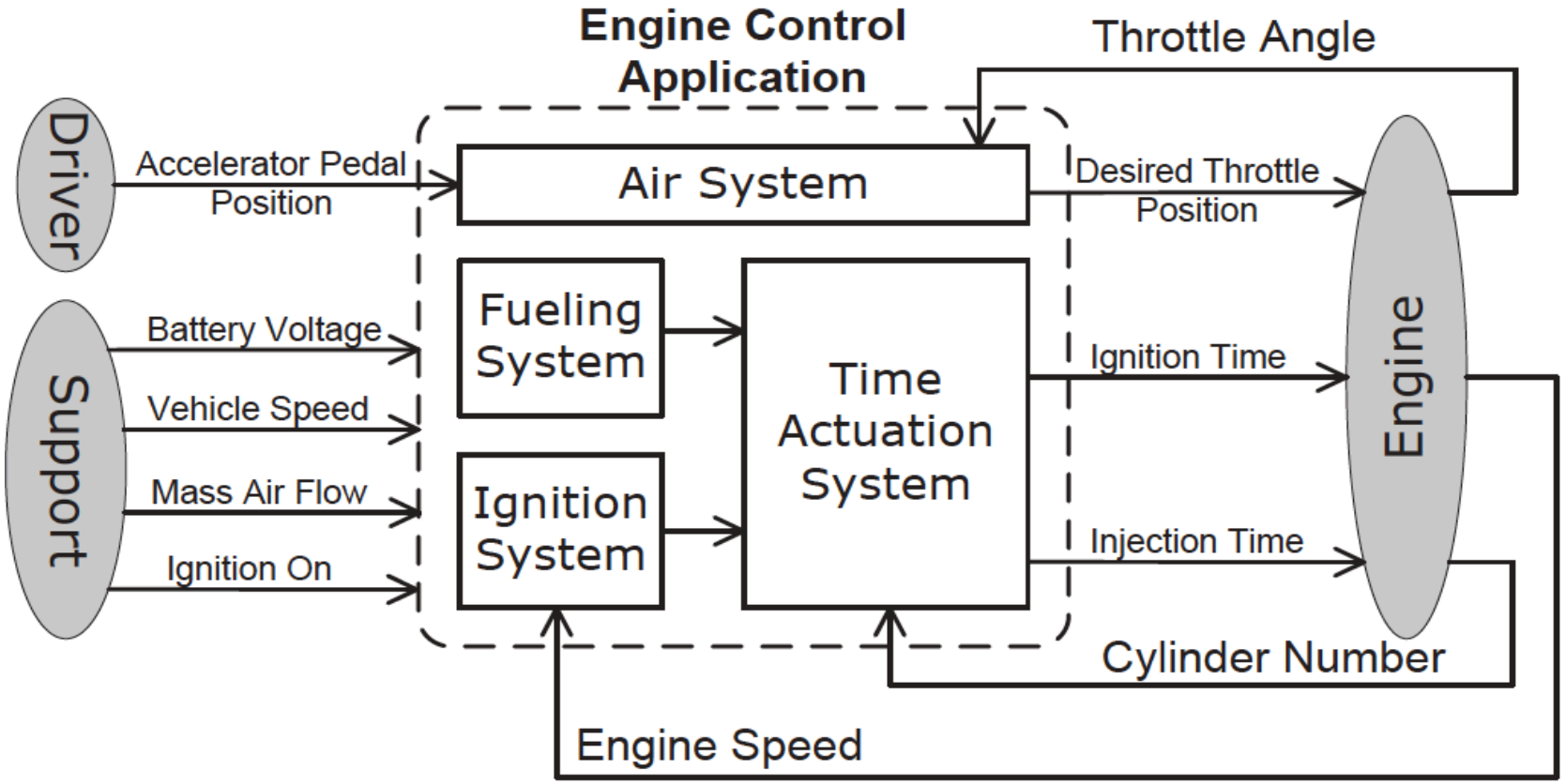
Outline

- Model-based design: modular framework
- Case-study: an automotive application
- Improving mapping decision: analysis & prediction
- Summary

