DE LA RECHERCHE À L'INDUSTRIE

# COUPLING EXECUTABLE UML MODELS WITH FMI

MODPROD 2017
Linköping, Sweden, February 7-8, 2017

openCPS ITEA European project


Sahar GUERMAZI, Sébastien REVOL, Arnaud CUCCURU, Saadia DHOUIB, Jérémie TATIBOUET, Sébastien GERARD

CEA LIST / DILS / LISE


www.cea.fr

INSTITUT CARNOT CEA LIST digiteo

# FMI FOR PAPYRUS / PAPYRUS FOR FMI

- **Synergy of two complementary standards for Complex system modeling and simulation**

- **FMI (Functional Mockup Interface)**
  - **Emerging standard for co-simulation**
  - **Enables multiple compliant modeling and simulation tools to interoperate**
  - **Particularly interesting for designing CPS (Cyber Physical Systems)**

- **UML in the FMI eco-system**
  - **UML (and its variants) can be used to design parts of CPS**
    - E.g., the high-level control logic of an embedded software
  - **Would be nice to have the possibility to assess the relevance of the UML-based parts with respect to their (simulated) environment**
    - Scenario exploration, early error detections.

- **Papyrus now provides FMI tool support**
  - **Based on Moka, the Papyrus module for model execution**

DE LA RECHERCHE À L'INDUSTRIE

## Papyrus is the official open-source Eclipse UML2 modeling tool:

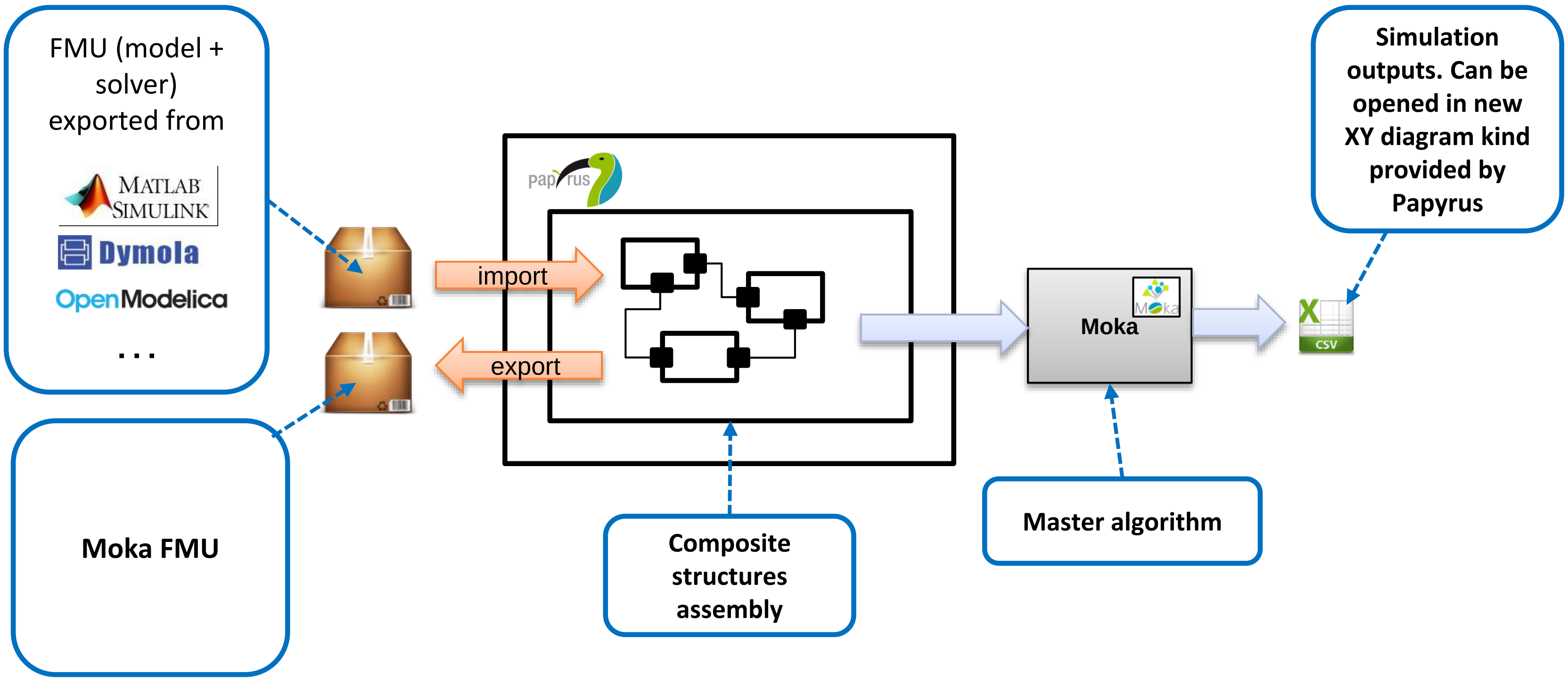
**www.eclipse.org/papyrus**

- Papyrus provides a complete graphical editor for both UML and SysML standards based on the MDT::UML2 component for its repository.
- Papyrus addresses the two key features expected from a UML2 graphical editor: modeling and profiling.
- Papyrus is highly customizable and extensible enabling DSML definitions based on standard UML profiles!
- Papyrus provides a support to MARTE 1.1 (including a rich text editor for VSL).

# MOKA: OVERVIEW

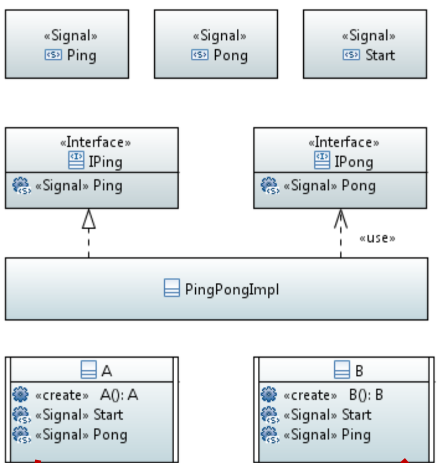
- **Papyrus module for model execution**
  - **Help designers to understand/orient their design choices**
  - **Basis for a straightforward, simulation-driven design process:**
    - (Model / Execute / Observe / Refine)+
  - **Front-end for integration of simulation tools and techniques**
- **Model Debugging capabilities**
  - **Control (start/stop, suspend/resume, breakpoints)**
  - **Observation (diagram animation, variables, threads)**
- **Complies with standard OMG semantics of UML**
  - **Implements the fUML execution model (State Machine extension coming)**
  - **Tool support for Alf, the standard textual notation of fUML**
- **Flexible/extendible**
  - **New execution engines can be plugged (to support multiple semantics and UML profiles)**
  - **Extension points to inject control/execution model libraries (to trigger the execution of external functions and procedures directly from a UML model)**

# OVERVIEW OF PAPYRUS TOOL SUPPORT FOR FMI

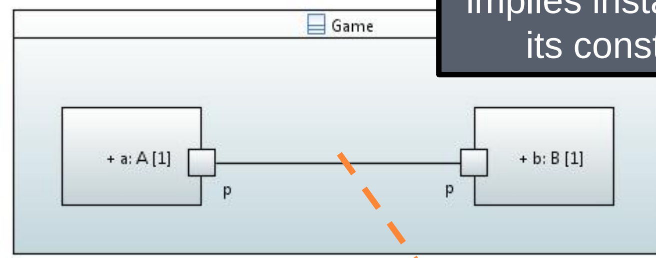


# FUML IN A NUTSHELL

Structure




1. Class diagram (~ BDD)

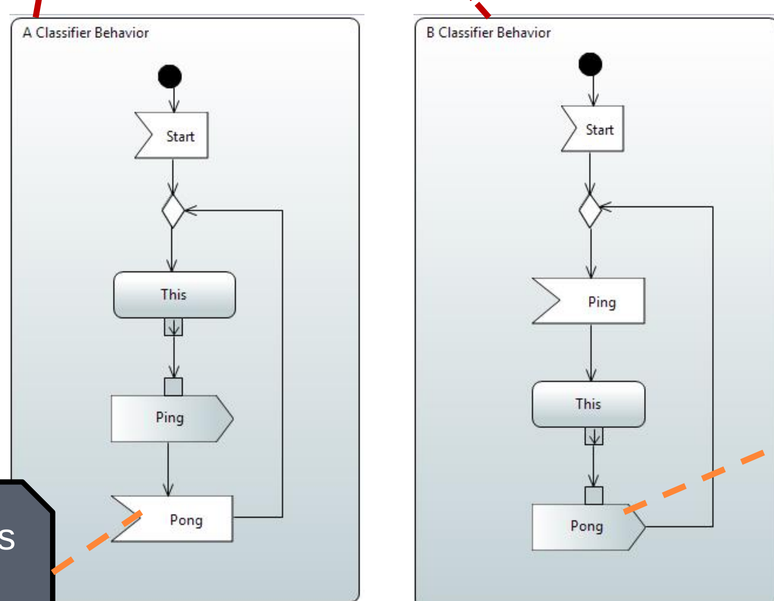



2. Composite structure diagram (~ IBD)

Instantiation of a composite structure implies instantiation of its constituents

Instantiation of an active class implies starting of its behavior

Behavior




3. Activity diagrams

SendSignalAction enable to specify asynchronous communications, which will flow through ports and connectors

AcceptEventActions enable to specify reactive behaviors

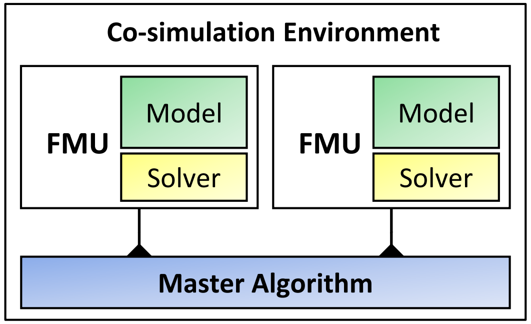
Event dispatching occurs at Run To Completion (RTC) steps

# KEY SEMANTIC QUESTION FOR FMI SUPPORT

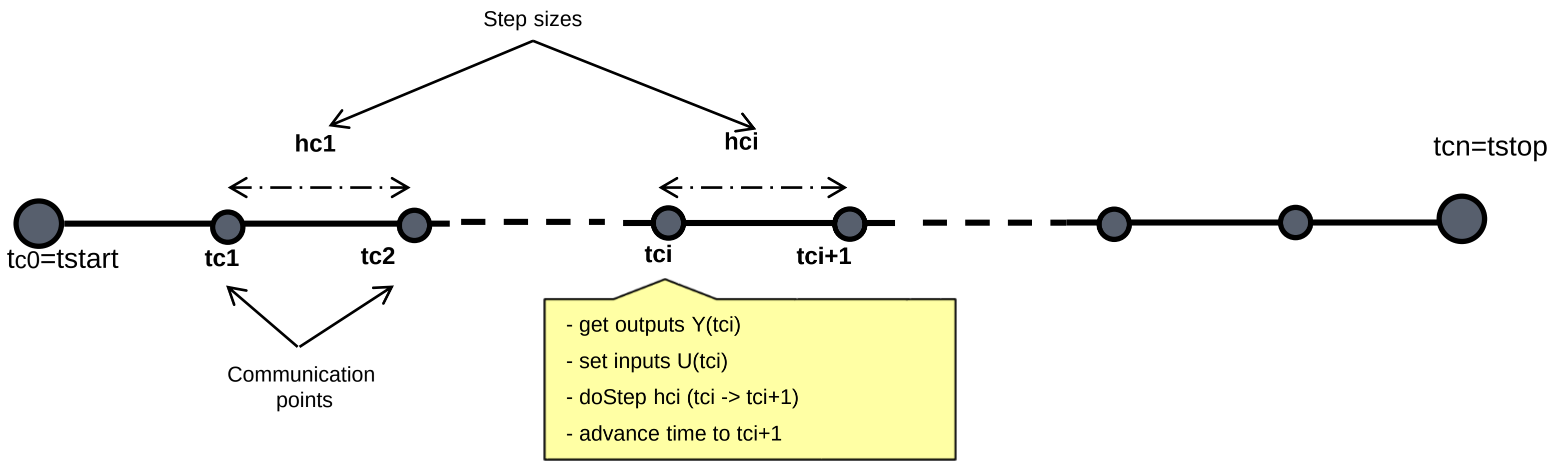
What is the meaning of an FMI co-simulation step in fUML?

