DE LA RECHERCHE À L'INDUSTRIE

PAPYRUS TOOL SUPPORT FOR FMI

TUTORIAL

MODPROD 2016
Linköping, Sweden, February 7-8, 2017

openCPS ITEA European project

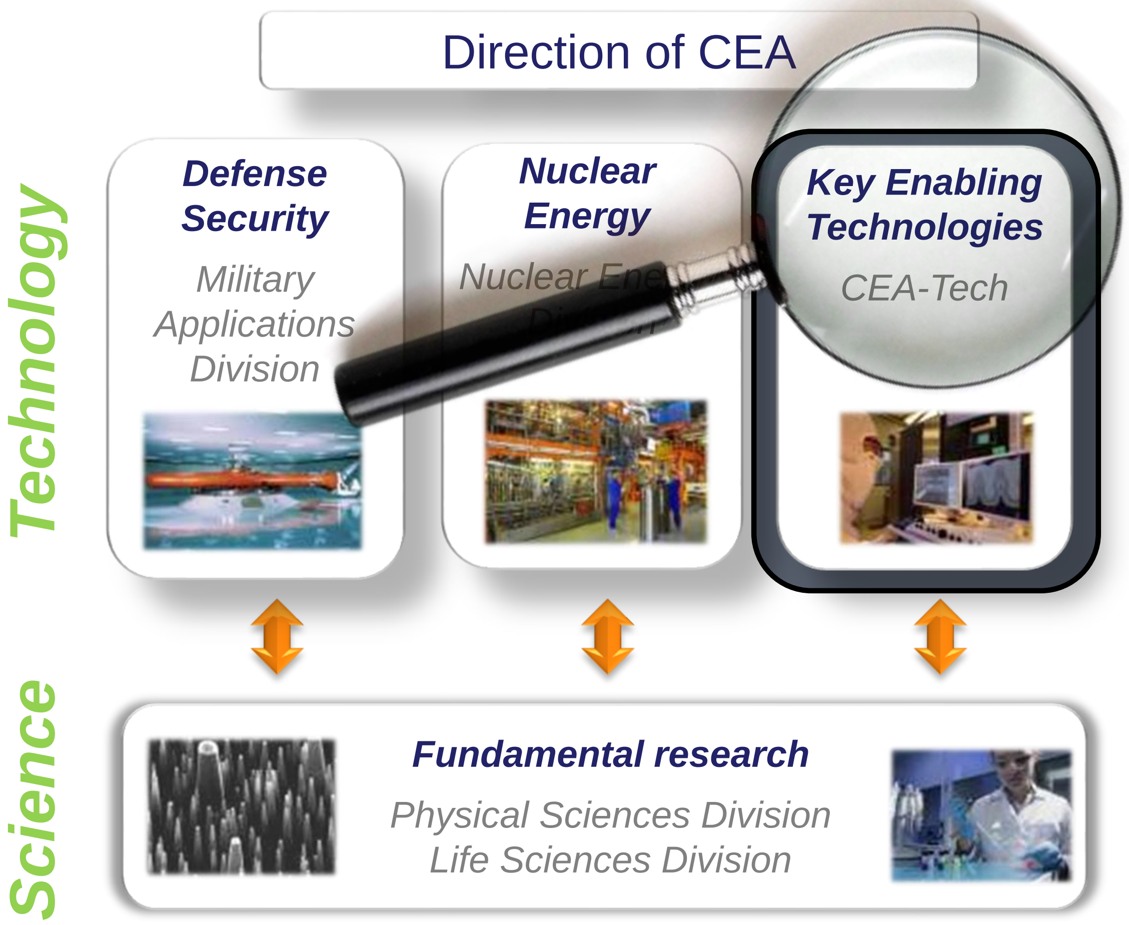
Sahar GUERMAZI, Sébastien REVOL, Arnaud CUCCURU, Saadia DHOUIB, Jérémie TATIBOUET, Sébastien GERARD

CEA LIST / DILS / LISE

www.cea.fr

CEA key figures

CEA is a major actor in research and innovation.



Direction of CEA

Technology

Defense Security

Military Applications Division

Nuclear Energy

Nuclear Ene

Di

Key Enabling Technologies

CEA-Tech

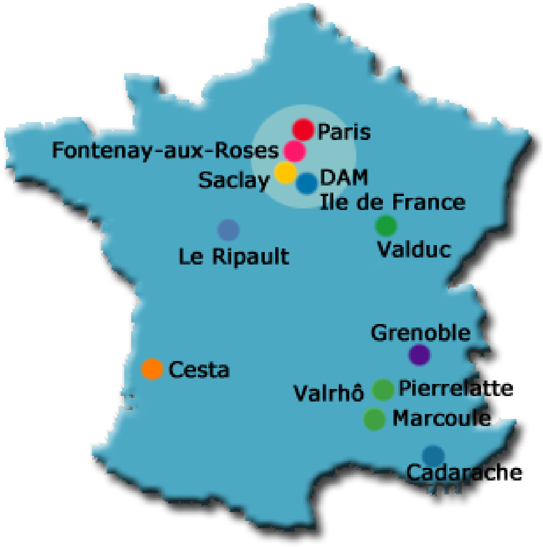
Science

Fundamental research

Physical Sciences Division

Life Sciences Division

- **16 000 people**
- **16 centers in France**
- Budget : 4,3€ bilions
- 1 600 patents
- **4 000 publications / year**
- 150 startup created since 1984