CASE STUDY

An automotive application

A case study: engine control system



- **Amalthea model**: 109 tasks, 1239 runnables, 10436 labels

A case study: engine control system (cont'd)

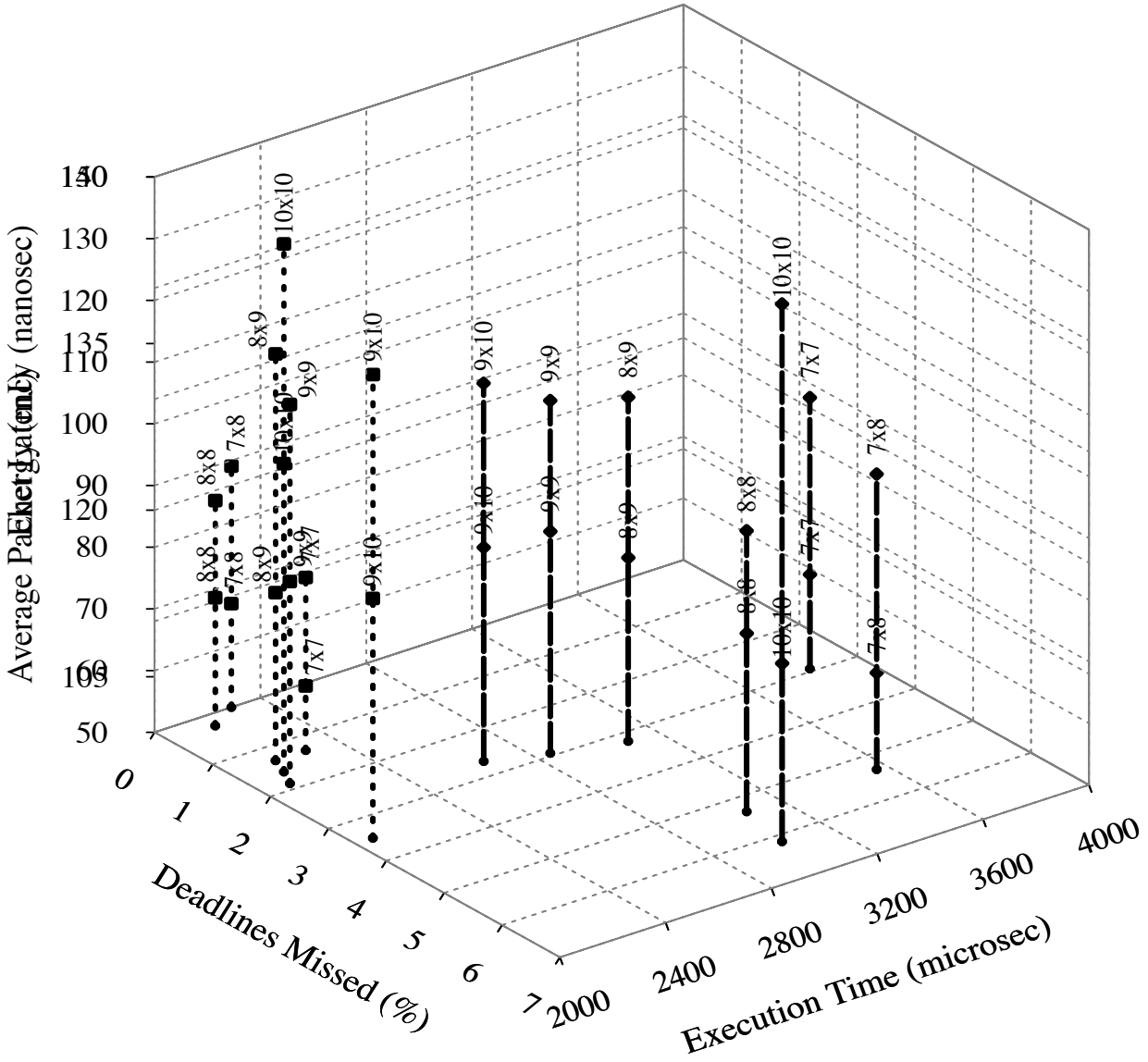
Setup

- McSim TLM NoC simulator
- Two simulated mappings: local-maximized (Loc.) & ZigZag (ZZ)
- Desktop machine: Intel 4-core i5-4670 (3.40GHz)

