Potsdam, Germany
02/06/2009

Le Fonds Européen de Développement Régional et la Région wallonne investissent dans votre avenir

UML-AADL'09: Towards a Model-Driven Approach for Mapping Requirements on AADL

Mathieu DELEHAYE
Christophe PONSARD

www.cetic.be

Your connection to ICT research

Outline

1. Motivation
2. NFR and tools survey
3. Goal-oriented Modelling
4. Requirements model mapping to AADL
5. Conclusions and perspectives

Running example: cruise control

Outline

1. **Motivation**
2. NFR and tool survey
3. Goal-oriented Modelling
4. Requirements model mapping to AADL
5. Conclusions and perspectives

Motivation: Bridging the Requirements to Architecture GAP

- Architectural design
  - Some styles, patterns
  - Remains much of an “art”

- Towards a model-driven approach
  - Architectural models, like AADL
  - Requirement models, goal-oriented approach, like KAOS

Functional Requirements vs Non-Functional Requirements

- Functional Requirements (FR)
  - define the functional effects the software-to-be is required to have on its environment
  - "WHAT" aspects
- Non-Functional Requirements (NFR)
  - define constraints on the way the software-to-be should satisfy its FR or on the way it should be developed
  - drive design choices
- **In real-time embedded systems, NFR are as much important as FR !**

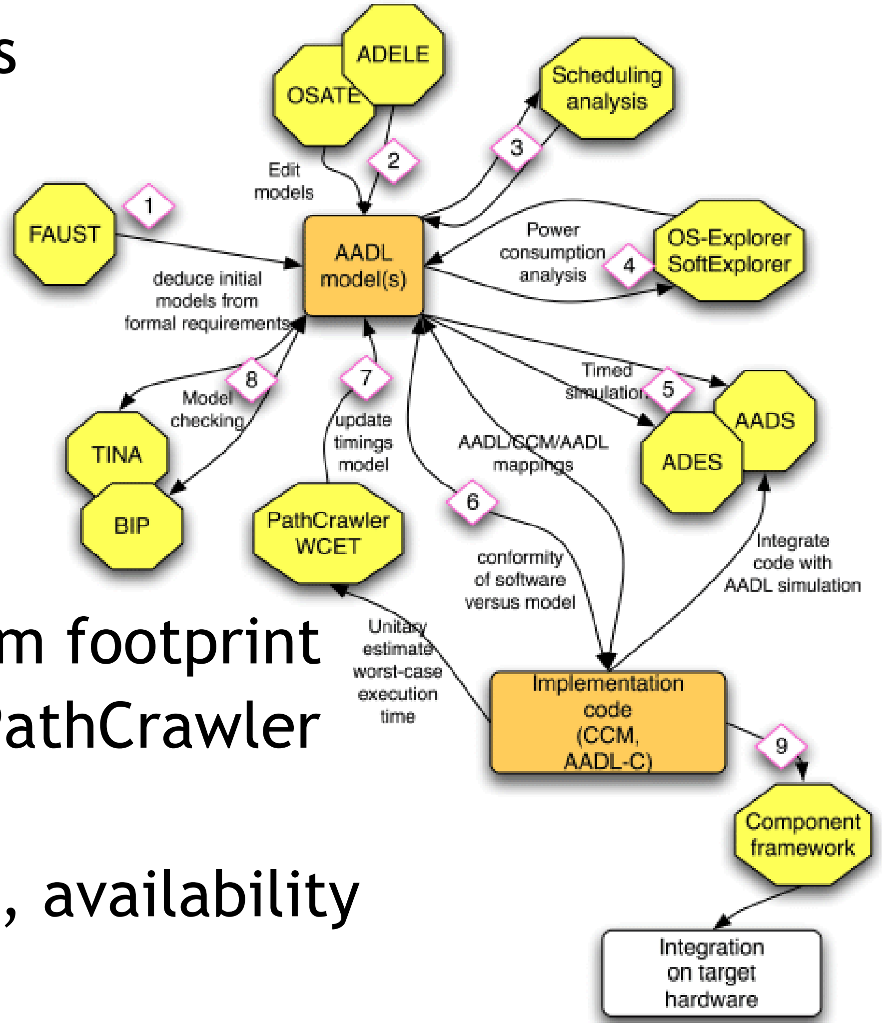
cetic

Outline

1. Motivation
2. **<u>NFR and tool survey</u>**
3. Goal-oriented Modelling
4. Requirements model mapping to AADL
5. Conclusions and perspectives

