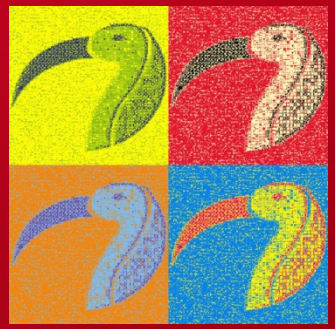
DE LA RECHERCHE À L'INDUSTRIE

cea



www.cea.fr

PAPYRUS TOOL SUPPORT FOR FMI

Ericsson Modeling Days 2016
Kista, Sweden, September 13-14, 2016

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

CEA LIST / DILS / LISE

INSTITUT CARNOT CEA LIST digiteo

FMI FOR PAPYRUS / PAPYRUS FOR FMI

- **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)
    - Heterogeneous systems
    - => many different skills and paradigms, each one being addressed using specific modeling and simulation tools.
- **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
    - Early consideration of environmental aspects in the design
    - => Early detection of possible flaws in the design
    - => Save time, save money, and make more robust designs.
  - Encompasses two complementary aspects:
    - 1. the ability to import / assemble FMUs, and co-simulate them (master tool)
    - 2. the ability to export FMUs from executable UML models.
- **Papyrus now provides FMI tool support**
  - Based on Moka, the Papyrus module for model execution
  - Encompasses both a master and an export functionality

OUTLINE

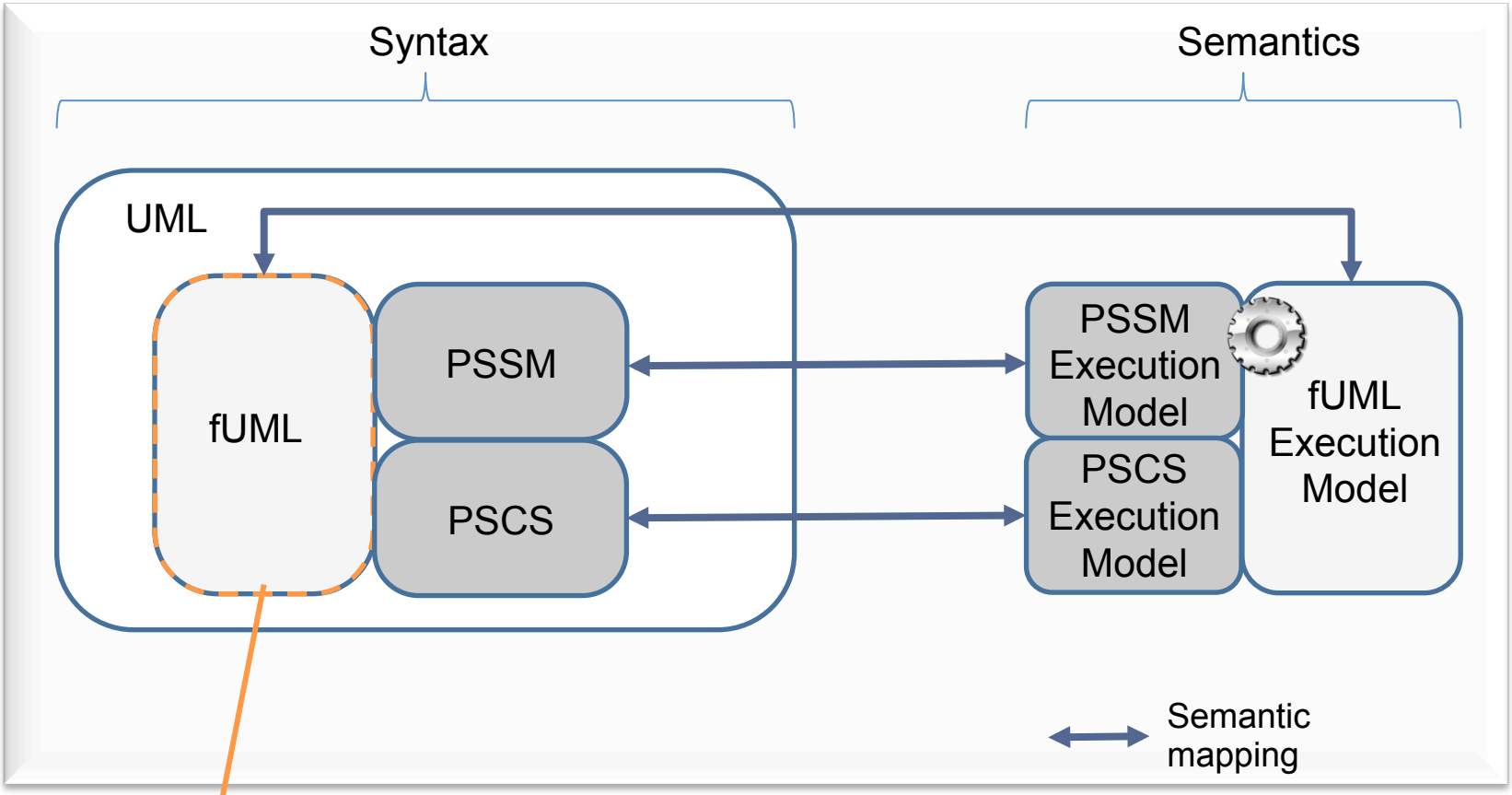
- Reminder on OMG standards for Executable Modeling
- Overview of Moka, the Papyrus module for model execution
- Papyrus tool support for FMI
  - Video demo
- Perspectives

