energie atomique • energies alternatives

AUTOSAR Timing Extension and a Case Study for Schedulability Analysis

ArtistDesign Workshop on Real-Time System Models for Schedulability analysis
University of Cantabria
7-8 February 2011

sara.tucci@cea.fr

INSTITUT CARNOT CEA LIST

digiteo

Agenda

- AUTOSAR quick look
- AUTOSAR development methodology
- AUTOSAR Timing Extension
- Case study, enabling schedulability analysis on AUTOSAR models (from “Enabling Schedulability Analysis for AUTOSAR Sytems” to appear ISORC11)

the AUTomotive Open System ARchitecture

Approach to engineering automotive systems that decouples the software architecture from the computing platform

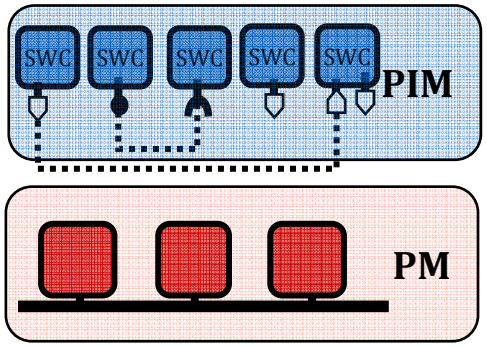
Rich set of standards in their fourth revision

The AUTOSAR initiative is in its seventh year

Goals of AUTOSAR aligns to MDA

- **AUTOSAR metamodel**
- **Development Methodology**
- **Timing Extension to support verification of timing constraints (newly added in last revision)**

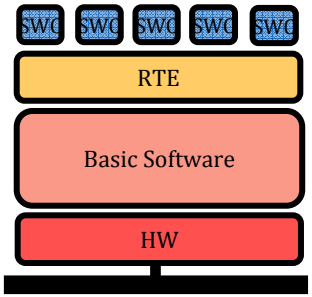
AUTOSAR: quick look

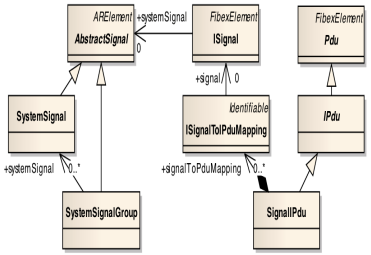


VFB Level

MDA principles

layered architecture

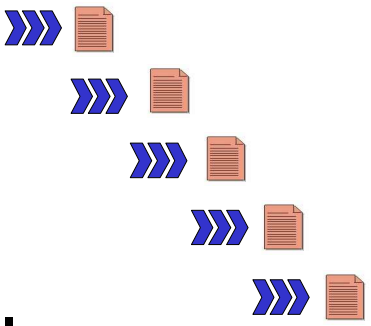




metamodel

AUTOSAR

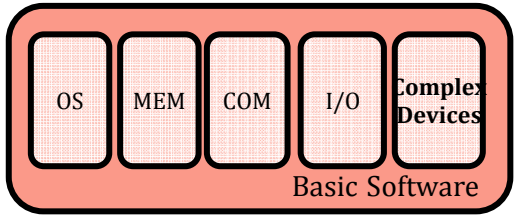
VFB Level + System



development methodology

Code!