## Allows to export each executable model as a standalone unit (FMU)

- **An FMU has to implement a standard binary interface as a shared library ( dll/.so)**
  - **Set Inputs**
  - **Get outputs**
  - **Do Step (stepSize)**




## The simulation Master synchronizes and orchestrates the FMUs

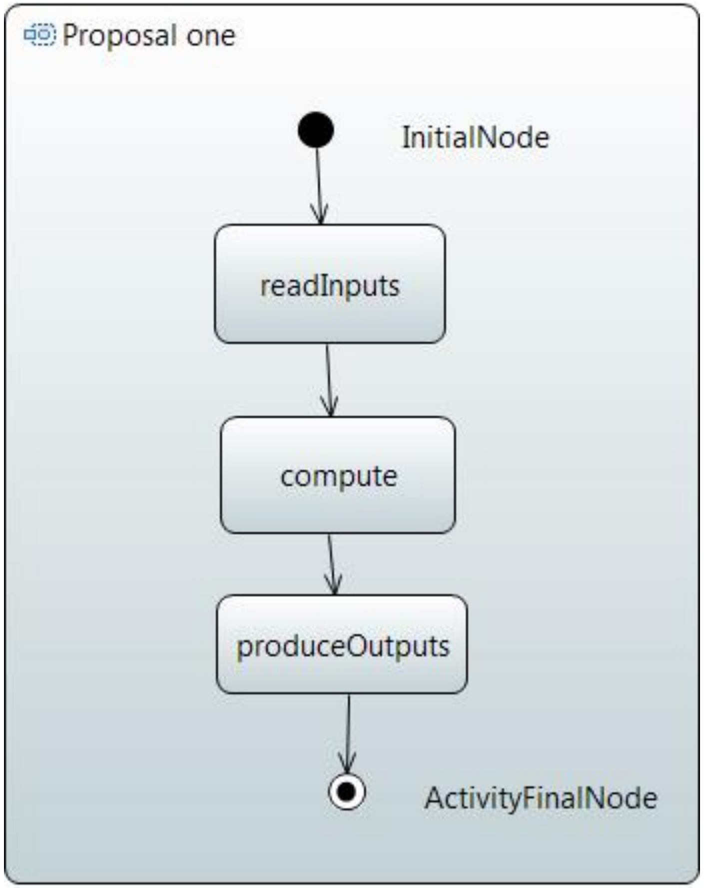


# HOW TO DESCRIBE AN FMU BEHAVIOR IN UML?

- **Proposal one : an FMU is a finished activity**
  - **Replayed at each simulation step**

- **Drawback : can't preserve internal states between simulation steps**

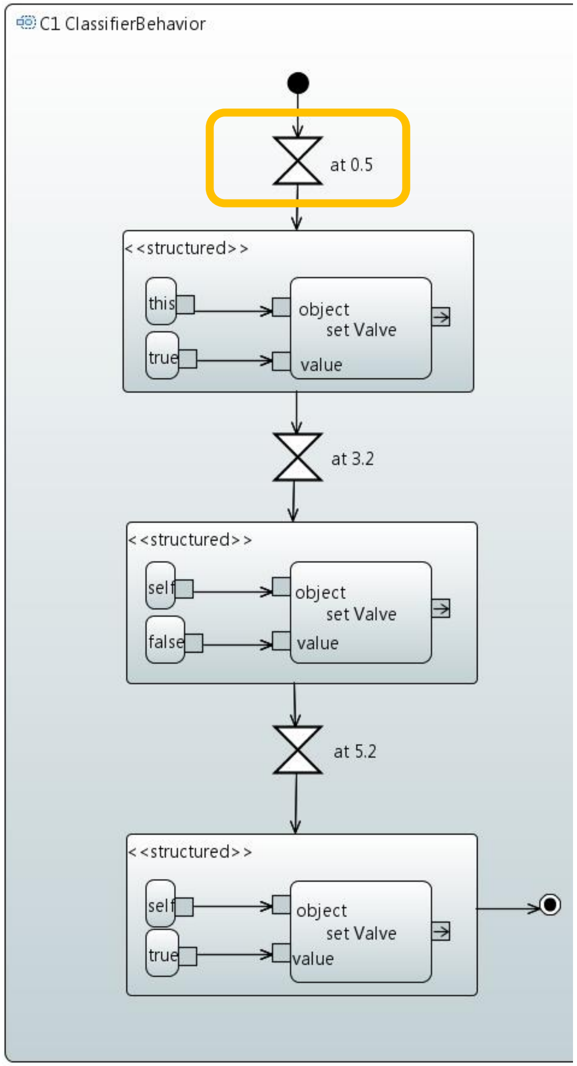
- **→ Need to be able to suspend the behavior between two co-simulation steps!**



# FUML EXTENSIONS FOR FMI

- **In fUML : only one kind of wait**
  - On signal event
  - Should be sent  by another active object
- **Our proposal : add to fUML two new ways to suspend/resume behaviors**
  - Wait for delays (Time Event)
  - Wait for FMU input changes (Change Event)
- **Time is managed by a central Discrete Event scheduler**
  - With its own event pool
- **An FMI to UML interface generates events at each simulation steps**
  - A StepEnd event @(currentTime + stepSize)
  - Change events for any new FMU input value

FMI Master




## DE Scheduler

Current time = 0.0

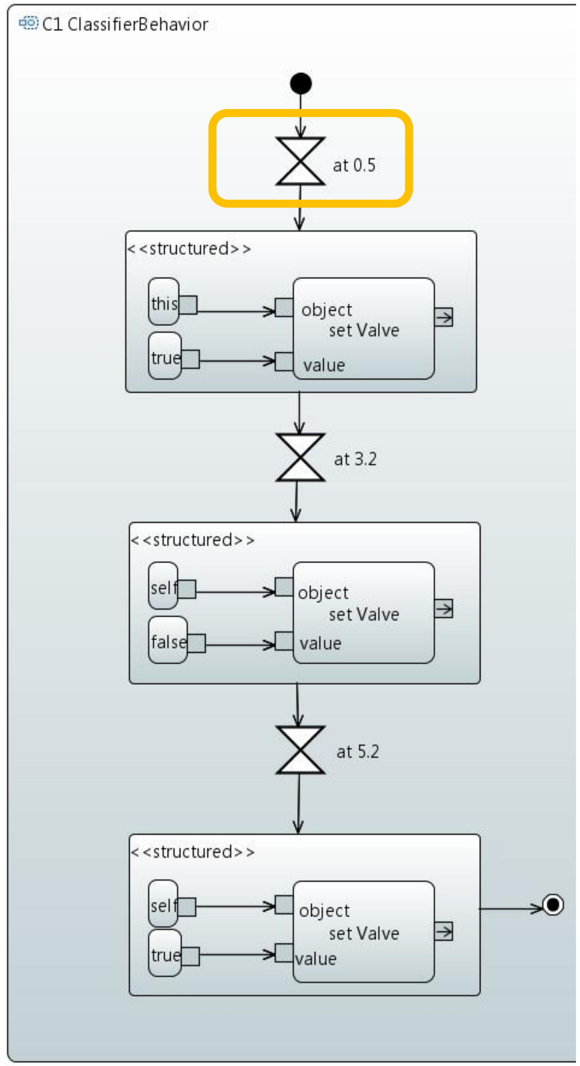
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




# DE Scheduler

Current time = 0.0

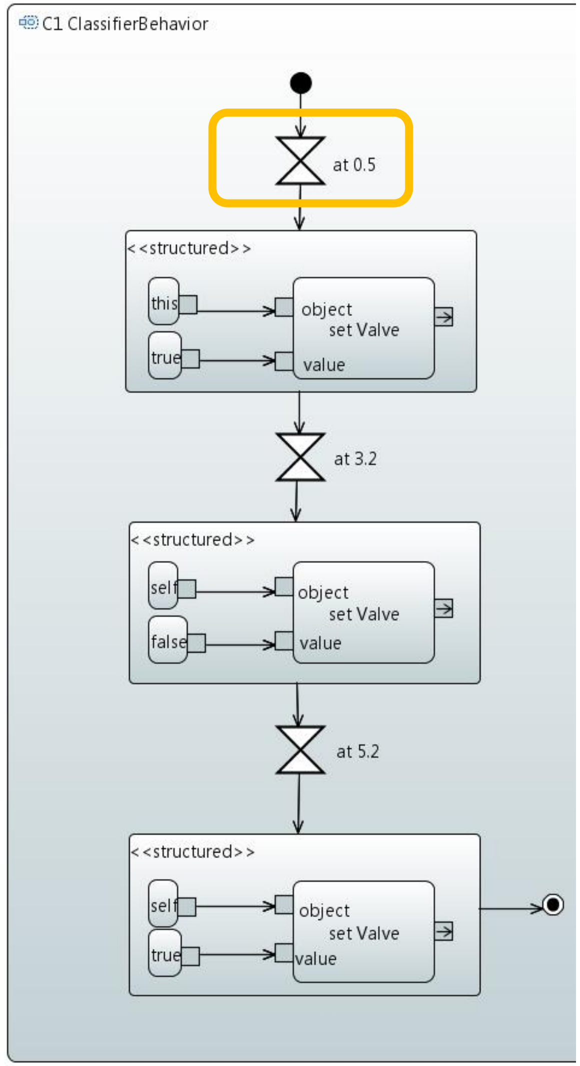
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




## DE Scheduler

Current time = 0.0

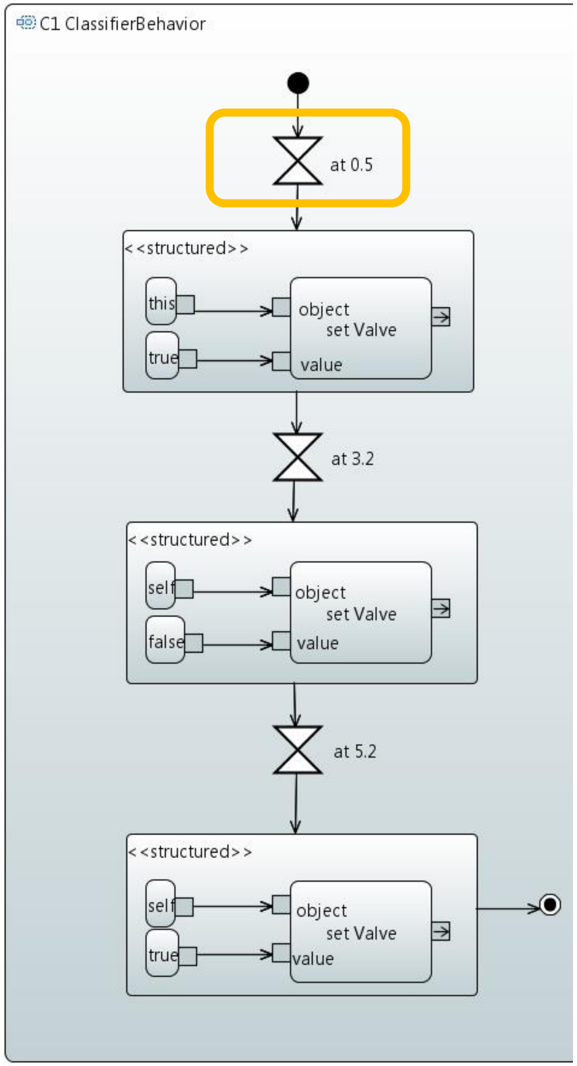
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Step End | 0.1 |
| Time Event | 0.5 |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




