Update on AADLInspector and Cheddar

P. Dissaux*, J. Legrand*, A. Schach*, S. Rubini+, J. Boukhobza+,
L. Lemarchand+, J.P. Diguet+, N. Tran+, M. Dridi+,
R. Bouaziz$, F. Singhoff+

* Ellidiss Technologies
+ Lab-STICC UMR CNRS 6285/UBO
$ ReDCAD Laboratory, University of Sfax

Summary

1. AADLInspector & Cheddar

2. AADLInspector 1.6 : new features

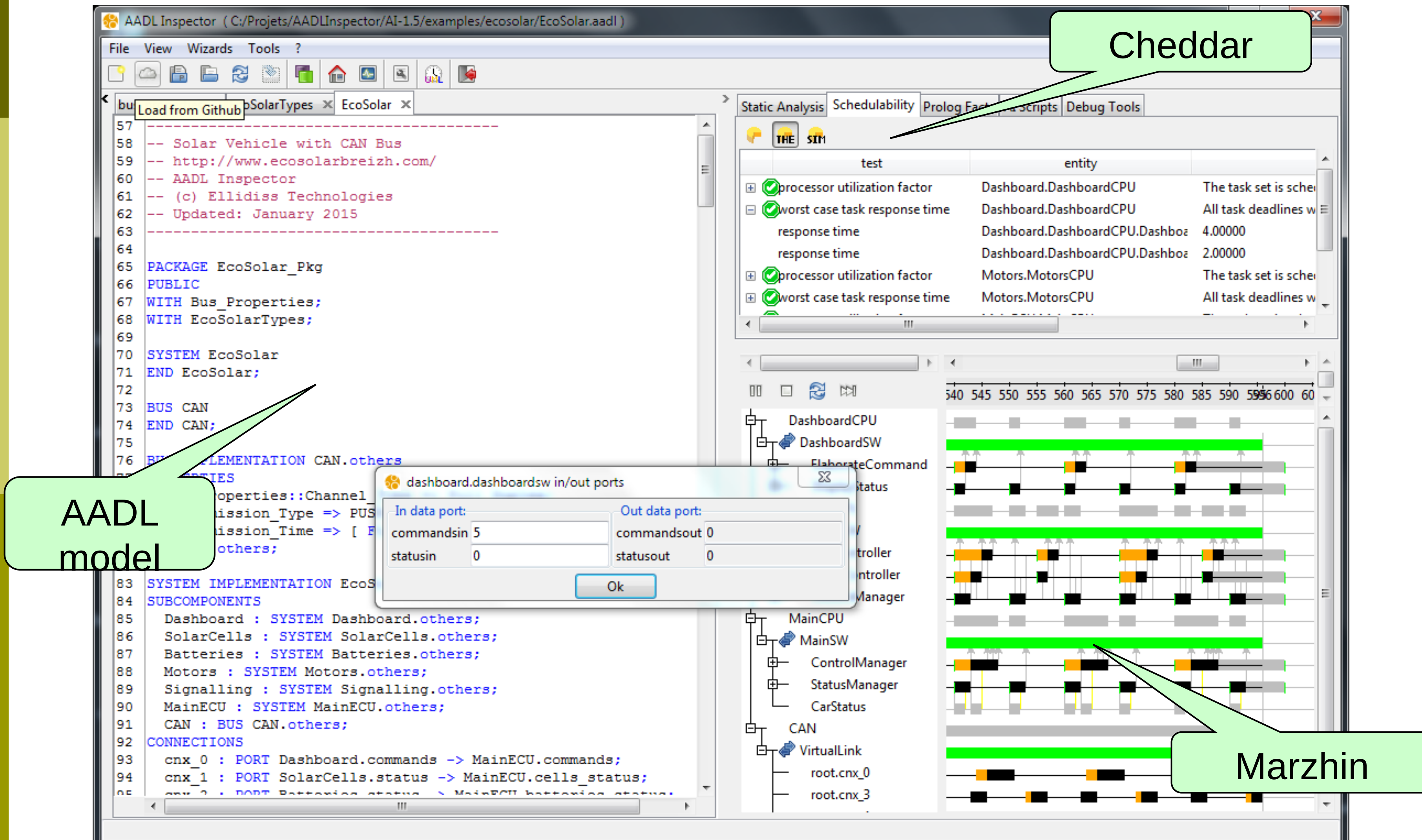
3. Multiprocessor analysis features
   1. Features in AADLInspector
   2. Current research activities outcomes (in Cheddar but not yet in AADLInspector)