PART I

–

REMINDER ON OMG STANDARDS FOR EXECUTABLE UML MODELING

EXECUTABLE UML OMG SPECIFICATIONS

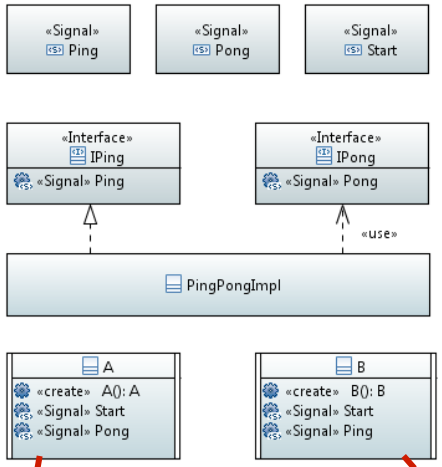


Alf (Action Language for fUML):
- Textual surface notation for the fUML subset

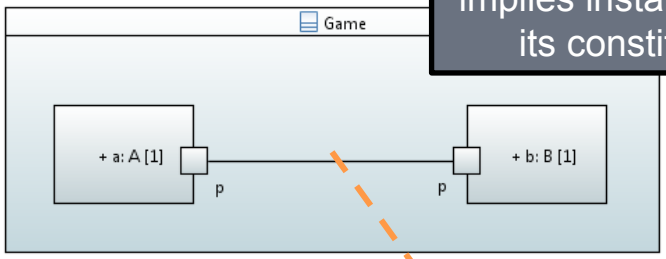
KEY SEMANTIC ASPECTS

Structure

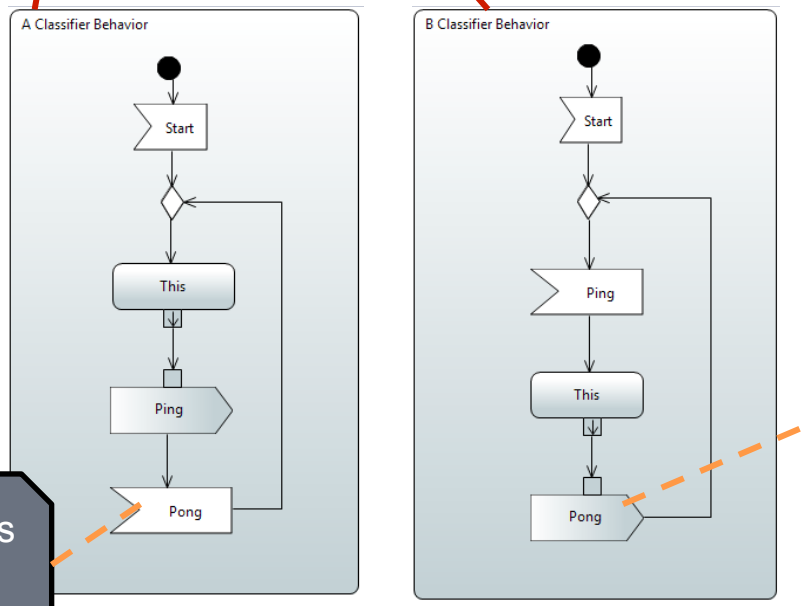
Behavior



1. Class diagram (~ BDD)



2. Composite structure diagram (~ IBD)



3. Activity diagrams

Instantiation of a composite structure implies instantiation of its constituents

Instantiation of an active class implies starting of its behavior

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

PART II

–

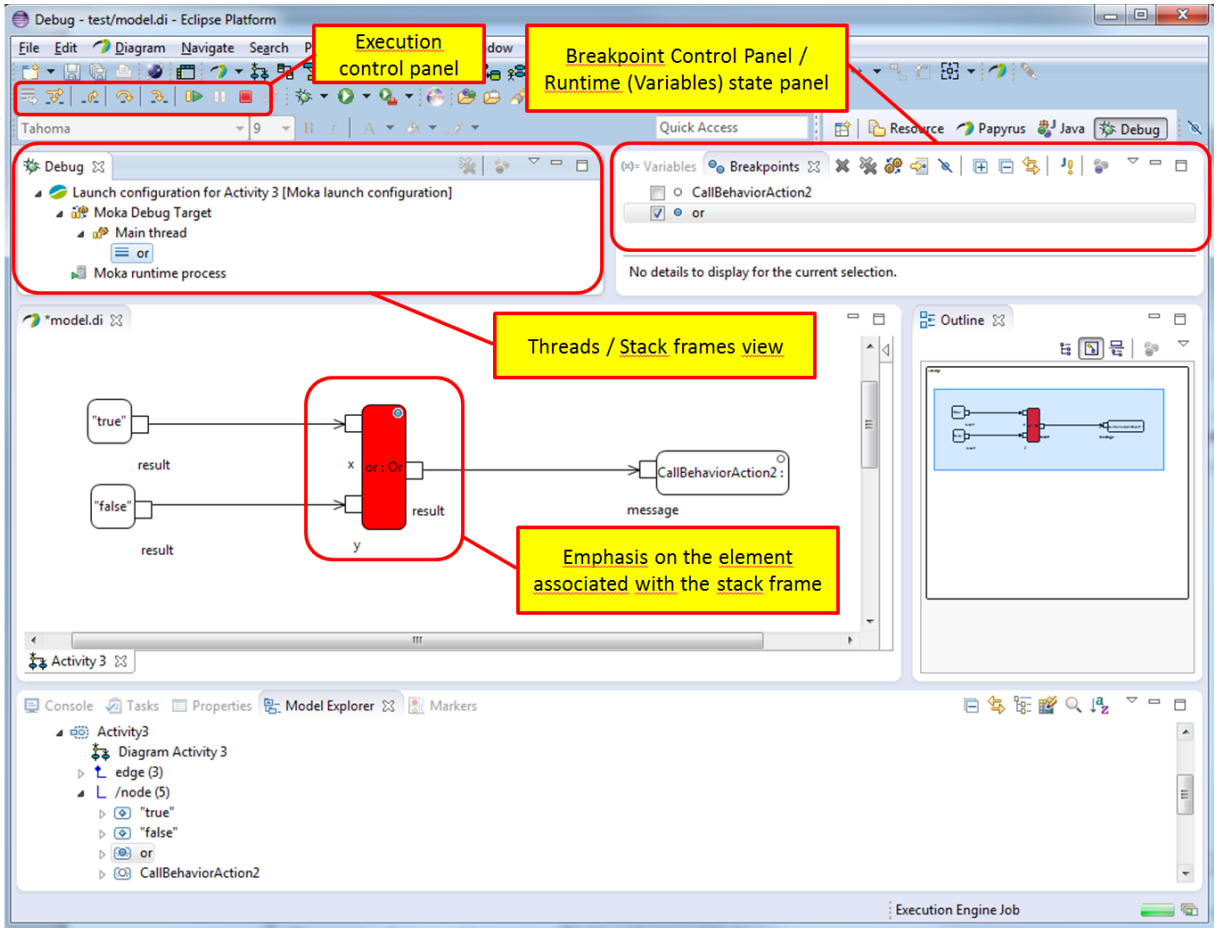
OVERVIEW OF MOKA, THE PAPYRUS MODULE FOR MODEL EXECUTION

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 and PSCS execution models (PSSM 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)