NoC Size	Simulation Time (sec.)		Execution Time (μs)		Deadline misses (%)		#Packets in NoC		APL (ns)	
	Loc.	ZZ	Loc.	ZZ	Loc.	ZZ	Loc.	ZZ	Loc.	ZZ
7×7	30	24	3833	2282	2.9	1.3	38724	39275	108	115
7×8	33	26	3560	2290	5.3	**Ref**	38711	40507	108	121
8×8	32	27	3069	2172	5.3	0.2	38797	40619	116	120
8×9	35	32	3090	2166	3.2	1.3	38964	40620	116	136
9×9	36	33	2852	2102	2.9	1.8	39009	40648	119	134
9×10	39	37	2659	2065	2.6	3.4	39025	40766	119	141
10×10	48	44	3026	2138	6.1	1.6	39073	40817	116	147

*APL = Average Packet Latency

A case study: engine control system (cont'd)



(▪---▪ : Zigzag, ◆——◆ : local-maximized)

Outline

- Model-based design: modular framework
- Case-study: an automotive application
- Improving mapping decision: analysis & prediction
- Summary

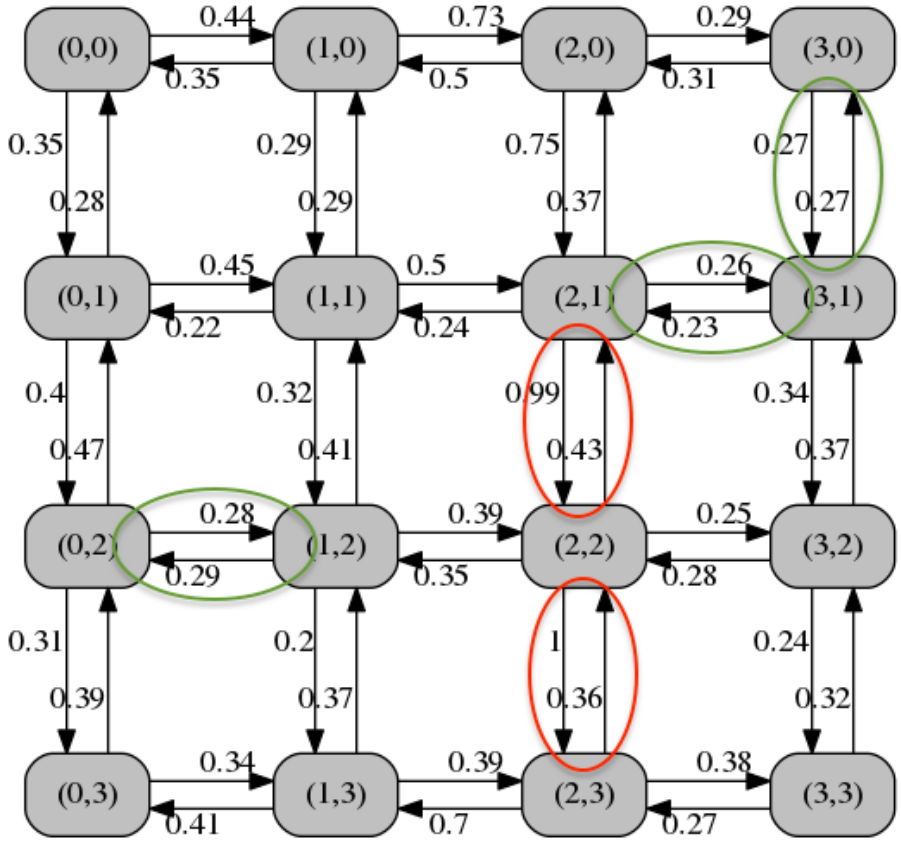
IMPROVING DYNAMIC MAPPING DECISIONS

Analysis & Prediction

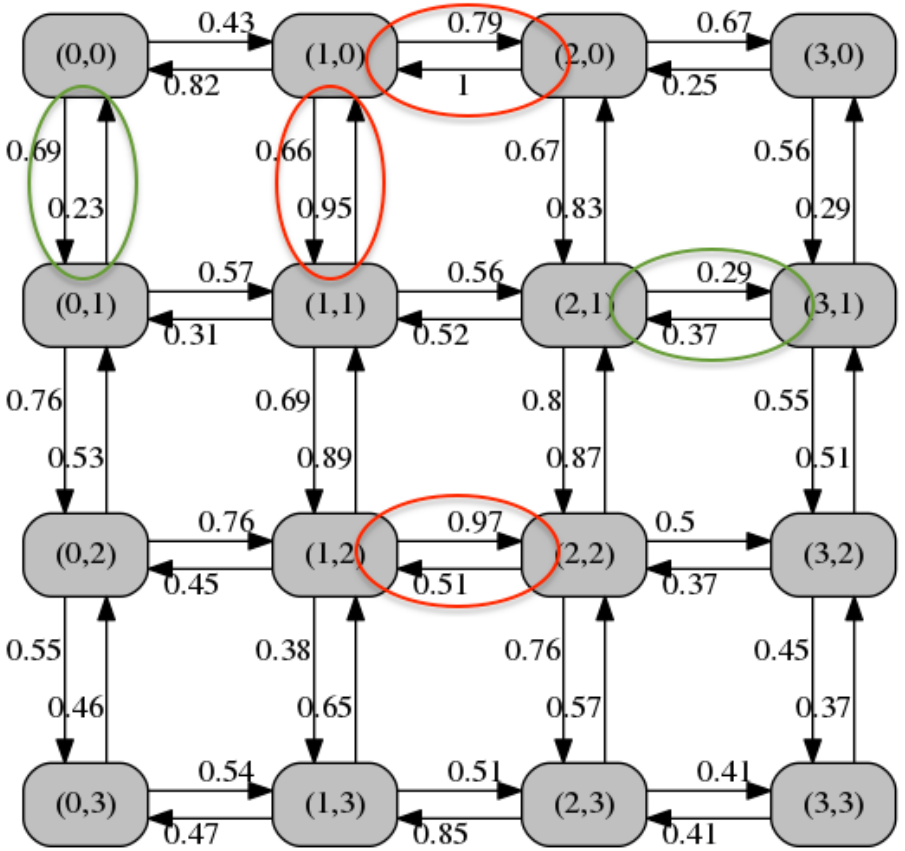
Analysis for behavior prediction

- Behavior prediction modeling
  - Why: help dynamic resource allocation algorithms to steer mapping policy
  - How: based on historical information, collected from execution traces
- Investigated approaches
  - Built-in trace analysis (visual outputs of McSim)
  - Supervised data-mining (not in this talk)