Ellidiss

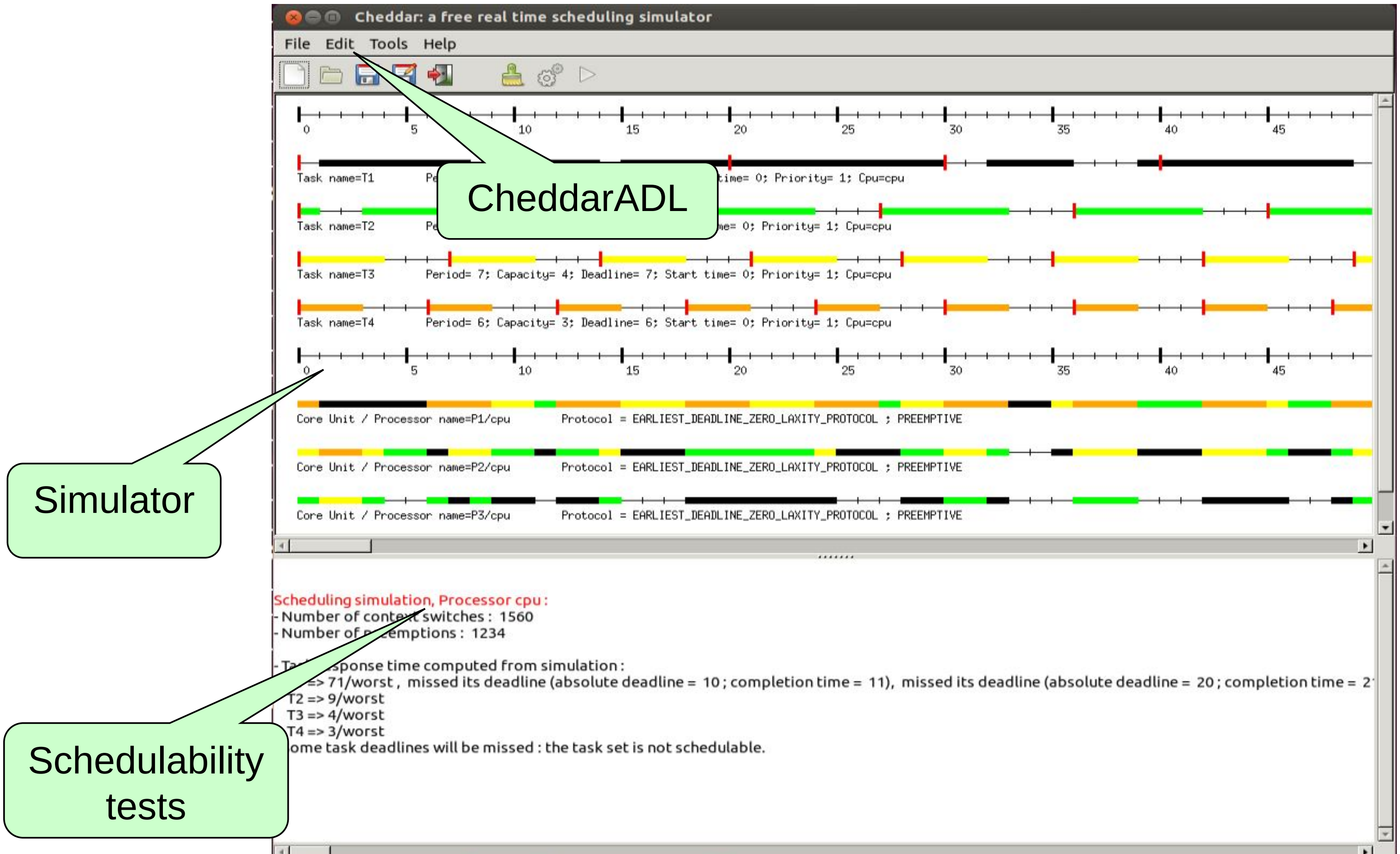
www.ellidiss.com

AADLInspector

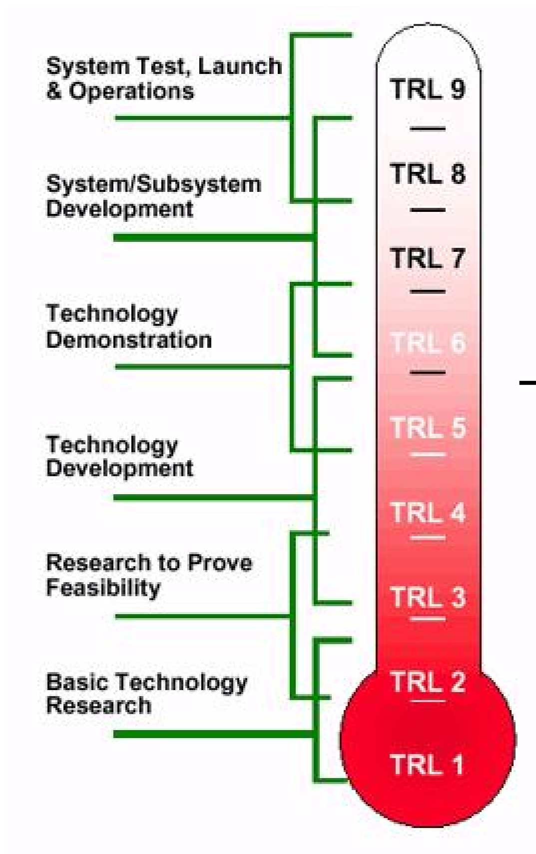
Model Processing Framework



Cheddar
Scheduling analysis framework



From AADLInspector to Cheddar



AADLInspector

Industrialization
Tool packaging
Commercial support (Ellidiss)