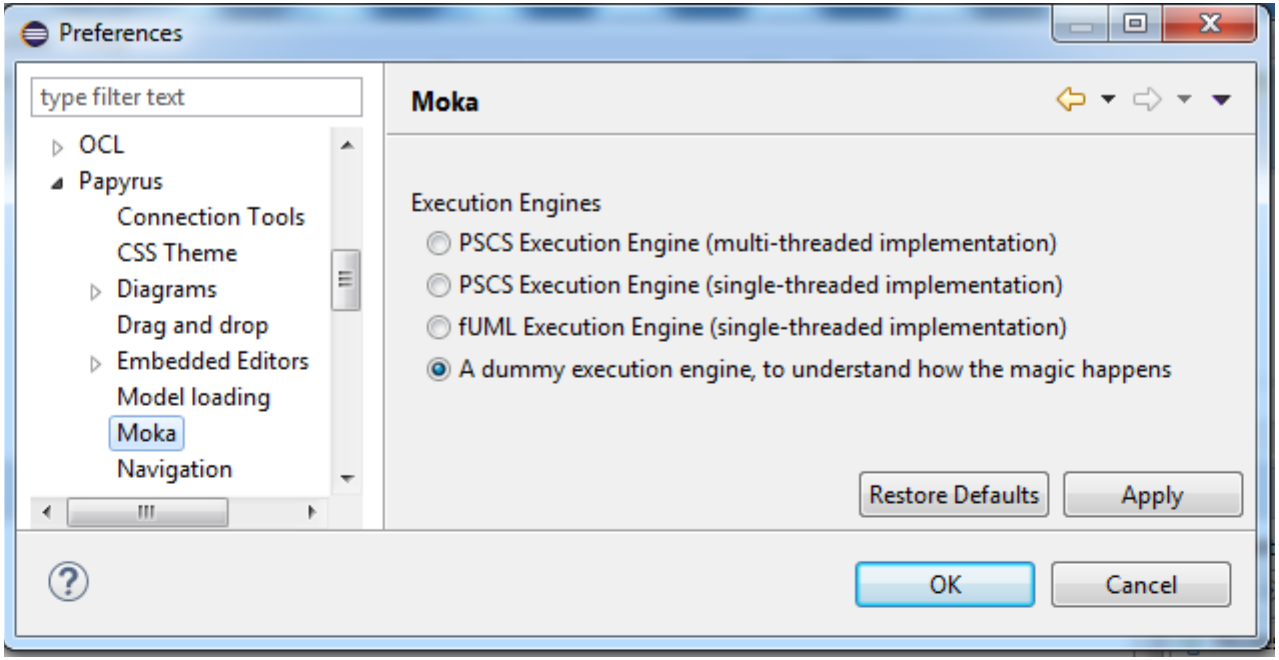
CONNECTION WITH THE ECLIPSE DEBUG FRAMEWORK

- **Controlling and Observing executions**



CONNECTION WITH THE ECLIPSE DEBUG FRAMEWORK

- **Multiple execution engines can be registered**



MOKA: OVERVIEW

- **Papyrus plug-in 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 and PSCS execution models (PSSM 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)
- **NEW : Support for FMI Co-Simulation standard**
  - Export of FMUs from executable UML models
  - Ability to import and assemble FMUs, co-simulate them with the built-in Moka master, and visualize simulation traces on XY charts.

PART III

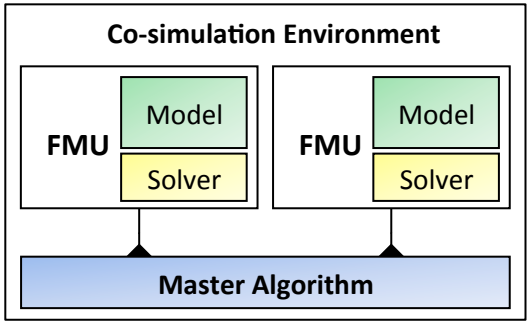
–

PAPYRUS TOOL SUPPORT FOR FMI

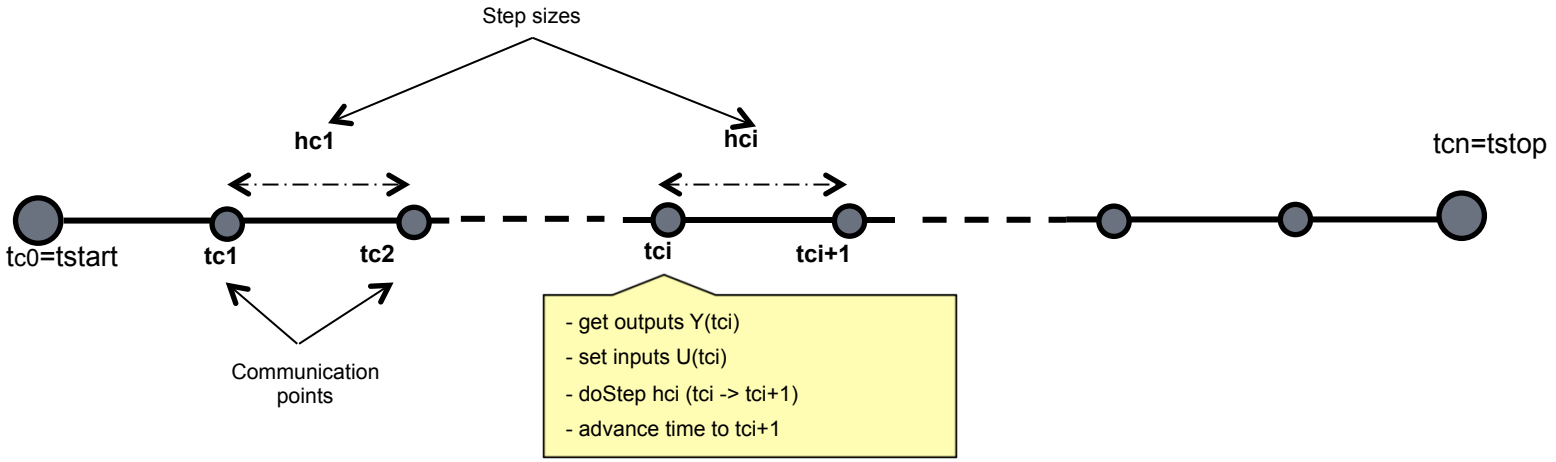
FUNCTIONAL MOCK-UP INTERFACE (FMI)