Example of built-in analysis: NoC link load

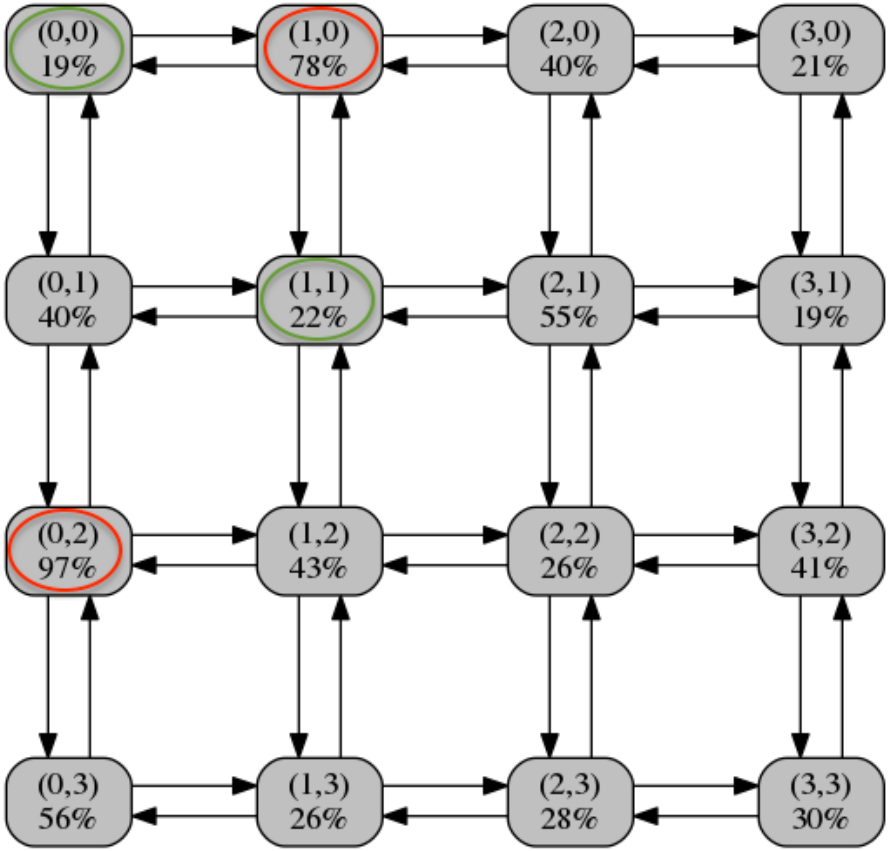


(ZigZag)

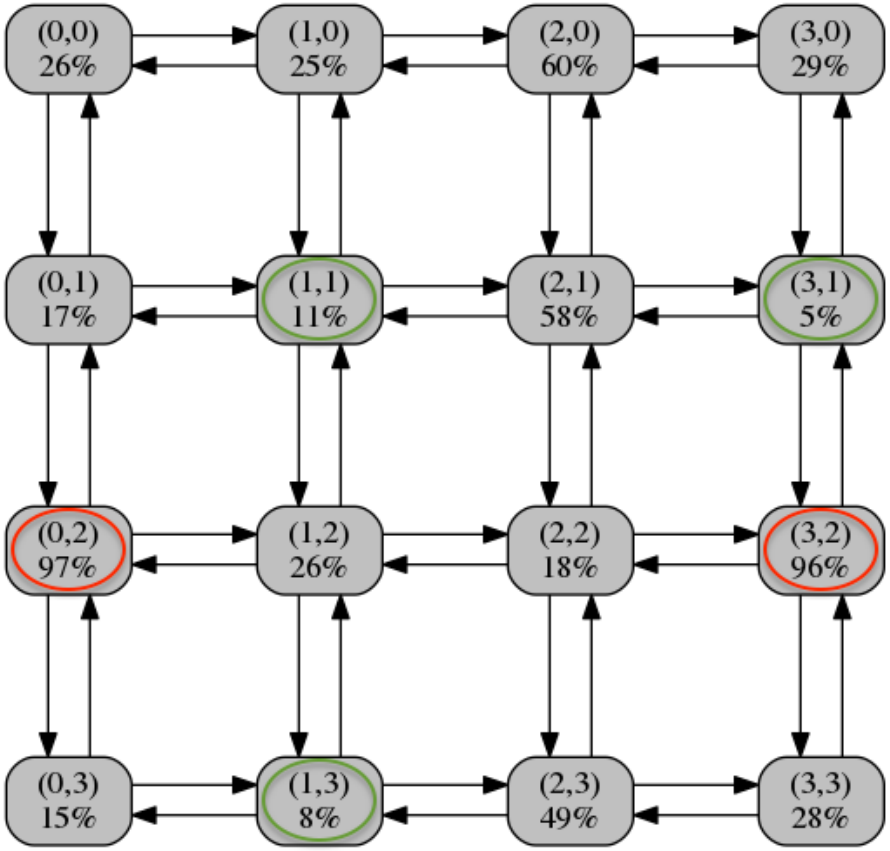


(Local-maximized)

Example of built-in analysis: core utilization



(ZigZag)