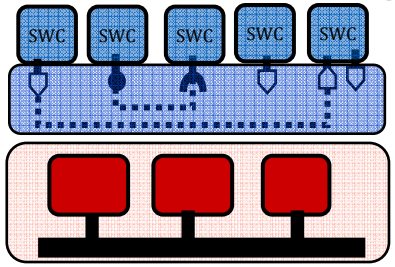
timing extension



standard basic software

AUTOSAR methodology

Vehicle Architecture Design



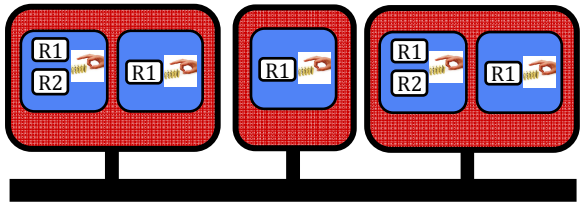
.XML System Configuration Input

SWCs and Connections

ECUs and topology

Mapping Constraints

System Configuration

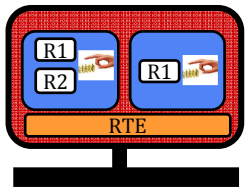


.XML System Configuration Description

Mapping of SWC to ECU

Component Internal Behavior

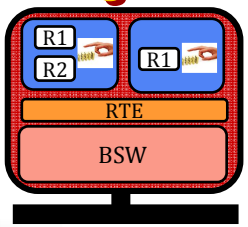
Extract ECU specific information



.XML Extract of System Configuration Description

information specific to each ECU is automatically extracted

Configure ECU



.XML ECU Configuration Description

tasks, mapping of runnables on tasks, BSW configuration

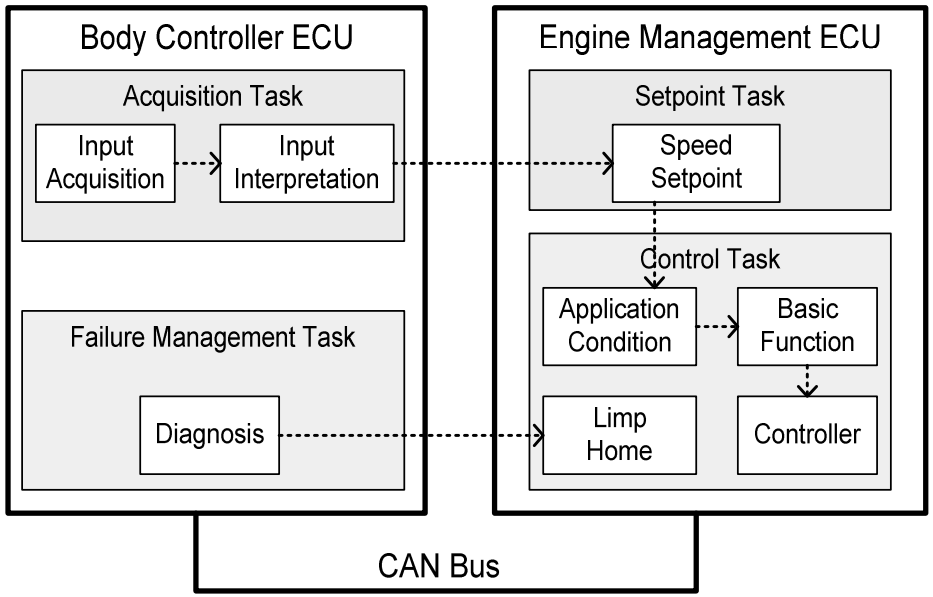
ECU Executable generation

.EXE

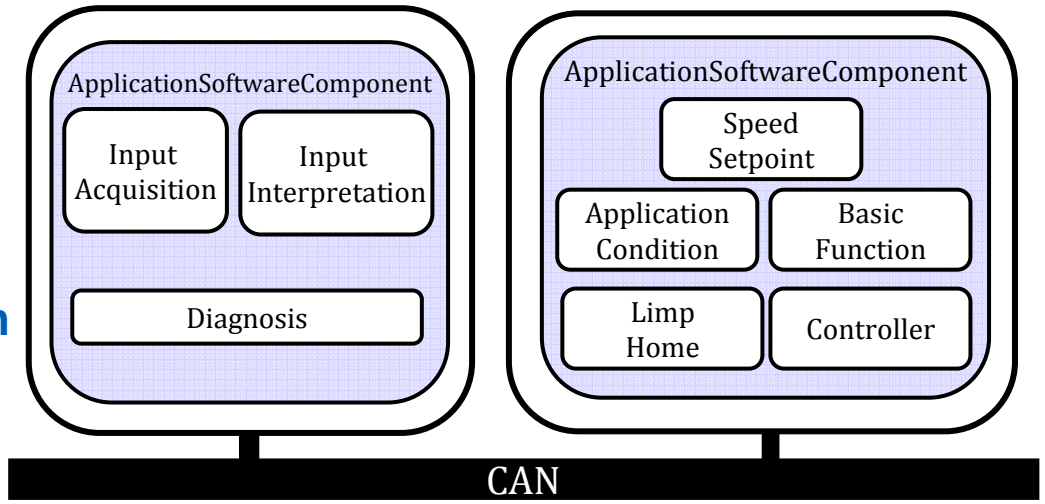
Timing Extension

– confidential

Case Study – cruise control system



System Level



SwcTiming

For each runnable: 1 basic event chain `TimingDescriptionDEventChainDescription`

- **Stimulus: `TimingDescriptionEvent=RunnableEntityActivated`**
- **Response `TimingDescriptionEvent=RunnableEntityTerminated`**

To model end-to-end flows, two event chains built by composing basic event chains

- *Control event chain and Failure management event chain*

To model end-to-end deadlines:

- **For** control event chain **`LatencyTimingConstraints=500ms`**
- **For** failure management event chain **`LatencyTimingConstraints=100ms`**

Case Study cruise control system

Table 1. Timing information of the cruise control functions

Functions	WCET (ms) **	Period (ms)	Deadline (ms)	Allocated to	Task priority
Input acquisition	2.5	10	20	Acquisition Task	1
Input interpretation	2.32	40	80		
Diagnosis	1.52	10	20	Failure Management Task	4 (highest priority)
Speed setpoint	3.5	40	80	Setpoint Task	2
Limp home	1.03	10	20	Control Task	3