Ellidiss
www.ellidiss.com

Cheddar

R&D,
collaborative projects,
prototyping
(UBO + Ellidiss + others)

Lab-STICC
université de bretagne occidentale

- -

Research activities (Lab-STICC/UBO)

Summary

1. AADLInspector & Cheddar

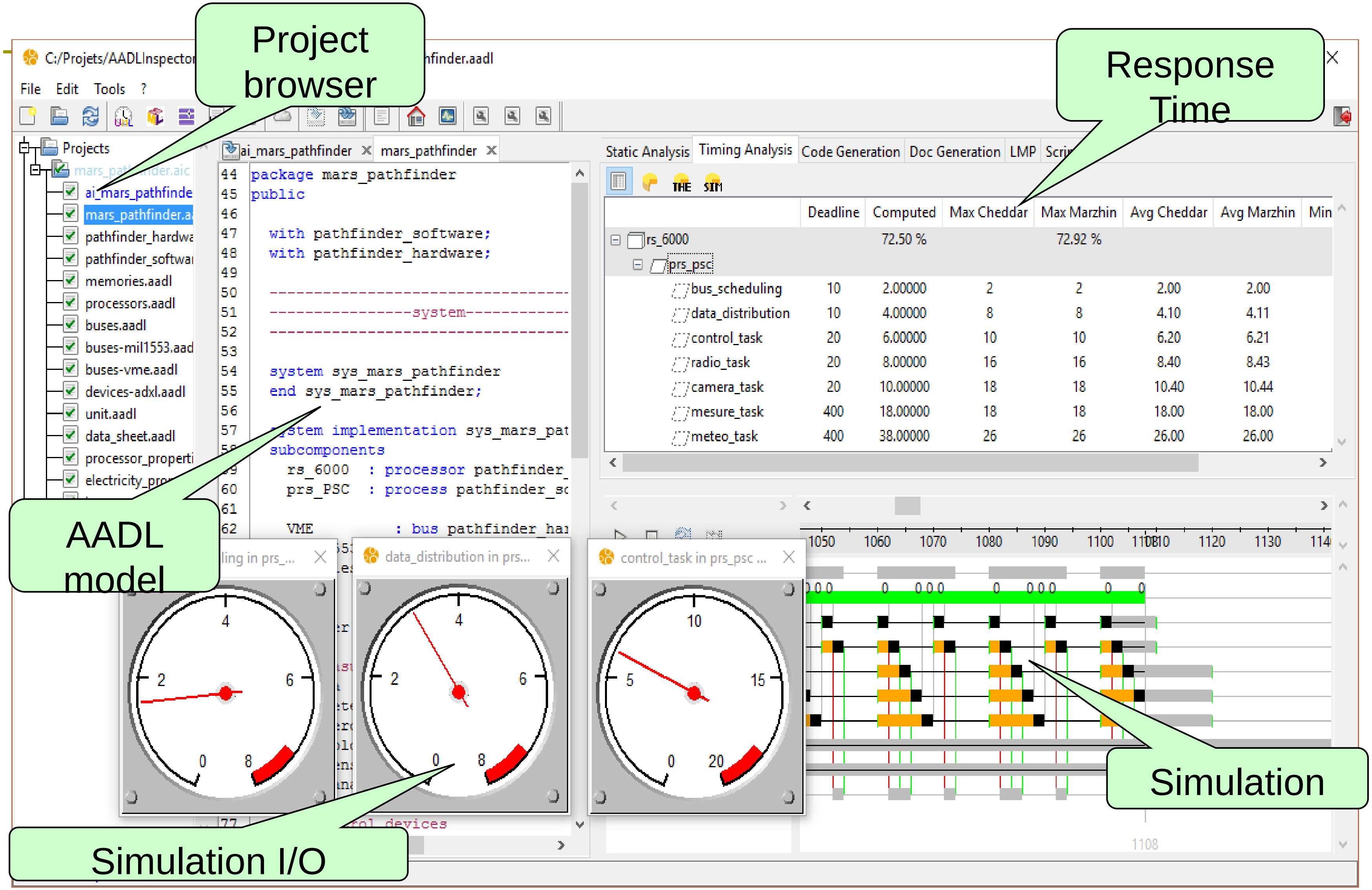
2. AADLInspector 1.6 : new features

3. Multiprocessor analysis features
    1. Features in AADLInspector
    2. Current research activities outcomes (in Cheddar but not yet in AADLInspector)