Non-Functional Requirements for Embedded Systems

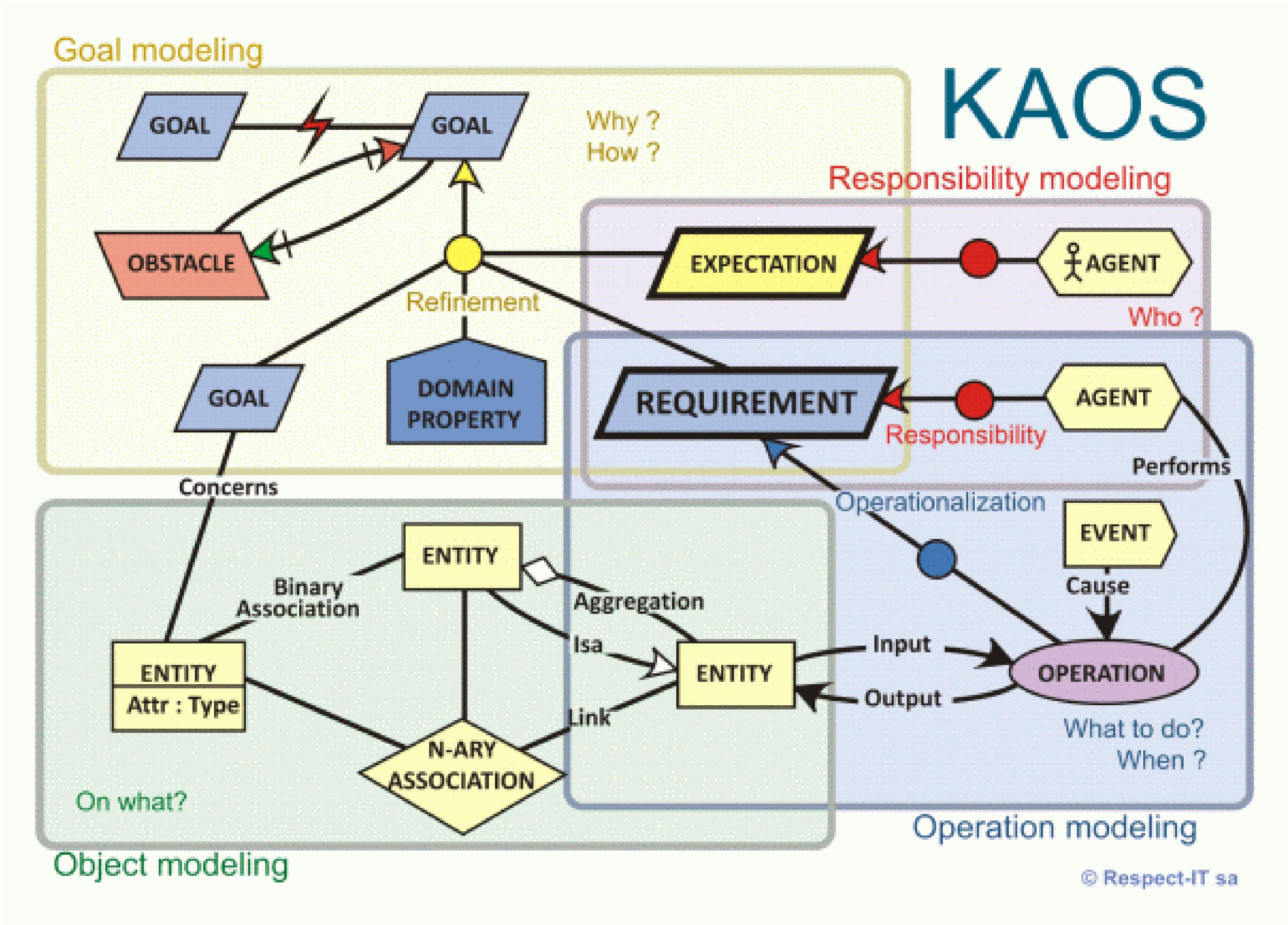
- Dependability requirements
  - RAMS, MTBF/MTTR/SIL
  - Formal verification
  - TOOL: TINA
- Performance requirements
  - Deadlock/starvation
  - Real-time constraints
  - Power consumption, mem footprint
  - TOOLS: CAT, Cheddar, PathCrawler
- Security
  - Confidentiality, security, availability