Application condition	3.92	40	80		
Basic function	2.08	40	80		
Controller	1.4	40	80		

** The WCETs used in this example were measured using internal methods and tools that for confidentiality reasons cannot be presented here

ECU Level

- To model OS Tasks and Runnables/Task Mapping, the ECU configuration description should be used.
- The mapping is done by creating a **`RteEventToTaskMapping`** that allows referencing the RteEvent (the runnable) and the OStask.
- On **`OStask`** priorities can be set.
- Worst Case Execution Time is specified by `ResourceConsumption` in each application software component implementation.

Swc Timing

- Periods for runnables are part of SwcTiming, where an **`EventTriggeringConstraint`** is specified in the stimulus event (**`RunnableEntityActivated`**) of the corresponding event chain for the runnable
- **`LatencyTimingConstraints`** are also specified for each runnable to specify deadline

Schedulability Analysis - MAST configuration

- **Offset-based technique used, technique table:**

Transaction	External Event period (ms)	Activity	Rate divisor	Local deadline (ms)	Global deadline (ms)	Scheduling server	Priority	Processor
Control	10	Input acquisition	1	20		Acquisition task	1	Body controller
		Input interpretation	4	80		Acquisition task		
		Send/receive interpretation				Acquisition message task	1	CAN bus
		Speed setpoint		80		Setpoint task	2	Engine management
		Application condition		80		Control task	3	
		Basic function		80				
		Controller		80	500			
Failure management	10	Diagnosis	1	20		Failure management task	4	Body controller
		Send/receive diagnosis				Diagnosis message task	4	CAN bus
		Limp home		20	100	Control task	3	Engine management

- **Results:**

Processors	Slack (%)
Engine Management	71.20
Body Controller	205.25
CAN bus	629.14
Transactions	**Slack (%)**
Failure management	260.94
Control	89.84