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

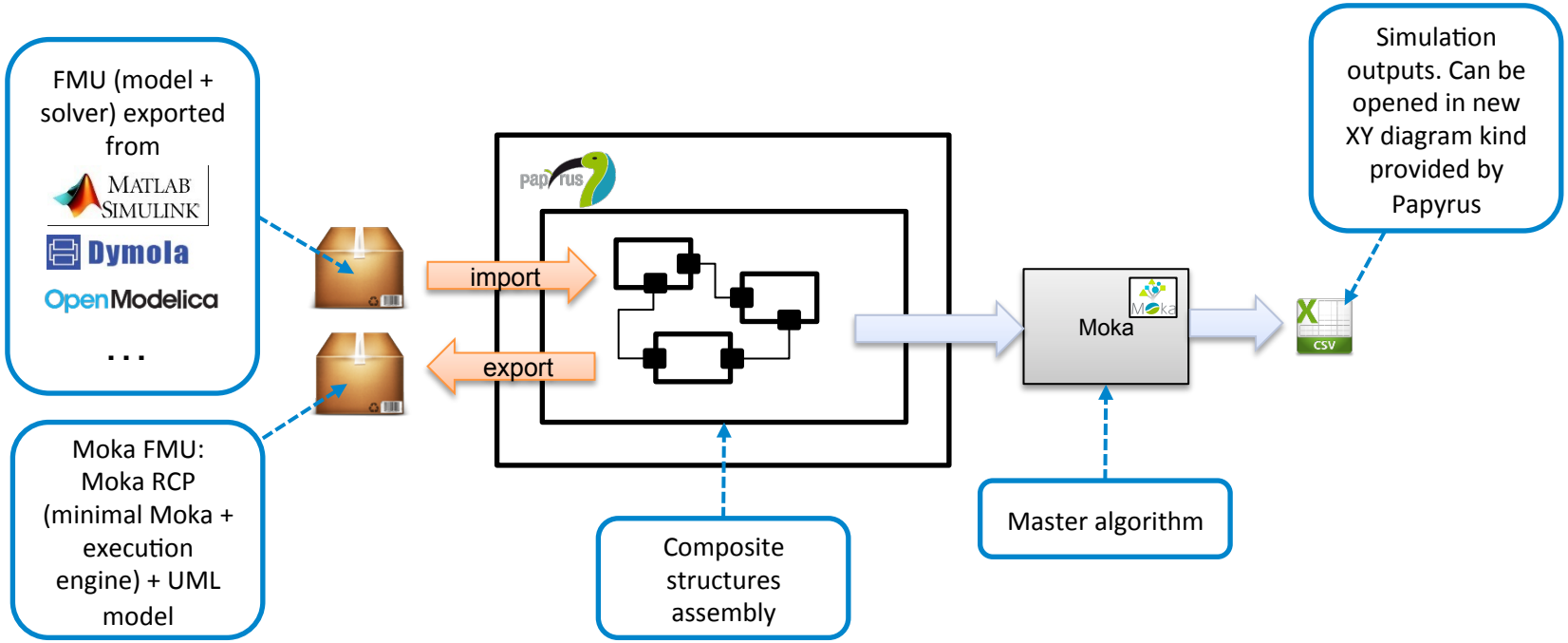
- **An FMU as to provide a standard binary interface as shared library (dll/.so)**
  - **Set Inputs**
  - **Get outputs**
  - **Do Step (stepSize)**