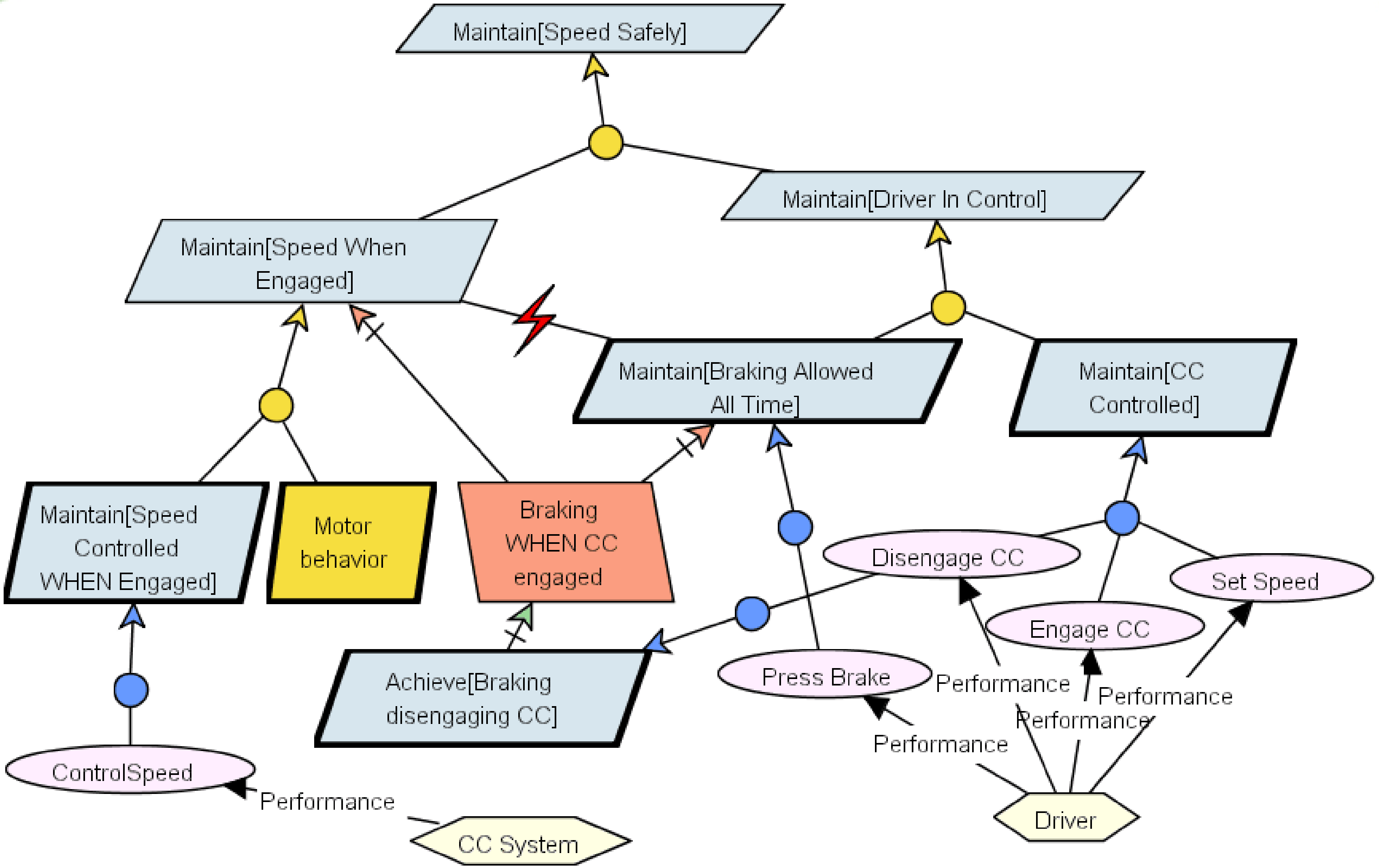
Outline

1. Motivation
2. NFR and tool survey
3. **<u>Goal-oriented Modelling</u>**
4. Requirements model mapping to AADL
5. Conclusions and perspectives

Goal-Oriented Requirements Engineering



Goal Model of the Cruise Control System



Formal level: goals

Requirement Maintain[SpeedControlledWHENEngaged]

FormalDef *cc.engaged*

$\Rightarrow |car.speed - cc.targetSpeed| < cc.margin$

Requirement Achieve[CruiseDisabledWHENBraking]

FormalDef *brakePedal.pressed*

$\Rightarrow \Diamond_{<max} \neg cruiseControl.state = ENABLED$

Formal level: operation

Operation ComputeTorque

Resp *CruiseController*

Input *car.speed,cc.targetSpeed*

Output *car.torque*

ReqPre for *Achieve*[*CruiseDisabledWHENBraking*] :

