Optimum: a MARTE-based methodology for schedulability analysis at early design stages

Third International Workshop on UML and Formal Methods
November 16th, 2010, Shanghai, China

chokri.mraidha@cea.fr
sara.tucci@cea.fr
sebastien.gerard@cea.fr
Model-based development of real-time systems

- Should define a threading strategy satisfying timing constraints
- Manual phase: relies on the designer experience
- Timing validation (schedulability analysis) is needed

- Timing validation at early design stages
- Optimum: a methodology for schedulability guided design of real-time systems
Outline

- The Modeling Language
- The Optimum Methodology for Early Stage Schedulability Analysis
- Conclusions and Future Work
MARTE UML Profile

- Modeling and Analysis of Real-Time and Embedded systems
- OMG standard: version 1.0 standardized in July 2009
- Structured in sub-profiles covering RTE systems development aspects
- Model libraries of RTE specific types
- Textual language for value specifications (VSL)

Public website: www.omg-marte.org
MARTE Concepts to Perform Analysis

Analysis Context

Evaluate situation

Workload Behavior

Evaluate capacity

Uses

Resources Platform

Resource allocation

Scenarios

Load

Protected resources

exec.host

broker

comm.host
MARTE4Optimum

- MARTE subset for Optimum methodology (out of MARTE 158 stereotypes)
- Restriction of stereotypes applicability (w.r.t UML base elements)
  - Reduces complexity of methodological rules validation
  - Reduces complexity of automation support

<table>
<thead>
<tr>
<th>MARTE4Optimum stereotype</th>
<th>Covered analysis concept</th>
<th>MARTE4Optimum UML extensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alloc::Allocate</td>
<td>Resource allocation</td>
<td>Abstraction</td>
</tr>
<tr>
<td>Alloc::Allocated</td>
<td>Resource allocation</td>
<td>CallAction, Property</td>
</tr>
<tr>
<td>GRM::SchedulableResource</td>
<td>Platform abstraction: task</td>
<td>Property</td>
</tr>
<tr>
<td>GQAM::GaPlatformResources</td>
<td>Platform abstraction: container</td>
<td>Class</td>
</tr>
<tr>
<td>GQAM::GaWorkloadBehavior</td>
<td>Workload behavior</td>
<td>Activity</td>
</tr>
<tr>
<td>GQAM::GaWorkloadEvent</td>
<td>Load: event with arrival pattern</td>
<td>AcceptEventAction</td>
</tr>
<tr>
<td>SAM::SaAnalysisContext</td>
<td>Analysis context</td>
<td>Activity</td>
</tr>
<tr>
<td>SAM::SaEndToEndFlow</td>
<td>Scenario with deadline</td>
<td>ActivityPartition</td>
</tr>
<tr>
<td>SAM::SaExecHost</td>
<td>Platform abstraction: exec host</td>
<td>Property</td>
</tr>
<tr>
<td>SAM::SaSharedResource</td>
<td>Platform abstraction: shared res.</td>
<td>Property</td>
</tr>
<tr>
<td>SAM::SaStep</td>
<td>Workload behavior: action</td>
<td>CallAction</td>
</tr>
</tbody>
</table>
Agenda

- Outline
- The Modeling Language
- The Optimum Methodology for Early Stage Schedulability Analysis
- Conclusions and Future Work
Optimum Methodology: Inputs

- Requirements
- Functional Model
An Automotive Example

System-level functions structure

ElectronicBrakeControl

- dp: DataProcessing
  - computedData: Data
- alc: AntilockControl
  - inputData: Data
  - outputCommand: Command
    - absCommand: Command
      - absFaultType: Fault
- sd: SelfDiagnosis
  - absEnabled: Boolean
    - isActive: Boolean
  - faultType: Fault

acquiredData: SensorData
An Automotive Example

System end-to-end scenarios

```
ElectronicBrakeControlScenarios

acquisitionForAbs -> preProcessingBehavior
acquisitionForDiagnosis -> diagnosisBehavior
antiLockControlBehavior
```

Textual requirements

- The acquisition for ABS occurs every 60ms
- The acquisition for diagnosis can occur every 100ms at minimum
- …
Build the Workload Model

