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
Time in MARTE Overview

- **SPT, UML 2 and Time**
  - UML::CommonBehaviors::SimpleTime

- **the MARTE Time domain view**
  - a.k.a. the MARTE Time meta-model
  - Concepts and relationships

- **the MARTE Time sub-profile**
  - a.k.a. UML view

- **Usage of the Time sub-profile**
UML profile for Schedulability, Performance, and Time (SPT)

- OMG UML profile formal/05-01-02 (v1.1)
- Based on UML 1.4
- Dealing with time and resources
- Quantitative time information
- Concepts
  - Instant, duration
  - Event bound to time, stimuli
- Timing mechanisms & services

To be aligned to UML 2

Metric time
UML::CommonBehaviors::SimpleTime

- **UML2 adds new metaclasses to represent**
  - Time
  - Duration
  - Observation (of time passing)
  - Some forms of time constraints

- **Simple (even simplistic) model of time**

- **Advice:** *Use a more sophisticated model of time provided by an appropriate profile*, if needed. [UML superstructure, chapter 13]

  e.g., MARTE
SimpleTime::TimeEvent

CommonBehaviors::
Communications::
Event

TimeEvent

0..1
when
1

isRelative: Boolean

Classes::
Kernel::
ValueSpecification

Absolute/relative
specification

Time specification
TimeEvent – usage (1)

UML state machine = behavior

Specification of a time-trigger

Informal semantics

stm blinker

On -> Off after 10

User
Meaning of “after 10”

Simple annotation $\rightarrow$ complex implied structure

Metaclasses involved in the modeling of a transition triggered by a `TimeEvent`
SimpleTime::Observation

Classes::
Kernel::
NamedElement

TimeObservation

firstEvent: Boolean

1

event

1..2

event

DurationObservation

firstEvent: Boolean[0..2]
Example of sequence diagram

MOS stands for MessageOccurrenceSpecification

Note that red and blue annotations are not part of the UML notation.
Instance model of the time constraint: receive CardOut in \{t .. t+d\}

Simple annotation → complex implied structure
Overview

- SPT, UML 2 and Time
  - UML::CommonBehaviors::SimpleTime

- the MARTE Time domain view
  - a.k.a. the MARTE Time meta-model
  - Concepts and relationships

- the MARTE Time sub-profile
  - a.k.a. UML view

- Usage of the Time sub-profile
Concepts in MARTE::Time (1/2)

- **Time structure** =
  set of time bases + time structure relations
  → Partially ordered set of instants

- **Access to time** = Clock

- **Principle**: associate Clocks with model elements
  - Behavioral elements → TimedEvent, TimedProcessing
  - Constraints → TimedConstraint
  - Data types and values → TimedValue
Main concepts introduced in Time modeling

1. **TimeValueSpecification**

2. **TimeStructure**
   - Concepts:
     - Time bases
     - Multiple Time Bases
     - Instants
     - Time structure relations

3. **TimeAccess**
   - Concepts:
     - Clocks
     - Logical clocks
     - Chronometric clocks
     - Current time

4. **TimeUsage**
   - Concepts:
     - Timed elements
     - Timed events
     - Timed actions
     - Timed constraints

**Not a UML diagram!**
MultipleTimeBase = set of TimeBases + Hierarchy + Constraints

TimeBase = oset of instants

TimeStructure = Relationships over instants of different TBs

Relationships over TBs
Access to Time: Clock

Units associated with a clock

TimeBase

1

Clock

nature: TimeNatureKind
resolution: Real=1.0
currentTime Real
maximalValue: Real[0..1]

Unit

acceptedUnits
timeBase

Event

clockTick

defaultUnit{subsets acceptedUnits}

Event occurring at each clock ticking

Access to the time structure
Chronometric/Logical Clocks

Two kinds of clocks

- **Implicit reference to physical time**
- **Possible reference to a repetitive event**

- NFPs measured against a reference clock

**ChronometricClock**
- referenceClock: 0..1
- standard: TimeStandardKind [0..1]
- stability: Real [0..1]
- offset: DurationValue [0..1]
- skew: Real [0..1]
- drift: Real [0..1]

**LogicalClock**
- definingEvent: 0..1

**Event**

**PhysicalTime**
Time Values

A TimeValue has a unit (default= clock unit)

A TimeValue must reference a clock

Instant/Duration two distinct concepts
The unifying concept: a TimedElement = a ModelElement + a Clock
Timed Entities: TimedEvent