The simulation Master synchronizes and orchestrates the FMUs

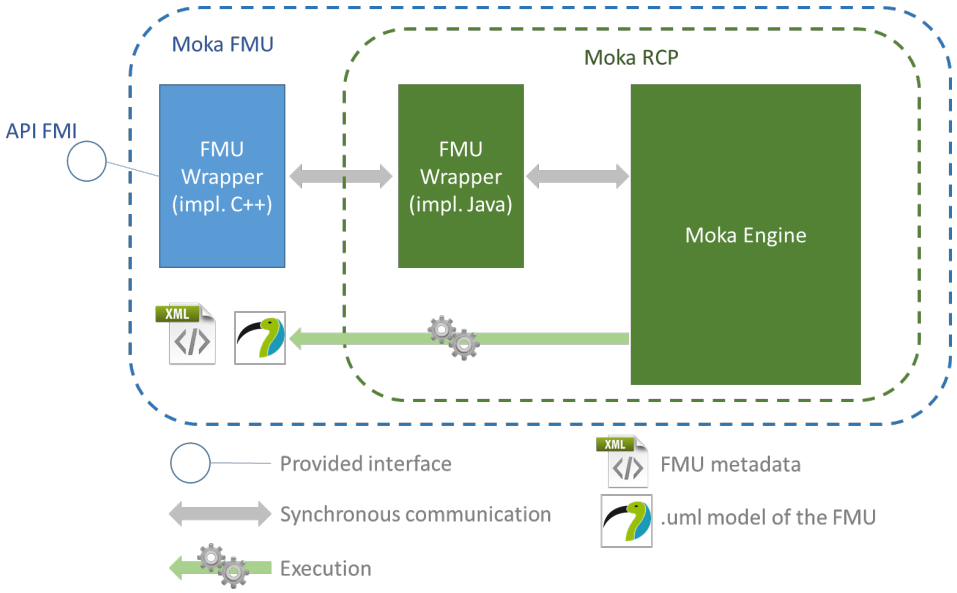


OVERVIEW OF PAPYRUS TOOL SUPPORT FOR FMI

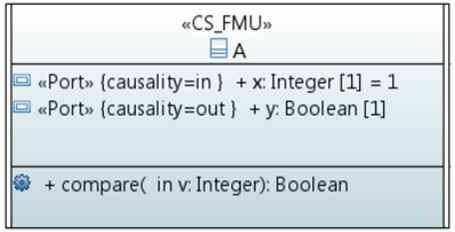


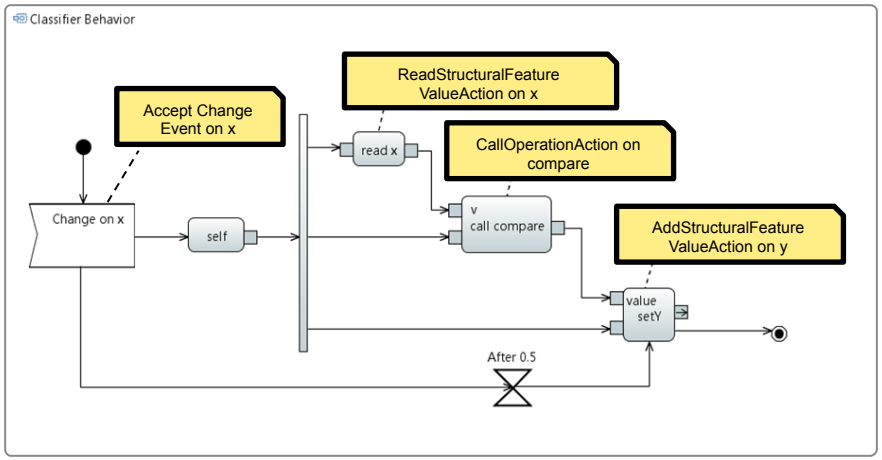
THE FMU EXPORT FUNCTIONNALITY

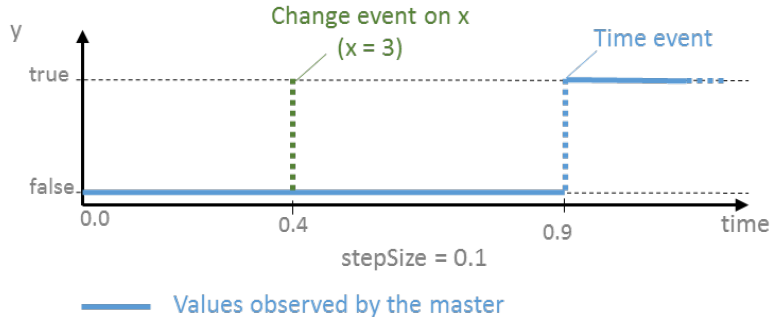
- **Architecture of exported FMUs**

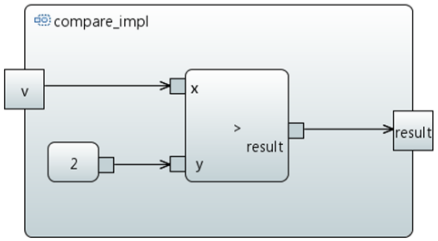


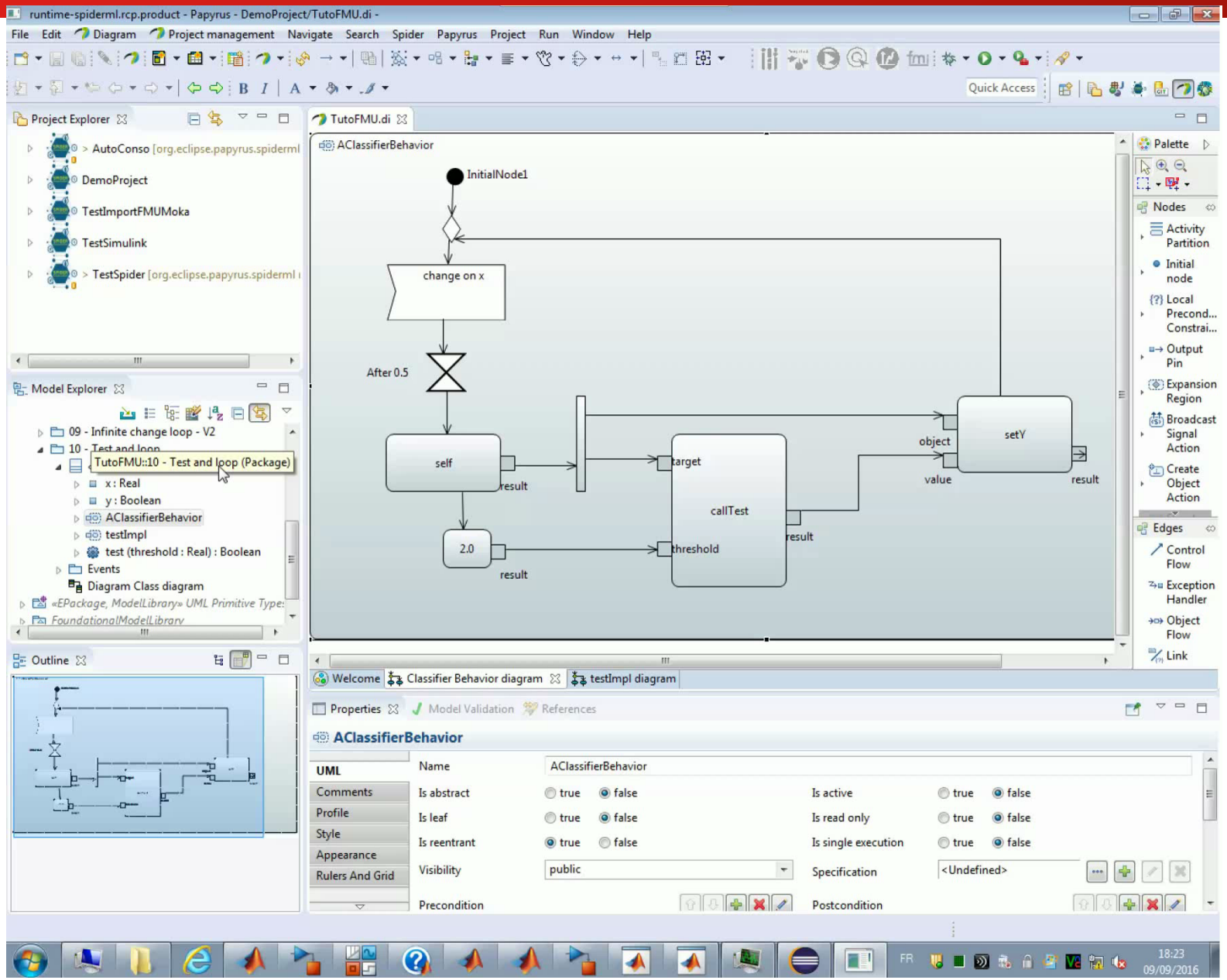
- **Models are made following some guidelines:**
  - Based on a minimal UML profile for FMI
  - FMUs modeled with active classes, and corresponding classifier behavior as an activity
  - Timing aspects handled with « TimeEvents » (requires some extensions to fUML)
  - Reactive aspects (i.e., changes on inputs of the FMU) handled with « ChangeEvents » (extensions to fUML)