- Workload behavior specification
  - End-to-end flows identification
  - End-to-end deadlines
  - Event arrival pattern specification
  - Time budgets (w.r.t execution host)

- Platform resources abstraction
  - Execution hosts
End-to-end flow:
- Activity partition
- Deadline
- Event arrival pattern

- Allocation on exec host
- Time budget

Build the Workload Model

<<gaWorkLoadBehavior>> ElectronicBrakeControlWorkload

- Allocating on exec host
- Time budget
**Build the Schedulability Analysis Model**

- **Shared resources**
  - Shared steps identification
  - Duplication of shared steps

- **Task mapping**
  - One task per event (scenario-based)

- **Platform resources refinement**
  - Tasks
  - Shared resources
Build the Schedulability Analysis Model

```
«saAnalysisContext»
workload=EuristicBrakeControlWorkload
platform=SaResources

SaTaskMapping

«gaWorkloadEvent»
arrivalPattern=periodic(60, ms)

task1
acquisitionForAbs
DataProcessingBehavior
AntiLockBehavior

«saStep»
execTime=(15, ms)
host=HECU
sharedRes=[AntiLock]

«gaWorkloadEvent»
arrivalPattern=aperiodic(100, ms)

task2
acquisitionForDiagnosis
DiagnosisBehavior
AntiLockBehavior

«saStep»
execTime=(10, ms)
host=HECU

gaPlatformResources

SaResources

<<saExecHost>>
hecu:HECU

«saExecHost»
schedPolicy=FixedPriority

«schedulableResource»
schedParams=fp(20)

task1: Task

«schedulableResource»
schedParams=fp(10)

task2: Task

<<saSharedResource>>
AntiLock:SharedResource

«saSharedResource»
protectKind=PriorityCeling
```
Schedulability Analysis Evaluation

- Requirements
- Functional Model
- Build Workload Model
- Workload Model
- Generate SAM
- Workload Model Rules
- Sched Analysis Model Construction Rules
- Early Timing Tests
- Early Feasibility Evaluation
- Schedulability Analysis Model
- Schedulability Analysis Evaluation
- Optimum Sched Analysis Output Model
- Sched. Analysis Test

Optimum
Evaluation and Results

Schedulability analysis input model

<table>
<thead>
<tr>
<th>task</th>
<th>$e_i$</th>
<th>$T_i$</th>
<th>$C_i$</th>
<th>$P_i$</th>
<th>$B_i$</th>
<th>$D_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>task1</td>
<td>acquisitionForAbs</td>
<td>60</td>
<td>25</td>
<td>20</td>
<td>15</td>
<td>60</td>
</tr>
<tr>
<td>task2</td>
<td>acquisitionForDiagnosis</td>
<td>100</td>
<td>25</td>
<td>10</td>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

Schedulability analysis output results (back annotated)

<table>
<thead>
<tr>
<th>task</th>
<th>response time</th>
<th>isSched</th>
</tr>
</thead>
<tbody>
<tr>
<td>task1</td>
<td>40</td>
<td>true</td>
</tr>
<tr>
<td>task2</td>
<td>50</td>
<td>true</td>
</tr>
</tbody>
</table>

Back annotated schedulability analysis models provides guidance to build a schedulable design model

- Schedulable task mapping
- Shared resources identified
Agenda

- Outline
- The Modeling Language
- The Optimum Methodology for Early Stage Schedulability Analysis
- **Conclusions and Future Work**
Conclusions

• **MARTE-based methodology for schedulability guided design of RTS**
  - Reduces design cycle
  - Eliminates unfeasible design at early stage
  - Gives correct by construction real-time design patterns

• **Tooling support**
  - Optimum framework is integrated in Papyrus UML modeling tool
  - Provides wizards for capturing real-time properties in a simple way
  - Automatic construction of the schedulability analysis model
  - Bridges to external schedulability analysis tools (MAST and Rt-Druid)
  - Provides basic schedulability analysis tests (RMA for fixed priority)

• **Future work**
  - Enrich the set of task mapping construction
  - Enrich internal schedulability analysis algorithms
  - Real-time component models generation