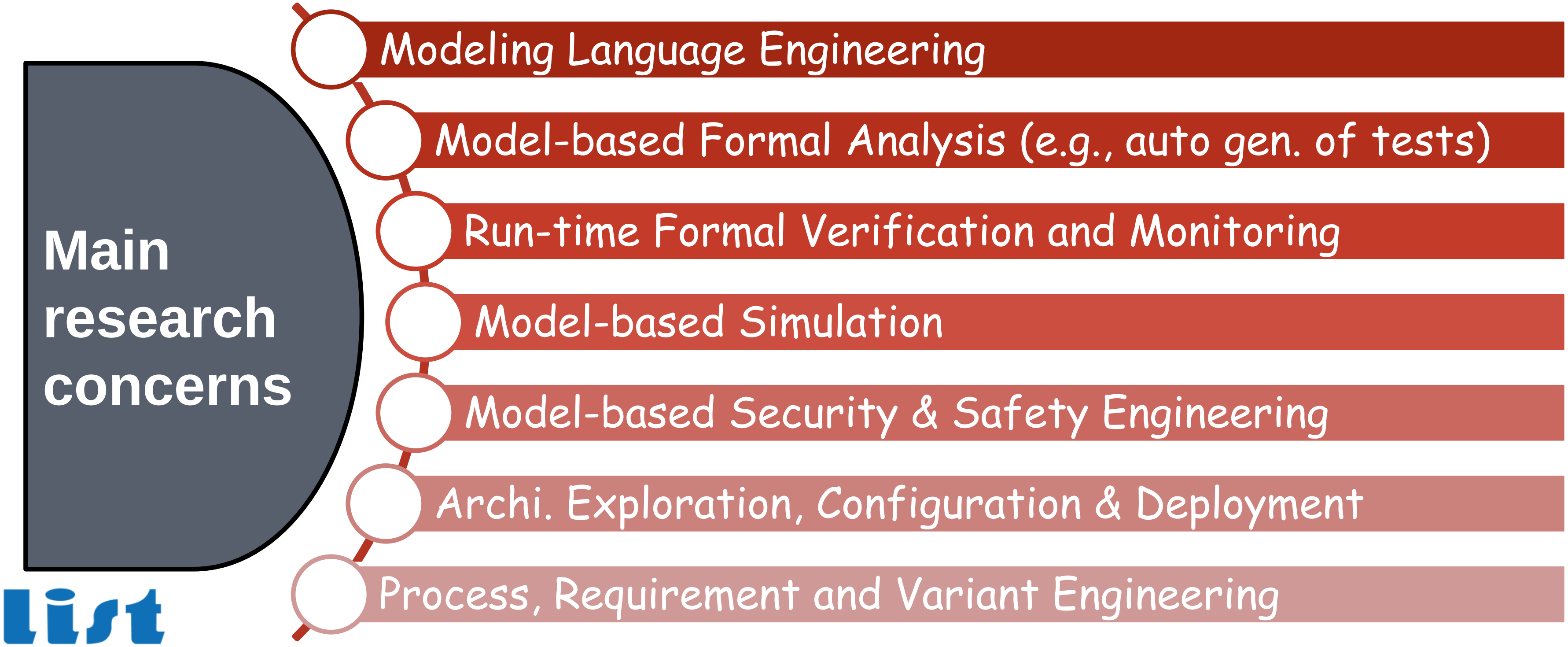


The LISE labs in a nutshell

Correct-by-construction design of safe CPS

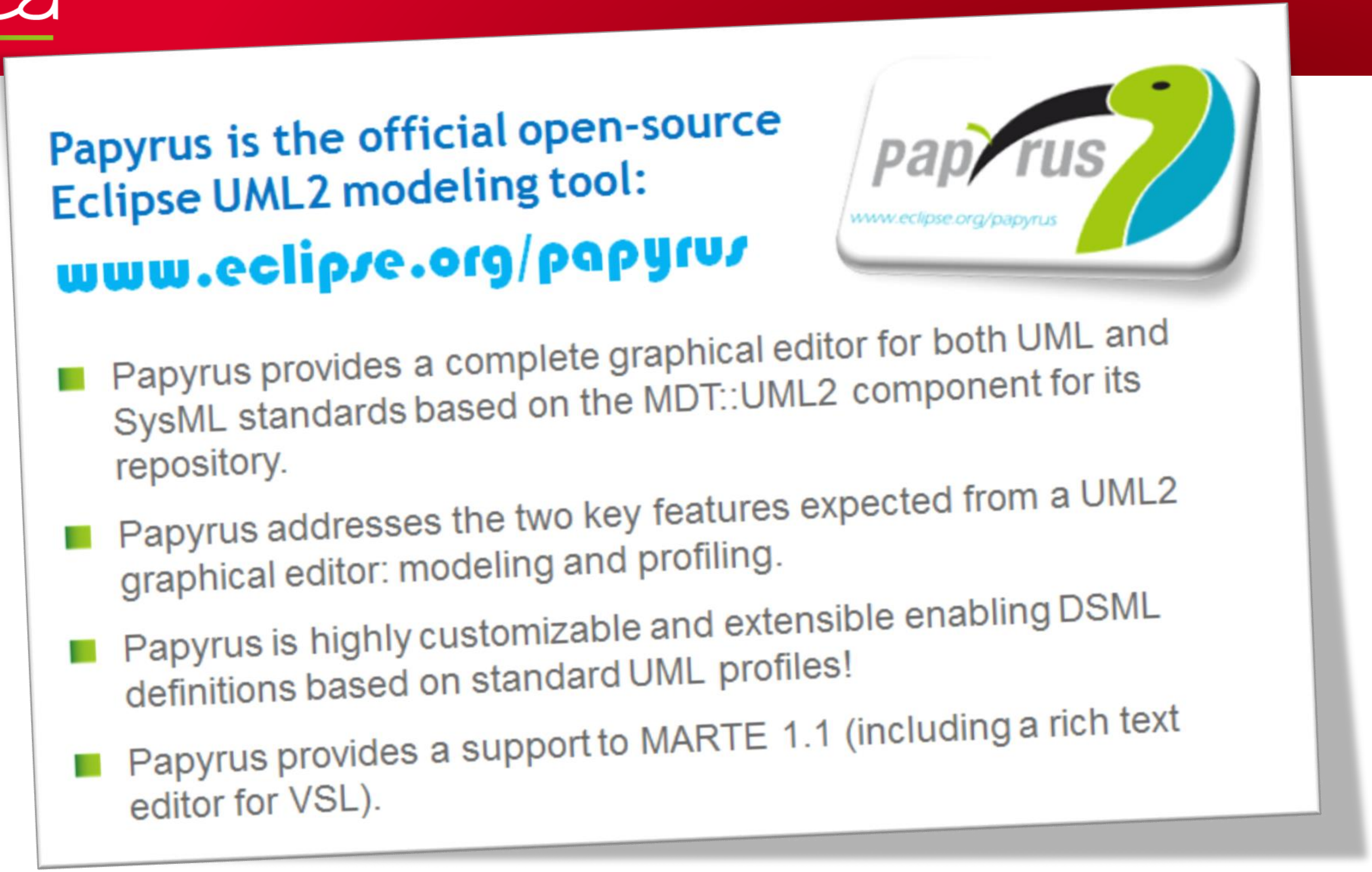
~50 persons

- 30 permanent members + 17 non-permanent members including PhD students, post-docs … (2015)



Main research concerns

- Modeling Language Engineering
- Model-based Formal Analysis (e.g., auto gen. of tests)
- Run-time Formal Verification and Monitoring
- Model-based Simulation
- Model-based Security & Safety Engineering
- Archi. Exploration, Configuration & Deployment
- Process, Requirement and Variant Engineering



papyrus
www.eclipse.org/papyrus

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).

