UML MARTE

www.omgmarte.org

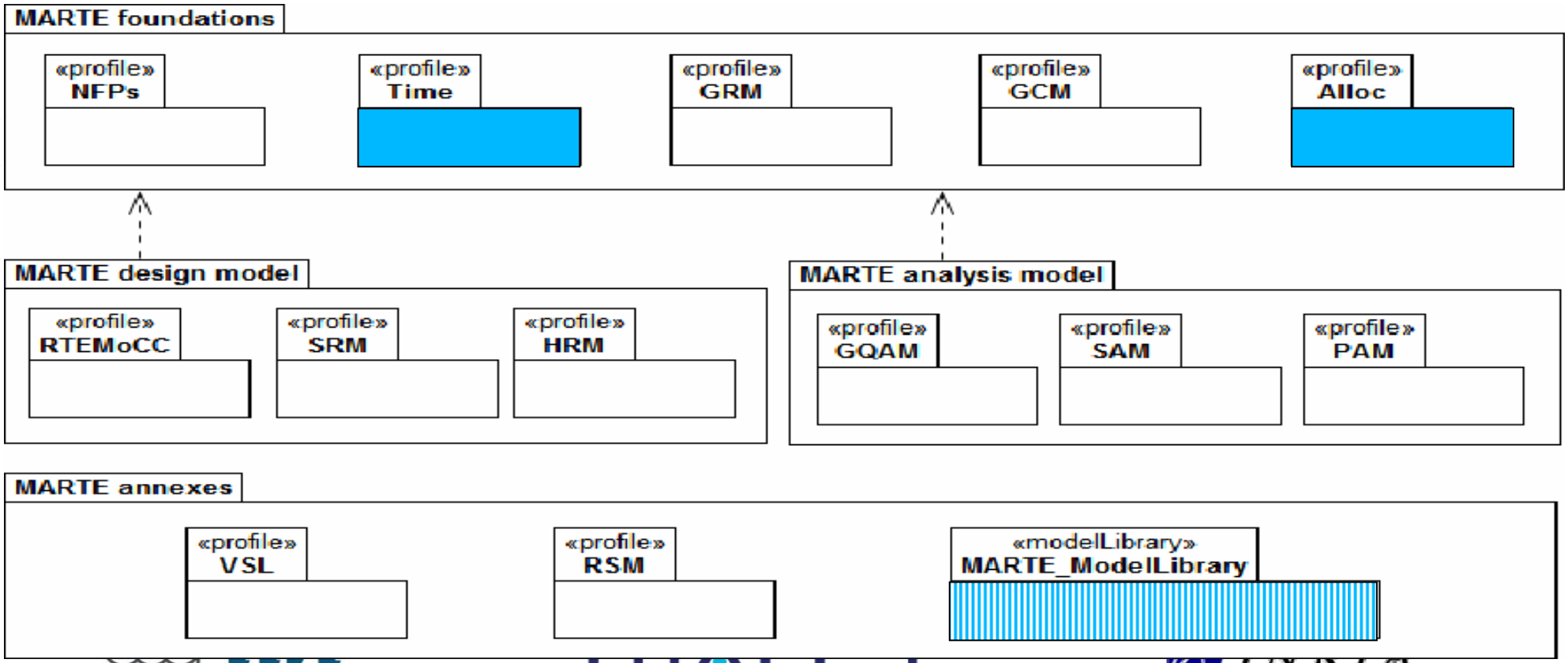
MARTE Time Model

AOSTE

F. Mallet – 06/03/08

cea list THALES INRIA

- **OMG UML2 Profile for Modeling and Analysis of Real-Time and Embedded systems**
  - OMG Adopted Specification (ptc/07-08-04) => FTF



Introduction

- **From antiquity to nowadays**
  - Several models of time (philosophy, religion, science)
- **In Real-time and Embedded Systems**
  - Multiple time domains (multicore, NoC, low power)
  - Highly heterogeneous: unify multiple models in a single framework (MoCC)
  - ⇨ Theory of tag systems [Lee & Sangiovanni-Vincentelli)
- **In MARTE**
  - Simple mechanisms to deal with “classical” time issues
  - Advanced mechanisms to build your own MoCC

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**

- OMG UML profile formal/05-01-02 (v1.1)
- Based on UML 1.4 — To be aligned to UML 2
- Dealing with time and resources
- Quantitative time information — Metric time
- Concepts
  - Instant, duration
  - Event bound to time, stimuli
- Timing mechanisms & services

UML::CommonBehaviors::SimpleTime

- **UML2 adds new metaclasses to represent**
  - Time (TimeEvent)
  - Duration
  - Observation: TimeObservation and DurationObservation
  - 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

