Agenda

- Part 1
  - Introduction to MDD for RT/E systems & MARTE in a nutshell
- Part 2
  - Non-functional properties modeling
  - Outline of the Value Specification Language (VSL)
- Part 3
  - The timing model
- Part 4
  - A component model for RT/E
- Part 5
  - Platform modeling
- Part 6
  - Repetitive structure modeling
- Part 7
  - Model-based analysis for RT/E
- Part 8
  - MARTE and AADL
- Part 9
  - Conclusions
Goals in Non-Functional (or Quantitative) Analysis

It offers a mathematically-sound way to calculate NFPs of interest based on other available NFPs and the system behavior

- Different Goals for Evaluate & Verify System Architectures
  - Point evaluation of the output NFPs for a given operating point defined by input NFPs
  - Search over the parameter space for feasible or optimal solutions
  - Sensitivity analysis of some output results to some input parameters
  - Scalability analysis: how the system performs when the problem size or the system size grow.
MARTE Features for Quantitative Analysis

- **Improvements w.r.t. SPT**
  - Extend implementation and scheduling models
    - e.g. distributed systems, hierarchical scheduling
  - Extend the set of analysis techniques supported
    - e.g. offset-based techniques
  - Extend timing annotations expressiveness
    - Overheads (e.g. messages passing)
    - Response times (e.g. BCET & ACET)
    - Timing requirements (e.g. miss ratios and max. jitters)

- **New features w.r.t. SPT**
  - Support for sensitivity analysis
  - Improve modeling reuse and component-based design.
  - Support of the “Y-chart” approach: application vs. platform models
UML-Based Analysis Foundations

- **GQAM Profile factorizes common constructs and NFPs**
  - Stereotypes define “analysis” abstractions
    - workload events, scenarios,…
    - schedulable entities, shared resources, processing nodes, schedulers…
  - Stereotype attributes define pre-defined NFPs
    - e.g. event arrival patterns, end-to-end deadlines, wcet-bcet-acet,…

- **The analysis sub-profiles define model well-formedness rules**
  - It includes “constraints” to construct “analyzable” models, w.r.t…
  - ”Analysis Model Viewpoints” (e.g., schedulability analysis viewpoint)
  - Specialized constraints must be refined by technique-specific approaches

The MARTE analysis sub-profiles provide standard constructs to map UML models on well-established analysis techniques

→ MARTE “Foundations” and “GQAM” allow for extending to further techniques
GQAM: Dependencies and Architecture

General NFP types

Processing & Scheduling model

Schedulability analysis (timeliness)

Performance analysis (non-deterministic performance)
GQAM: Analysis Modeling Structure

Analysis Context

- evaluate situation
- evaluate capacity

Workload Behavior

- uses
- scenarios
- load

Resources Platform

- exec.host
- protected resources
- comm.host
- broker

resource allocation
Processing Schema for Analysis

- Tech. space for UML modeling
  - « profile » MARTE
  - UML2 editor
  - Annotated model
  - Result/Diagnostic model

- Tech. space for analysis
  - Domain model
  - Analysis tool
  - Analysis results

- Model converter
- Results converter
Schedulability Analysis

Provides the ability to evaluate time constraints and guarantee worst-case behavior of a system or particular piece of software

- Schedulability analysis offers:
  - Offline guarantees. E.g., worst-case latencies and worst-case resource usage.
  - At different development stages.
    - Early analysis: to detect potentially unfeasible real-time architectures.
    - Later analysis: to discover temporal-related faults, or to evaluate the impact of migrations (e.g., scheduling strategies).

- Provide answer to questions such as for example...
  - Will we miss any deadline if we switch a processor from a normal operation mode to a lower-consumption mode?
  - If yes, how can we modify task workloads for allowing our system to still work?
Three main analysis approaches for verify timeliness:

- Critical instant calculation
- Utilization bound test
- Response time calculation
SAM: Integration Different Approaches

- Classic RMA
- Extended RMA
- Holistic Approach

- Timed Automata with Tasks
- AEIOLTS

- RMA-Style

- Modular Analysis

- SAM (MARTE)

- MDA

- Object Oriented

- Compositional Analysis

- Active Object Semantic
- Event Priorities vs. Thread Priorities

- PIM, PSM, PDM

Other Sched. Analysis tools: Livedevices’ Real-Time Architect, CoMET from VaST, Vector’s CANAlyzer…
An “End-To-End Flow” is the basic workload unit to be evaluated by schedulability analysis tools.