FMI FOR PAPYRUS / PAPYRUS FOR FMI

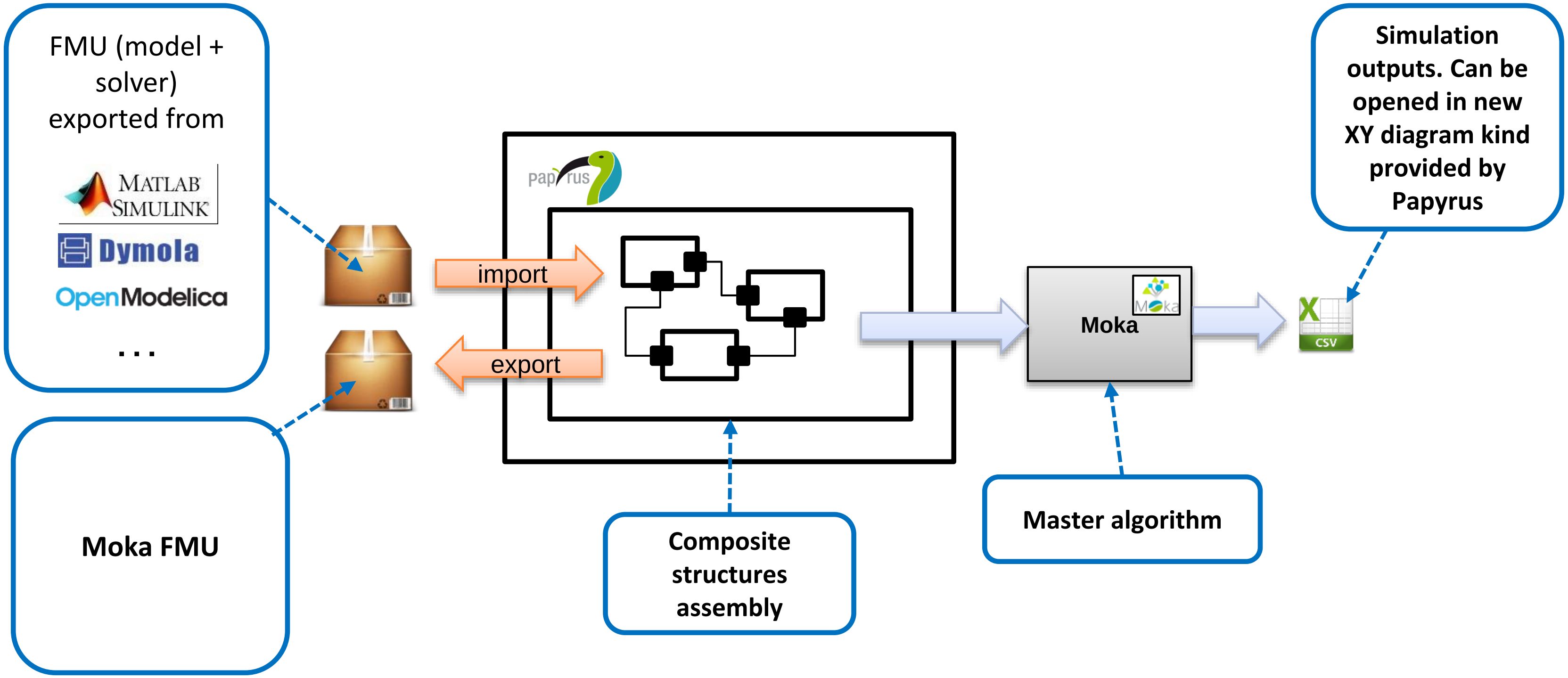
- **Synergy of two complementary standards for Complex system modeling and simulation**

- **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**
  - **Early results, still in incubation phase**

OVERVIEW OF PAPYRUS TOOL SUPPORT FOR FMI



FMU (model + solver) exported from

Moka FMU

Composite structures assembly

Master algorithm

Simulation outputs. Can be opened in new XY diagram kind provided by Papyrus

TUTORIAL OUTLINE

- Papyrus MOKA overview
- Short Reminder on FMI/FMU

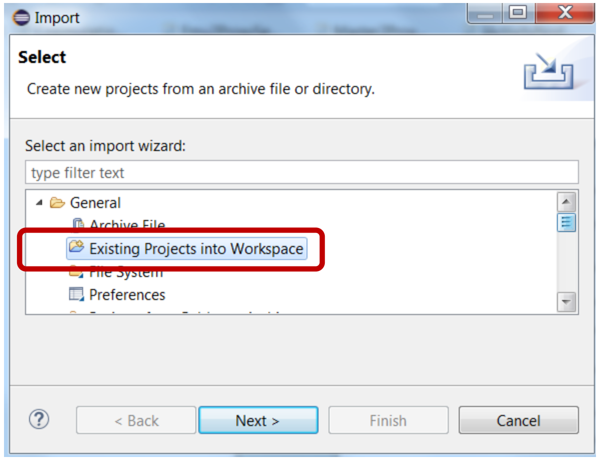
- Papyrus as FMI co-simulation Master :
  - FMU modelling in Papyrus
  - Import of a simple FMU in Papyrus
  - Run a simple simulation
  - Visualize results
- Papyrus as FMU provider :
  - Reminder on OMG standards for Executable Modeling
  - Study and debug a simple UML-based FMU model
  - Export FMU
  - Analysis of generated FMU
- Integration :
  - Integration and co-simulation of the newly exported FMU

SYSTEM REQUIREMENTS

- Papyrus is based on Eclipse
  - Most common platforms are supported (Windows/Linux/Mac…)
  - Requires JAVA 8
- Papyrus for FMI cosimulation
  - JAVA imposes restrictions on 32bits/64bits DLL loading
  - DLL should have the same architecture as the running JVM
  - → 64 bits JVM can only load 64 bits FMUs (and 64 bits eclipse distributions)
  - Running mix of 32 bits/64 bits FMUs is not possible
  - But running 32 bits FMUs on a 64 bits machine is possible
    - → Install 32 bits JVM and Eclipse/Papyrus distribution
- Papyrus as FMU provider :
  - Generated FMUs can run on Linux 32/64 bits and Windows 64 bits
  - Other architectures may be supported on-demand
  - Generated FMUs may requires a JVM on the running machine
- For this tutorial
  - We only provide a Windows 64 bits Papyrus distribution and FMU example

TUTORIAL SETUP

- **Papyrus.zip : papyrus distribution to unzip on your machine**
  - **Includes pre-installed MOKA FMI plugins**
  - **Run Papyrus.exe after unzip**
  - **Select a workspace where your projects will be saved**
- **TutorialProjects.zip : zipped projects**
  - **No need to unzip**
  - **In eclipse : File →Import … →Existing project …→ select archive**