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

- **Time structure =**

  set of time bases + time structure relations

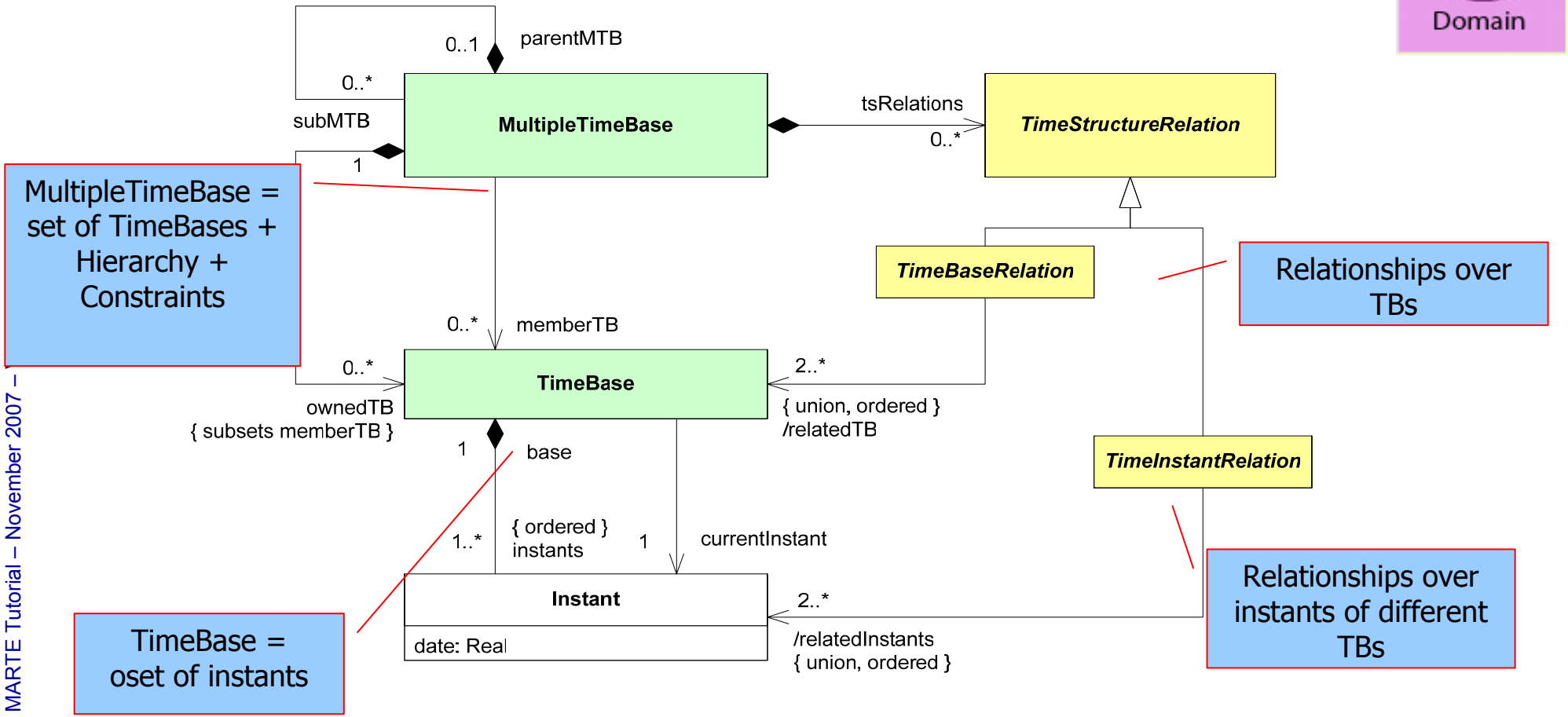
  → Partially ordered set of instants

- **Access to time = Clock**

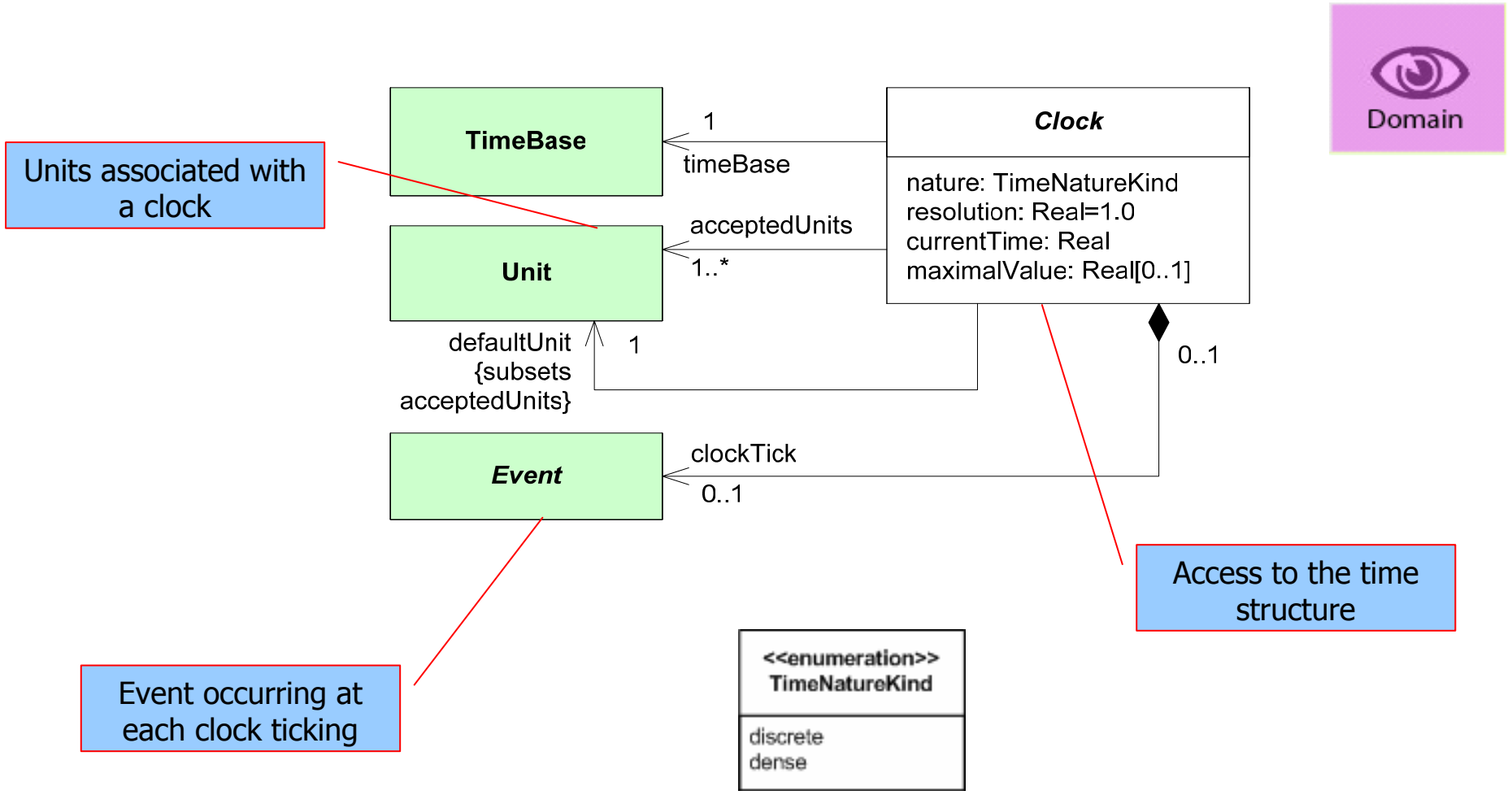
- **Principle: associate Clocks with model elements**
  - Behavioral elements → TimedEvent, TimedProcessing
  - Constraints → TimedConstraint
  - Data types and values → TimedValue

Time Structure

Domain



Access to Time: Clock

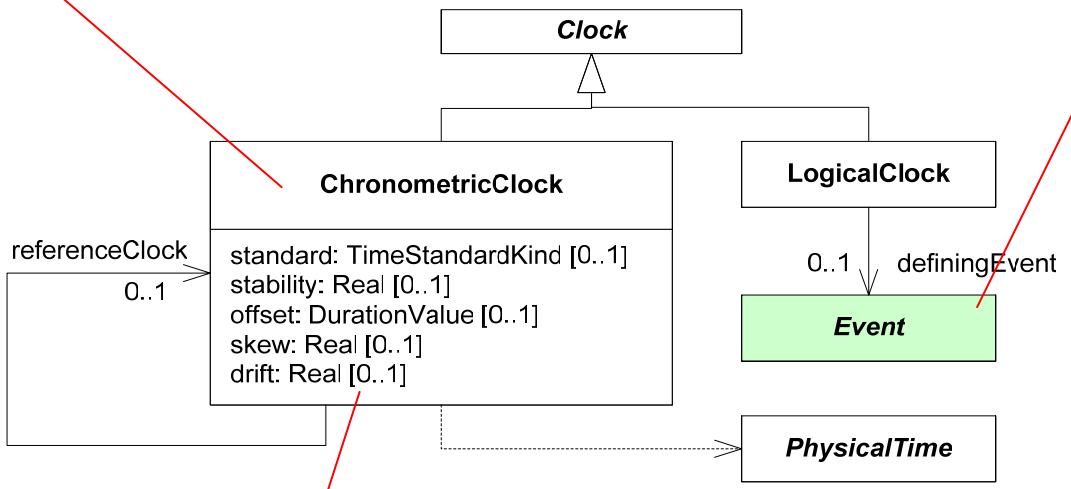


Chronometric/Logical Clocks

Two kinds of clocks

Domain

Implicit reference to physical time

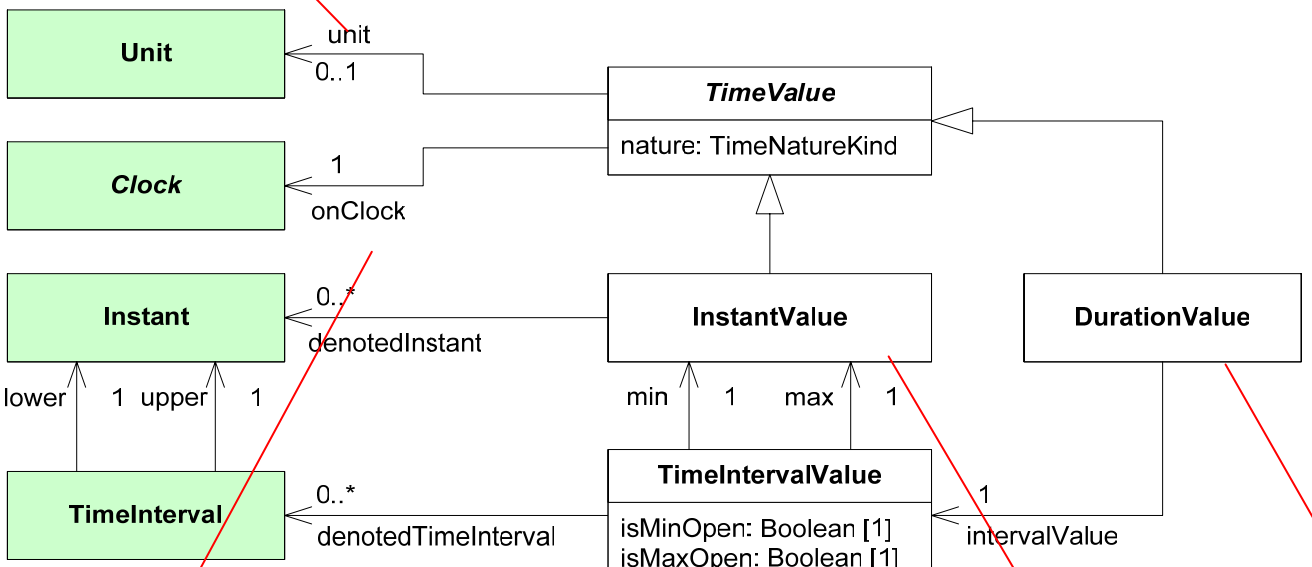


Possible reference to a repetitive event

NFPs measured against a reference clock

Time Values

A TimeValue has a unit (default= clock unit)