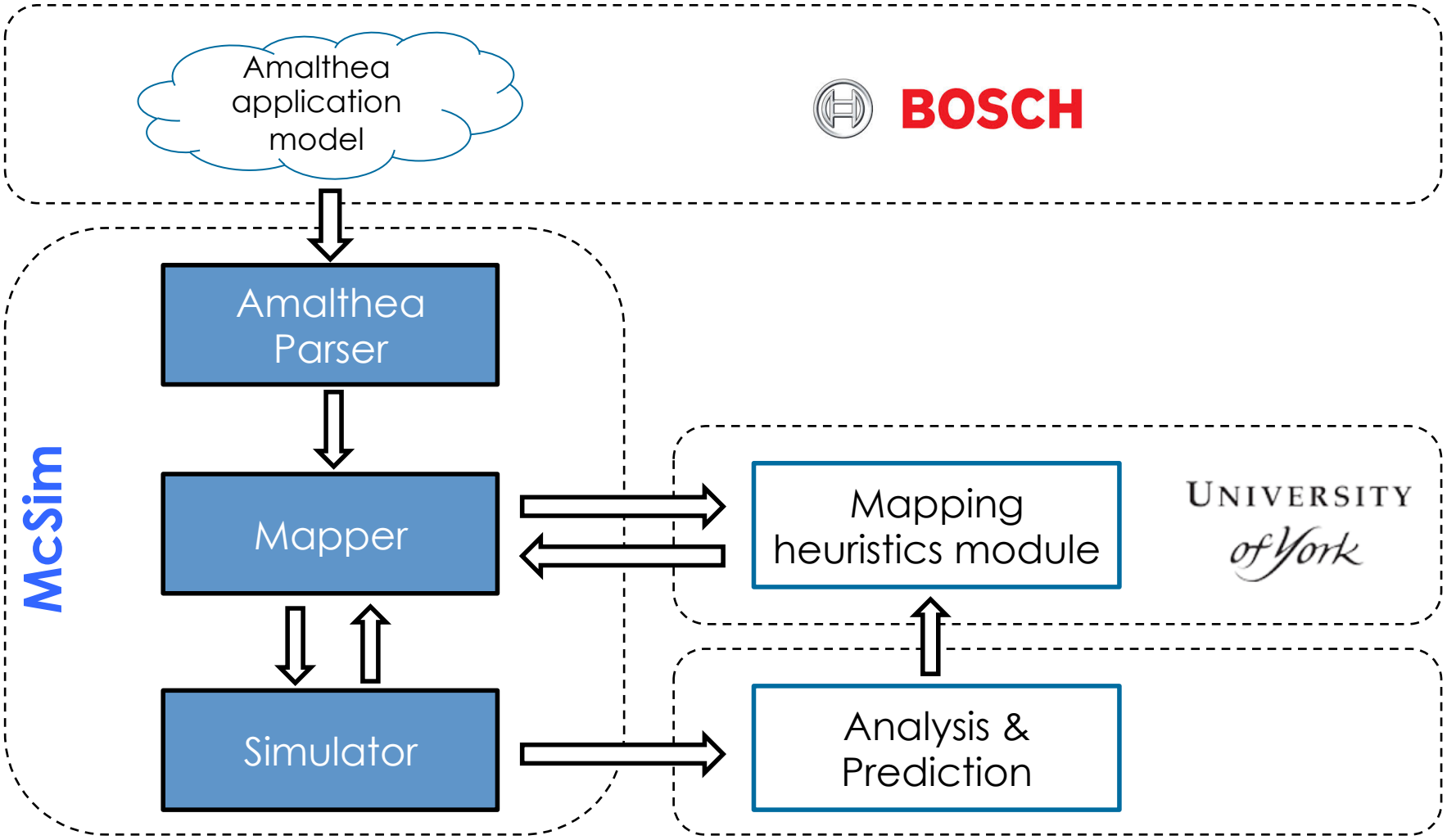
(Local-maximized)

Outline

- Model-based design: modular framework
- Case-study: an automotive application
- Improving mapping decision: analysis & prediction
- Summary

SUMMARY

Overvall orchestration



McSim tool-suite: **https://github.com/DreamCloud-Project**

MODEL-BASED DESIGN AND ANALYSIS OF AUTOMOTIVE APPLICATIONS ON MULTICORE PLATFORMS: AN EFFECTIVE APPROACH

Abdoulaye Gamatié

***** Joint work with: C. Effiong, K. Latif, L. Ost, G. Sassatelli, M. Selva and R. Ursu, L. Zordan, P. Dziurzanski, L. S. Indrusiak...**

LIRMM cnrs UNIVERSITÉ MONTPELLIER