T uise control transactions response times

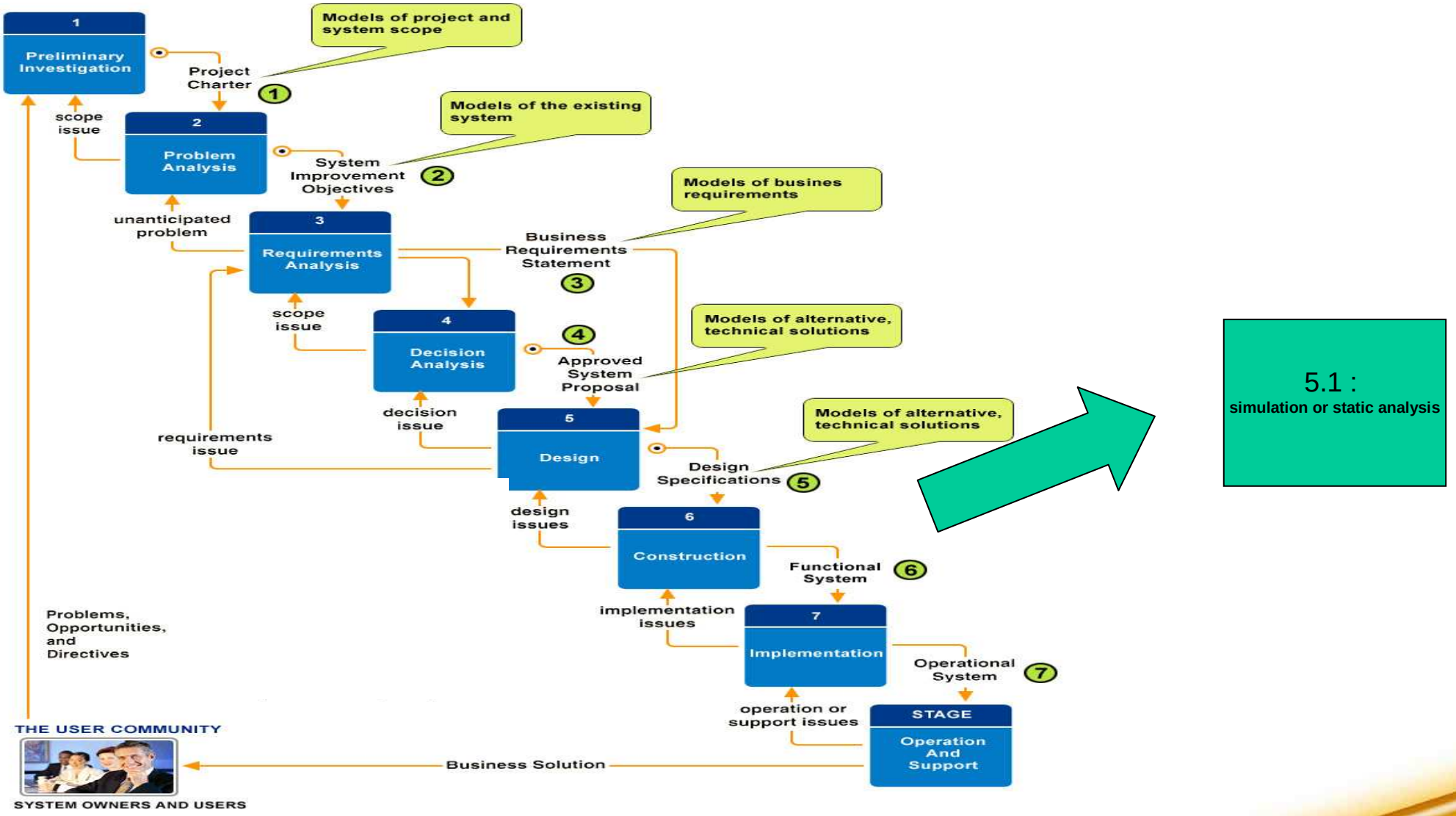
Transactions	Worst end-to-end response time (ms)	End-to-end deadline (ms)
Failure management	10.95	100
Control	59.29	500

Cruise control activities worst response times

Activity	Worst response time (ms)	Deadline (ms)
Input acquisition	4.84	20
Input interpretation	34.84	80
Speed setpoint	49.82	80
Controller	59.29	80
Diagnosis	1.52	20
Limp home	10.95	20