# DE Scheduler

Current time = **0.1**

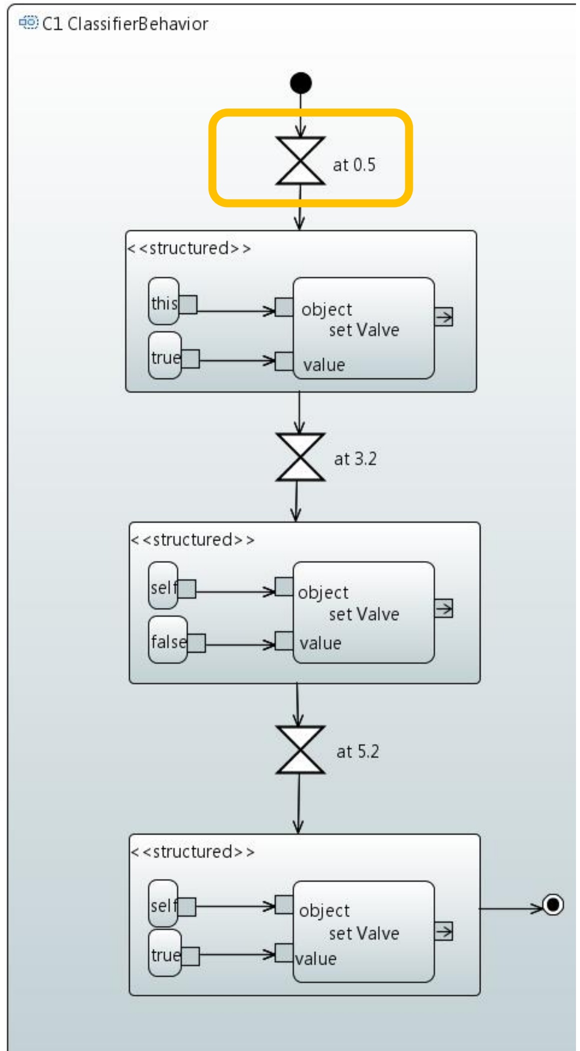
Current event = **Step End**

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




## DE Scheduler

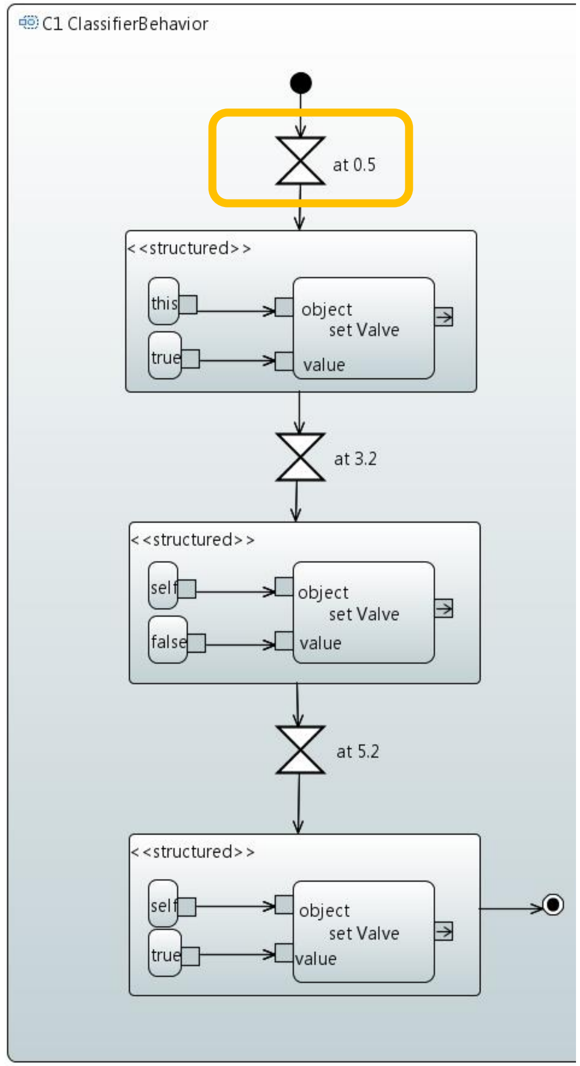
Current time = **0.1**

Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master




# DE Scheduler

Current time = 0.1

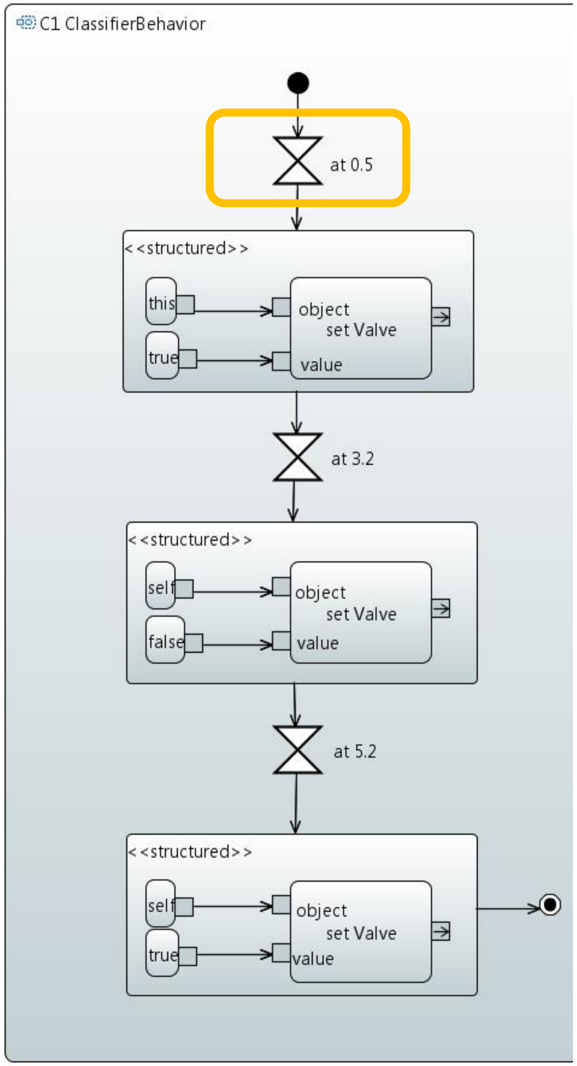
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




## DE Scheduler

Current time = 0.1

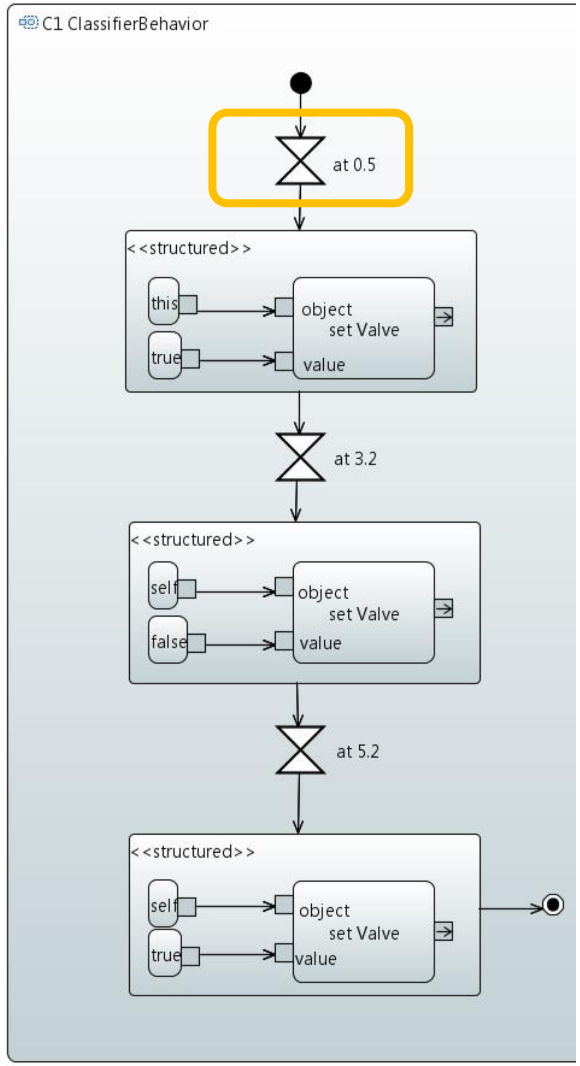
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




## DE Scheduler

Current time = 0.1

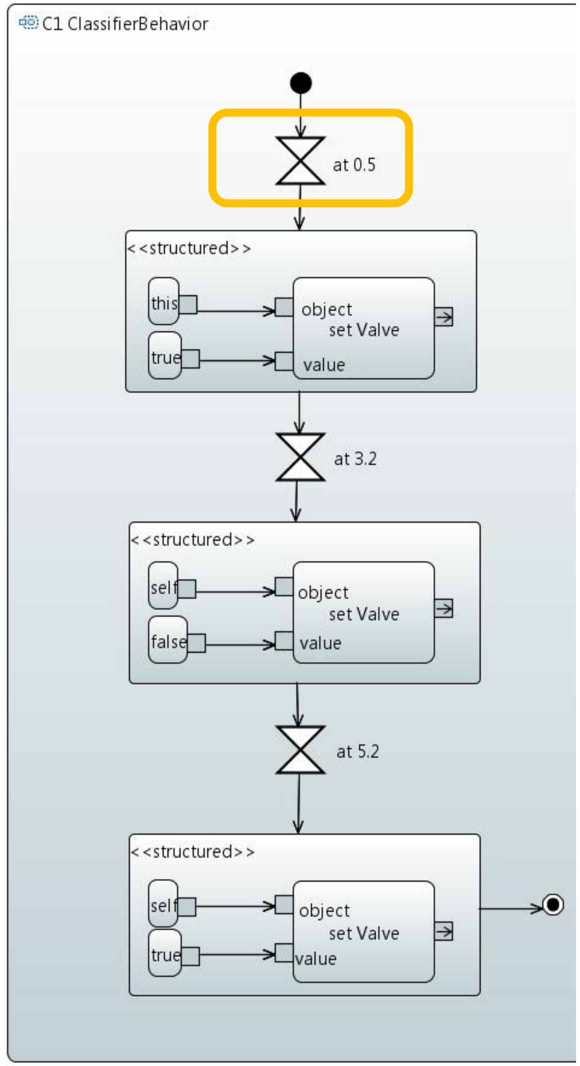
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Step End | 0.2 |
| Time Event | 0.5 |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




## DE Scheduler

Current time = **0.2**

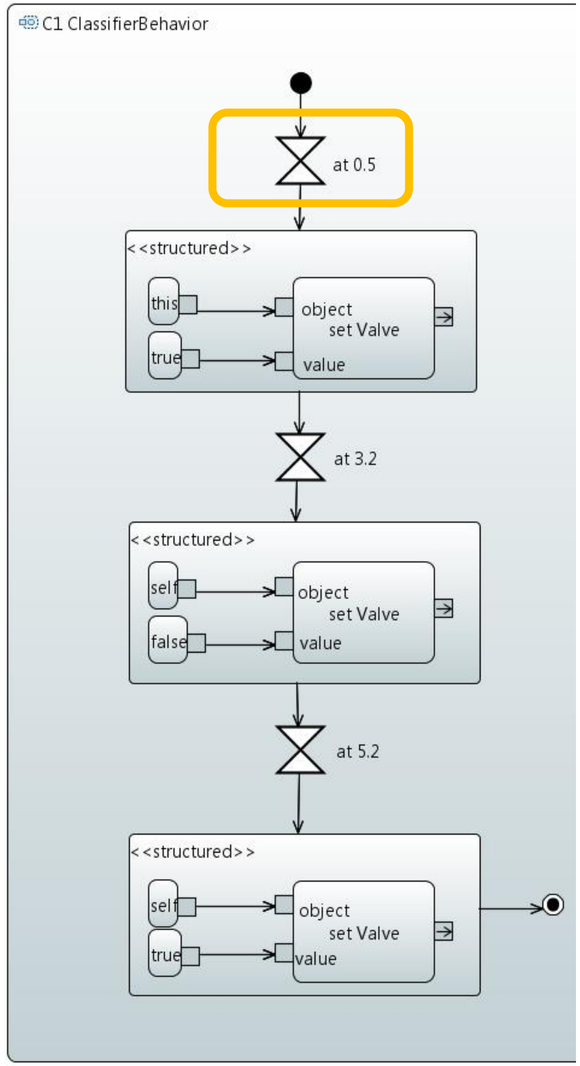
Current event = **Step End**

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




## DE Scheduler

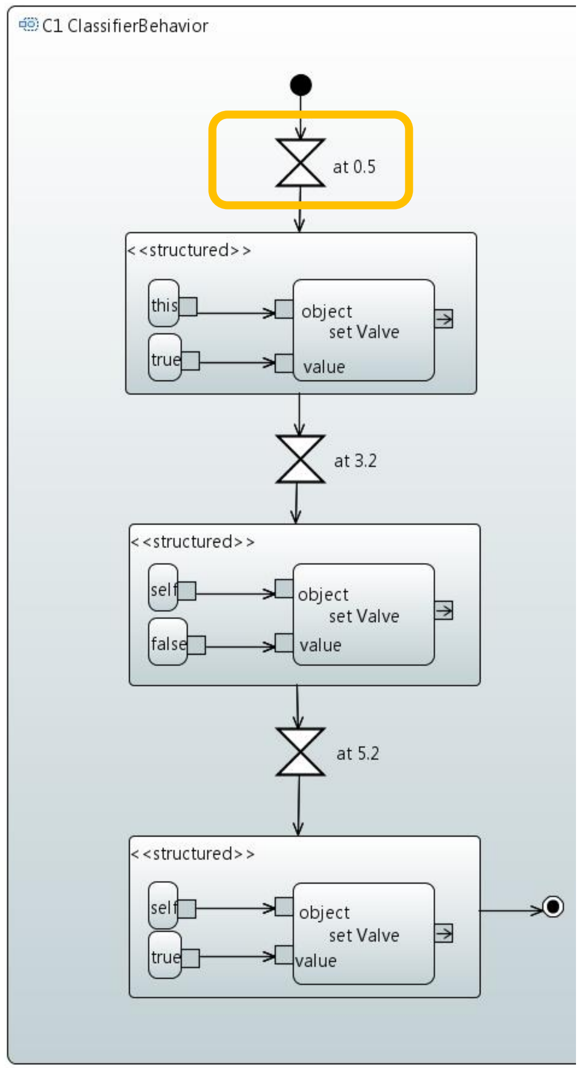
Current time = **0.2**

Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master




## DE Scheduler

Current time = 0.4

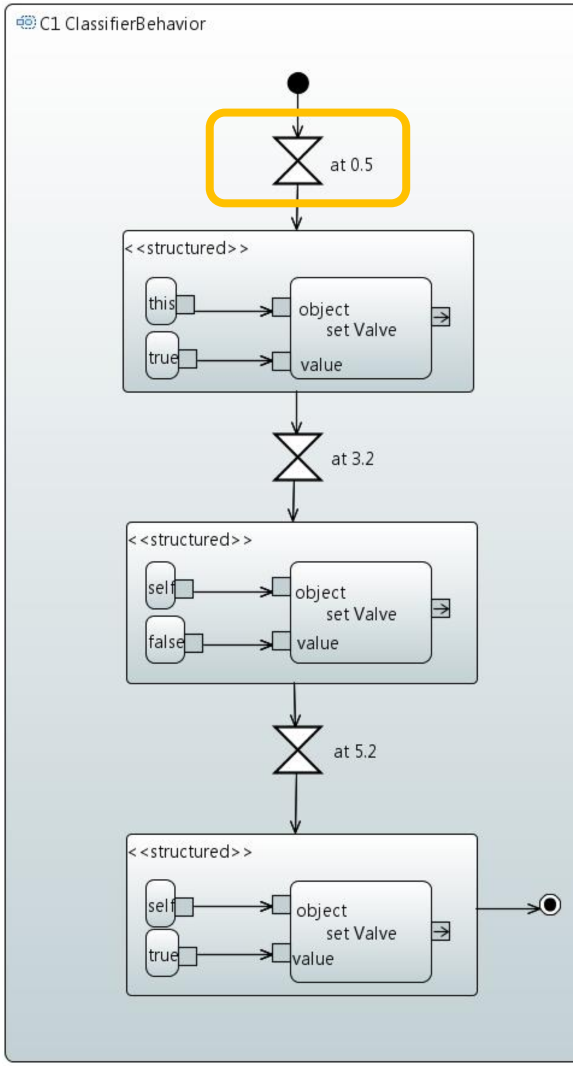
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