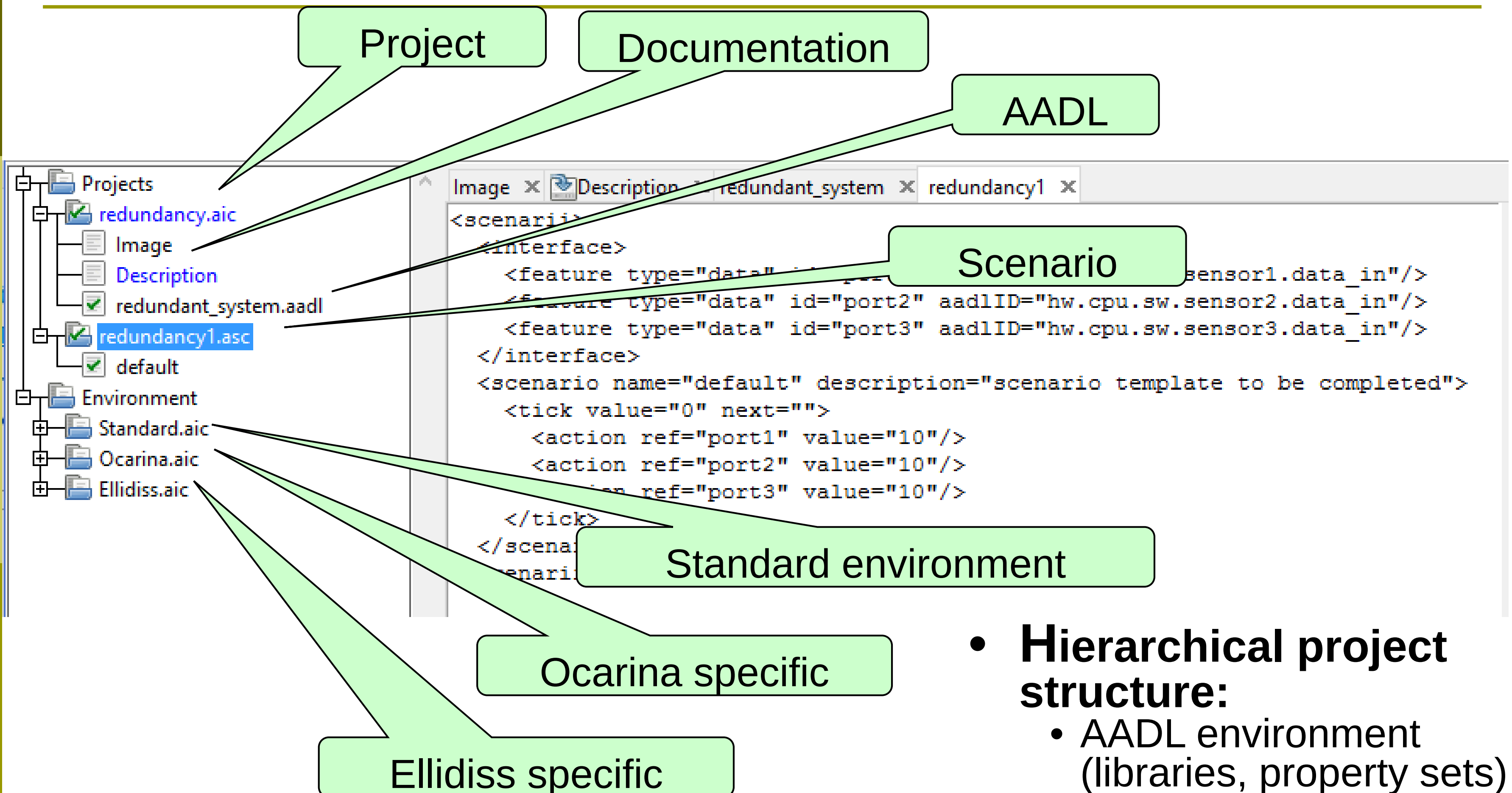
AI 1.6 Features

- **Imports XML/XMI models:**
  - generic transformation process for ECore based models using LMP
  - existing prototypes for UML/MARTE, SysML, Capella, …
- **AADL model processing:**
  - turnkey embedded tools:
    - Cheddar (scheduling analysis)
    - Marzhin (event based simulation)
    - Ocarina (AADL compliancy analysis, code generation)
  - customizable plugins using the LMP AADL toolbox:
    - AADL parser (aadlrev)
    - AADL processing libraries (instance model, legality rules, …)
- **AADL projects manager:**
  - core 2.2 + annex sub-languages EMV1, EMV2, BA 2.0
  - interface with other AADL editors (Osate, Stood, …) and github
  - hiearchical
- **Improved simulation interface**

AADL Inspector 1.6



AADL Projects manager



- **Hierarchical project structure:**
  - AADL environment (libraries, property sets)
  - Sharable sub-projects
  - Simulation scenarii
  - Documentation sections