→ An end-to-end flow refers to the entire causal set of steps triggered by one or more external workload events.

SAM: The Notion of End-To-End Flow

Step: basic behavioral unit (e.g., execution actions, call actions, messages,...)

Workload event: basic stimuli unit (e.g., timers, external occurrences, internal events,...)

Processing times (worst and best case)
Execution and communication steps may be causally related by one of the following precedence relations:

- **Sequential**: $a_i \rightarrow a_j$

- **Merge OR**: $a_i + a_j$

- **Join**: $a_i \times a_j$

- **Decision OR**: $a_i + a_j$

- **Fork**: $a_i \times a_j$
SAM: Workload Domain Metamodel (end-to-end)

SAM_Workload

- « dataType »
- « choiceType »
- ArrivalPattern
  - periodic: PeriodicPattern
  - aperiodic: AperiodicPattern
  - sporadic: SporadicPattern
  - burst: BurstPattern
  - irregular: IrregularPattern
  - closed: ClosedPattern
  - open: OpenPattern

GQAM_Workload:: WorkloadBehavior

- isSchedulable: NFP_Boolean
- schedulabilitySlack: NFP_Real
- endToEndTime: NFP_Duration
- endToEndDeadline: NFP_Duration
- endToEndStimuli 1..*
- endToEndResponse 1

EndToEndFlow

- GQAM_Workload:: WorkloadEvent
  - pattern: ArrivalPattern
  - inputStream
  - effect

SAM_Observers:: TimingObserver

- Timing

Predictions provided by analysis tools

Stimuli information

End-to-end response and deadline times

_predictions provided by analysis tools

Stimuli information

End-to-end response and deadline times
SAM: Workload Domain Metamodel (detailed behav.)

Processing unit (execution or communication)

Execution units accessing shared resources

SAM_Workload

GQAM_Workload:: BehaviorScenario
  hostDemand: NFP_Duration
  respTime: NFP_duration [*]
  utilization: NFP_Real [*]

GQAM_Workload:: Step
  behavior
  outputRel
  inputRel
  succes
  predec

GQAM_Workload:: PrecedenceRelation
  connectors

GQAM_Workload:: ReleaseStep
  (redefines concurRes)
  concurRes

GQAM_Workload:: AcquireStep
  0..1

GQAM_Workload:: CommunicationStep
  msgSize: NFP_DataSize

GQAM的工作负载:: CommunicationChannel

GRM::ResourceUsages:: ResourceUsage

gsamres:: Scheduling:: SchedulableResource

gsamres:: Resources:: SharedResource

SaStep
  deadline : NFP_Duration
  spareCapacity: NFP_Duration
  schedulabilitySlack: NFP_Real
  preemptedTime: NFP_Duration
  readyTime: NFP_Duration
  delayTime: NFP_Duration

SaCommunicationStep
  deadline : NFP_Duration
  spareCapacity: NFP_Duration
  schedulabilitySlack: NFP_Real

GRM::Scheduling:: SchedulableResource

Copyright © Thales, CEA and INRIA 2007 All rights reserved, commercial use strictly prohibited.
## SAM: Example of Stereotype Extensions Usage

<table>
<thead>
<tr>
<th>SAM Domain Model</th>
<th>SAM Stereotype</th>
<th>UML Metaclasses</th>
<th>Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>WorkloadBehavior</td>
<td>GaWorkloadBehavior</td>
<td>UML::Interactions::Fragments::CombinedFragments</td>
<td>Modeled in a high-level interaction</td>
</tr>
<tr>
<td>EndToEndFlow</td>
<td>SaEnd2EndFlow</td>
<td>UML::Interactions::Fragments::InteractionOperand</td>
<td>Modeled in a high-level interaction</td>
</tr>
<tr>
<td>WorkloadEvent</td>
<td>GaWorkloadEvent</td>
<td>UML::Interactions::BasicInteractions::Message</td>
<td>Modeled in a high-level interaction</td>
</tr>
<tr>
<td>BehaviorScenario</td>
<td>GaScenario</td>
<td>UML::Interactions::BasicInteractions::Interaction</td>
<td>Modeled as a low-level interaction nested within a higher-level interaction</td>
</tr>
<tr>
<td>Step</td>
<td>SaStep</td>
<td>UML::Interactions::BasicInteractions::Message</td>
<td>Messages in low-level interactions</td>
</tr>
<tr>
<td>CommunicationStep</td>
<td>SaCommStep</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ReleaseStep</td>
<td>GaRelStep</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AcquireStep</td>
<td>GaAcqStep</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SAM: Examples of Behavior Annotations