Import

Select

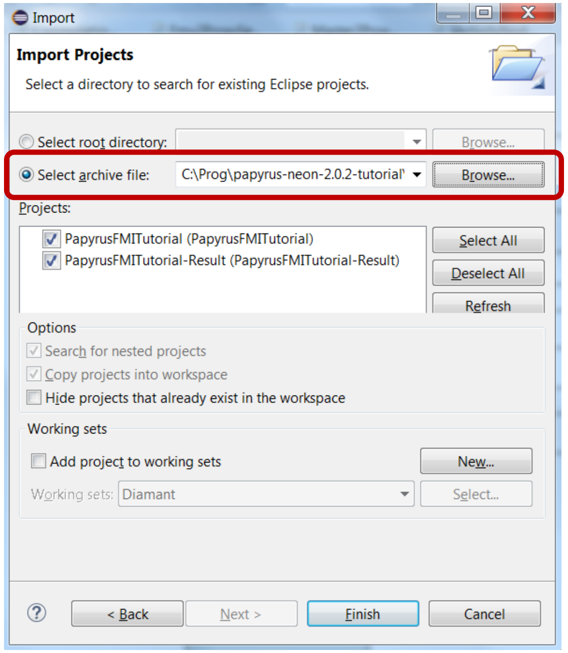
Create new projects from an archive file or directory.

Select an import wizard:

type filter text

- General
  - Archive File
  - Existing Projects into Workspace
  - File System
  - Preferences

< Back | Next > | Finish | Cancel



Import

Import Projects

Select a directory to search for existing Eclipse projects.

○ Select root directory: Browse...

● Select archive file: C:\Prog\papyrus-neon-2.0.2-tutorial\ Browse...

Projects:

☑ PapyrusFMITutorial (PapyrusFMITutorial)
☑ PapyrusFMITutorial-Result (PapyrusFMITutorial-Result)

Select All | Deselect All | Refresh

Options

☑ Search for nested projects
☑ Copy projects into workspace
☐ Hide projects that already exist in the workspace

Working sets

☐ Add project to working sets | New...

Working sets: Diamant | Select...