New presentation of the analysis results

Cheddar Theoritical Tests

Cheddar Simulation

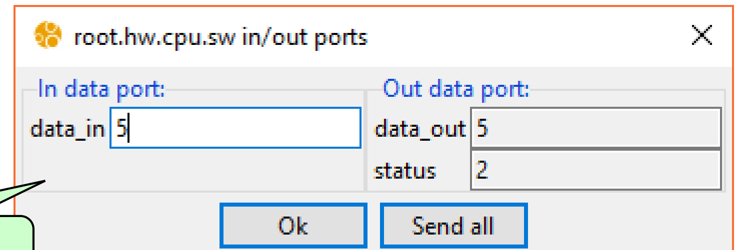
Marzhin Simulation

	Deadline	Computed	Max Cheddar	Max Marzhin	Avg Cheddar	Avg Marzhin	Min Cheddar	Min Marzhin
rs_6000		72.50 %		74.90 %				
prs_psc								
bus_scheduling	10	2.00000	2	2	2.00	2.00	2	2
data_distributio	10	4.00000	8	8	4.10	4.16	4	4
control_task	20	6.00000	10	10	6.20	6.32	6	6
radio_task	20	8.00000	16	16	8.40	8.64	8	8
camera_task	20	10.00000	18	18	10.40	10.64	10	10
mesure_task	400	18.00000	18	18	18.00	18.00	18	18
meteo_task	400	38.00000	26	26	26.00	26.00	26	26

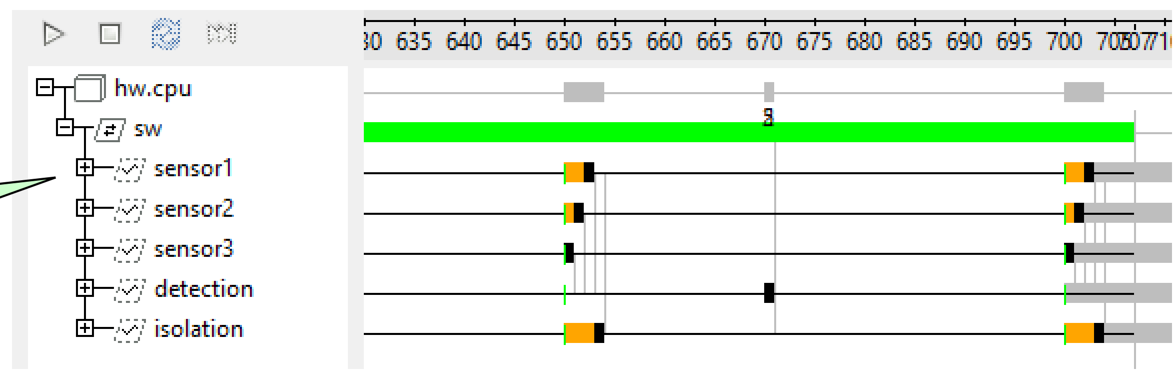
New features to interact during simulation

```
<scenarii>
  <interface>
    <feature type="data" id="port1" aadlID="hw.cpu.sw.sensor1.data_in"/>
    <feature type="data" id="port2" aadlID="hw.cpu.sw.sensor2.data_in"/>
    <feature type="data" id="port3" aadlID="hw.cpu.sw.sensor3.data_in"/>
  </interface>
  <scenario name="default" description="scenario template to be completed">
    <tick value="0" next="">
      <action ref="port1" value="10"/>
      <action ref="port2" value="10"/>
      <action ref="port3" value="10"/>
    </tick>
  </scenario>
</scenarii>
```