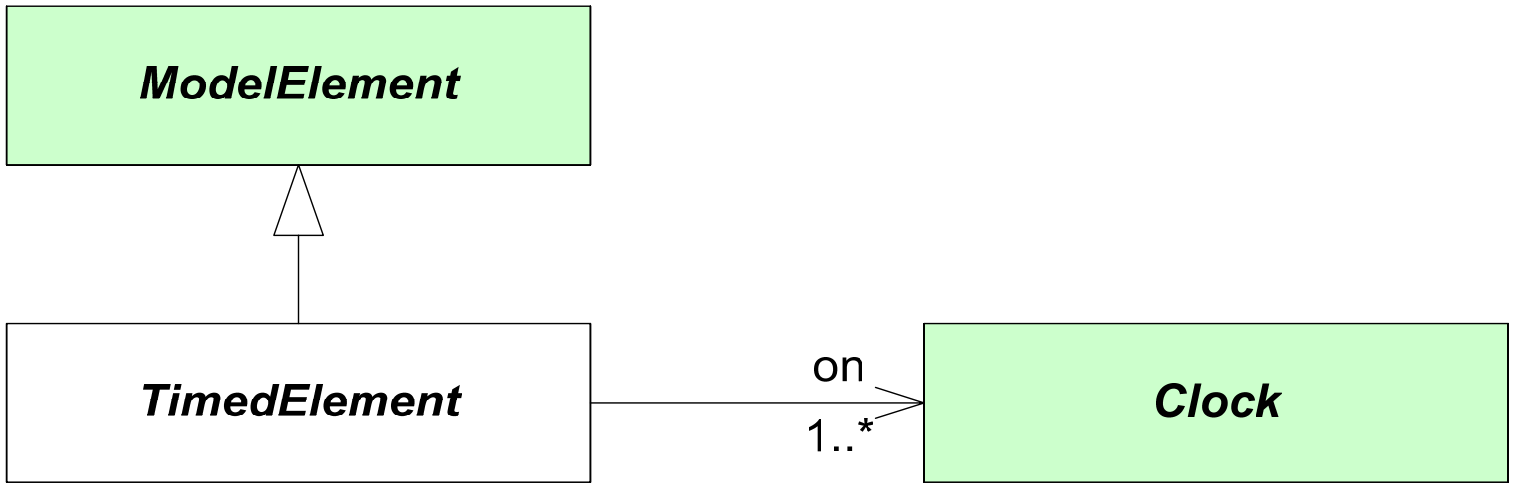
A TimeValue must reference a clock

Instant/Duration two distinct concepts

Domain

Timed Entities: TimedElement

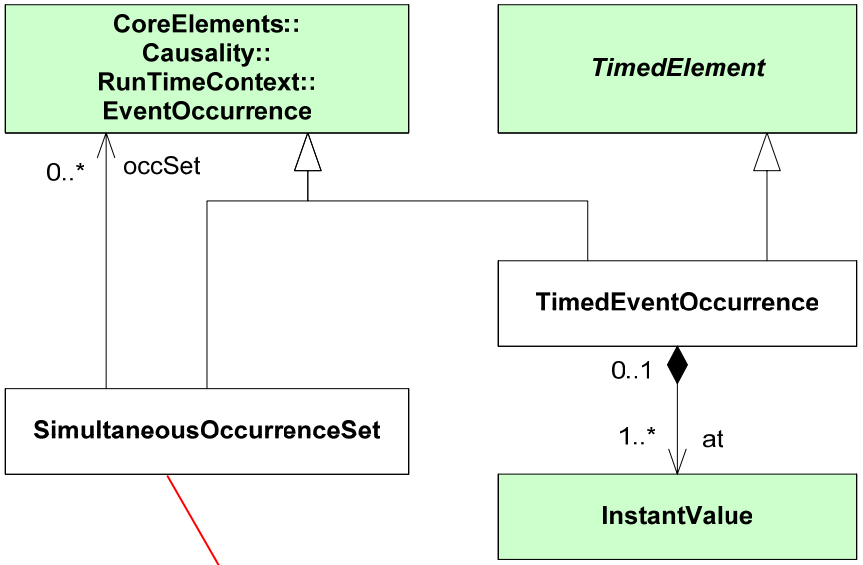
The unifying concept: a TimedElement = a ModelElement + a Clock



Timed Entities: TimedEvent

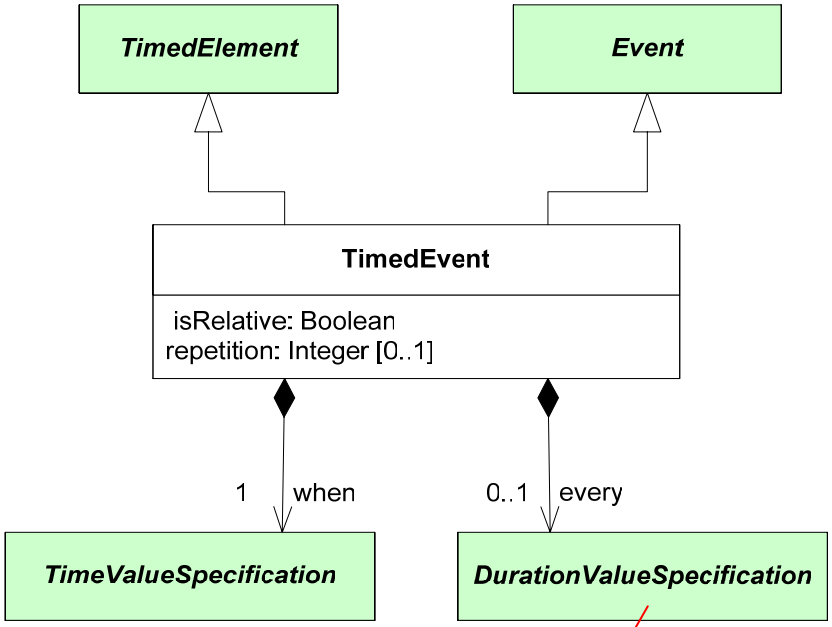
Domain

occurrences



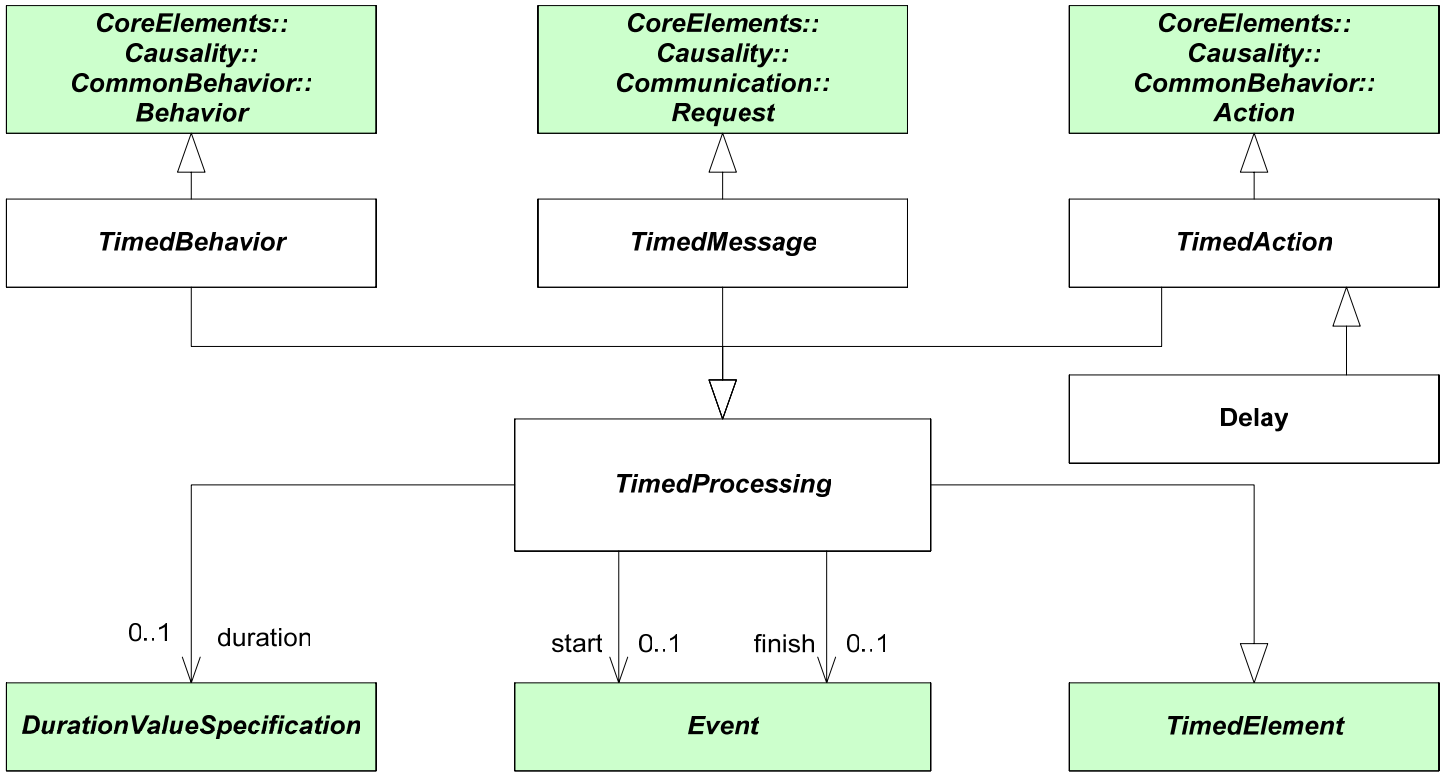
Provision for simultaneity

events



Facility to specify multiple occurrences

Timed Entities: TimedProcessing



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

Central stereotypes: ClockType & Clock

Chronometric clock → "physical " time; units ∈{s,ms,us,…}

Logical clock → any repetitive event; units ∈ {tick} U PhysicalUnits

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

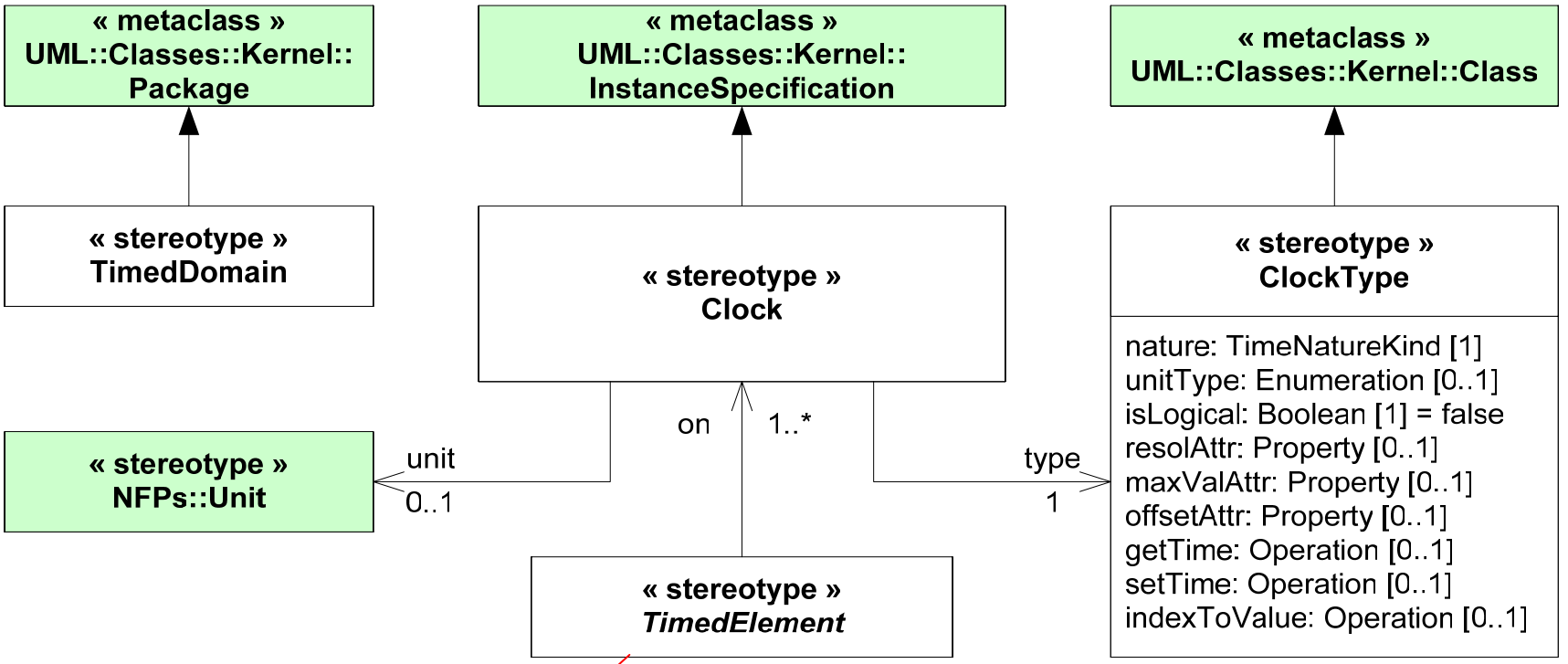
Stereotype properties:
Special semantics