< Back | Next > | Finish | Cancel

PART I

–

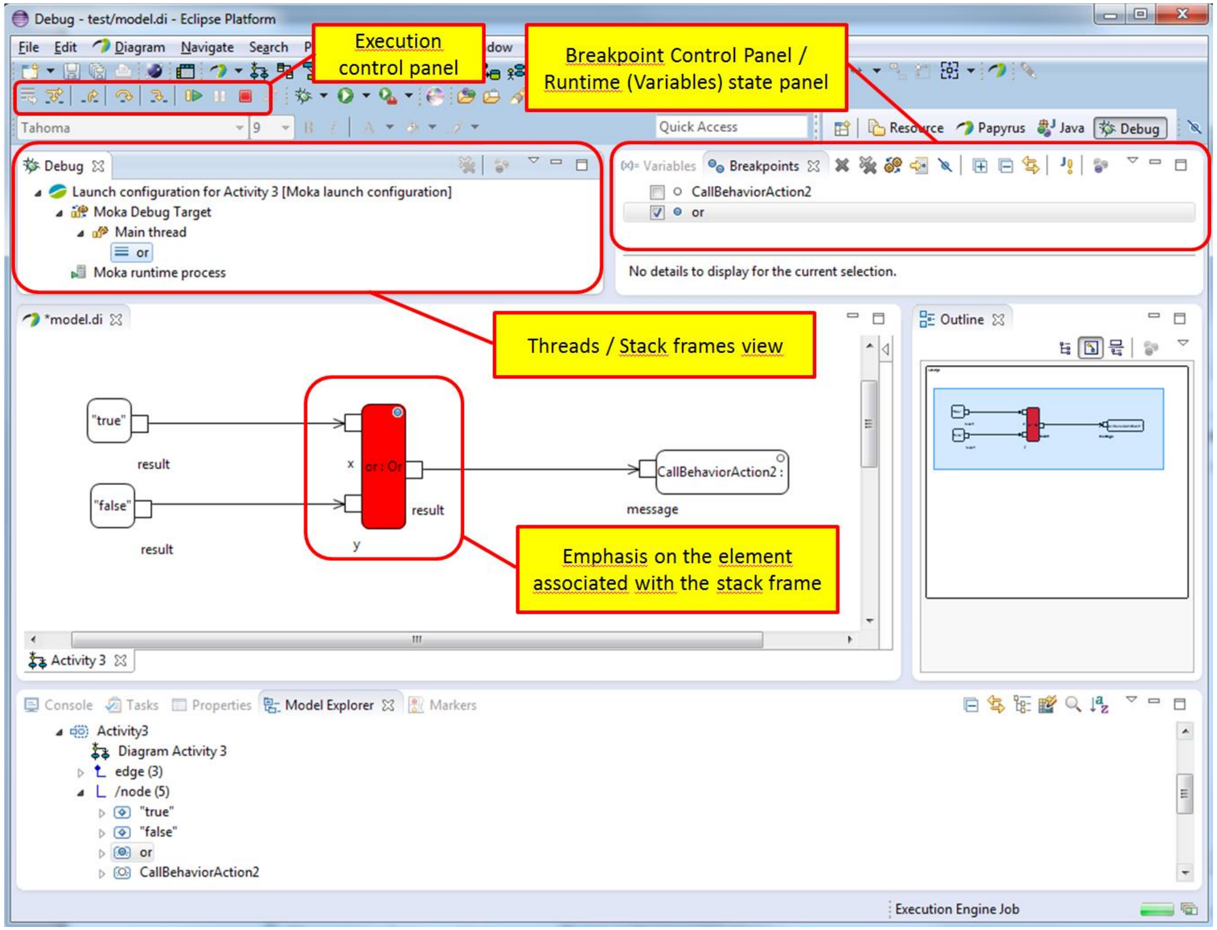
OVERVIEW OF MOKA, THE PAPYRUS MODULE FOR MODEL EXECUTION

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)**
  - **Experimental 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**


Execution control panel

Breakpoint Control Panel / Runtime (Variables) state panel

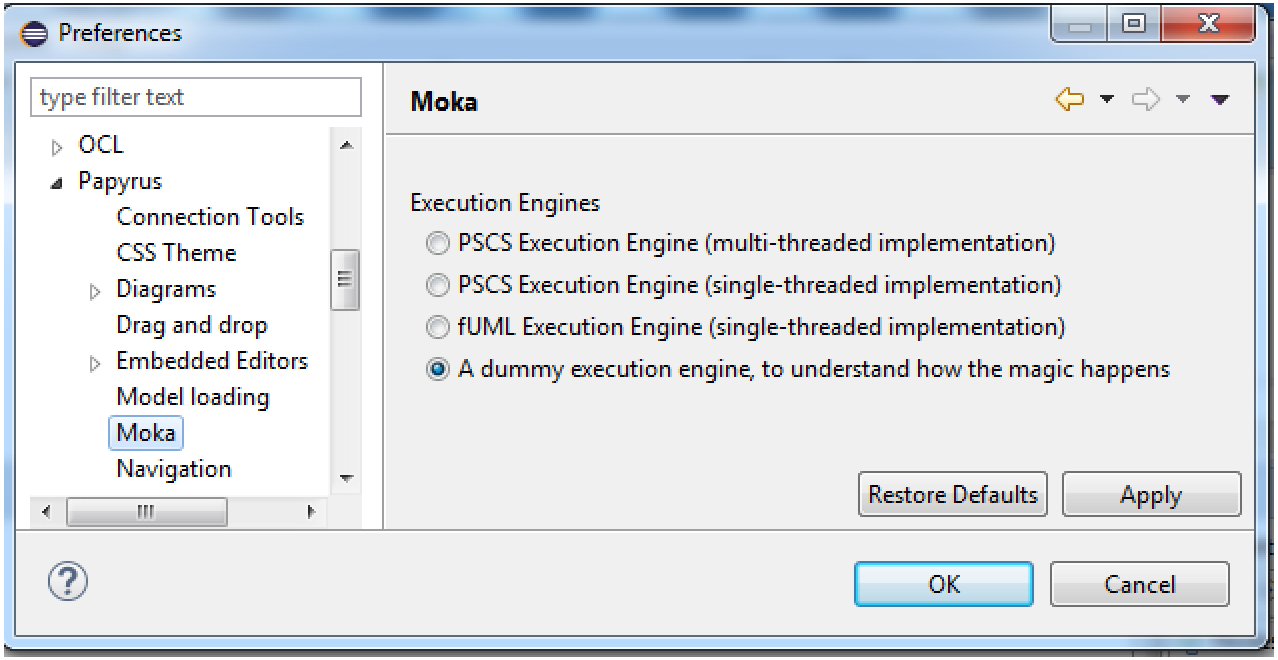
Threads / Stack frames view

Emphasis on the element associated with the stack frame

No details to display for the current selection.

CONNECTION WITH THE ECLIPSE DEBUG FRAMEWORK

- Multiple execution engines can be registered



Execution Engines

○ PSCS Execution Engine (multi-threaded implementation)

○ PSCS Execution Engine (single-threaded implementation)

○ fUML Execution Engine (single-threaded implementation)

● A dummy execution engine, to understand how the magic happens

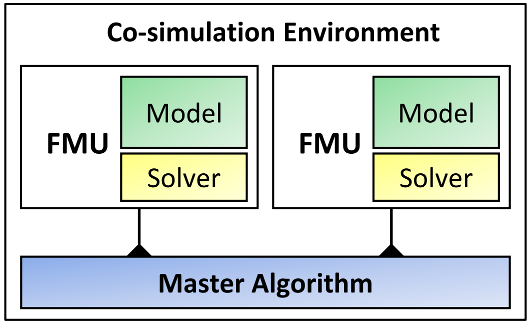
MOKA: OVERVIEW

- 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.