## DE Scheduler

Current time = 0.4

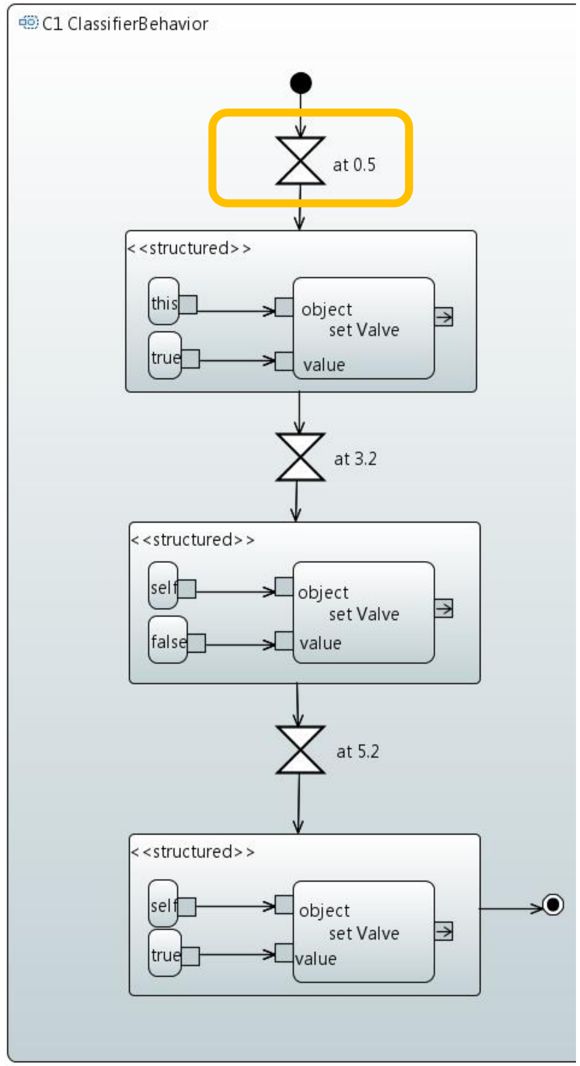
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




# DE Scheduler

Current time = 0.4

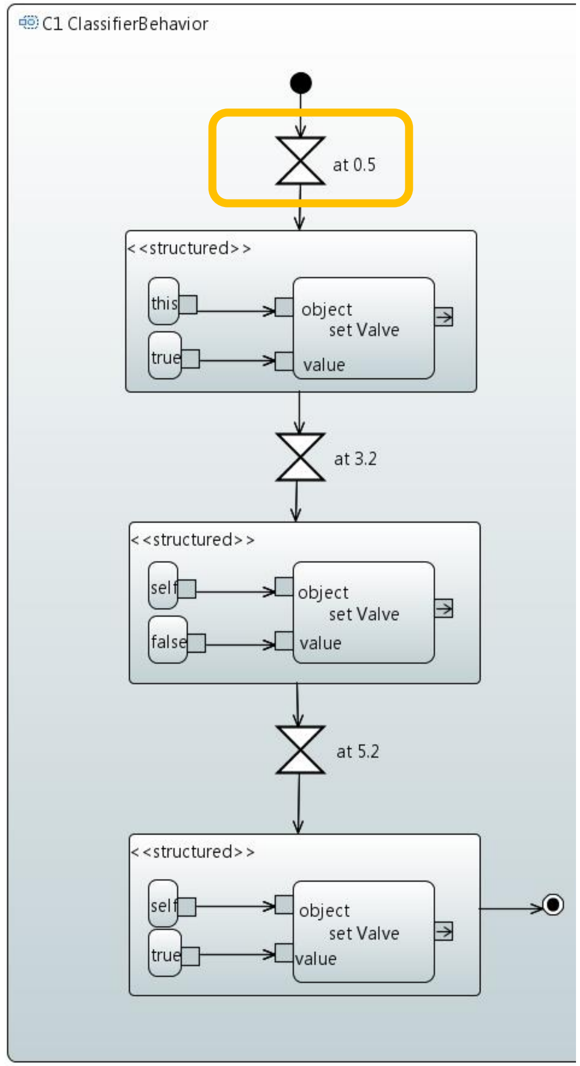
Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 0.5 |
| Step End | 0.5 |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




# DE Scheduler

Current time = **0.5**

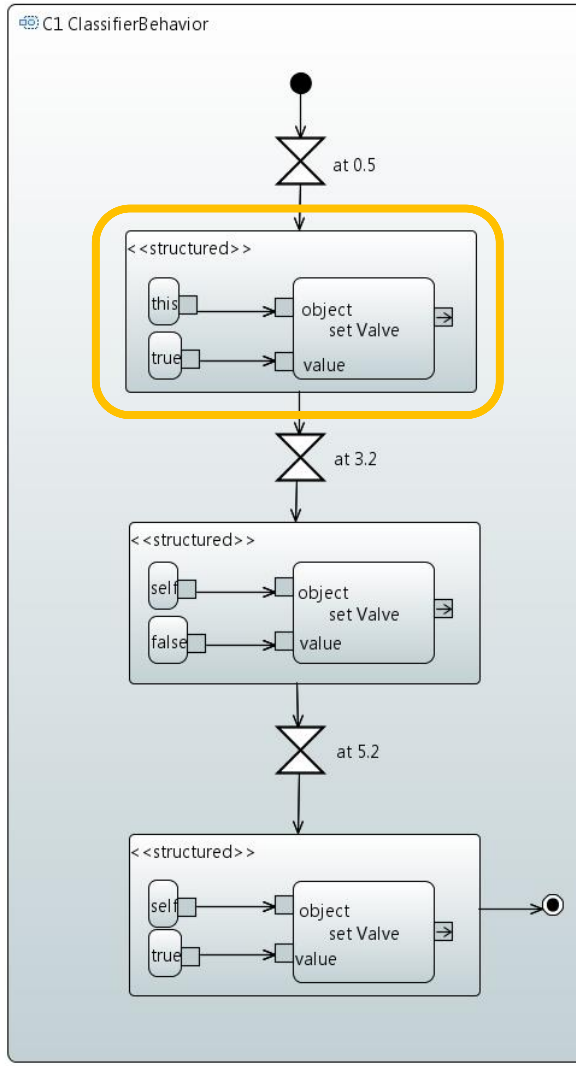
Current event = **Time Event**

| Scheduler Events | Time stamp |
|---|---|
| Step end | 0.5 |
| | |
| | |
| | |

Valve = false

FMI Master

doStep (0.1)




## DE Scheduler

Current time = **0.5**

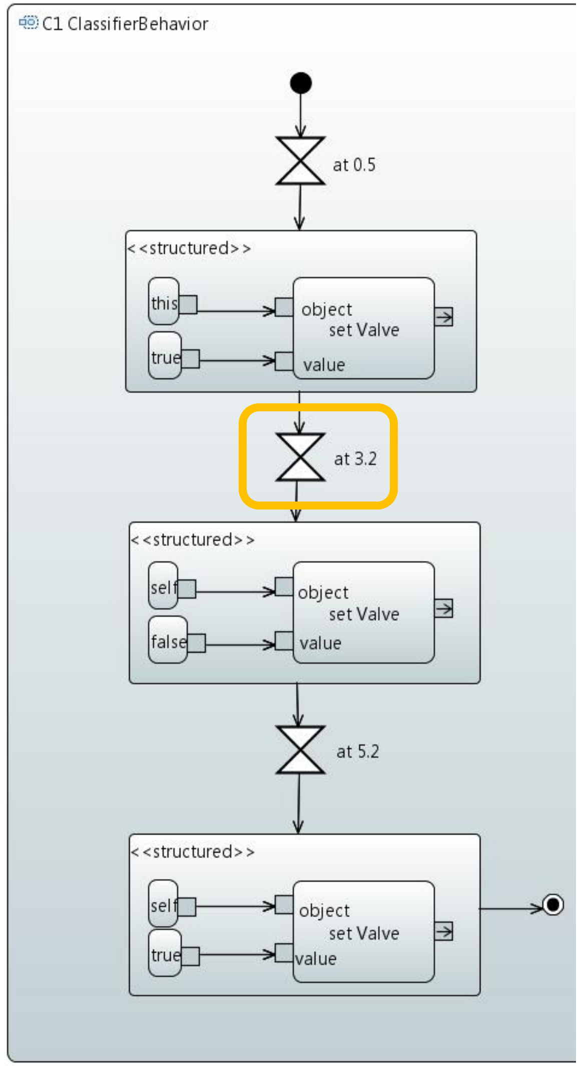
Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| Step end | 0.5 |
| | |
| | |
| | |

Valve = **true**

FMI Master

doStep (0.1)




## DE Scheduler

Current time = **0.5**

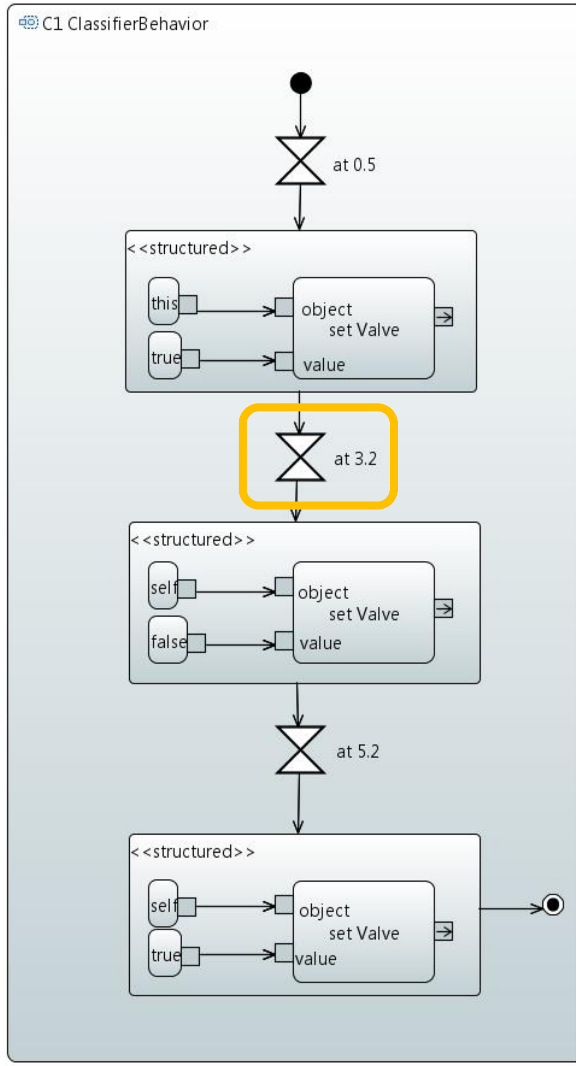
Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| Step end | 0.5 |
| Time event | 3.2 |
| | |
| | |

Valve = **true**

FMI Master

doStep (0.1)




# DE Scheduler

Current time = **0.5**

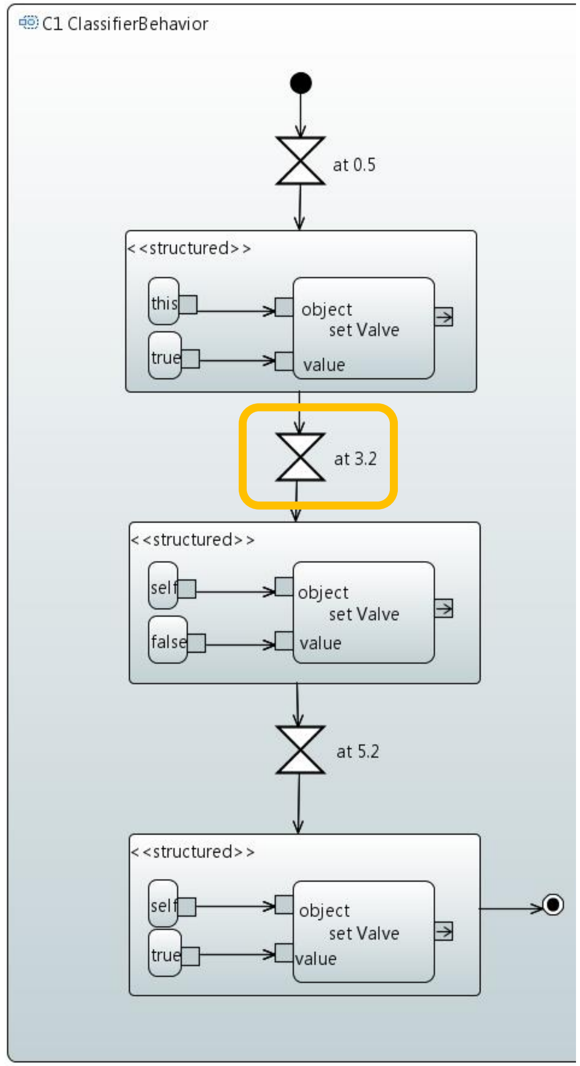
Current event = **Step end**

| Scheduler Events | Time stamp |
|---|---|
| Time event | 3.2 |
| | |
| | |
| | |

Valve = **true**

FMI Master

doStep (0.1)