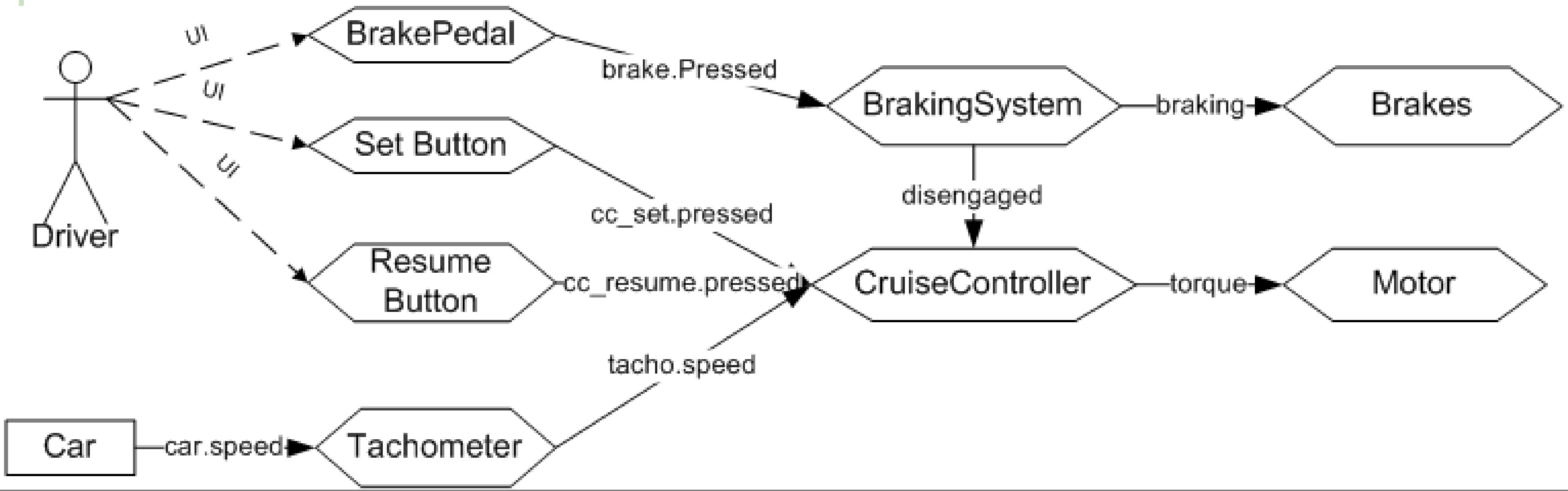
cc.engaged

ReqPost for *Maintain*[*TorqueComputed*] :

car.torque=f(car.speed,cc.targetSpeed)

Agent Model: a High Level Architecture

- Outcome of the goal-oriented process
- Information flow between agent
- Based on responsibilities (goals) and capabilities to monitor/control information (operation)