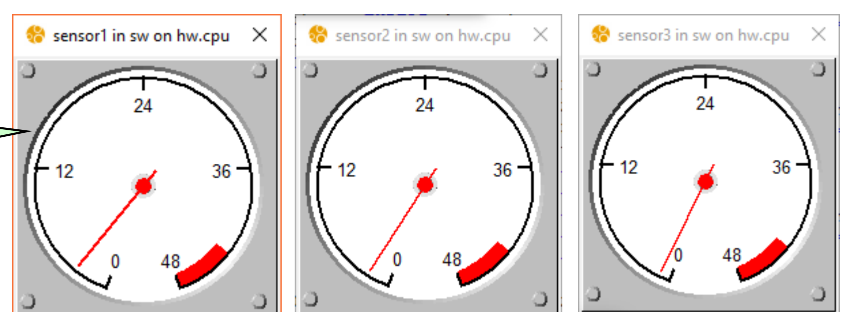
Scenarii



Process I/O



Event bases simulation



Threads Response Time

Multiprocessor with AADLInspector & Cheddar

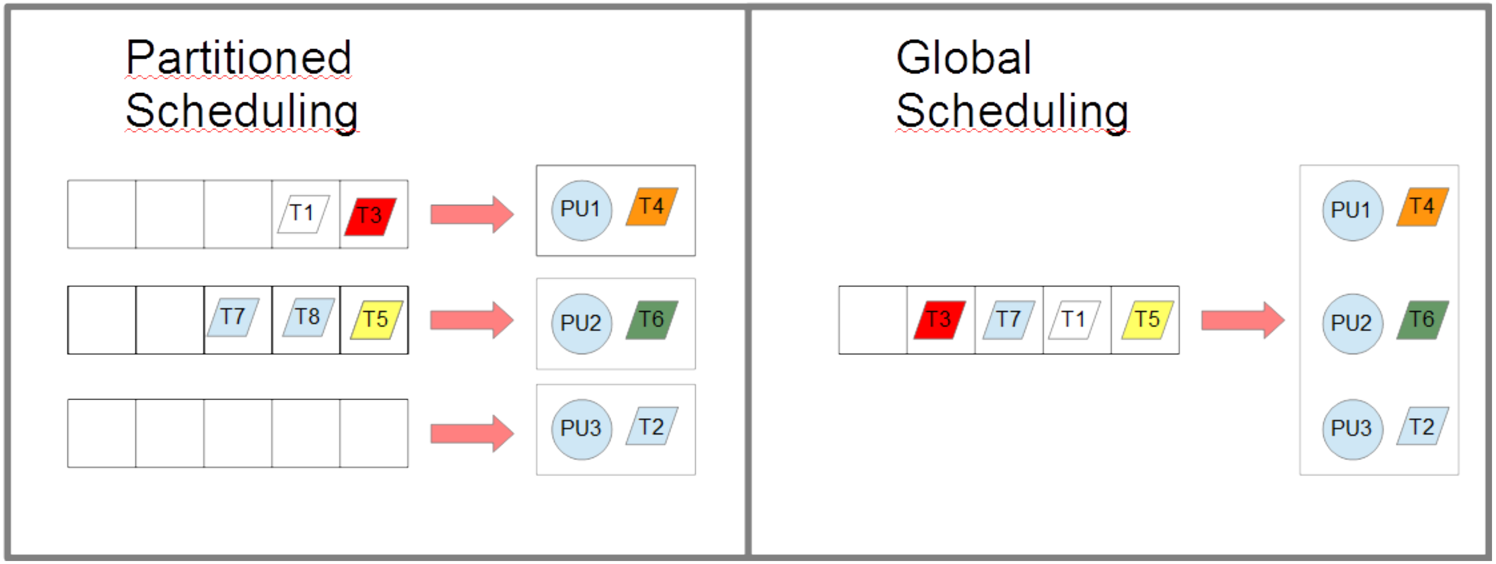
1. AADLInspector & Cheddar
2. AADLInspector 1.6 : new features
3. Multiprocessor analysis features
    1. Features in AADLInspector
    2. Current research activities (in Cheddar but not yet in AADLInspector)

Multiprocessor with AADLInspector & Cheddar

- **SMART project and later:**
  - Define typical distributed/multiprocessor architectures AADLInspector should support
  - How to model such distributed/multiprocessor architectures with AADL
  - Choose or design scheduling analysis features for those patterns
  - Prototype in Cheddar, to be made available in AADLInspector

- **Focus on:**
  - Classical multiprocessor scheduling algorithms : partitioned vs global scheduling algorithms
  - Shared resources between processing units, e.g. cache, memory bus, NoC

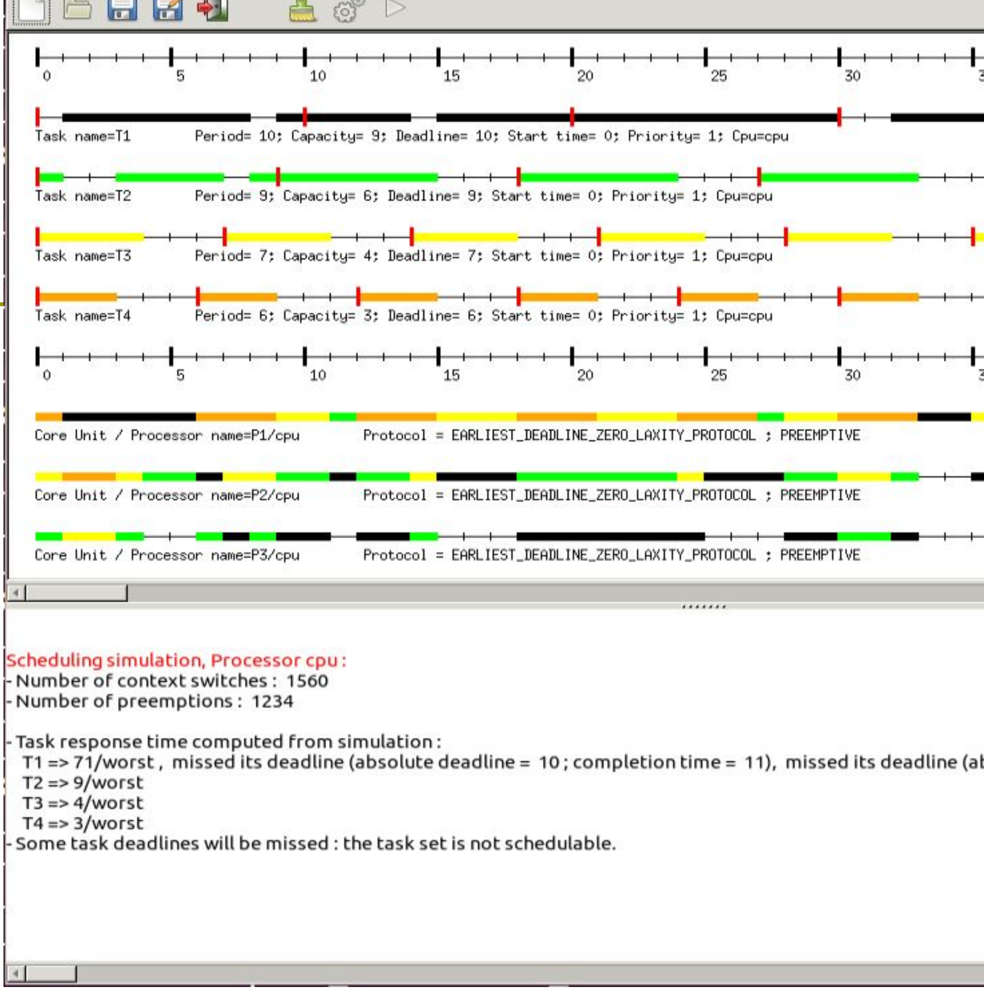
Multiprocessor with AADLInspector & Cheddar