# DE Scheduler

Current time = **0.5**

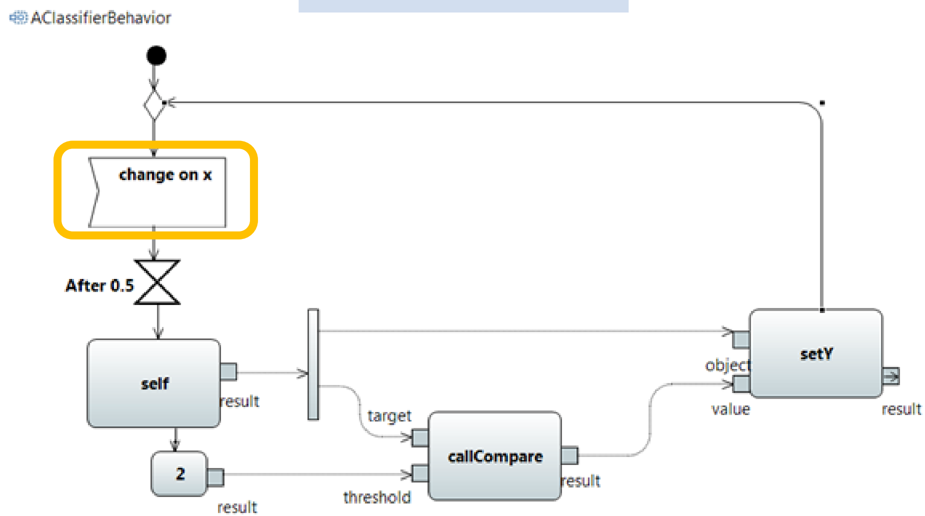
Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| Time Event | 3.2 |
| | |
| | |
| | |

Valve = **true**

FMI Master

**Object event pool**




# DE Scheduler

Current time = 0.0

Current event = NONE

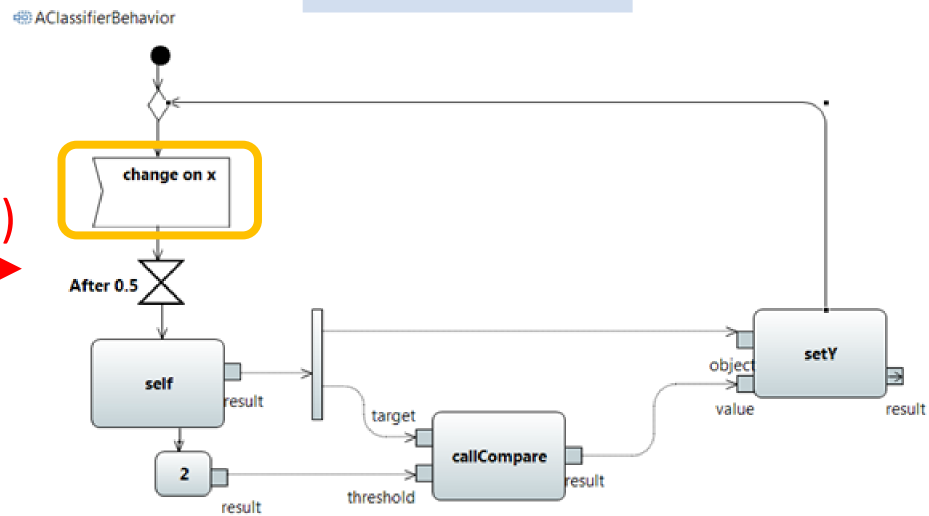
| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 1
Y = false

| Object event pool |
|---|
| |

FMI Master

setX (4)



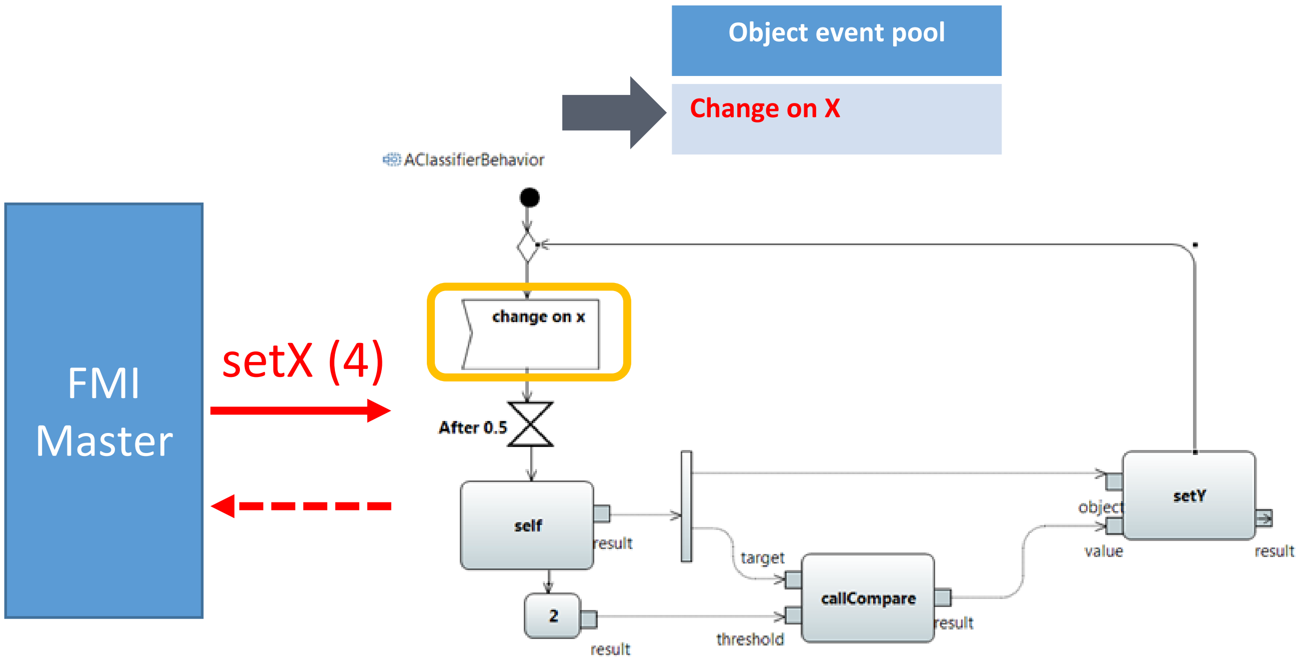

## DE Scheduler

Current time = 0.0

Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 1
Y = false




# DE Scheduler

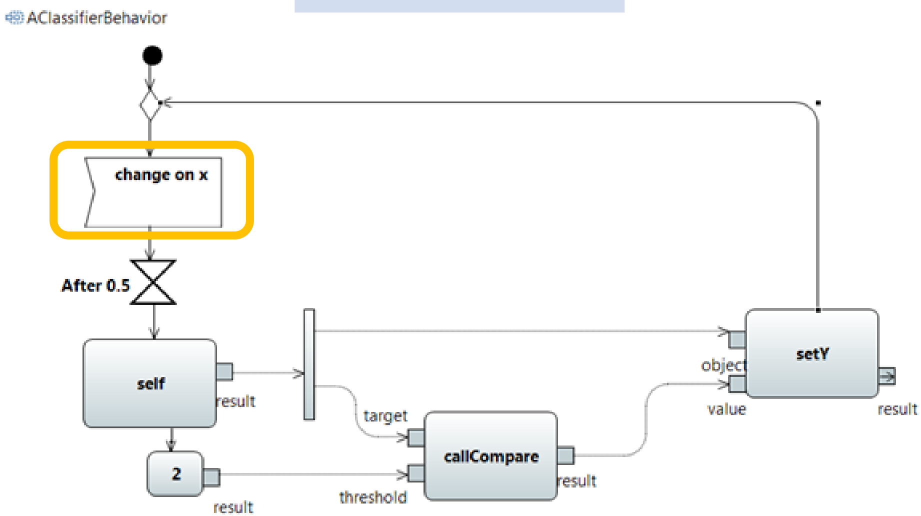
Current time = 0.0

Current event = NONE

| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = **4**
Y = false

| Object event pool |
| --- |
| Change on X |




FMI Master

## DE Scheduler

Current time = 0.0

Current event = NONE

| Scheduler Events | Time stamp |
| --- | --- |
| | |
| | |
| | |
| | |

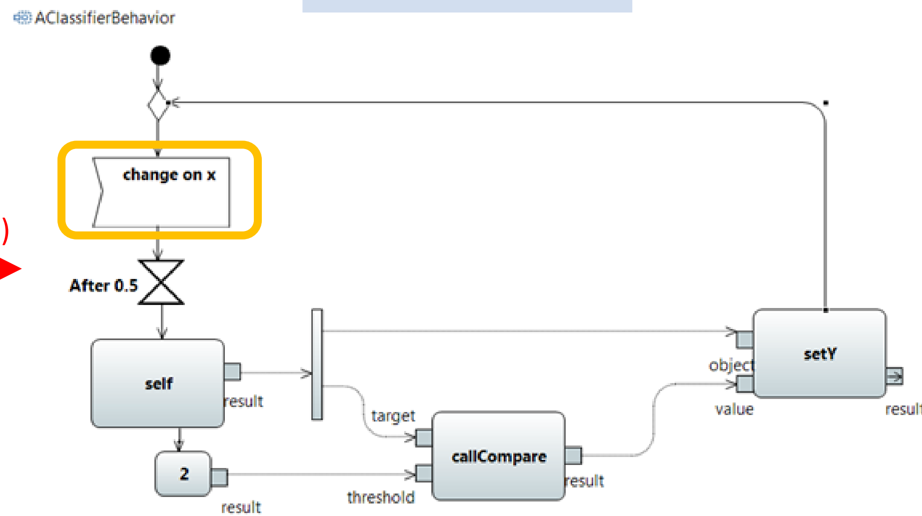
X = 4
Y = false

**Object event pool**

Change on X

FMI Master

doStep (0.1)




# DE Scheduler

Current time = 0.0

Current event = NONE

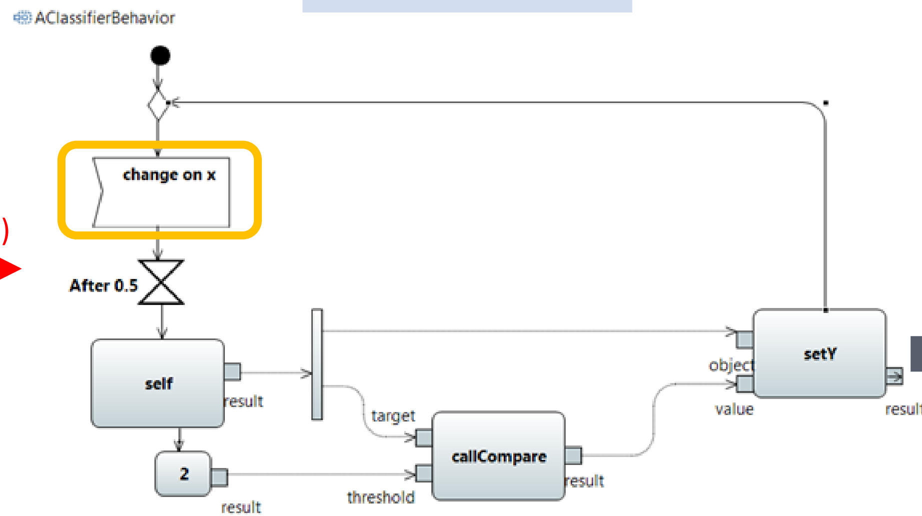
| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 4
Y = false

FMI Master

doStep (0.1)

| Object event pool |
|---|
| Change on X |




## DE Scheduler

Current time = 0.0

Current event = NONE

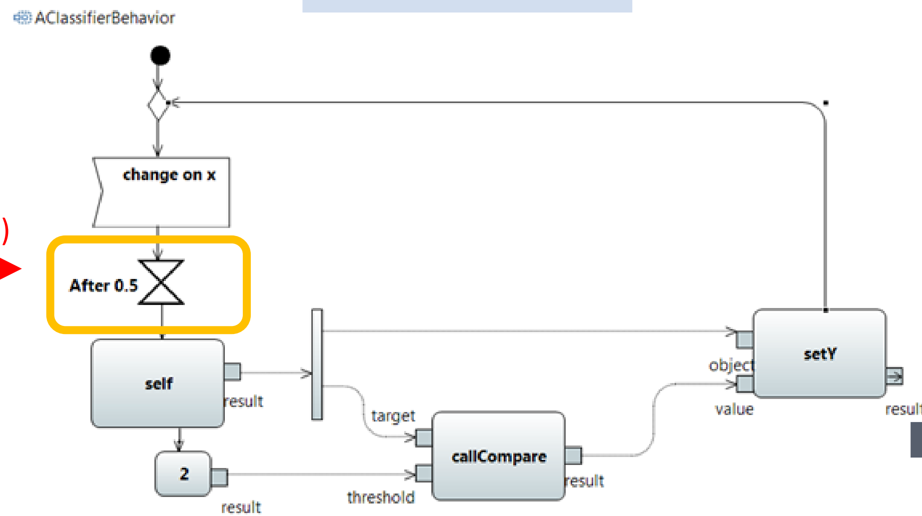
| Scheduler Events | Time stamp |
|---|---|
| Step end | 0.1 |
| | |
| | |
| | |

X = 4
Y = false

## Object event pool

FMI Master

doStep (0.1)