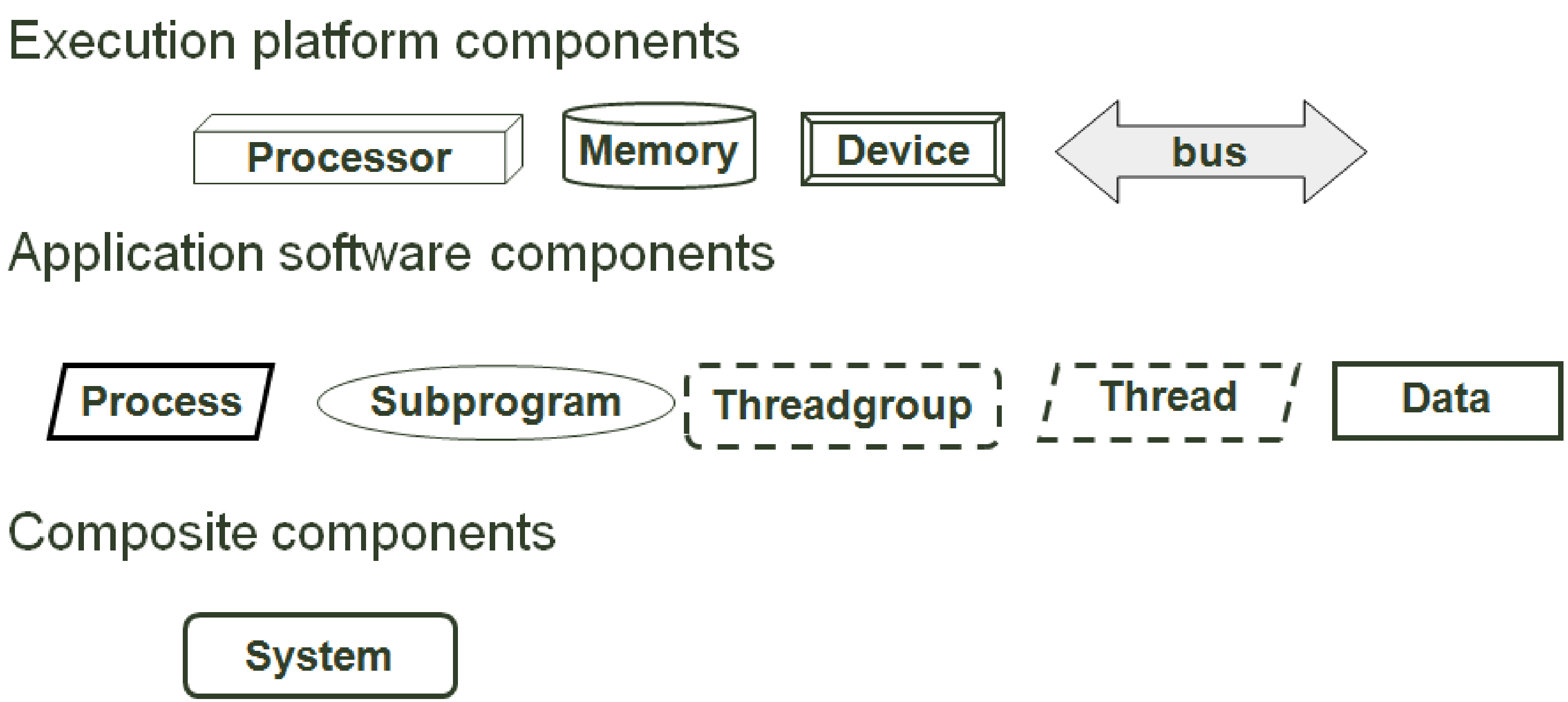


Outline

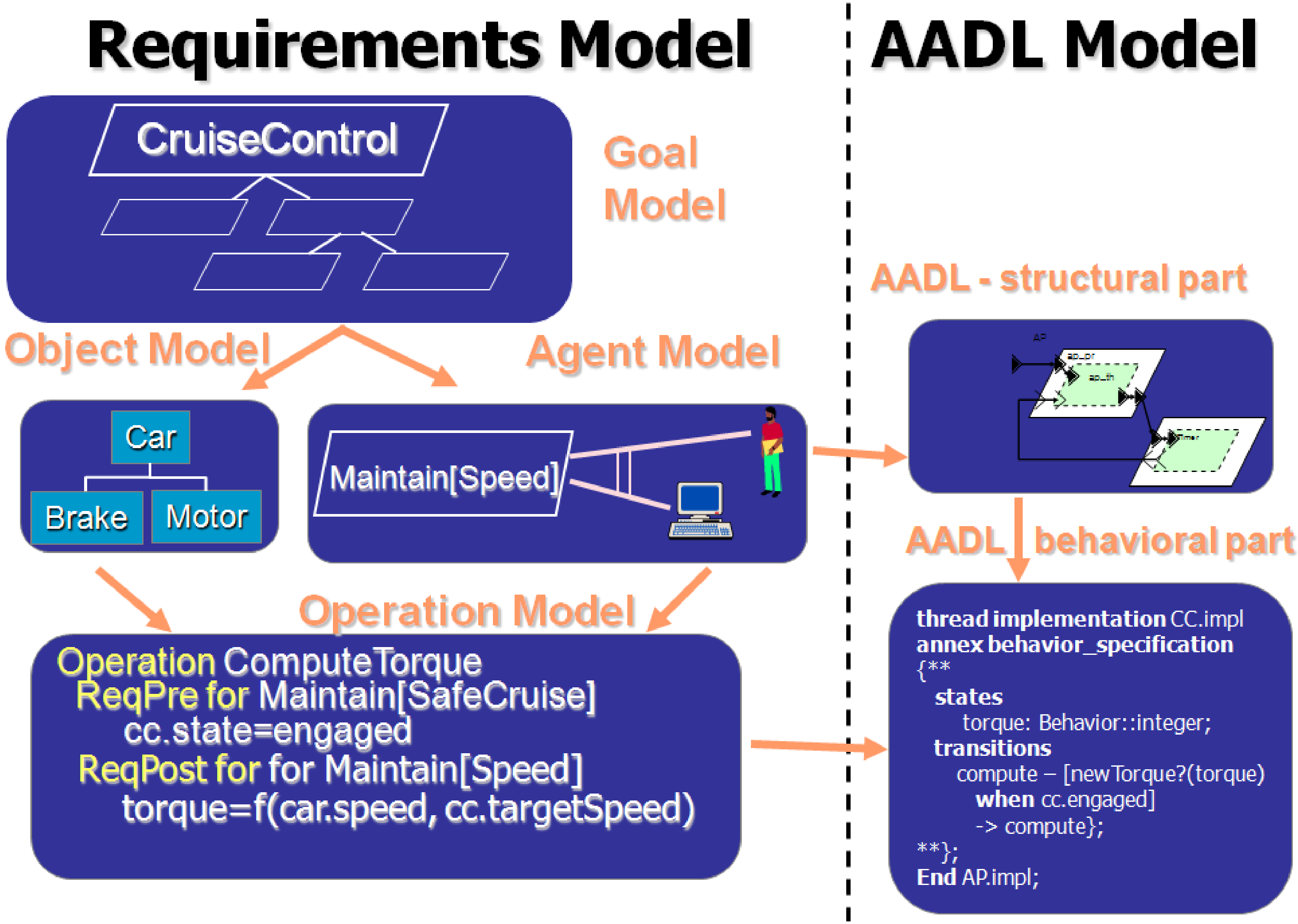
1. Motivation
2. NFR and tool survey
3. Goal-oriented Modelling
4. **<u>Requirements model mapping to AADL</u>**
5. Conclusions and perspectives

Quick reminder on (A)ADL

- Components: HW or SW, 2 levels of description
- Connectors: data and control flows