+ optional

- **set of properties**
- **set of operations**

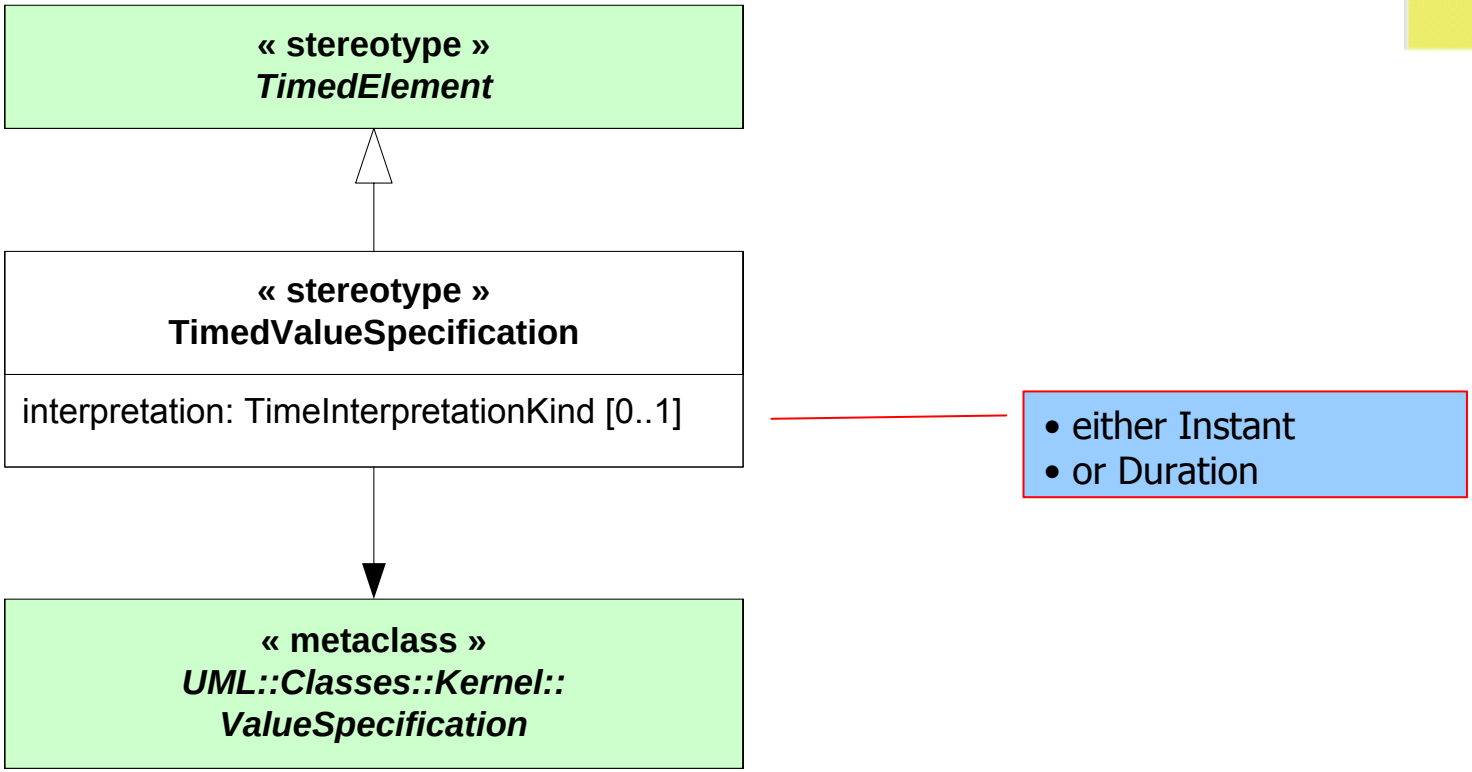
nature / isLogical	discrete	dense
true	**Logical clock**	Not used
false	**Chronometric clock**	
	discrete	dense

Clock and TimedElement



TimedValueSpecification

- Dedicated Language:
  - Clocked Value Specification Language (CVSL)



Time Values: Concrete syntax

Examples of Clocked value expressions

- **Simple time values**
  (value=3.5, unit=ms, onClock='idealClk');
  3.5 ms on idealClk;

  tuple, *a la* VSL

  short form

- **Homogemeous expressions**
  (value=1.5, unit=ms, onClock='idealClk') +
  (value=150, unit=us, onClock='idealClk');
  → (value=1650, unit=us, onClock='idealClk')

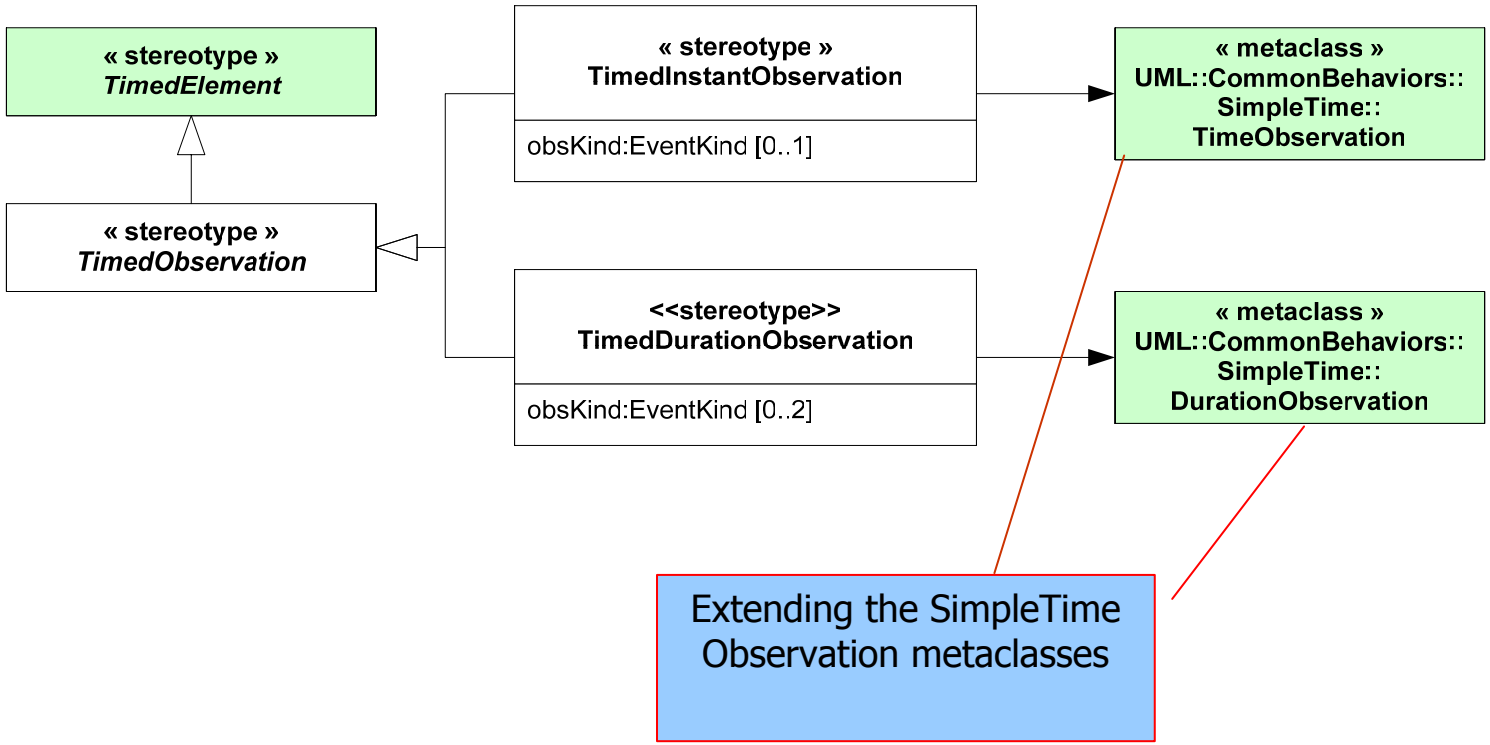
  Can be evaluated, because convFactor between units

- **Heterogeneous expressions**
  min (15 tick on prClk, 5 ms on idealClk);

  Clock relation between prClk and idealClk must be provided

- **Additional capabilities with VSL**
  - Occurrence number, jitter,…
  - but implicitly on idealClk