# DE Scheduler

Current time = 0.0

Current event = NONE

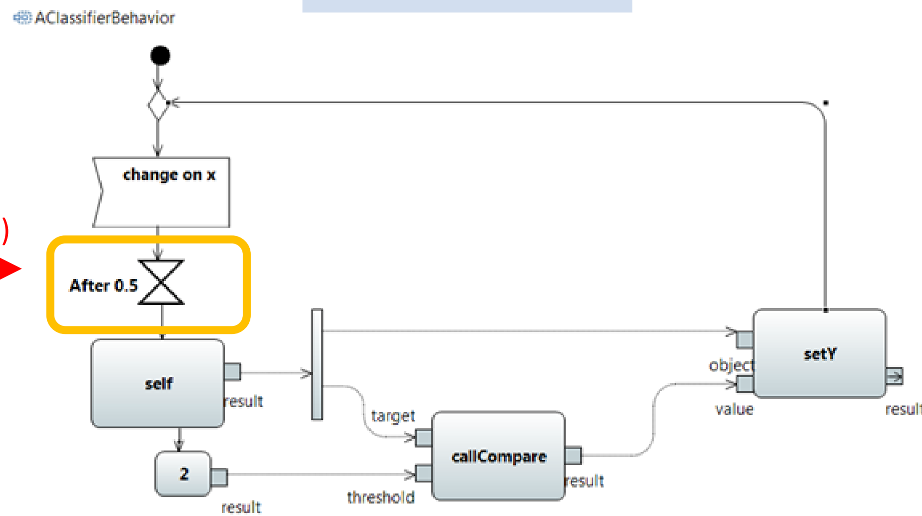
| Scheduler Events | Time stamp |
|---|---|
| Step end | 0.1 |
| Time event | 0.5 |
| | |
| | |

X = 4
Y = false

**Object event pool**

FMI Master

doStep (0.1)




# DE Scheduler

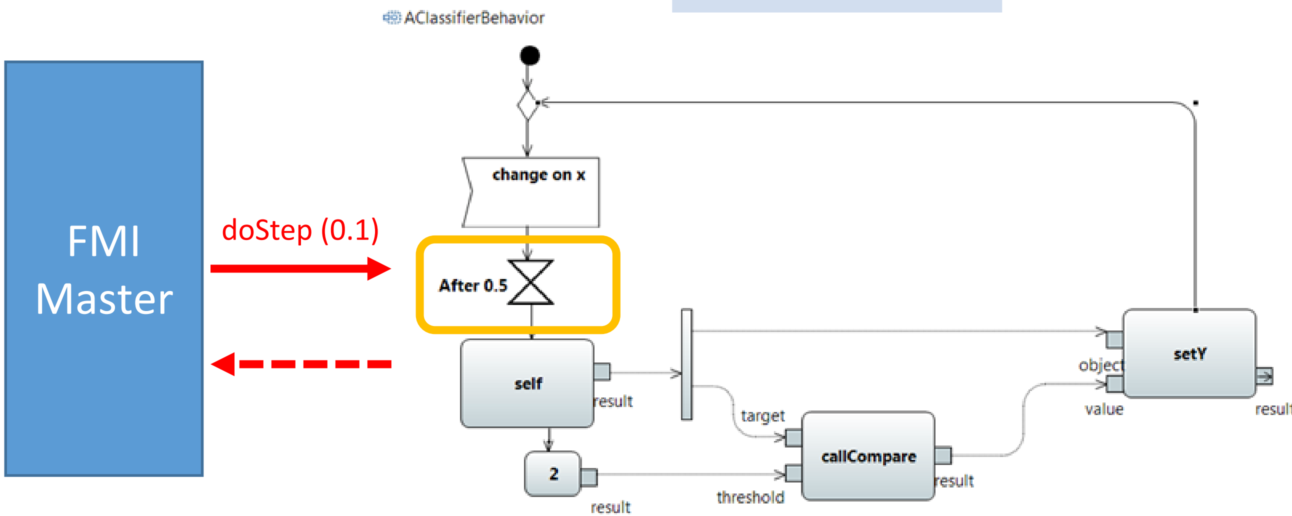
Current time = **0.1**

Current event = **Step end**

| Scheduler Events | Time stamp |
|---|---|
| Time event | 0.5 |
| | |
| | |
| | |

X = 4
Y = false

Object event pool




# DE Scheduler

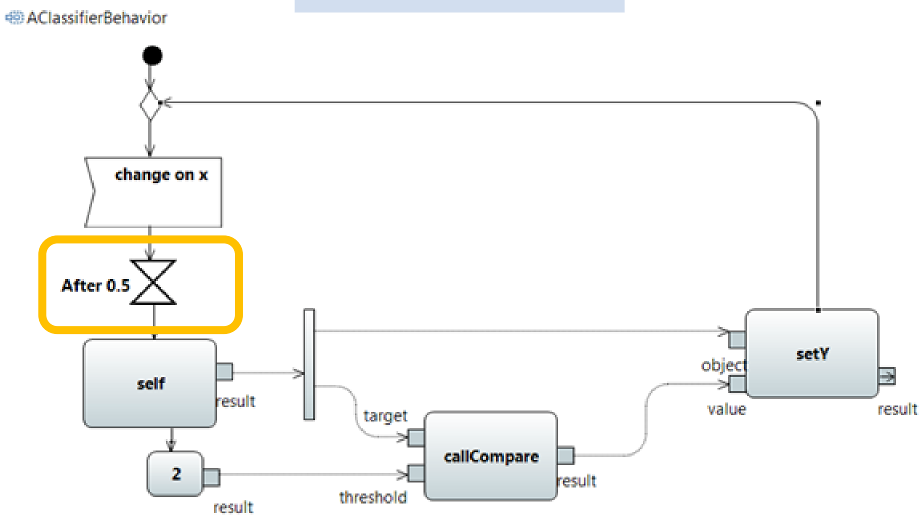
Current time = **0.1**

Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| Time event | 0.5 |
| | |
| | |
| | |

X = 4
Y = false

Object event pool




FMI Master

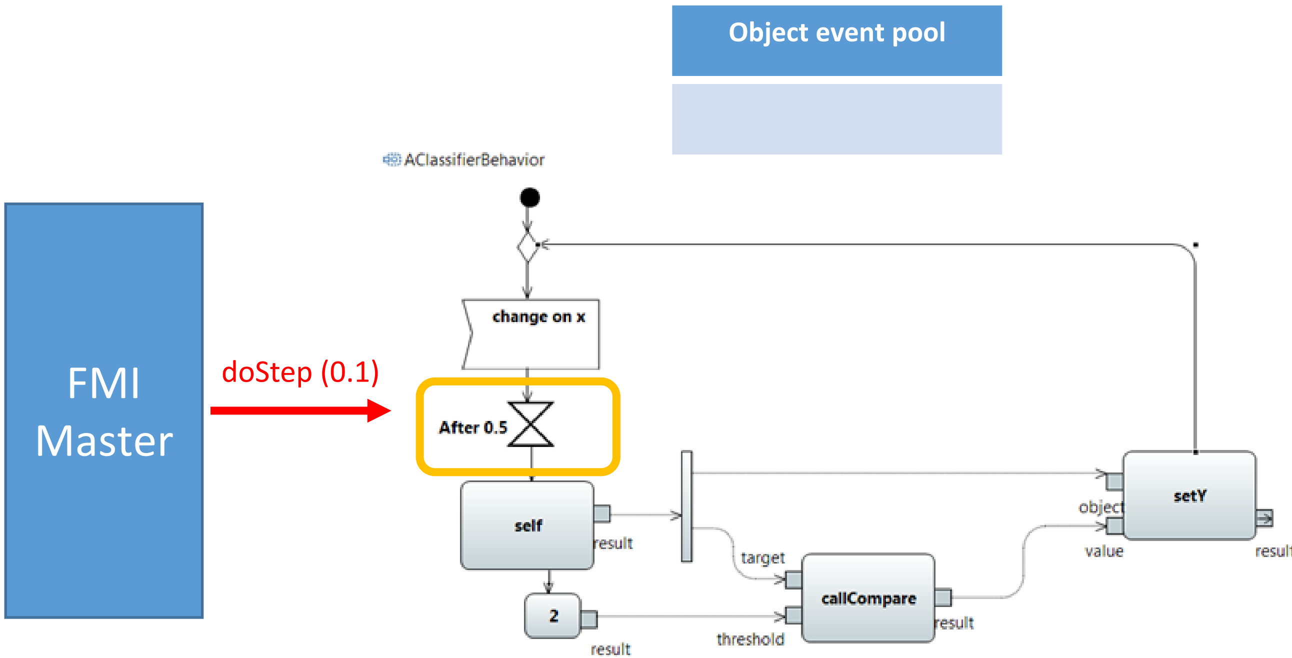
## DE Scheduler

Current time = **0.4**

Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| Time event | 0.5 |
| | |
| | |
| | |

X = 4
Y = false




# DE Scheduler

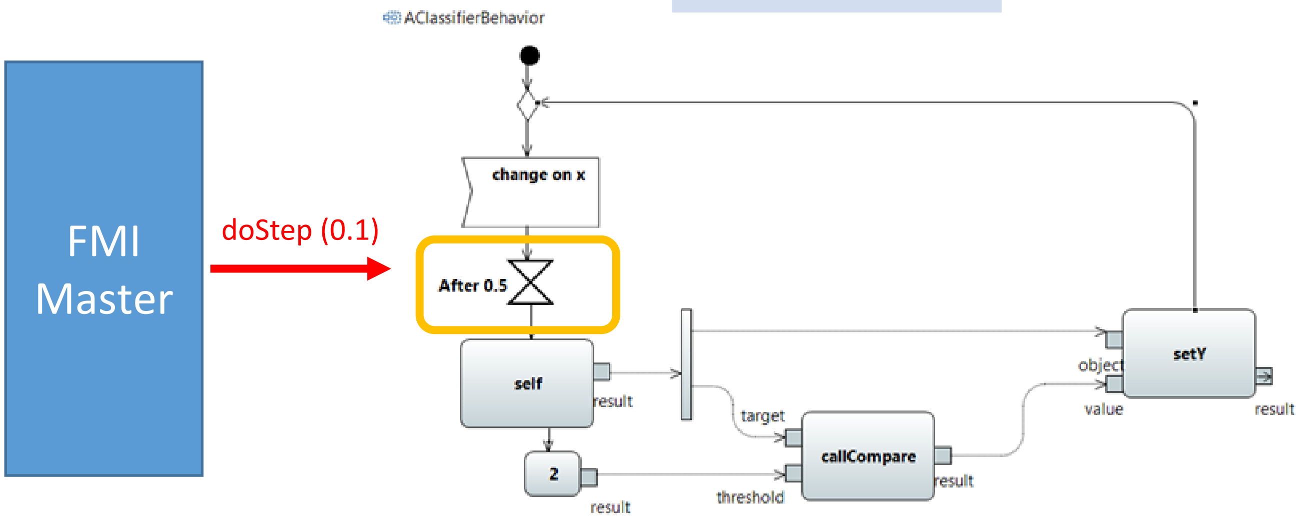
Current time = **0.4**

Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| **Time event** | **0.5** |
| | |
| | |
| | |

X = 4
Y = false

**Object event pool**




# DE Scheduler

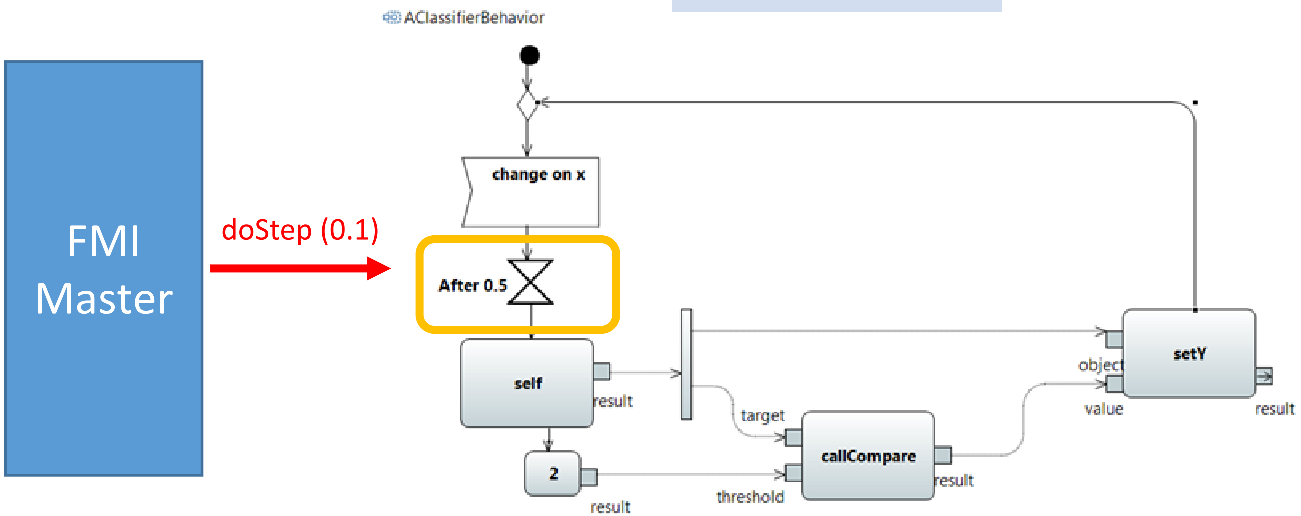
Current time = **0.4**

Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| **Time event** | **0.5** |
| **Step end** | **0.5** |
| | |
| | |