**SAM: Examples of Behavior Annotations**

![Diagram showing behavior scenario, async. message transmission, syncr. execution message, lock and unlock of a shared resources]
SAM: Example of Precedence Relations Annotation

**SAM: Example of Precedence Relations Annotation**

![Diagram showing precedence relations]

**Sequencial**

- **a** \rightarrow **a**

**Fork**

- **a** \rightarrow **a**
- **a** \rightarrow **a**

**Join**

- **a** \rightarrow **a**
SAM: Example of Workload Annotations

end-to-end flow

triggering events (async. messages)

use of an interaction

concurrent fragments
SAM: Resources Concepts

- Provide additional (analysis-specific) annotations to annotate resources platform models

**Domain**

Processing resources (execution and communication)

**SCH**

- Scheduler

- Shared Resources

**ShaR**

- Shared resource

**Chn**

- Channels

**τ**

- Task

**ε**

- Execution host

**κ**

- Communication host

**ς**

- Shared resource

**τ_1, τ_2, τ_3**

- Schedulable resources (e.g., threads, channels, ...)

**τ_1, τ_2, τ_3, τ_4, τ_5**

- Scheduler

**Sch**

- Scheduler

**ShaR_1**

- Shared resource
SAM: Resources Domain Metamodel
# SAM: Examples of the Stereotypes Usage

## SAM Domain Model
- **ResourcesPlatform**
- **SaExecutionHost**
- **SaCommunicationHost**
- **GRM::Scheduler**
- **GRM::SchedulableResource**
- **SaCommChannel**

## SAM Stereotype
- GaResourcesPlatform
- SaExecHost
- SaCommHost
- Scheduler
- SchedulableRes
- SaCommChannel

## UML Metaclasses
- UML::StructuredClasses::StructuredClass
- UML::StructuredClasses::Property
- UML::StructuredClasses::Property

## Context
- Main container of resources
- Parts of the resources platform
- Parts of processing resources
SAM: Example of Resources Stereotype Usage

Concurrence resources as nested parts

Scheduler as UML part

Resources platform under analysis

Processing Resources as UML parts

« gaResourcesPlatform »
RP1

« saCommHost »
commH1

« saCommChannel »
ch1 : CH

« saCommChannel »
ch2 : CH

« saExecHost »
execH1

« schedulableRes »
schR1 : SCHR

« schedulableRes »
schR2 : SCHR

« scheduler »
sch1 : SCH

Resources platform under analysis

Processing Resources as UML parts
SAM: Analysis Context concepts

- An analysis context is the root concept used to collect relevant quantitative information for performing a specific analysis scenario.

- An analysis context integrates workload behavior models and resources platform models.
SAM: Analysis Context Domain Metamodel

SAM

GQAM:: AnalysisContext

GQAM_Workload:: WorkloadBehavior

GQAM_Resources:: ResourcesPlatform

SaAnalysisContext

« enumeration » OptimalityCriterionKind

isSchedulable: NFP_Boolean
optimalityCriterion: optimalityCriterionKind

meetHardDeadlines
minimizeMissedDeadlines
minimizeMeanTardiness
undef
other
SAM: Example of Analysis Context Stereotype Applic.

Interaction representing an analysis context

Allocation to Schedulable resources (link to platform Resources)
Example of Global Development Process

**Requirements Model**
- Use cases
- Scenarios

**Logical Model**
- Structure
- Interactions
- Behaviour

**Prototype Model**
- Components
- Activities

**Platform Model**
- Components
- Hardware
- Software

**Schedule & Performance Model**
- Workload
- Allocation
- Behaviour

**Profile**
- SAM
- HRM&SRM

**QoS**

**WCET Calculation**
General Procedure to Use the SAM Profile

1. Design Models
2. Annotated Behavior Models
3. Workload Behavior Models (PIM)
4. Specify Parameterized Analysis Context Model
5. Resources Platform Models (PDM)
6. Annotate Resources Models

Non-functional values for specific analysis contexts

Analysis Tools

Determine Desired NFPs of Interest (given and predicted parameters)
Example: A Teleoperated Robot