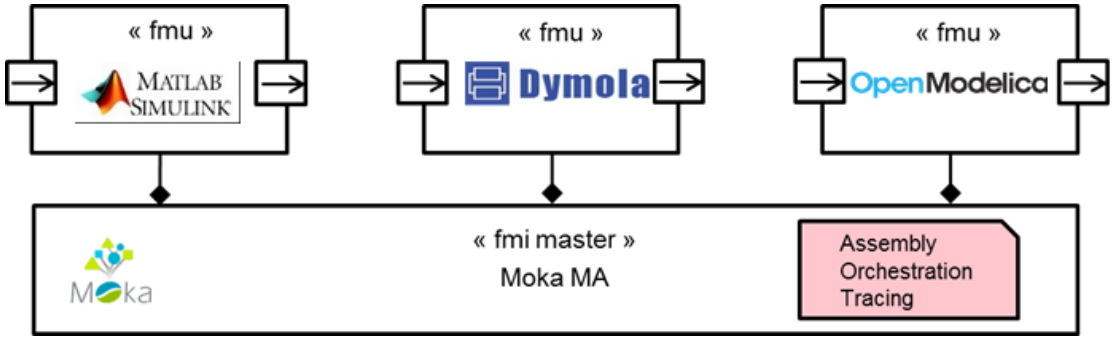

«CS_FMU»
A
«Port» {causality=in } + x: Integer [1] = 1
«Port» {causality=out } + y: Boolean [1]
+ compare(in v: Integer): Boolean


Classifier Behavior
Accept Change Event on x
ReadStructuralFeature ValueAction on x
CallOperationAction on compare
AddStructuralFeature ValueAction on y
Change on x
self
read x
v
call compare
value
setY
After 0.5

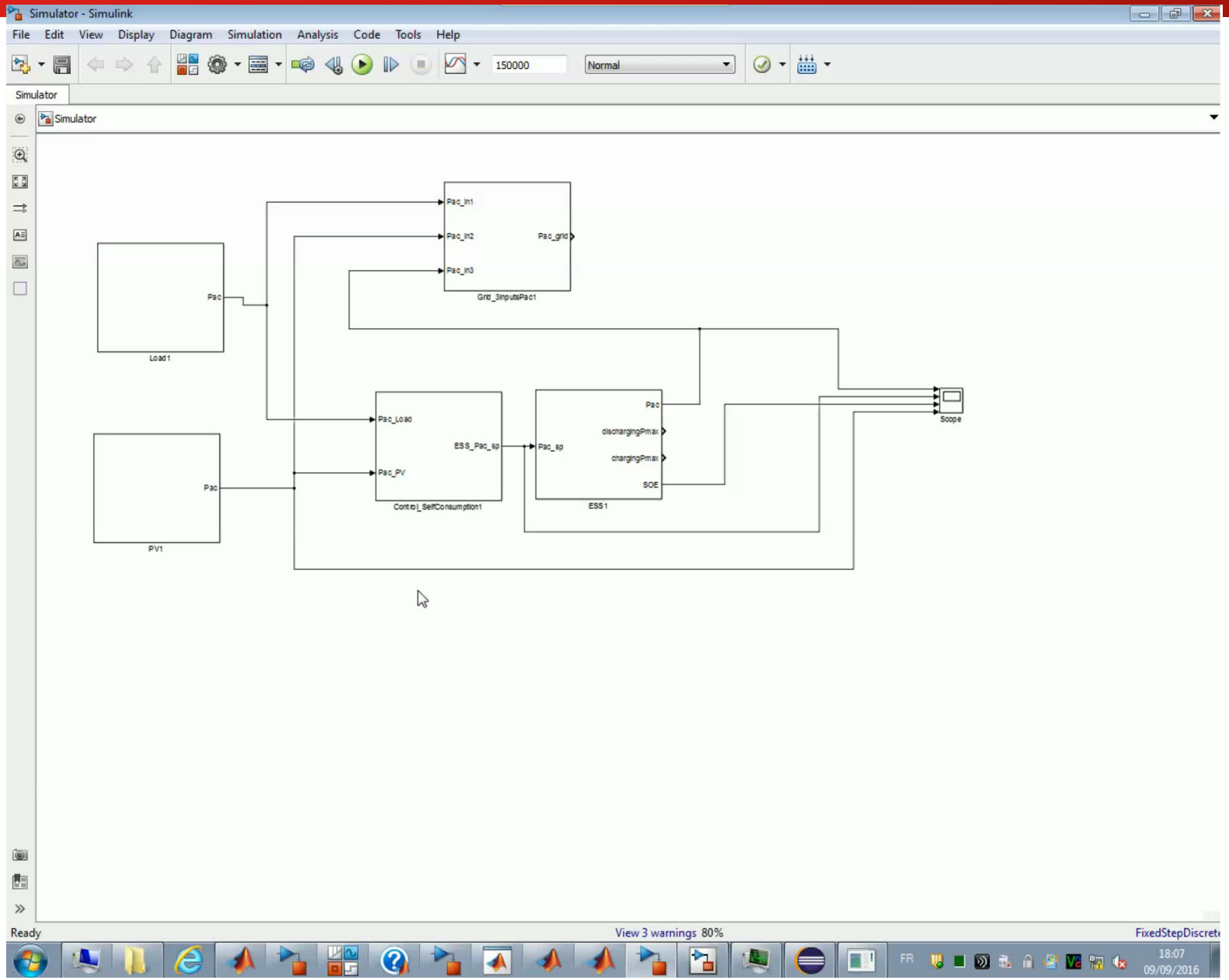

y
Change event on x
(x = 3)
Time event
true
false
0.0
0.4
0.9
time
stepSize = 0.1
Values observed by the master


compare_impl
v
x
>
result
result
2
y


runtime-spiderml.rcp.product - Papyrus - DemoProject/TutoFMU.di -
File Edit Diagram Project management Navigate Search Spider Papyrus Project Run Window Help
Quick Access
Project Explorer
AutoConso [org.eclipse.papyrus.spiderml
DemoProject
TestImportFMUMoka
TestSimulink
TestSpider [org.eclipse.papyrus.spiderml
Model Explorer
09 - Infinite change loop - V2
10 - Test and loop
TutoFMU::10 - Test and loop (Package)
x : Real
y : Boolean
AClassifierBehavior
testImpl
test (threshold : Real) : Boolean
Events
Diagram Class diagram
Outline
TutoFMU.di
AClassifierBehavior
InitialNode1
change on x
After 0.5
self
result
target
callTest
result
object
setY
value
result
2.0
result
threshold
Palette
Nodes
Activity Partition
Initial node
Local Precond... Constrai...
Output Pin
Expansion Region
Broadcast Signal Action
Create Object Action
Edges
Control Flow
Exception Handler
Object Flow
Link
Welcome
Classifier Behavior diagram
testImpl diagram
Properties
Model Validation
References
AClassifierBehavior
UML
Comments
Profile
Style
Appearance
Rulers And Grid
Name
AClassifierBehavior
Is abstract
Is leaf
Is reentrant
Is active
Is read only
Is single execution
Visibility
public
Specification
<Undefined>
Precondition
Postcondition
FR
18:23
09/09/2016