- **occurrences**
  - CoreElements:
    - Causality:
    - RunTimeContext:
    - EventOccurrence
  - SimultaneousOccurrenceSet
  - Provision for simultaneity

- **events**
  - TimedElement
    - TimedEventOccurrence
      - InstantValue
        - 0..1 at
        - 1..* occSet
  - TimedEvent
    - isRelative: Boolean
    - repetition: Integer [0..1]
  - Event
    - TimeValueSpecification
    - DurationValueSpecification
    - Facility to specify multiple occurrences

Provision for simultaneity
Timed Entities: TimedProcessing

CoreElements: Causality: CommonBehavior: Behavior

TimedBehavior

CoreElements: Causality: Communication: Request

TimedMessage

CoreElements: Causality: CommonBehavior: Action

TimedAction

TimedProcessing

DurationValueSpecification

Event

TimedElement

Delay

0..1 duration

start 0..1 finish 0..1
Timed Entities: TimedObservation

- **TimedElement**
  - **TimedObservation**
    - **TimedInstantObservation**
      - obsKind: EventKind[0..1]
    - **TimedDurationObservation**
      - obsKind: EventKind[0..2]
    - **CoreElements**: Causality: RunTimeContext: EventOccurrence
    - **CoreElements**: Causality: RunTimeContext: Communication: Request
    - **<<enumeration>>** EventKind
      - start
      - finish
      - send
      - receive
      - consume
See:

Overview

- **SPT, UML 2 and Time**
  - UML::CommonBehaviors::SimpleTime

- **the MARTE Time domain view**
  - a.k.a. the MARTE Time meta-model
  - Concepts and relationships

- **the MARTE Time sub-profile**
  - a.k.a. UML view

- **Usage of the Time sub-profile**
Providing extensions to UML

- Through a UML profile
  - New Stereotypes

- Facilities
  - Model libraries
  - Dedicated languages (especially for expressions)
Two other sub-profiles of MARTE

- « profile » NFPs
  - <<import>>
- « profile » Time
  - « modelLibrary » TimeTypesLibrary
  - « apply >>
  - <<import>>
- « profile » VSL::DataTypes
  - « import >>

- « modelLibrary » TimeLibrary
  - « import >>

User’s model library
Central stereotypes: ClockType & Clock

**Chronometric clock** → "physical" time; units $\in \{s, ms, us, \ldots\}$

**Logical clock** → any repetitive event; units $\in \{\text{tick}\} \cup \text{PhysicalUnits}$

<table>
<thead>
<tr>
<th>nature</th>
<th>discrete</th>
<th>dense</th>
</tr>
</thead>
<tbody>
<tr>
<td>isLogical</td>
<td></td>
<td></td>
</tr>
<tr>
<td>true</td>
<td>Logical clock</td>
<td>Not used</td>
</tr>
<tr>
<td>false</td>
<td>Chronometric clock</td>
<td></td>
</tr>
<tr>
<td></td>
<td>discrete</td>
<td>dense</td>
</tr>
</tbody>
</table>

- **Accepted units**
- **Default unit**

Stereotype properties:
- Special semantics
- + optional
- - set of properties
- - set of operations
Notice that this abstract stereotype has no base metaclass
TimedValueSpecification

```
TimedValueSpecification

< stereotype >
TimedElement

< stereotype >
TimedValueSpecification

interpretation: TimeInterpretationKind[0..1]

< metaclass >
UML::Classes::Kernel::ValueSpecification

• either Instant
• or Duration
```
Extending the TimeEvent metaclass of SimpleTime

TimeEvent

TimedEvent

«stereotype»

0..1

every

0..1

repetition: Integer [0..1]

ValueSpecification

CommonBehaviors:: SimpleTime:: TimeEvent

UML::Classes::Kernel:: ValueSpecification

UML::::

«metaclass»

Extending the TimeEvent metaclass of SimpleTime

TimedEvent

«stereotype»

TimedElement

TimedElement

«stereotype»

0..1

0..1

every

repetition: Integer [0..1]

ValueSpecification

CommonBehaviors:: SimpleTime:: TimeEvent

UML::Classes::Kernel:: ValueSpecification

UML::::

«metaclass»

Extending the TimeEvent metaclass of SimpleTime

TimedEvent

«stereotype»