Personal perspective

MDD – is about the concept of correct by construction…

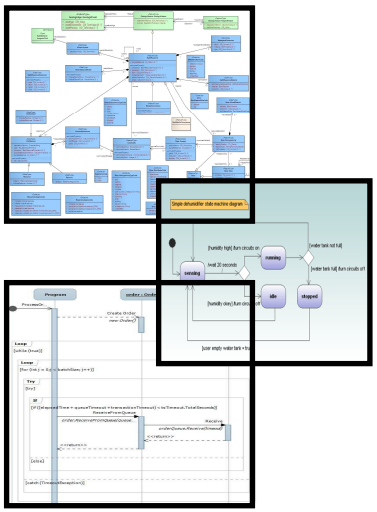


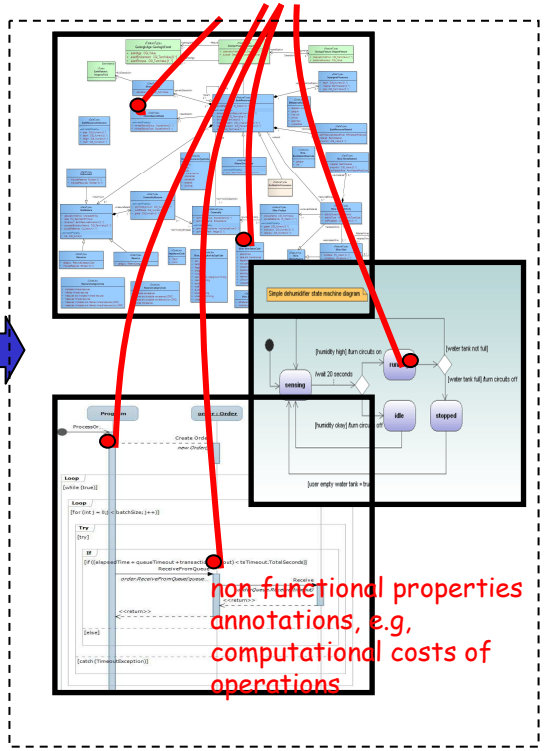
Personal Perspective

In the ideal world (for industries), this should be the path for quantitative analysis

UNIFIED MODELING LANGUAGE model

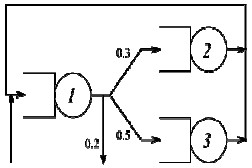
UML MARTE model





non functional properties annotations, e.g, computational costs of operations

Mathematical models

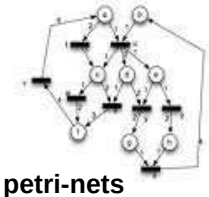


queueing network

Task Set:

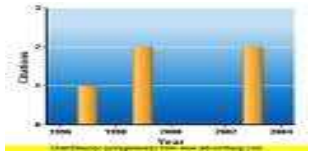
Task	ThrID	Type	Period(msec)	Priority
TaskA	01	T	0.010	High
TaskB	02	T	0.025	Medium
TaskC	03	T	0.040	Low

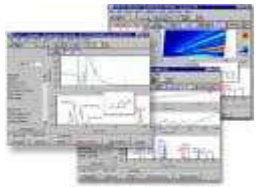
schedulability task model



petri-nets

Results





system design

5.1 :
simulation or static analysis

results /design issues

Is this transformation possible????

Personal Perspective

- **Methodologies for schedulability analysis should be defined at different level of abstractions (see AUTOSAR Timing views)**
- **The entry design model should specify a precise semantics, it is not a free model, e.g. for MAST**
  - only a subset of UML Activity elements should be used

Properties on the Activity diagram elements are:

- **P1** – The subset of activity diagram elements used is: {AcceptEventAction nodes, CallBehaviorAction nodes, FinalFlowNode, MergeNode, ControlFlow, ActivityPartition}. All other activity diagram elements are not used, and the following properties hold for this subset.
- **P2** - Events are modeled as UML *AcceptEventActions* that have UML *Triggers* referencing the Events. These events are modeled with UML *SignalEvents*.
- **P3** - Actions invoked in response to an event (connected to the accept event action with a control flow) are modeled as UML *CallBehaviorActions* calling UML *Behaviors* representing the functions.
- **P4** - The last action to be executed in response to an event is always followed by exactly one UML *FinalFlowNode*.

  - The structure of the Activity Diagram is also constrainted (e.g. no cycles)
- **A methodology is useless without a tool**
- **The tool allows the construction of only well-formed design models, i.e. design model expressing precise semantics**
- **Transformations towards the schedulability model should be transparent to the designer, this is important when presenting results (e.g. an artificial task added to handle a shared task between two transactions should be transparent to the designer)**
- **Domain-specific front-ends (e.g. AUTOSAR) should be also supported, MARTE can be a pivot language in this sense**

Thanks! & Questions