X = 4
Y = false

**Object event pool**




# DE Scheduler

Current time = **0.5**

Current event = **Time event**

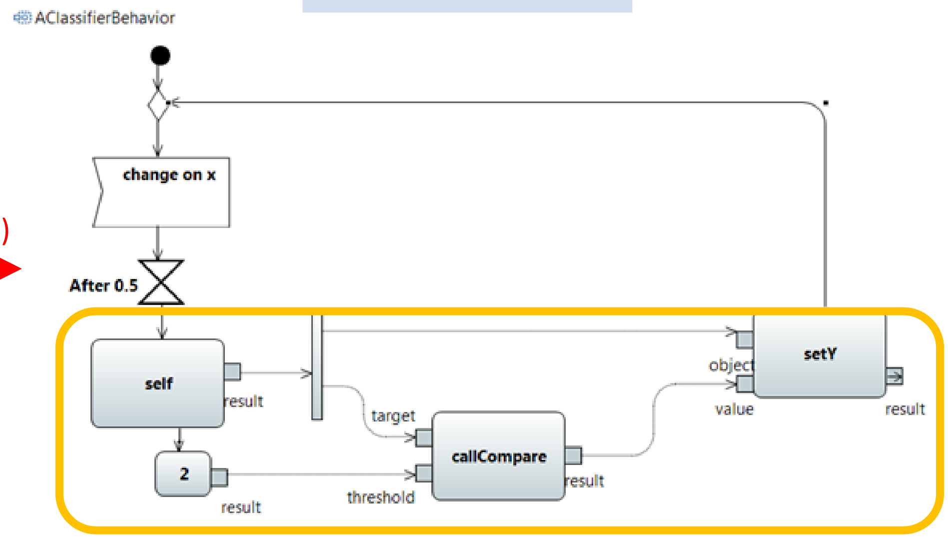
| Scheduler Events | Time stamp |
|---|---|
| Step end | 0.5 |
| | |
| | |
| | |

X = 4
Y = false

FMI Master

doStep (0.1)

**Object event pool**




## DE Scheduler

Current time = **0.5**

Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| Step end | 0.5 |
| | |
| | |
| | |

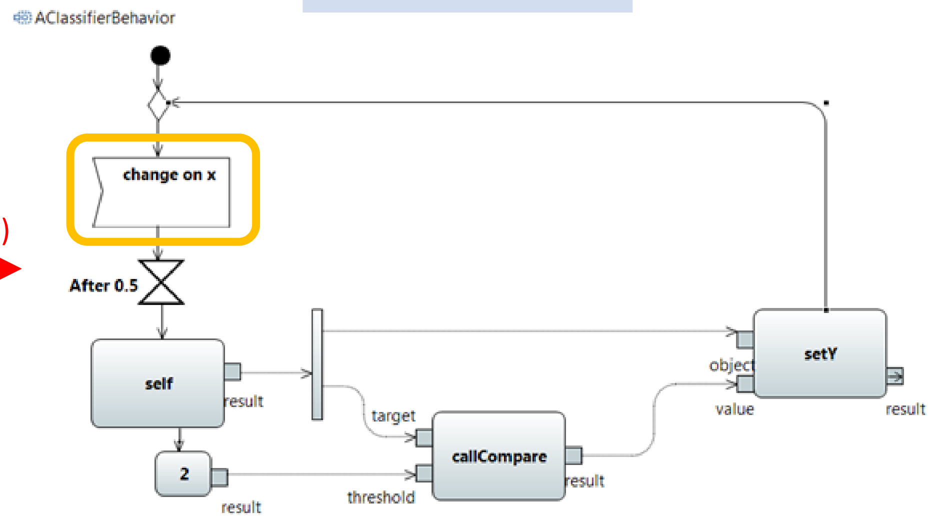
X = 4

Y = **True**

FMI Master

doStep (0.1)

**Object event pool**




# DE Scheduler

Current time = **0.5**

Current event = **NONE**

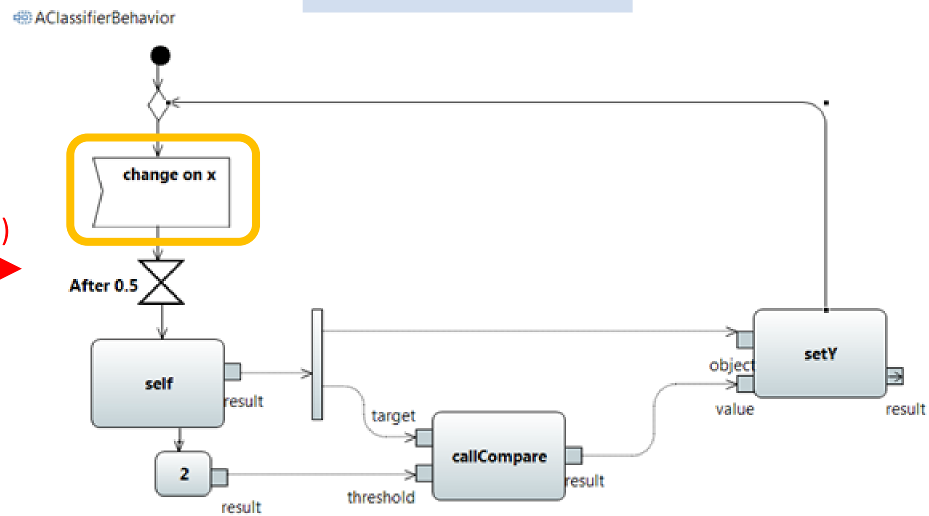
| Scheduler Events | Time stamp |
|---|---|
| Step end | 0.5 |
| | |
| | |
| | |

X = 4
Y = **True**

| Object event pool |
|---|
| |

FMI Master

doStep (0.1)




## DE Scheduler

Current time = **0.5**

Current event = **Step end**

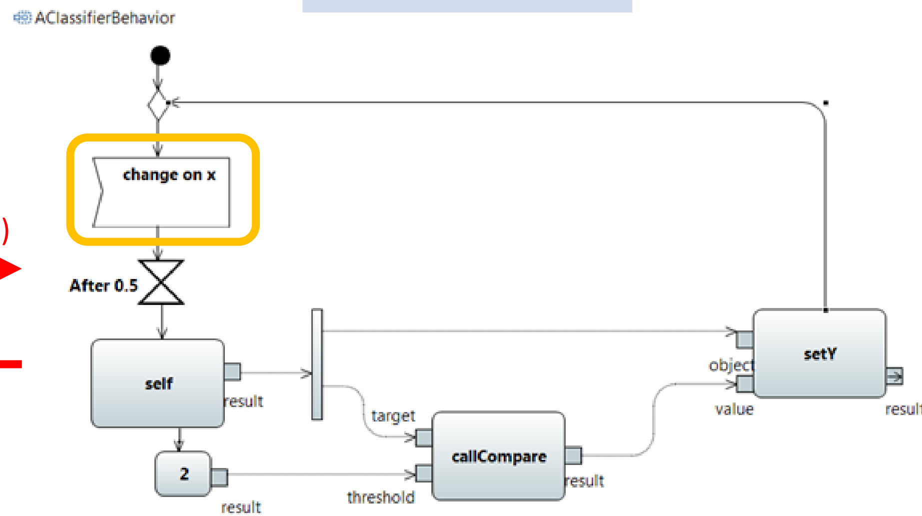
| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 4
Y = **True**

**Object event pool**

FMI Master

doStep (0.1)




# DE Scheduler

Current time = **0.5**

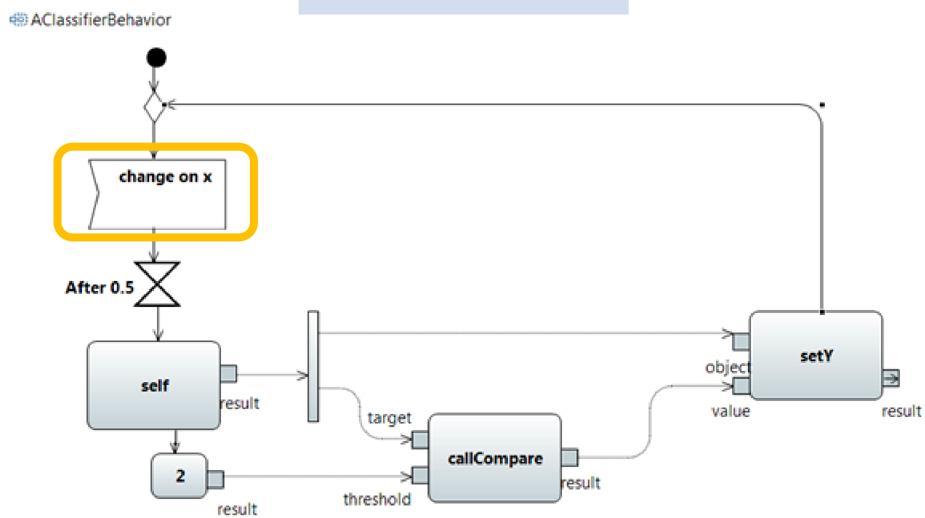
Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 4
Y = **True**

FMI Master

| Object event pool |
|---|
| |




# DE Scheduler

Current time = 0.5

Current event = NONE

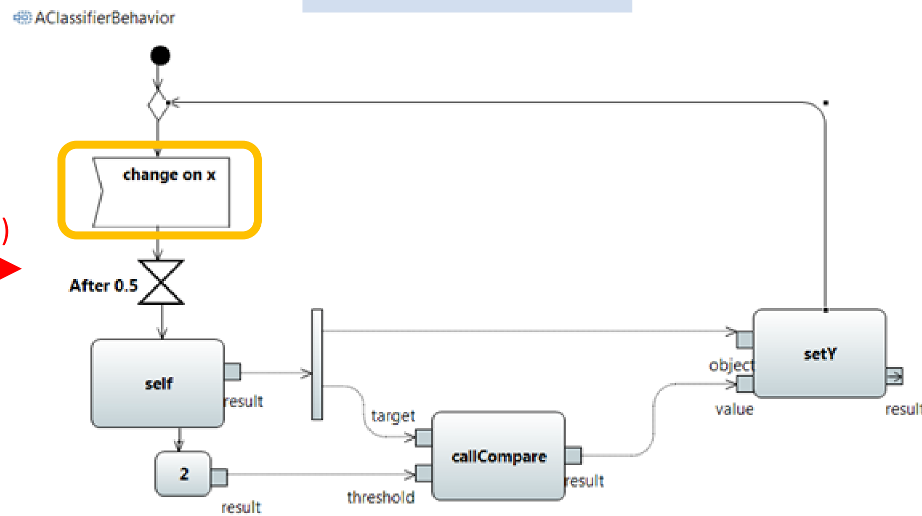
| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 4
Y = True

FMI Master

doStep (0.1)

| Object event pool |
|---|
| |




# DE Scheduler

Current time = 0.5

Current event = NONE

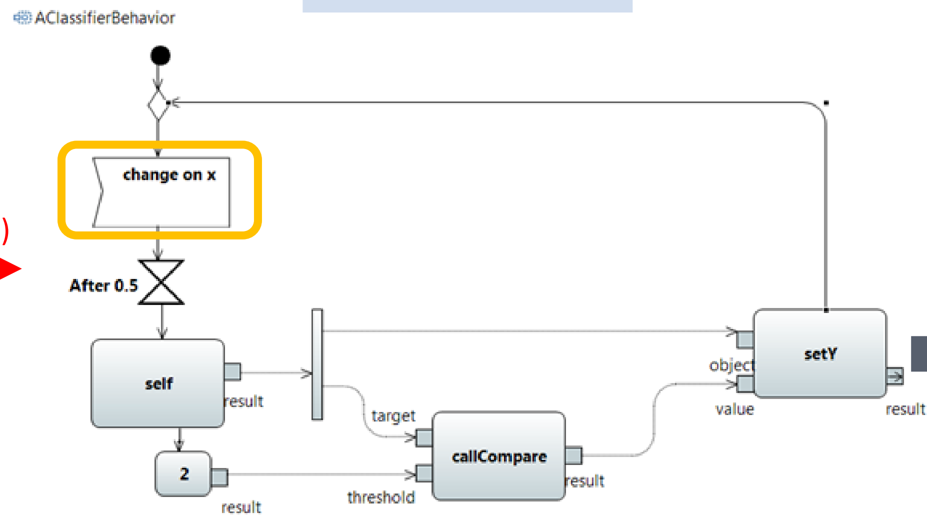
| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 4
Y = True

| Object event pool |
| --- |
| |

FMI Master

doStep (0.1)




## DE Scheduler

Current time = 0.5

Current event = NONE

| Scheduler Events | Time stamp |
| --- | --- |
| Step end | 0.6 |
| | |
| | |
| | |

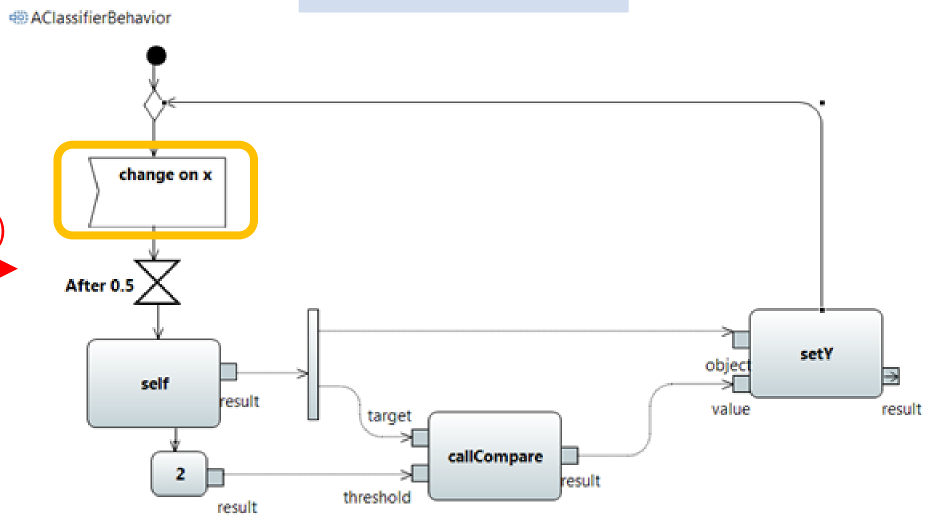
X = 4

Y = True

**Object event pool**

FMI Master

doStep (0.1)