TimedElement

TimedElement

«stereotype»
```
<metaclass>
UML::Actions: Action
</metaclass>

<metaclass>
UML::CommonBehaviors: Behavior
</metaclass>

<metaclass>
UML::Interactions: BasicInteractions: Message
</metaclass>

<metaclass>
UML::Classes: Kernel: ValueSpecification
</metaclass>

<stereotype>
TimedProcessing
</stereotype>

<stereotype>
TimedElement
</stereotype>

start
0..1
finish
0..1
duration
0..1
0..1
```
Extending the SimpleTime Observation metaclasses
« stereotype »
TimedConstraint

interpretation: TimeInterpretationKind

« stereotype »
ClockConstraint

« stereotype »
NFPs::
NfpConstraint

« stereotype »
TimedElement
Time-related GRM stereotypes

Sterotypes defined in the Generic Resource Modeling sub-profile

- « stereotype » Resource
- « stereotype » Time::ClockType
- « stereotype » TimingResource
- « stereotype » TimerResource
- « stereotype » ClockResource

Resources for time management
## Time-related libraries: TimeTypesLibrary

### TimeNatureKind
- discrete
- dense

### TimeStandardKind
- TAI
- UTC
- Local
- ...
- GPS

### TimeInterpretationKind
- duration
- instant

### EventKind
- start
- finish
- send
- receive
- consume

---

**UML**:

- « modelLibrary »
- TimeTypesLibrary

---

**User**
**Time-related libraries: TimeLibrary**

- **TimeLibrary**
  - **enumeration** TimeUnitKind
    - unit: s
    - ms (baseUnit=s, convFactor=0.001)
    - us (baseUnit=ms, convFactor=0.001)
    - ns (baseUnit=us, convFactor=0.001)
    - min (baseUnit=s, convFactor=60)
    - hrs (baseUnit=min, convFactor=60)
    - dys (baseUnit=hrs, convFactor=24)
  - **tupleType** TimedValueType
    - value: Real
    - expr: ClockedValueSpecification
    - unit: TUK
    - onClock: String
  - **primitive** ClockedValueSpecification
    - <<clockType>>
      - (nature = dense, wireType = TimeUnitKind, onClock = IdealClock)
        - IdealClock
          - currentTime(): Real

Two usual sets of Time Units

Model of ideal "physical time"
Time-related NFP types

- **NFP_CommonType**
  - expr: VSL_Expression
  - source: SourceKind
  - statQ: StatisticalQualifierKind
  - dir: DirectionKind

- **NFP_Duration**
  - value: Real

- **NFP_Real**
  - value: Real

- **NFP_DateTime**
  - value: DateTime

- **NFP_Frequency**
  - unit: FrequencyUnitKind
  - precision: Real

**Time-related types. Often used.**
Expressing time values with EXPLICIT clocks

- **ClockedValueSpecification**
  - **ValueSpecification**
    - **InstantValueSpecification**
      - **DurationValueSpecification**
        - **InstantIntervalSpecification**
          - **DurationIntervalSpecification**
            - **Translation**
              - **Scaling**
                - **InstantExpression**
                  - **DurationExpression**
Time specific languages: VSL Time Expressions

Expressing time values with EXPLICIT clocks

Extended capabilities:
- Occurrence index
- Time intervals
- Jitter

```
VSL::TimeExpressions

ValueSpecification
  conditionExpr 0..2
  occurIndexExpr 0..1

TimeExpression
  ValueSpecification
  DurationExpression

InstantExpression
  InstantInterval

JitterExpression

Observation

DurationInterval

CompositeValues: IntervalSpecification

Extended capabilities:
- Occurrence index
- Time intervals
- Jitter

Expressing time values with EXPLICIT clocks
```
Time Values: Concrete syntax

Examples of Clocked value expressions

- **Simple time values**
  
  \[(\text{value}=3.5, \text{unit}=\text{ms}, \text{onClock}=\text{idealClk}');\]
  
  3.5 ms on \text{idealClk};

- **Homogeneous expressions**
  
  \[(\text{value}=1.5, \text{unit}=\text{ms}, \text{onClock}=\text{idealClk}') +
  (\text{value}=150, \text{unit}=\text{us}, \text{onClock}=\text{idealClk}');\]
  
  \[\rightarrow (\text{value}=1650, \text{unit}=\text{us}, \text{onClock}=\text{idealClk}');\]

- **Heterogeneous expressions**
  
  \[\text{min} \ (15 \text{ tick on prClk}, 5 \text{ ms on idealClk});\]

- **Additional capabilities with VSL**
  
  - Occurrence number, jitter,…
  - but implicitly on \text{idealClk}
Time specific languages: VSL Time Constraints