- **Partitioned scheduling :** first assign offline each task on a processing unit ; each processing unit schedules its own task set.
  - No migration. Both online and off-line.
- **Global scheduling:** choose the next task to run on any available processing unit (or preempt if all busy).
  - With migration. Fully on-line.

Multiprocessor analysis features

- ❑ **AADLInspector 1.6 :**
  - ❑ Partitioned scheduling only
  - ❑ Available partitioning policies : Best fit, First Fit, Next Fit, GT, SF



- ❑ **Cheddar 3.x only (not in AI yet):**
  - ❑ Global scheduling : any uniprocessor policies + specific policies such as EDZL, LLREF, Pfair, … **(finished now)**
  - ❑ Shared resources on multiprocessor architectures : Cache**,** NoC, memory **(on going)**
  - ❑ Partitioning optimization approaches based on PAES **(on going)**

Cache/CRPD-Aware Priority Assignment Algorithm

- In fixed priority preemptive scheduling context, tasks can preempt and evict data of other tasks in the cache.
- Cache related preemption delay **(CRPD)**: additional time to refill the cache with the cache blocks evicted by the preemption.

- **Problem statement:**
  - CRPD is high, non-negligible preemption cost. It can present up to 44% of the WCET of a task (Pellizzoni et al., 2007)
  - No fixed priority assignment algorithm takes CRPD into account.

Cache/CRPD-Aware Priority Assignment Algorithm

- Extend Audsley's priority assignment algorithm (Audsley, 1995) to take into account CRPD.
- CRPD-aware priority assignment algorithms (**CPA**) that assign priority to tasks and verify theirs schedulability.
- 5 algorithms with different levels of schedulability efficiency (1) and complexity (2,3).
- Implemented into Cheddar

	CPA-PT-Simplified	CPA-PT	CPA-Tree	Exhaustive Search
(1)	0.65	0.72	0.80	0.87
(2)	Low	Medium	High	
(3)	100 tasks	30 tasks	10 tasks	

Cache-Aware Scheduling Simulator

- **Problem Statement:**
  - Various parameters need to be taken into account in scheduling analysis of systems with cache.
  - Lack of tool addressing all parameters in the state-of-the-art work.
  - Theoretical issues (feasibility interval, sustainability)

Scheduling analysis for systems with cache

Cache Configuration	Control Flow Graph	Worst-Case Execution Time
Memory Layout	Cache Access Profile	Scheduling Policy

- **Approach:**
  - Extend Cheddar component modeling related cache entities.
  - Extend Cheddar scheduling simulator.

Networks-on-Chip Aware Scheduling Analysis

- **Context :**
  - Networks-on-Chip (NoC)
  - Communication infrastructure based on links and routers that interconnect PU or memory unit, providing packet-based data transfer.

- **Problem statement :**
  - Relationships between thread models and communication models
  - Various AADL thread communication design patterns
  - Various NoC designs
  - Today : AADL data port & 4x4 Wormhole XY NoC

  - How to model both thread and communication models in order to enforce schedulability?