- **Key features:**
  - **Ability to import FMUs from FMI 2.0 compliant tools**
  - **Definition of the co-simulation graphs (i.e., assembly of FMUs + configuration of simulation runs)**
  - **Master algorithm specified by an executable UML models, along with a dedicated model library**
    - Fixed step size, no usage of rollbacks, but we have some plans to go further…
  - **Visualization of co-simulation results with XY charts**


Simulator - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
150000
Normal
Simulator
Simulator
Pac_In1
Pac_In2
Pac_In3
Pac_grid
Grid_3InputsPac1
Pac
Load1
Pac_Load
Pac_PV
ESS_Pac_sp
Control_SelfConsumption1
Pac_sp
Pac
dischargingPmax
chargingPmax
SOE
ESS1
Scope
Pac
PV1
Ready
View 3 warnings 80%
FixedStepDiscrete
FR
18:07
09/09/2016

PART IV

–

PERSPECTIVES

PERSPECTIVES (SHORT / MID TERM)

- **FMU modeling and export:**
  - Support of rollback
  - Support of state machines
- **Master tool:**
  - Native cosimulation of fUML and FMI
  - Architecture to ease the integration of new master algorithms (might be based on models on those masters, but raises performance issues)
- **Tooling:**
  - Papyrus customization for modeling of FMUs and configuration of simulation runs
  - Consolidate integration of the XY visualisation tool
  - Debug capabilities at the master level (similar to those available on unitary FMUs. Cf. video demo)

PERSPECTIVES (LONGER TERM)

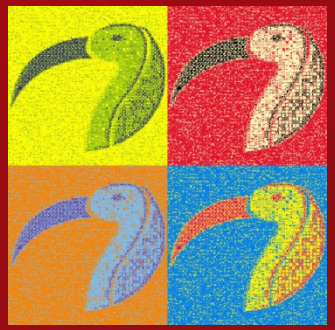
- **Focus so far: Simulation engineering, with technological bricks getting maturity on:**
  - Execution and debugging of UML models (and their variants)
  - Ergonomy of the modeling environement
  - Support for cosimulation aspects
  - Visualisation capabilities

- **Next steps: Leverage the simulation engineering capabilities in a more global Model-Based System Engineering approach, relying on other technological bricks of the Papyrus ecosystem:**
  - Requirement engineering (Links between simulation models and requirement coverage / satisfaction)
  - Testing engineering (Links between simulation models/runs and test case generation/verdict computation provided by model-based testing techniques)
  - Software engineering (Refinement of executable models into deployable software artifacts)
  - Etc.