Principle of Mapping Requirements and AADL Models



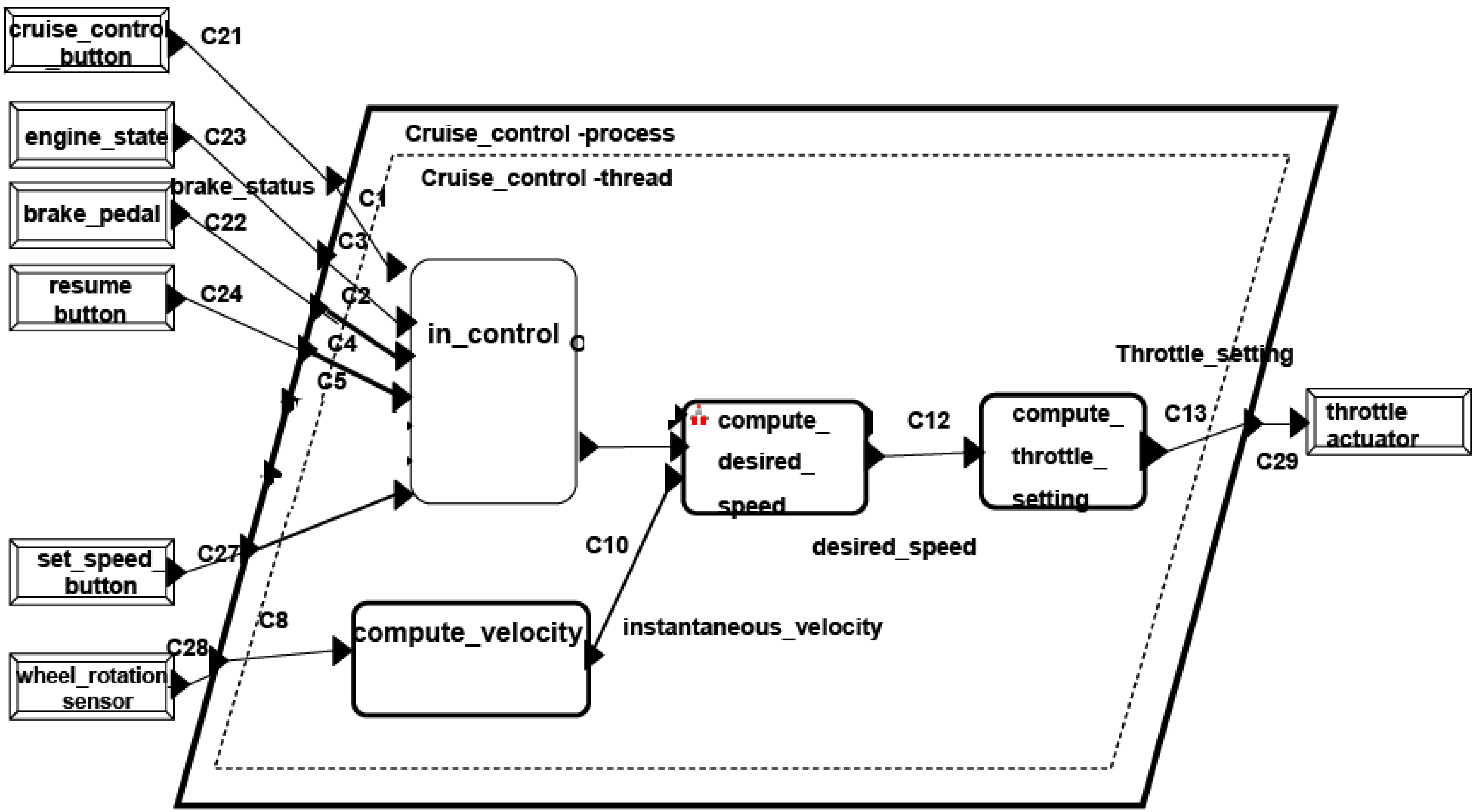
Global mapping process

- producing the AADL structural description
  - mostly based on the agent model,
  - complemented by the goal and object models.
- producing the AADL behavioral part,
  - based on the operation model
- refining the initial architecture
  - by injecting NFR based on a pattern library.

Deriving the structural part

- each of this software process corresponds to an agent of the requirements model.
- for each of those agent, the inner operations under its operation are depicted as components with:
  - a data input port defined for each input of the operation
  - a data output port defined for each input of the operation
  - a control port is defined for each variable of the trigger part of the operation
- additionally, use AADL properties for specific requirements (e.g. temporal constraints as deadlines)