Networks-on-Chip Aware Scheduling Analysis

- **Dual Task and Flow Model (DTFM) :**
  - Computes the flow model from a task model, task mapping and precedence constraints
  - Identify/compute delays induced by the NoC architecture and perform scheduling analysis

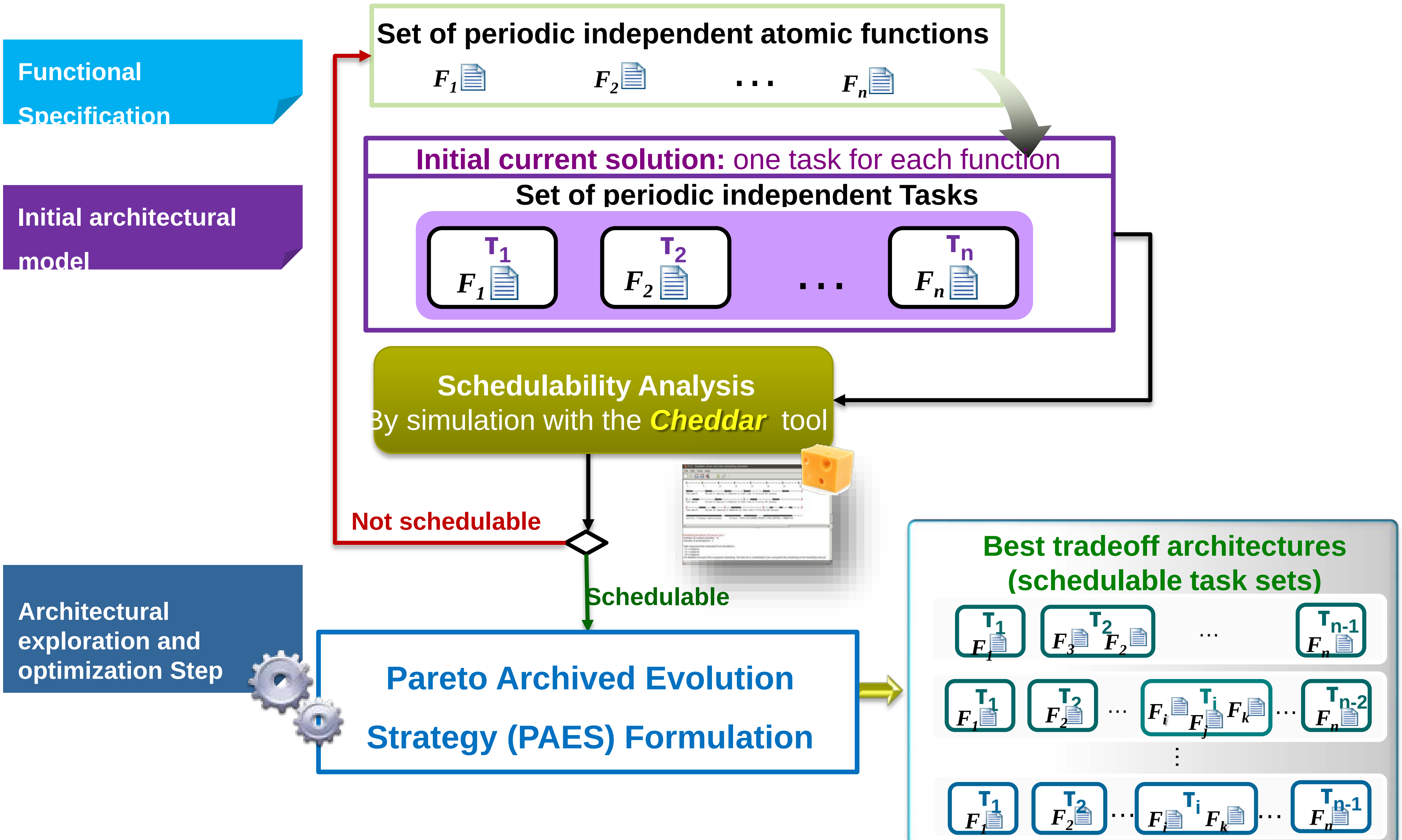
- **DTFM Implemented into Cheddar**

- **DTFM is evaluated with a multiscale toolset** composed of a tick accurate real-time scheduling tool (Cheddar) and a cycle accurate SystemC NoC simulator (SHOC).

Partitioning methods, multi-objective optimization

- **Problem Statement:**
  - Mapping functions to software architectures (i.e. RTOS tasks)
  - Conflicting objective functions, e.g. number of preemptions vs laxity
    - Tradeoffs between large number of candidate software architectures
- **Contributions:**
  - Formulation based on PAES (Pareto Archived Evolution Strategy) to explore possible functions to tasks assignments
  - Implementation into Cheddar, both sequential and parallel implementation
  - Uniprocessor only right now

Partitioning methods, multi-objective optimization



Conclusion

- **New features in AADLInspector & Cheddar: about multiprocessors analysis:**
  1. Typical multiprocessor architectures (SMART project)
  2. Classical multiprocessor scheduling algorithms: partitioning vs global scheduling algorithms
  3. New analysis features when (hardware) shared resources between computing units

- **AADL models handled during those activities**
  - Bindings and Implemented_As ??
  - Need a white paper

- **Questions ?**