ClassesView_TeleoperatedRobot

- **DisplayData**
  - data: Integer [*]
  - read (): Data
  - write (D: Data)

- **DisplayRefresher**
  - updateDisplay ()
  - updateGraphics ()

- **CommandInterpreter**
  - processEvent ()
  - planTrajectory ()

- **StationCommunication**
  - sendCommand (C: Command)
  - awaitStatus (): Status

- **ControllerCommunication**
  - sendStatus (S: Status)
  - awaitCommand (): Command

- **Reporter**
  - report ()

- **ServosData**
  - Data: Integer [*]
  - get (): Data
  - set (D: Data)

- **ServosController**
  - controlServos ()
  - controlAlgorithms ()
  - doControl ()

DeploymentView_TeleoperatedRobot

- **Station**
- **CAN_Bus**
- **Controller**
- **RobotArm**
Example of Annotated Scenario with SAM

Step (exec. time)

Shared Resource

« gaScenario » Report

:ContrClock :Reporter

« saSharedRes » ServosData
{ protectKind= priorityCeiling,
  priority=16 }

:ControllerComm :CANBus

« StationComm »

:DisplayRefresher

« SaSharedRes » DisplayData
{ protectKind= priorityCeiling,
  priority=31 }

« gaAcqStep » lock ()
{ execT= (1.1, 1.22, ms) }

« gaRelStep » unlock ()
{ execT= (0.004, 0.02, ms) }

« saStep » sendStatus ()
{ execT= (0.031, 0.031, ms) }

« saCommStep »transmit ()
{ msgSize=(100, kB) }

« saCommStep »transmit ()
{ msgSize=(100, kB) }

« saCommStep »transmit ()
{ msgSize=(100, kB) }

<saStep> transmitCommand ()
{ execT= (2.56, 2.56, ms) }

<saStep> transmitDisplay ()
{ execT= (0.24, 0.4, ms) }

<saStep> read ()

<saStep> write ()

<saStep> updatesGraphics ()
{ execT= (5, 10, ms) }

Example of Annotated Scenario with SAM
Example of Annotated Resources Model with SAM

Threads owned by the processing resource
Example of Analysis Context Model

```
«saAnalysisContext » {isSched= ($isSch, calc) } sd AnalysisScenario01

End To End Flows (end2end deadlines and predicted times)
```

```
Workload Behavior
```

```
Scenario (response times, hosts utilization…)
```

```
Workload Events (arrival patterns)
```

```
Example of Analysis Context Model
```

```
User
```

```
End To End Flows (end2end deadlines and predicted times)
```

```
Workload Events (arrival patterns)
```

```
Scenario (response times, hosts utilization…)
```

```
Workload Behavior
```

```
Example of Analysis Context Model
```
Example of Parametric Analysis Context

TeleoperatedRobotSAM

Schedulability: TeleoperatedRobotSAM

isSched_System= (true, $v0, calc)
wct_Report= (5, ms, determ)
procRate_CAN= (1, determ)
period_Report= (30, ms, determ)

Sensitivity Analysis: TeleoperatedRobotSAM

isSched_System= (true, req)
wct_Report= (50, $v1, ms, max, calc)
procRate_CAN= (0.2, $v2, min, calc)
period_Report= (10, $v3, ms, min, calc)

Context under Analysis

Context-specific variables

Instance of a WorkloadBehavior model

Simple Schedulability Analysis context

Sensitivity Analysis context
Current Implementations supporting MARTE

- Full MARTE Profile & Libraries for Eclipse UML2
- VSL edition assistant and type checker as a Eclipse plug-in for the UML Papyrus tool and RSA 7.0

On-going work:

- Eclipse plug-ins to transform UML models annotated with the SAM profile to input files of MAST, SymTA/S, Cheddar and RapidRMA tools

MARTE Open Source Implementation in

UML Papyrus:  www.papyrusuml.org
IBM RSA:  www.omgmar.te.org
Conclusions on MARTE’s Analysis

- **Industrial Use of V&V can benefits from MDE**
  - Analysis task must be cohesively integrated with Design tasks
  - Application of individual analysis techniques should be regarded as an essential part of an integrated V&V methodology

- **Methodological support is still under way:**
  - Complex analysis scenarios for Interface-Based Design, Multiobjective Design Space Exploration…
  - Means to manage NFP measurement models
  - Methods to map/transform MoCCs into analysis models