AADL Structural Model of the Cruise Control

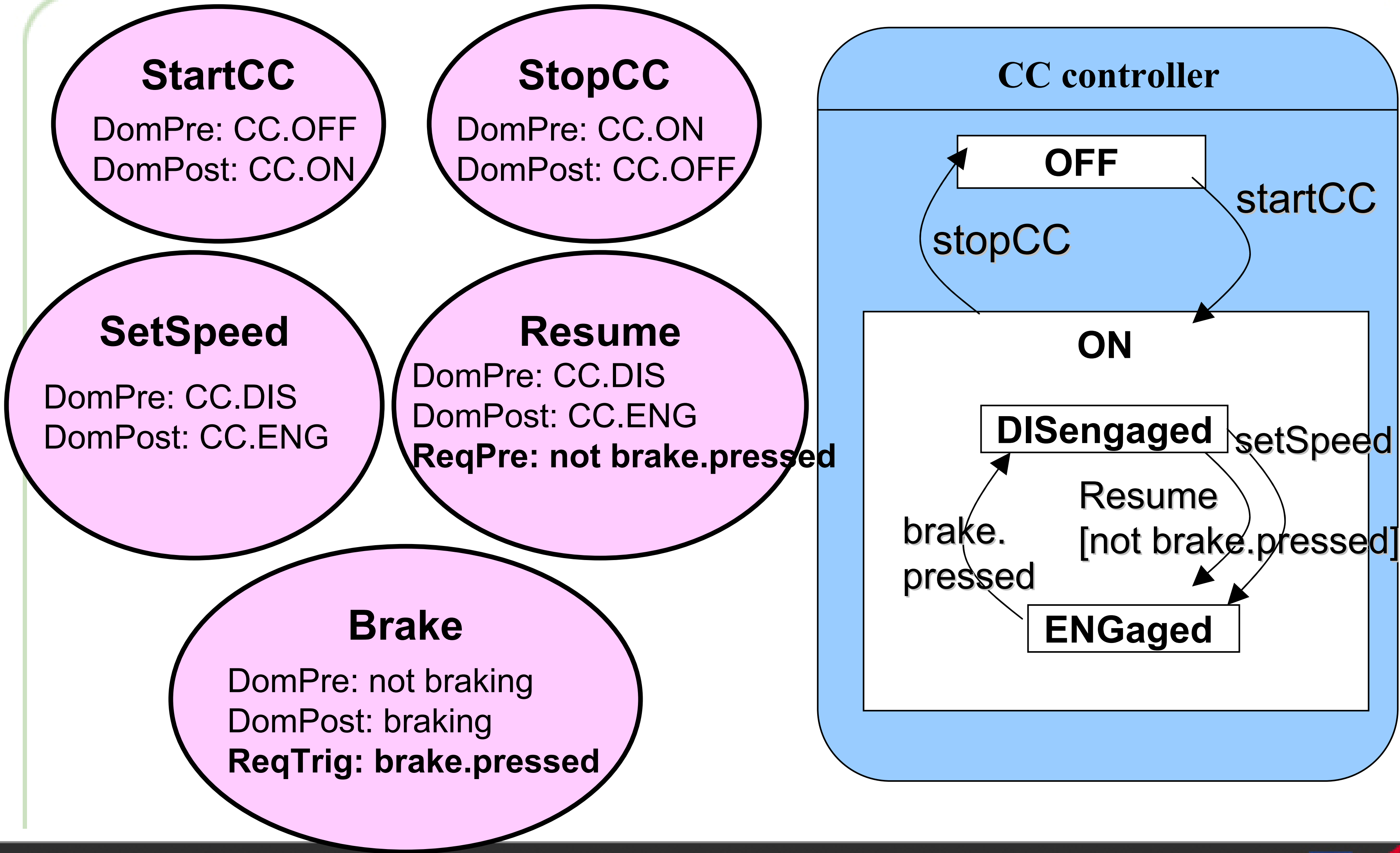


cetic

Generating the behavioural part

1. Generation of Finite State Machine (FSM) by a goal based generator
   - Input: a collection of operations under the responsibility of an agent
   - Output: FSM describing the states and transitions logic expressing the objects and operations.
2. Generation of AADL mode-transition based on the generated FSM. The structure of the FSM is recoded in AADL style.

Generating the behavioural part



Typical Behavioral Description of the Cruise Control in AADL

```
thread CruiseController
features
   speed: in data port;
   brake: in data port;
   ccPressed: in event port;
   resumePressed: in event port;
   torque: out data port;
end CruiseContoller;

thread implementation CruiseController.i
annex behaviorspecification{**
states
   OFF: initial states;
   ON: complete join state;
```