TimedObservation

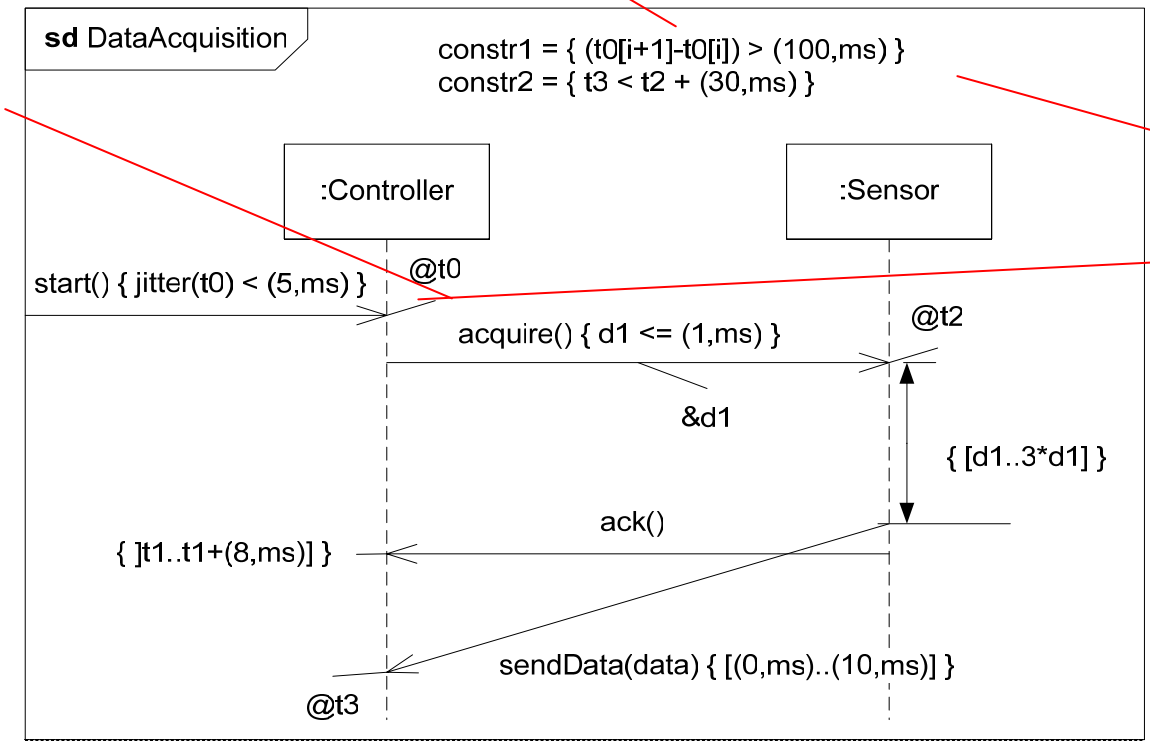


Time specific languages:VSL Time Constraints

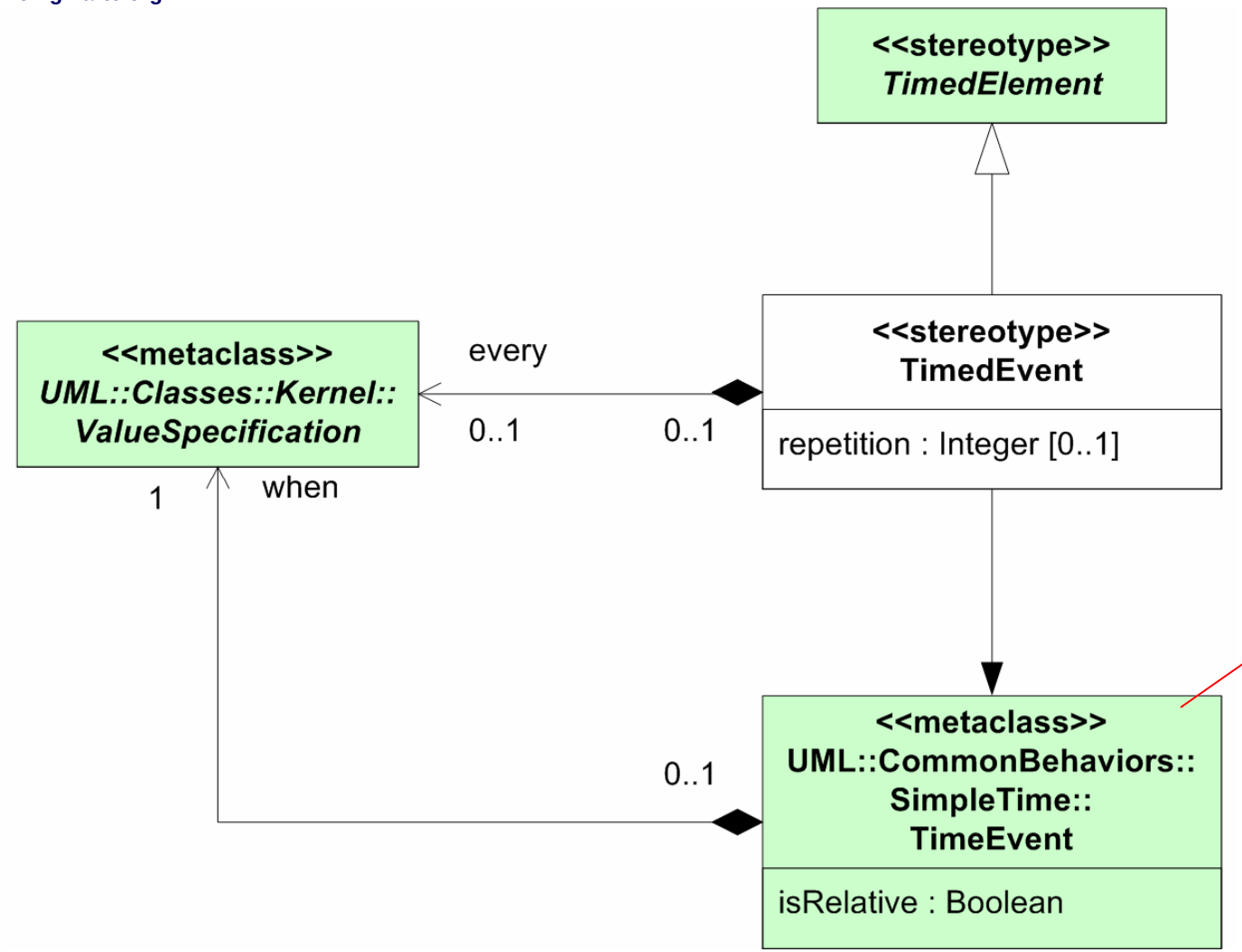
t0[i] denotes the i-th occurrence of

t0: observation of the message: start

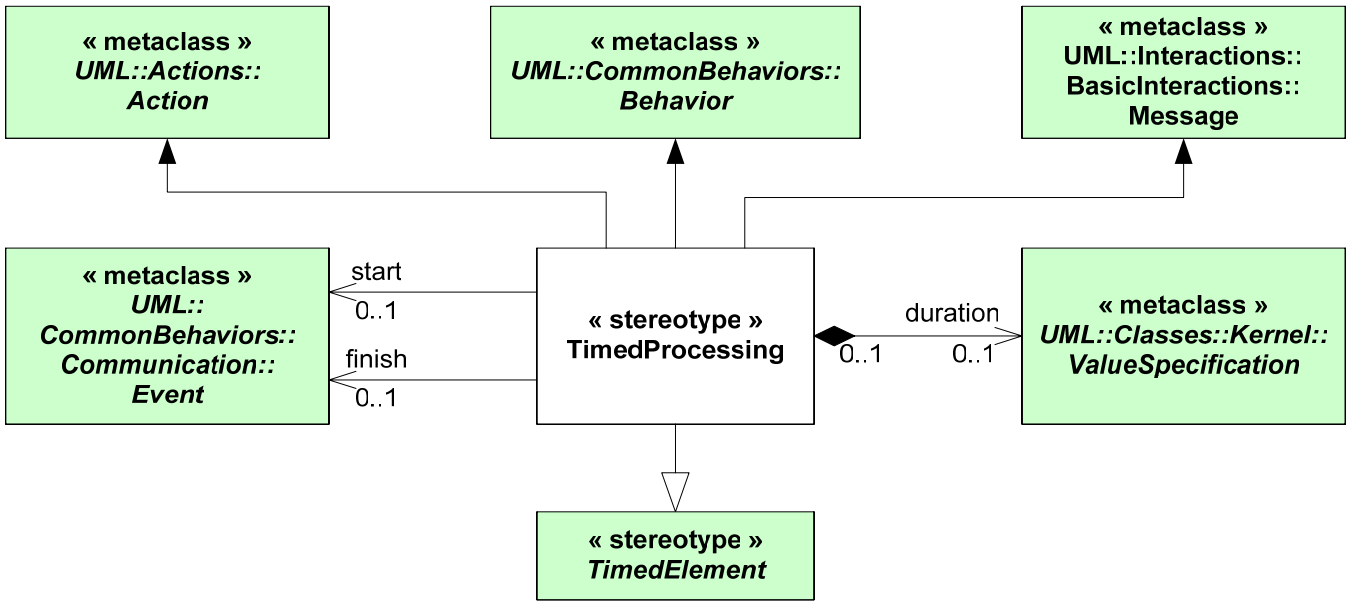
t0 is periodic, period 100ms with a jitter less than 5ms