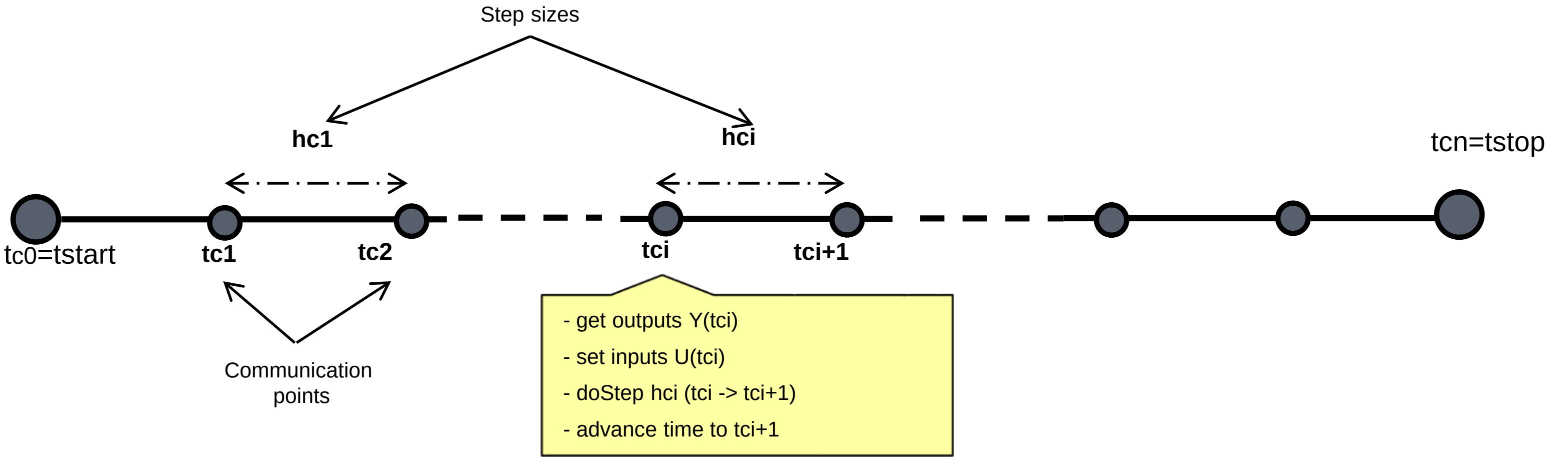
FUNCTIONAL MOCK-UP INTERFACE (FMI)

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



- get outputs Y(tci)
- set inputs U(tci)
- doStep hci (tci -> tci+1)
- advance time to tci+1

