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
• AUTOSAR quick look

• AUTOSAR development methodology

• AUTOSAR Timing Extension

• Case study, enabling schedulability analysis on AUTOSAR models (from “Enabling Schedulability Analysis for AUTOSAR Systems” 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

**Layered Architecture**
- SWC (Software Component)
- RTE (Runtime Environment)
- Basic Software
- HW (Hardware)

**MDA Principles**
- VFB (Virtual Function Block)
- Level + System
- PIM (Platform Independent Model)
- PM (Platform Model)

**Development Methodology**
- Code!
- Timing Extension

**Metamodel**
- OS (Operating System)
- MEM (Memory)
- COM (Communication)
- I/O
- Complex Devices
- Basic Software

**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
For each runnable: 1 basic event chain

- **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

- 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.

#### Table 1. Timing information of the cruise control functions

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

**The WCETs used in this example were measured using internal methods and tools that for confidentiality reasons cannot be presented here.
Schedulability Analysis - MAST configuration

- Offset-based technique used, technique table:

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

- Results:

<table>
<thead>
<tr>
<th>Processors</th>
<th>Slack (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engine Management</td>
<td>71.20</td>
</tr>
<tr>
<td>Body Controller</td>
<td>205.25</td>
</tr>
<tr>
<td>CAN bus</td>
<td>629.14</td>
</tr>
<tr>
<td>Transactions</td>
<td>Slack (%)</td>
</tr>
<tr>
<td>Failure management</td>
<td>260.94</td>
</tr>
<tr>
<td>Control</td>
<td>89.84</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cruise control activities worst response times</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity</td>
</tr>
<tr>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>Input acquisition</td>
</tr>
<tr>
<td>Input interpretation</td>
</tr>
<tr>
<td>Speed setpoint</td>
</tr>
<tr>
<td>Controller</td>
</tr>
<tr>
<td>Diagnosis</td>
</tr>
<tr>
<td>Limp home</td>
</tr>
</tbody>
</table>
Personal perspective

MDD – is about the concept of correct by construction...

5.1: simulation or static analysis
Personal Perspective

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

System design
Is this transformation possible????

Results /design issues

5.1 : simulation or static analysis

Mathematical models
- queueing network
- schedulability task model
- petri-nets

Non-functional properties annotations, e.g., computational costs of operations
Personal Perspective

- **Methodologies for schedulability analysis** should be defined at different levels 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

  The structure of the Activity Diagram is also constrained (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