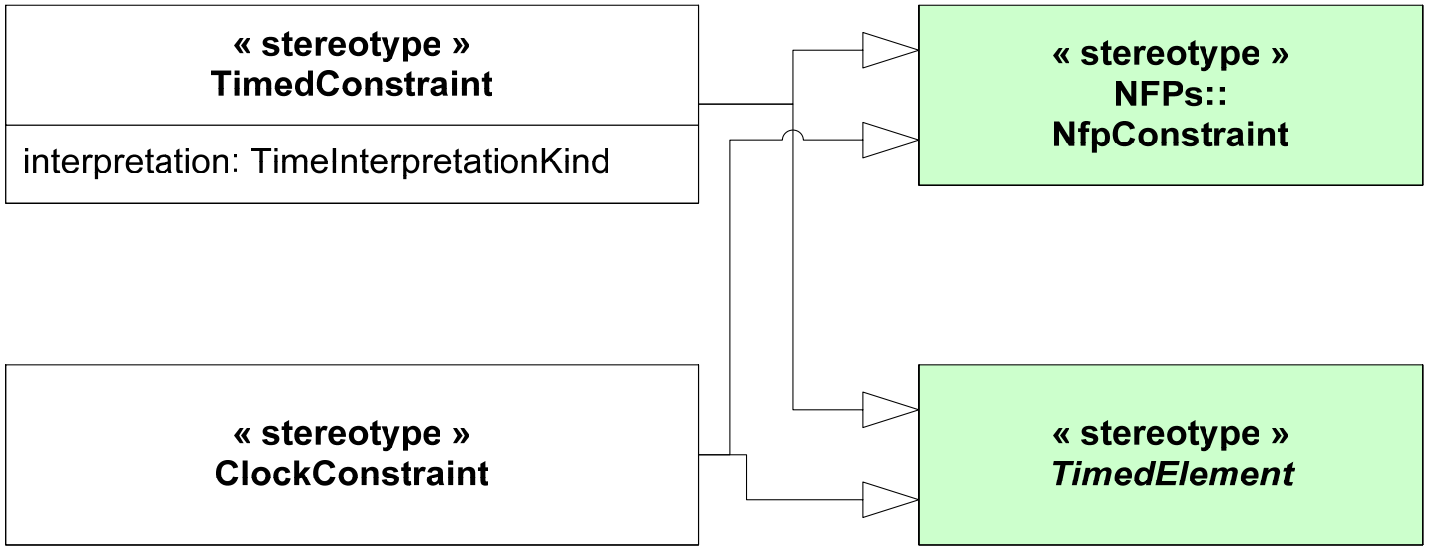
TimedEvent



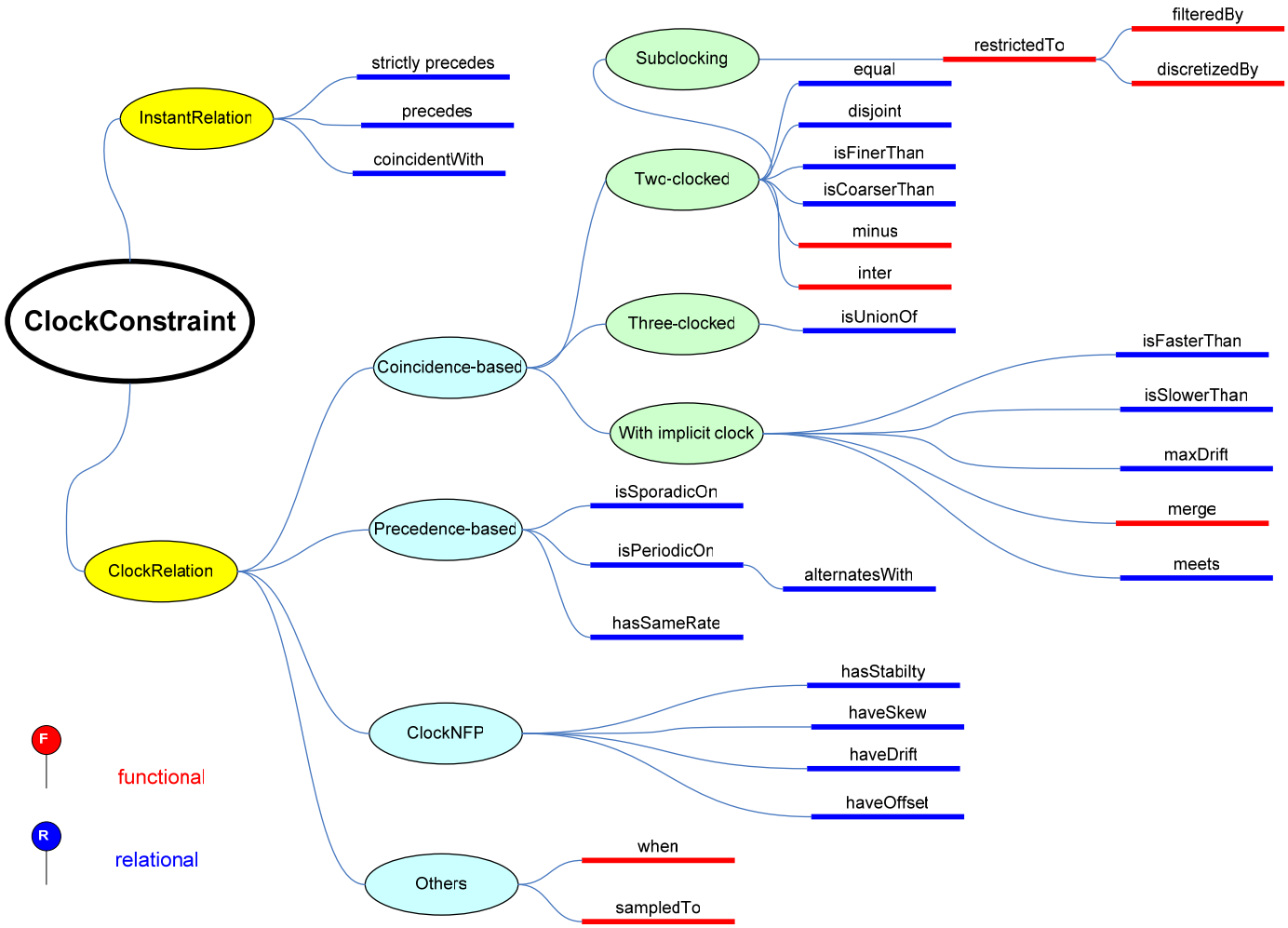
TimedProcessing



TimedConstraint & ClockConstraint



Clock Constraint Specification



Pre-defined Clock Constraints

Each relation has a mathematical specification

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**

Chronometric Clocks (1)

UML MARTE
www.omgmarte.org

How to specify chronometric clocks

« clockType »
{ nature = discrete, unitType = TimeUnitKind, resolAttr=resolution, getTime = currentTime }
Chronometric

resolution: Real {readOnly}

currentTime( ): Real

« clockType »
{ nature = dense, unitType = TimeUnitKind, getTime = currentTime }
IdealClock

currentTime( ): Real

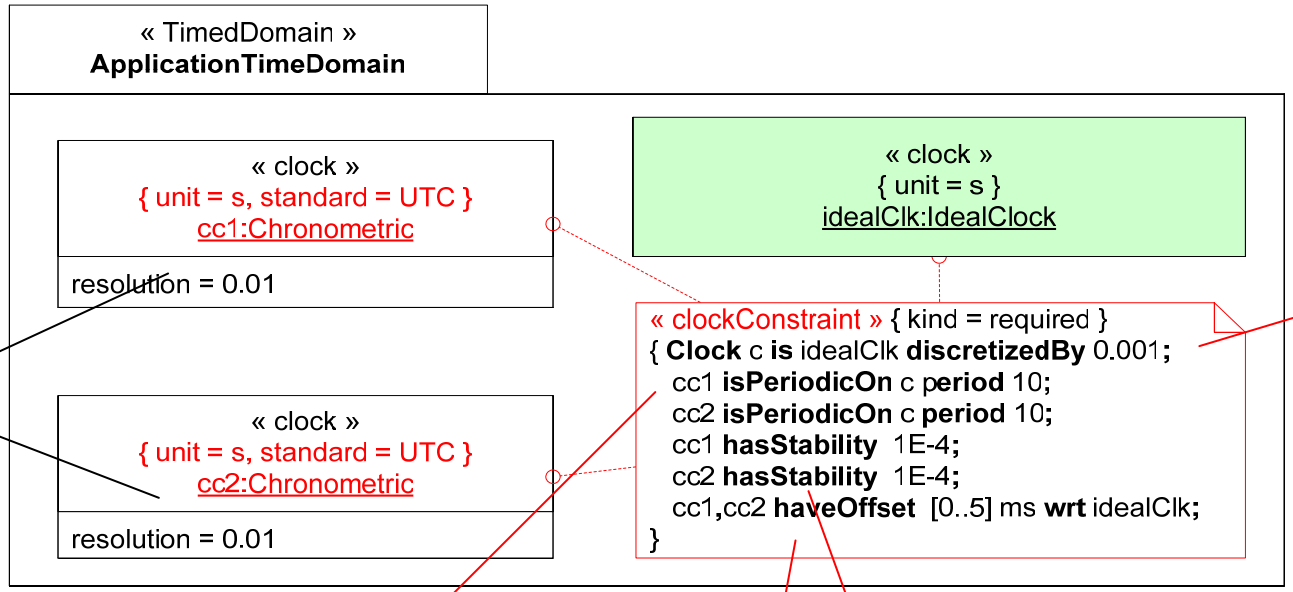
Imported from MARTE::TimeLibrary

An user's defined ClockType

cea list THALES INRIA

Chronometric Clocks (2)

Specifying NFP of (non ideal) chronometric clocks



User

Two instances

c: local ideal discrete clock – 1kHz

Exists d such that for all k:
c[d+10*(k-1)]<cc1[k]≤c[d+10*k]
⇒ 0<cc1[k+1]-cc1[k]<20 **ms**

A Non-functional property:**Stability**
10-0.001≤cc1[k+1]-cc1[k]≤10+0.001 in **ms**