# DE Scheduler

Current time = **0.6**

Current event = **Step end**

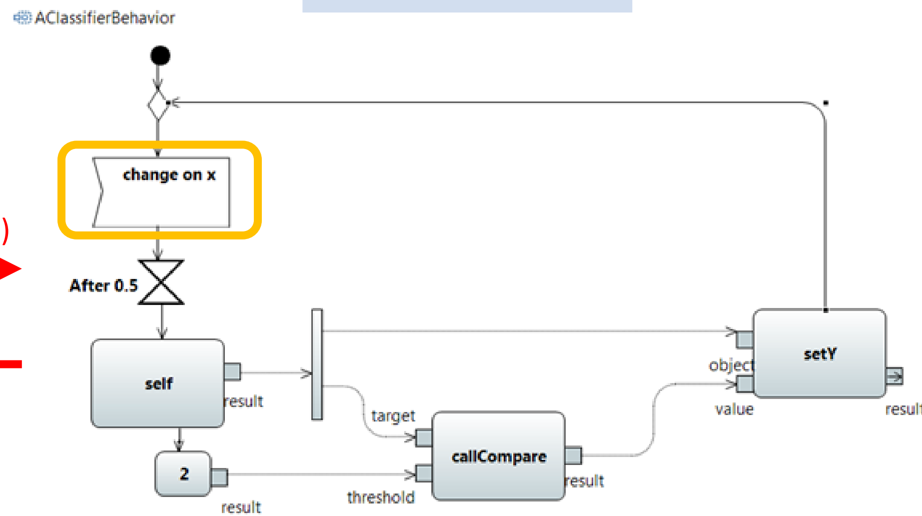
| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 4
Y = True

Object event pool

FMI Master

doStep (0.1)




# DE Scheduler

Current time = **0.6**

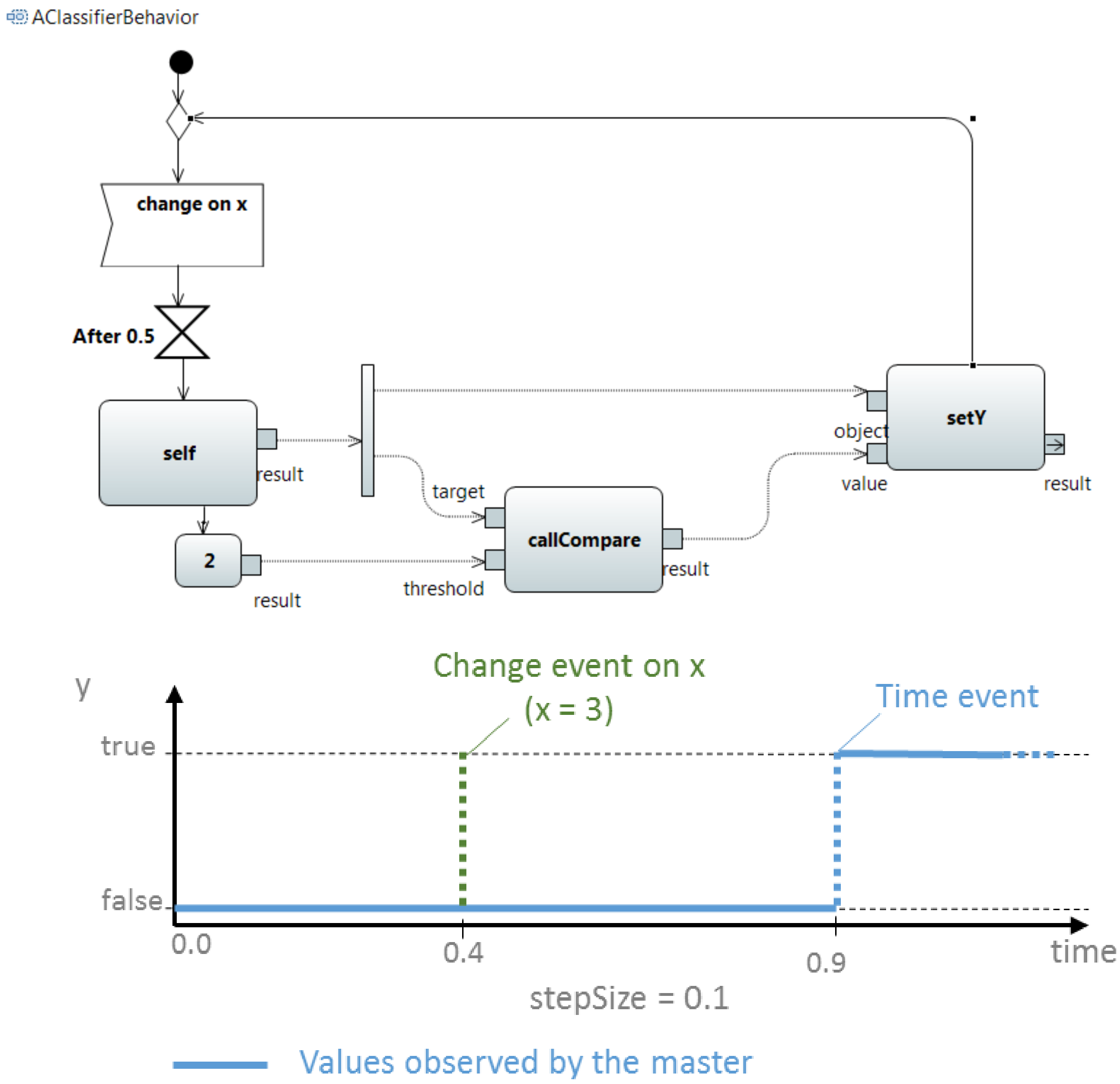
Current event = **NONE**

| Scheduler Events | Time stamp |
|---|---|
| | |
| | |
| | |
| | |

X = 4

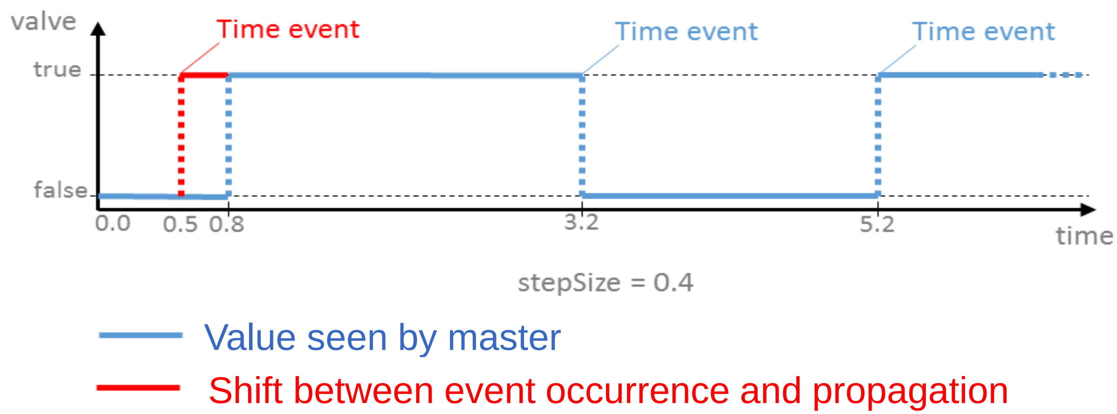
Y = True

# RESULTING SIMULATION TRACE



# OPEN QUESTIONS

- **FMI is not fully satisfying for Discrete Event simulations**




- **With a 0,4 step size, an event that occurs at T=0,5 will only be visible by other FMUs at T=0,8**
  - **Even worse : an other opposite event can occur at T=0,6 …**

- **Should we reject the step and ask for a smaller one?**
  - **Requires rollback support**
  - **Potentially : 0-time simulation steps : other FMUs would be stuck**

# INVESTIGATION TRACKS…

- **Try to group all the Discrete Event (UML) control in a single FMU**
  - **Inside the FMU, during a simulation step, rely on well defined DE execution model**
  - **Implies to generate FMU from a fUML hierarchical model**

- **Need for FMI standard API evolutions**
  - **On-going works :**

  J. P. Tavella *et al*., "Toward an accurate and fast hybrid multi-simulation with the FMI-CS standard," *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, Berlin, 2016, pp. 1-5.
  - **Specific Discrete FMUs implementing a new *hybridDoStep***
  - **Presented to FMI standard consortium**

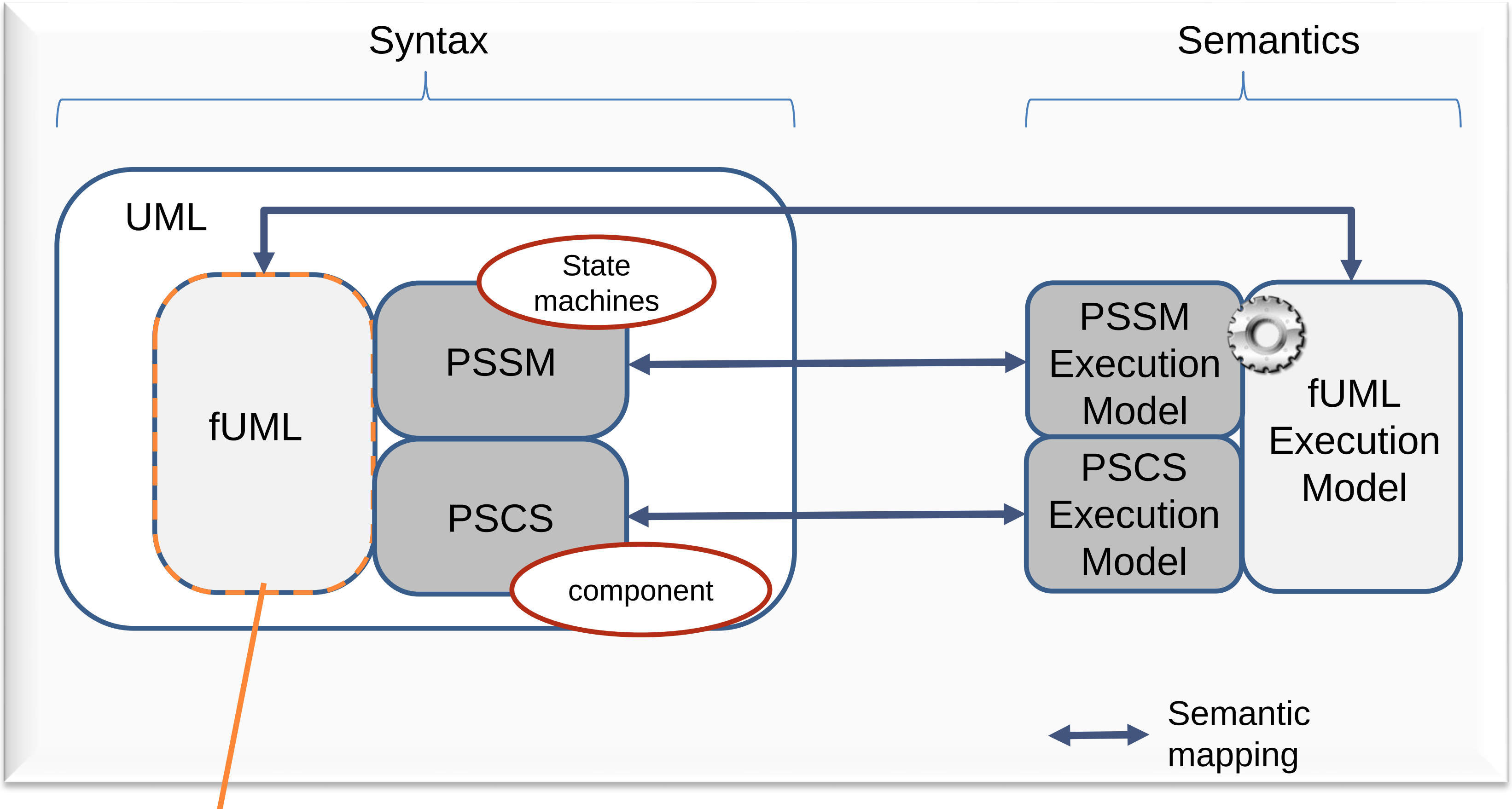
# THANK YOU

Acknowledgments to the LISE team for their direct and indirect contributions to this presentation.

**GETTING STARTED WITH MOKA:**
**HTTPS://WIKI.ECLIPSE.ORG/PAPYRUS/USERGUIDE/MODELEXECUTION**

**VIDEO TUTORIALS :**
**HTTPS://WWW.YOUTUBE.COM/CHANNEL/UCXYPOBLZC_RKLS7_K2DTWYA**

# EXECUTABLE UML OMG SPECIFICATIONS




Alf (Action Language for fUML):
- Textual surface notation for the fUML subset