**t0[i]** denotes the i-th occurrence of

**t0**: observation of the message: start

- **t0** is periodic, period 100ms with a jitter less than 5ms

**Constraints**:
- \( \text{constr1} = \{ (t0[i+1]-t0[i]) \geq (100, \text{ms}) \} \)
- \( \text{constr2} = \{ t3 < t2 + (30, \text{ms}) \} \)

**Example**: Data Acquisition

- **Controller**
  - `start()` \{ jitter(t0) < (5, ms) \}
  - `acquire()` \{ d1 \leq (1, ms) \}
  - \{ t1..t1+(8,ms) \}

- **Sensor**
  - `ack()`
  - `sendData(data)` \{ ([0,ms..(10,ms)] \}

- \&d1

\( t0 \) denotes the i-th occurrence of **t0**
Time specific languages: Clock Constraint Specification

Expression of Clock dependencies
Clock Constraint Specification

Each relation has a mathematical specification

Pre-defined Clock Constraints
Overview

- SPT, UML 2 and Time
  - UML::CommonBehaviors::SimpleTime

- the MARTE Time domain view
  - a.k.a. the MARTE Time meta-model
  - Concepts and relationships

- the MARTE Time sub-profile
  - a.k.a. UML view

- Usage of the Time sub-profile
How to specify chronometric clocks

Chronometric

currentTime(): Real

resolution: Real {readOnly}

currentTime( ): Real

IdealsClock

currentTime( ): Real

« clockType »

{ nature = dense, unitType = TimeUnitKind,
  resAttr = resolution, getTime = currentTime }

Imported from MARTE:TimeLibrary

An user’s defined ClockType

« clockType »

{ nature = discrete, unitType = TimeUnitKind,
  resAttr = resolution, getTime = currentTime }

Chronometric

{ nature = dense, unitType = TimeUnitKind,
  getTime = currentTime }

IdealClock
Specifying NFP of (non ideal) chronometric clocks

There are two instances:

- `cc1`: Chronometric clock
  - `resolution = 0.01`

- `cc2`: Chronometric clock
  - `resolution = 0.01`

For all `k`, there exists `d` such that:

\[ c[d+10*(k-1)] < cc1[k] < c[d+10*k] \]

\[ 0 < cc1[k+1] - cc1[k] < 20 \text{ ms} \]

A non-functional property: **Stability**

\[ 10 - 0.001 \leq cc1[k+1] - cc1[k] \leq 10 + 0.001 \text{ in ms} \]

Another non-functional property: **Offset**

- `c`: Local ideal discrete clock – 1kHz

- `idealClk`: Ideal clock

- `ApplicationTimeDomain`: Timed Domain

- `Chronometric`: Clock

- `discretizedBy 0.001`
Logical Clocks (1/4)

How to specify logical clocks:
1) Start with a standard UML class diagram

Explicit model elements not usual in Class Diagrams

Period of the PID controller: uses a NFP-type

A Voltage-Scaling processor. Assume 2 frequencies for simplicity
Logical Clocks (2/4)

2) Apply MARTE stereotypes

The pid code is triggered by tev and takes 45 cycles of Processor.

Event tev is periodic on idealClock, the period is the value of the controller's attribute.

The class Processor is stereotyped by ClockType.
3) Instantiate user’s model elements

An instance of the system with an instance of Processor supporting two instances of Controller

Each controller instance has its own period
Logical Clocks (4/4)

4) Introduce clock (by stereotyping)

This instance of Processor is used as a logical clock

This clock constraint binds processor clock cycle to physical time, taking account of the power mode
Automotive application

For ignition and injection, the position of the camshaft or the crankshaft is a “natural” reference frame for events and behaviors.

=> Define logical clocks dealing with angular positions.

Note the possible use of an OCL rule

(1) define a set of units

(2) define a clock type

(3) optional define the labeling function

(4) instantiate clocks
Example of usage of an “AngleClock”

Stereotyped State Machine. Makes reference to a Clock

Reference to a (logical) clock, the unit of which is °CAM (elsewhere defined)

Semantics: 90 °CAM after entering state Compression leave this state and enter state Combustion
Another example of usage of an “AngleClock”: Enhanced timing diagram used in specification
Combining logical clocks:
ck is an AngleClock used to specify the ignition of a cylinder
c is the clock used to specify ignitions in a 4-cylinder engine

These values are not imposed, this is an arbitrary (but rather natural) choice. Any clock finer than c1,c2,c3,c4 is allowed.