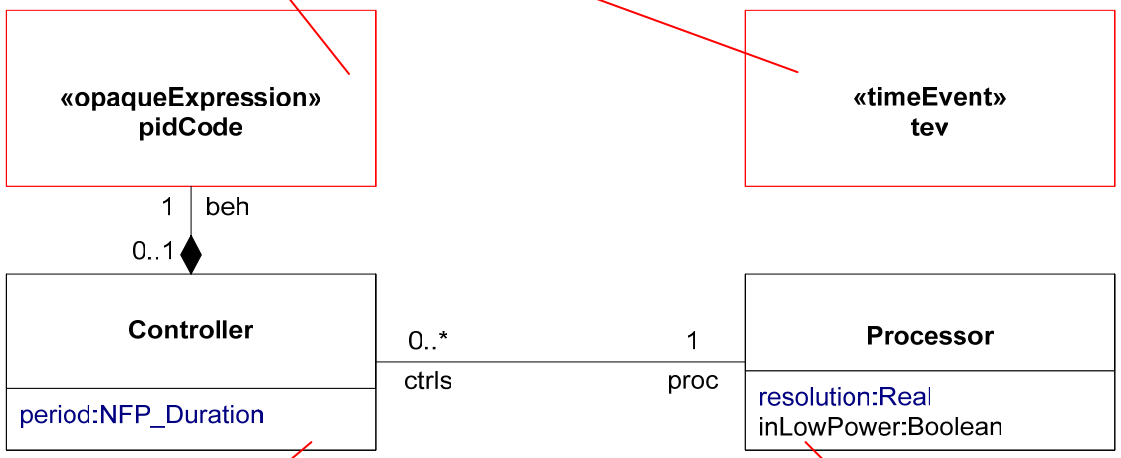
Another Non-functional property:**Offset**

Logical Clocks (1/4)

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

User

Explicit model elements not usual in Class Diagrams

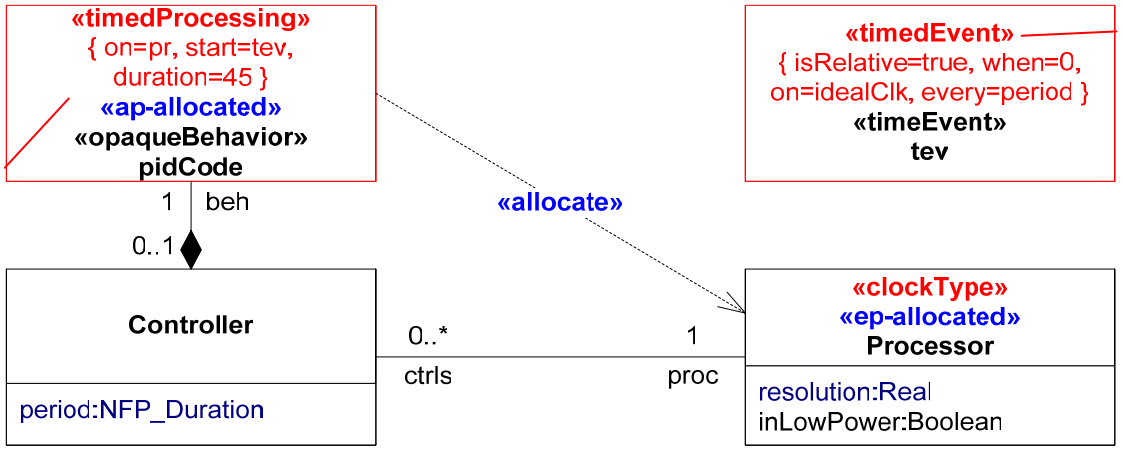


Period of the PID controller: uses a NFP-type

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

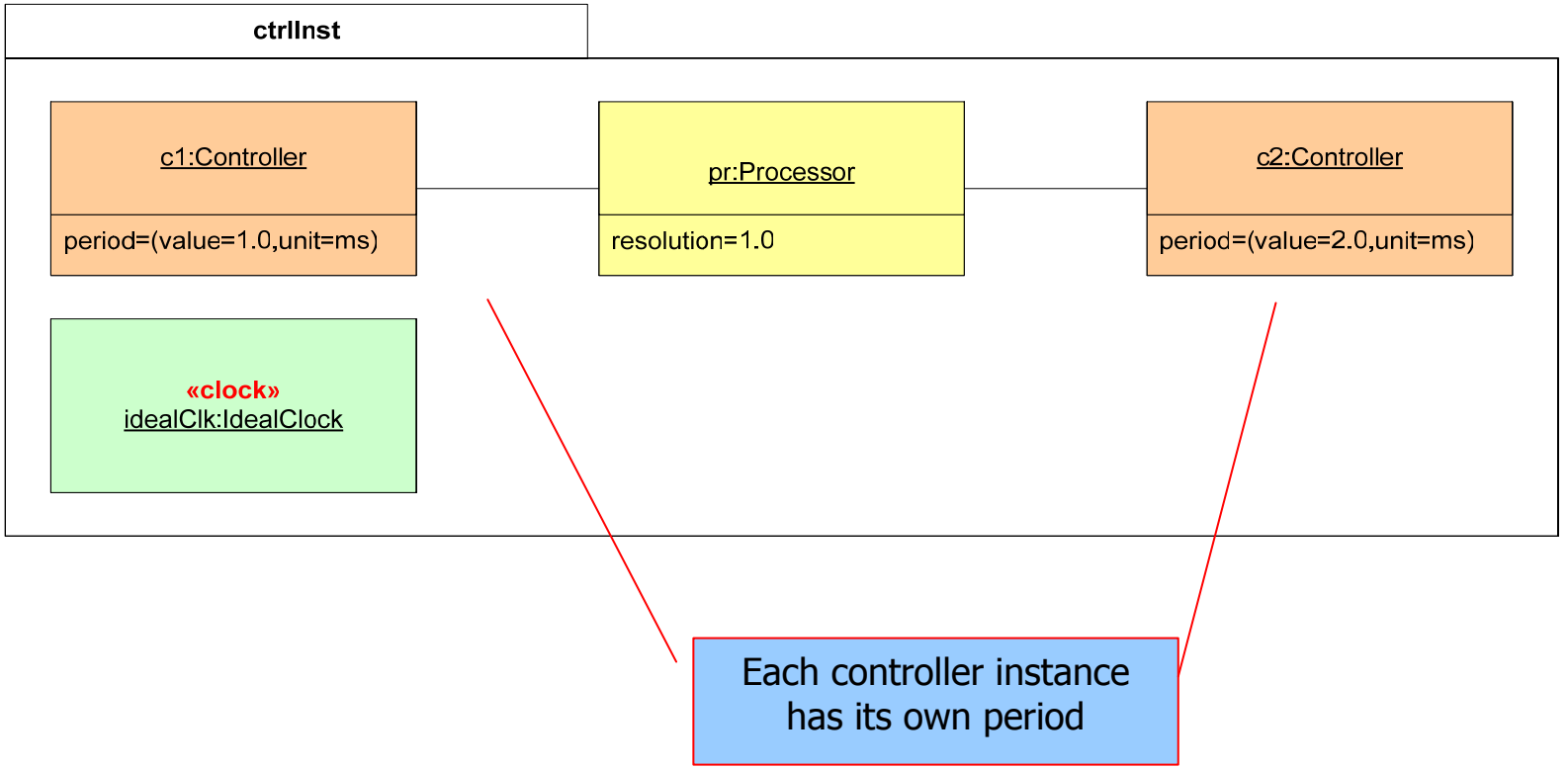
The class Processor is stereotyped by ClockType

Logical Clocks (3/4)

3) Instantiate user's model elements

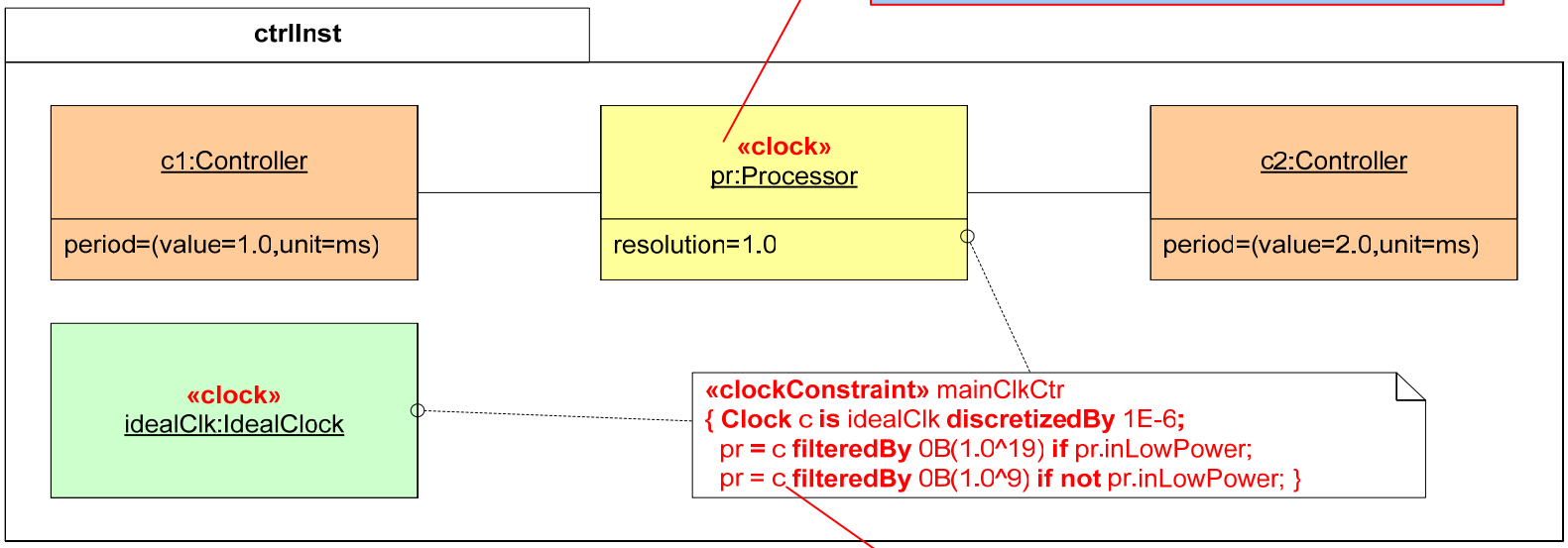
User

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



Logical Clocks (4/4)

4) Introduce clock (by stereotyping)