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

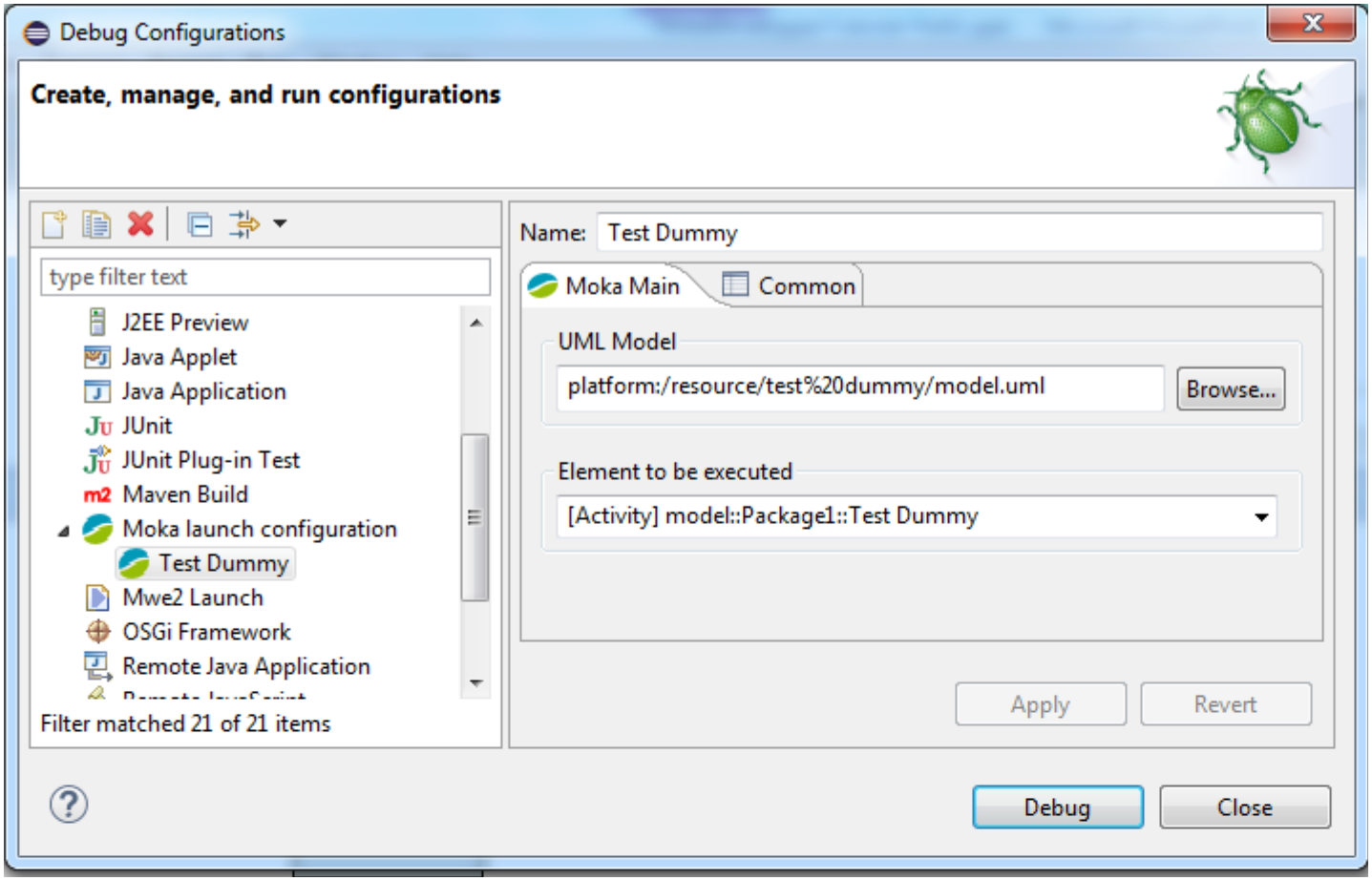
VIDEO TUTORIAL AVAILABLE SOON



BACKUP

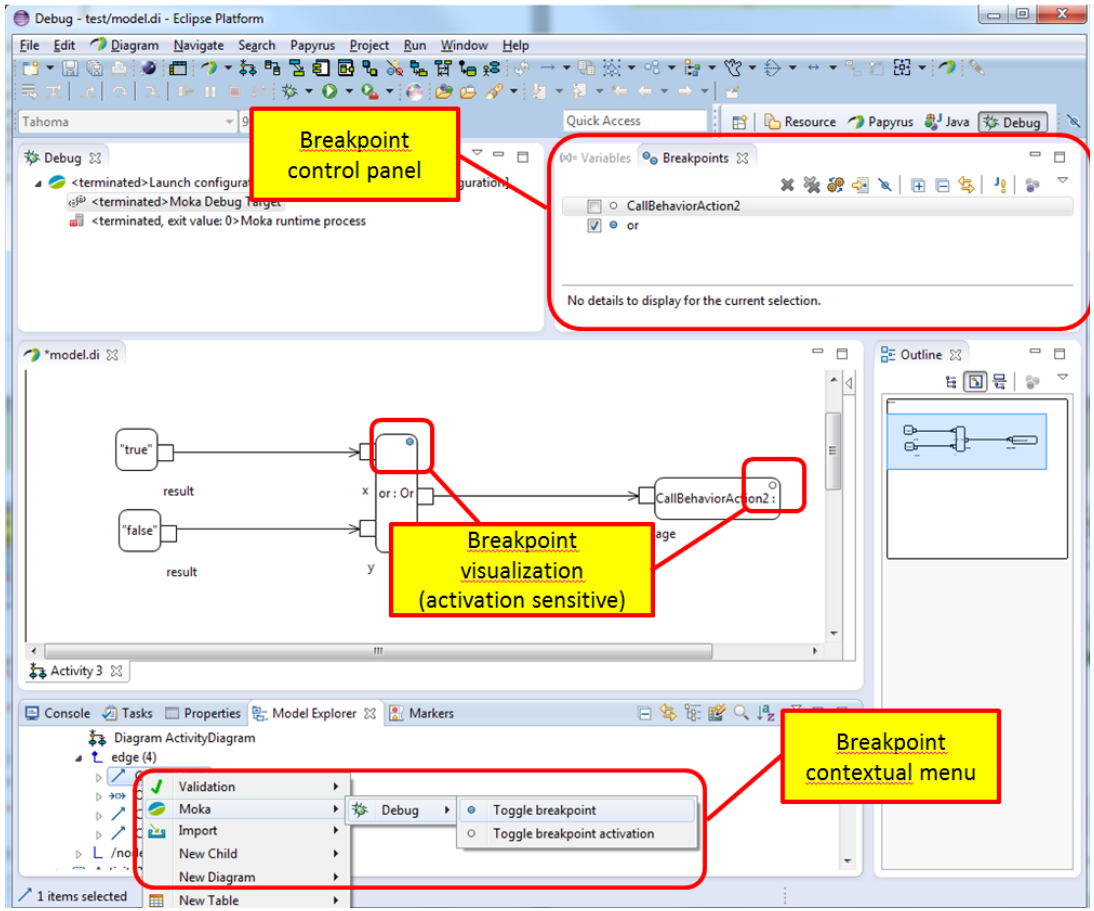
CONNECTION WITH THE ECLIPSE DEBUG FRAMEWORK

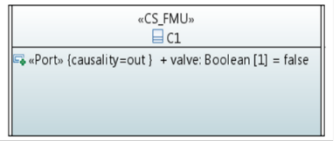
- **Managing launch configurations**

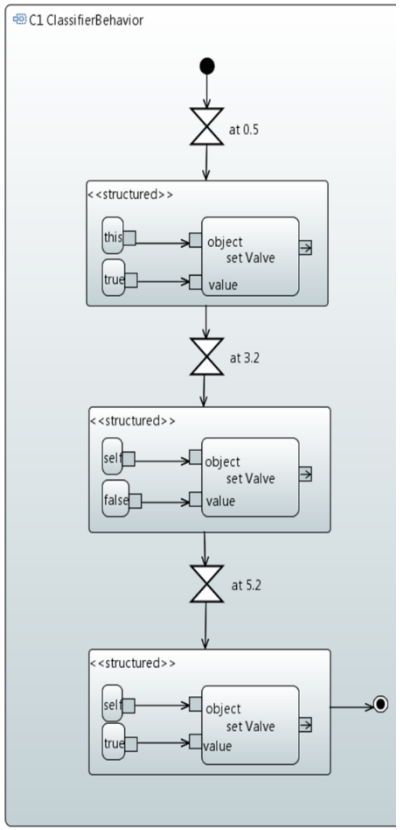


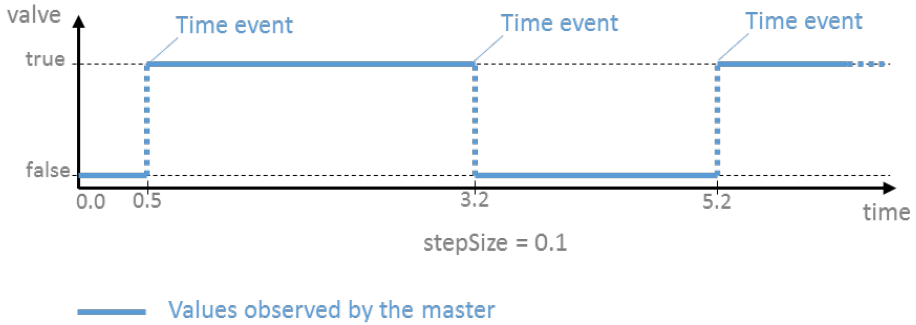
CONNECTION WITH THE ECLIPSE DEBUG FRAMEWORK

- **Managing breakpoints**




«CS_FMU»
C1
«Port» {causality=out } + valve: Boolean [1] = false


C1 ClassifierBehavior
at 0.5
<<structured>>
this
true
object
set Valve
value
at 3.2
<<structured>>
self
false
object
set Valve
value
at 5.2
<<structured>>
self
true
object
set Valve
value


valve
true
false
Time event
Time event
Time event
0.0
0.5
3.2
5.2
time
stepSize = 0.1
Values observed by the master