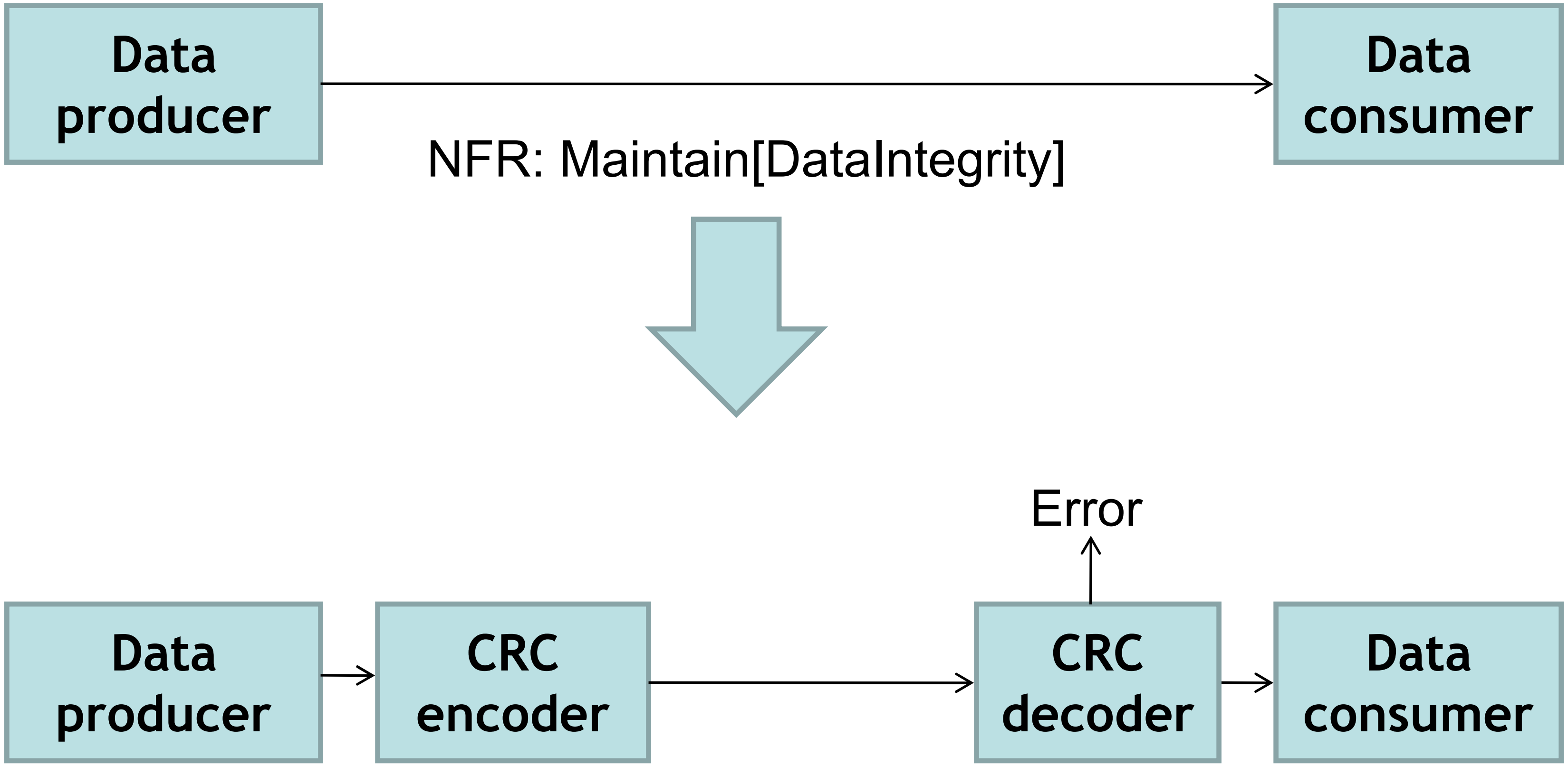
```
transitions
   OFF-[ccPressed?]-> ON;
   ON-[ccPressed?]-> OFF;
Composition state ON
   states
      DIS: initial state;
      ENG: complete state;
   transistions
      DIS-[resume?]-> ENG;
      ENG-[on brake>0]->DIS;
      ENG-[on |speed-
targetSpeed|>margin]->ENG

{torque=fn(speed,targetSpeed)};
end ON ;
**};
end CruiseController
```

NFR-driven refinement (on-going)

- Refining architecture by adding/replacin/constraining components and connectors
- Based on pattern approach (reusable)
  - Category specific patterns: availability, accuracy, fault tolerance, interoperability, security
  - Domain specific patterns (e.g. embedded systems): Actuator-Sensor, Controller Decompose(splitting responsibilities), Monitor-Actuator (reliability), Fault Handler (fault tolerance), Watchdog (reliability)

Example: data integrity connector



Outline

1. Motivation
2. NFR and tool survey
3. Goal-oriented Modelling
4. Requirements model mapping to AADL
5. **<u>Conclusions and perspectives</u>**

Conclusions

- Model-driven method to derive initial architecture from requirements models
- Benefits of requirement models:
  - early reasoning on FR (completeness, consistency)
  - identification of NFR and connection with FR
- NFR taken into account in this process, also taking into account specificities of embedded systems, allowing the analyst to stay in control of the way he wants to transition from the problem to the solution space.
- This work was restricted to the context of goal-based requirements and AADL model but can be generalized.

cetic

Future Work

- More research on architecture refinement driven by NFR, especially related to embedded systems.
- Implement a prototype of the mapping based on the Objectiver and TOPCASED tools
- Encode and enrich pattern library.
- Moreover the alignment of the requirements approach with existing standards such as UML/MARTE or SysML must also be considered.

cetic

Questions ?