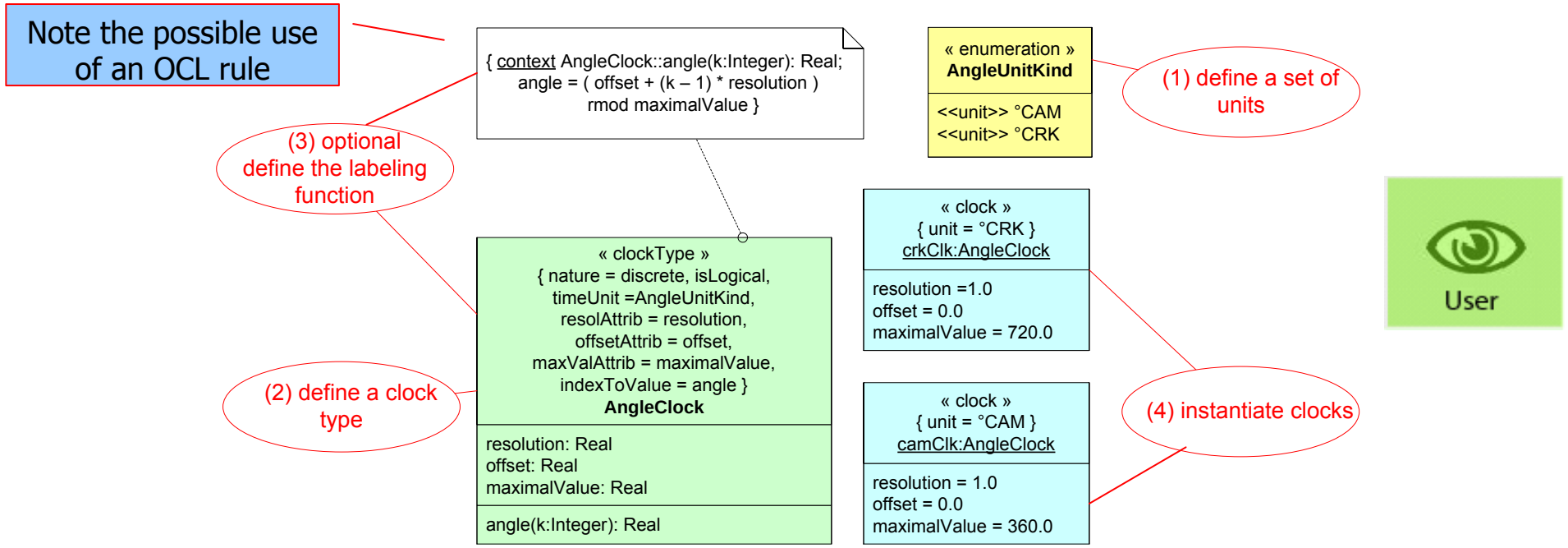
Multiform time (1/4)

Another example of logical clocks

Automotive application

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

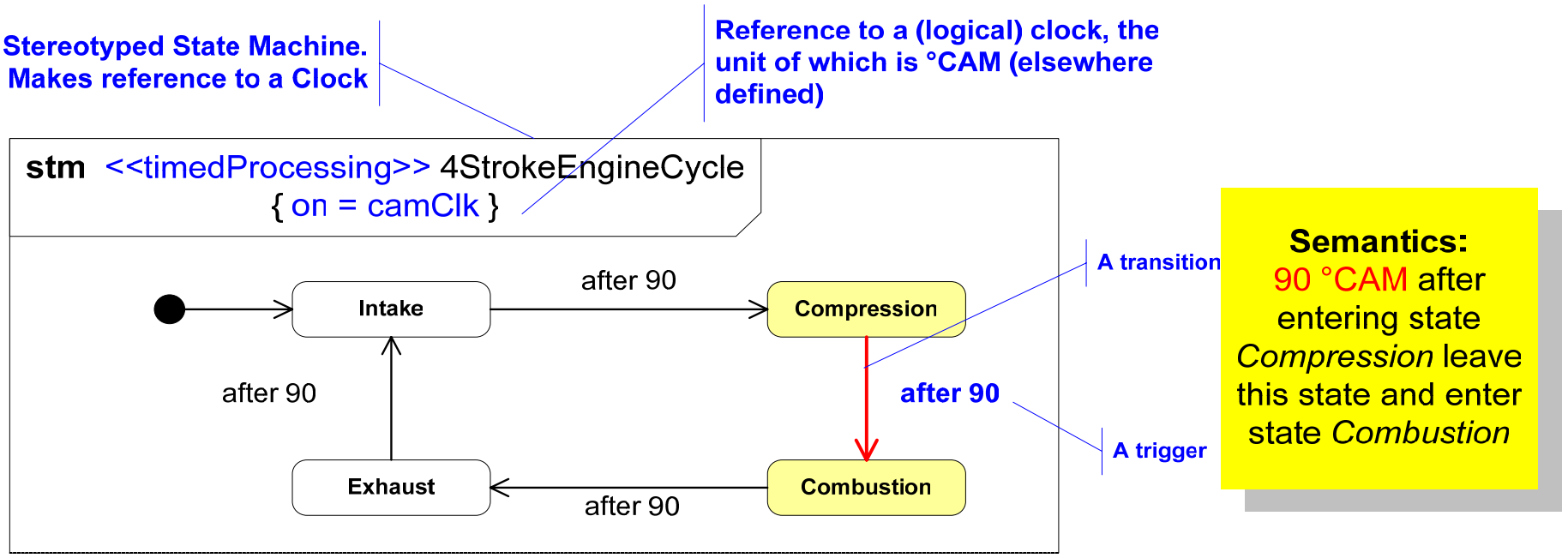
=> Define logical clocks dealing with angular positions.



Multiform time (2/4)

Example of usage of an “AngleClock”

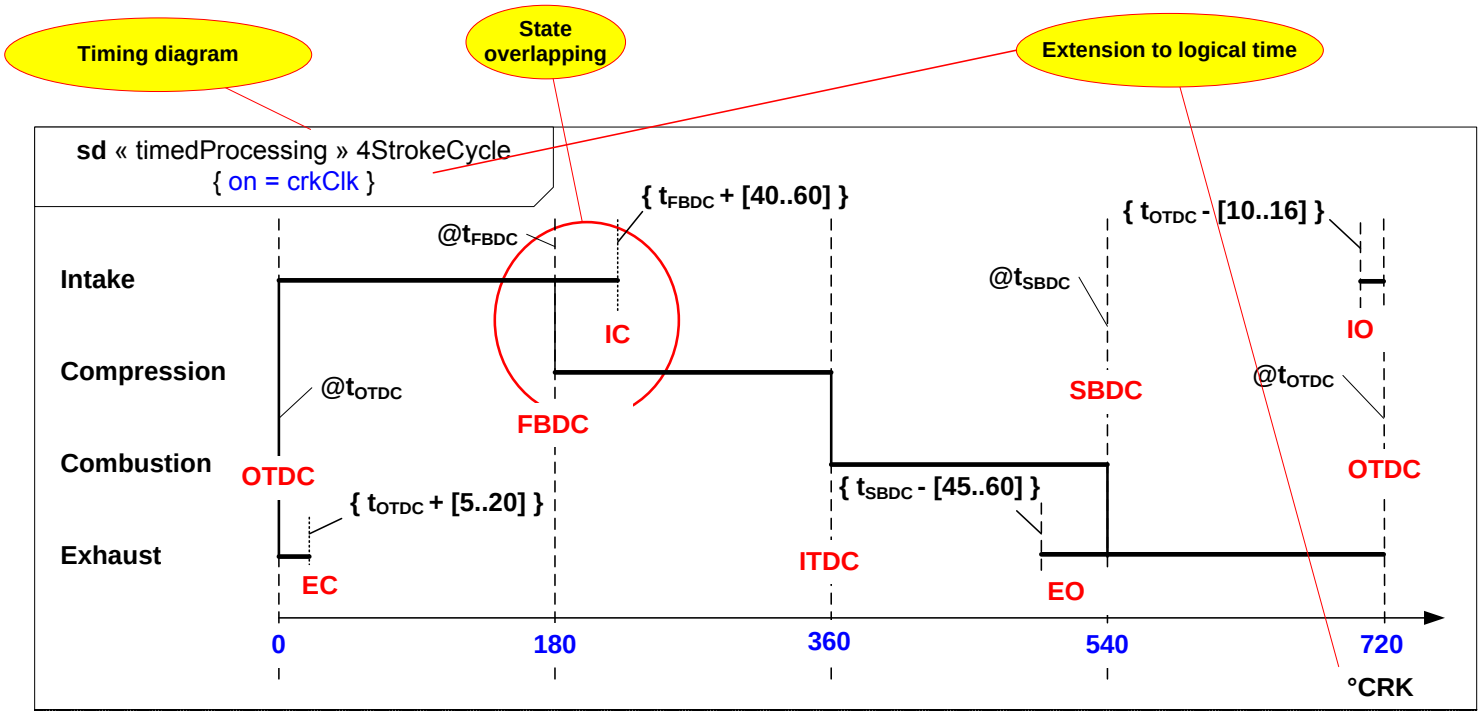
User



Multiform time (3/4)

Another example of usage of an “AngleClock”:
Enhanced timing diagram used in specification

User



crkClk: crankshaft Clock
°CRK: degree crank

TDC: Top Dead Center
ITDC: Ignition TDC
OTDC: Overlap TDC

BDC: Bottom Dead Center
FBDC: First BDC
SBDC: Second BDC

IC: Intake closes
IO: Intake opens
EC: Exhaust closes
EO: Exhaust opens

Multiform time (4/4)

Combining logical clocks:
c*k* is an AngleClock used to specify the ignition of a cylinder
c is the clock used to specify ignitions in a 4-cylinder engine

User