PART II

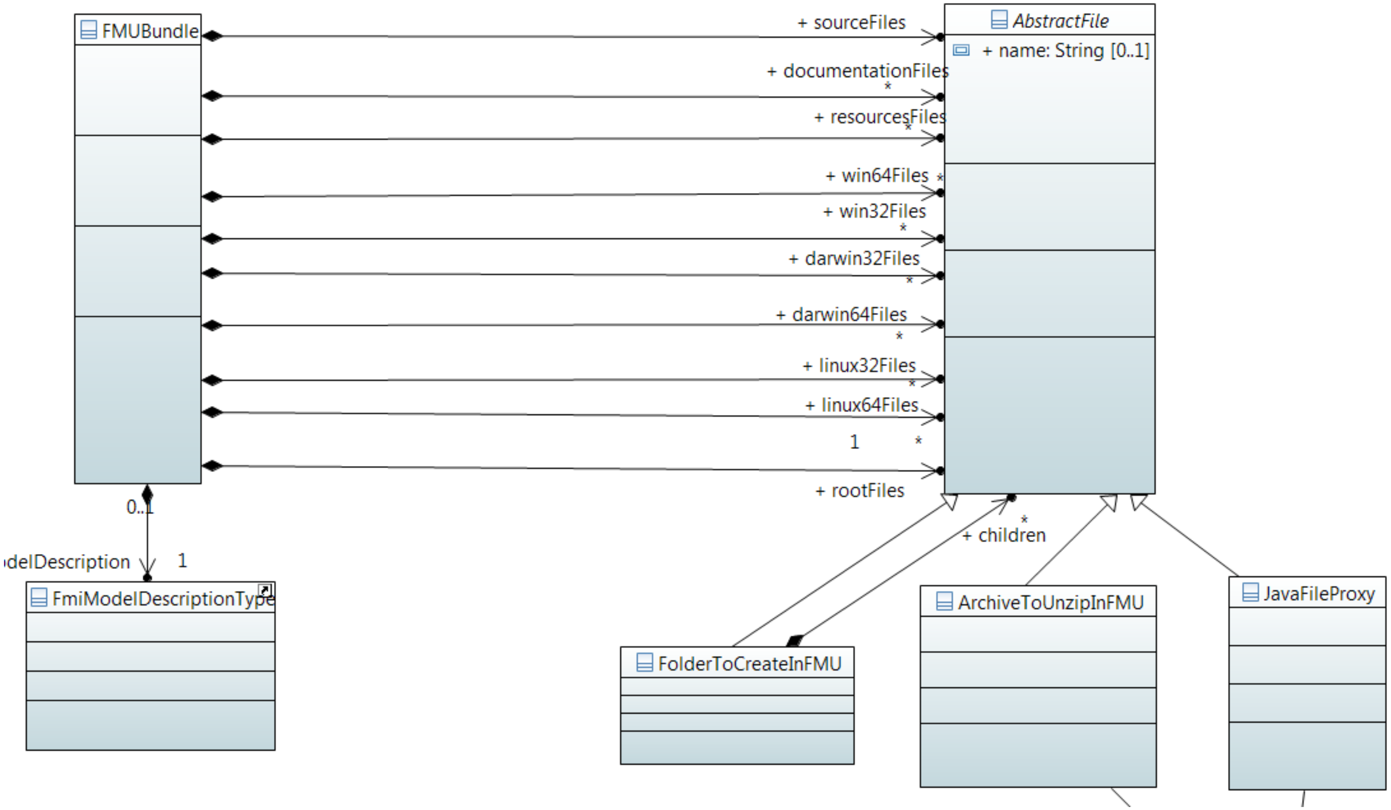
–

PAPYRUS AS FMI CO-SIMULATION MASTER

FMU MODELING IN ECLIPSE/PAPYRUS

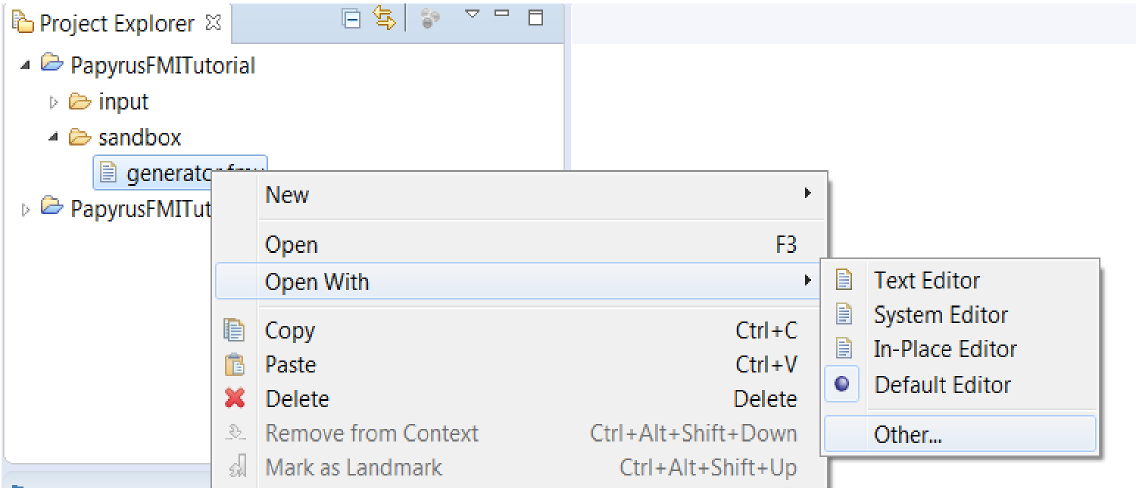
FMU loading/saving integrated in Eclipse Modeling Framework

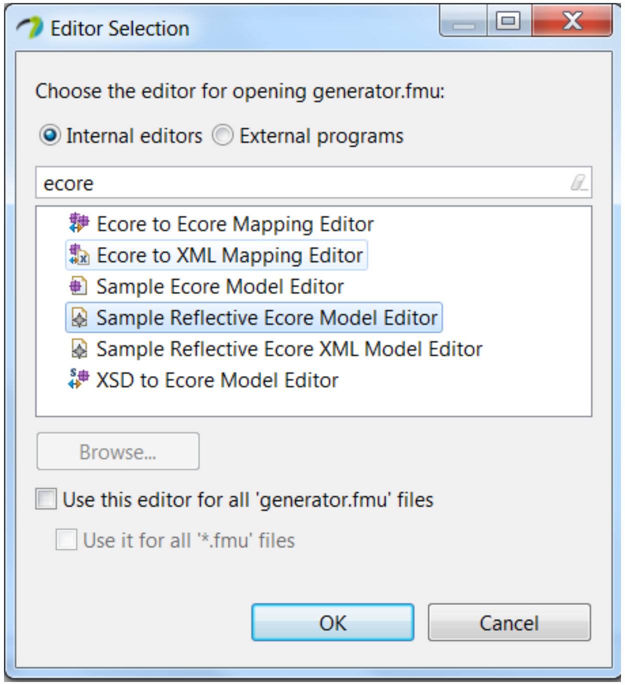
- **FMUs are considered as « Models »**
- **Automatically unzip/zip FMU archive**
- **FMUs can be used as inputs or outputs of model transformation techniques**



EXERCISE 1: OPEN AN FMU IN ECLIPSE

FMUs can be edited with default Ecore Reflective editor




Editor Selection

Choose the editor for opening generator.fmu:

● Internal editors ○ External programs

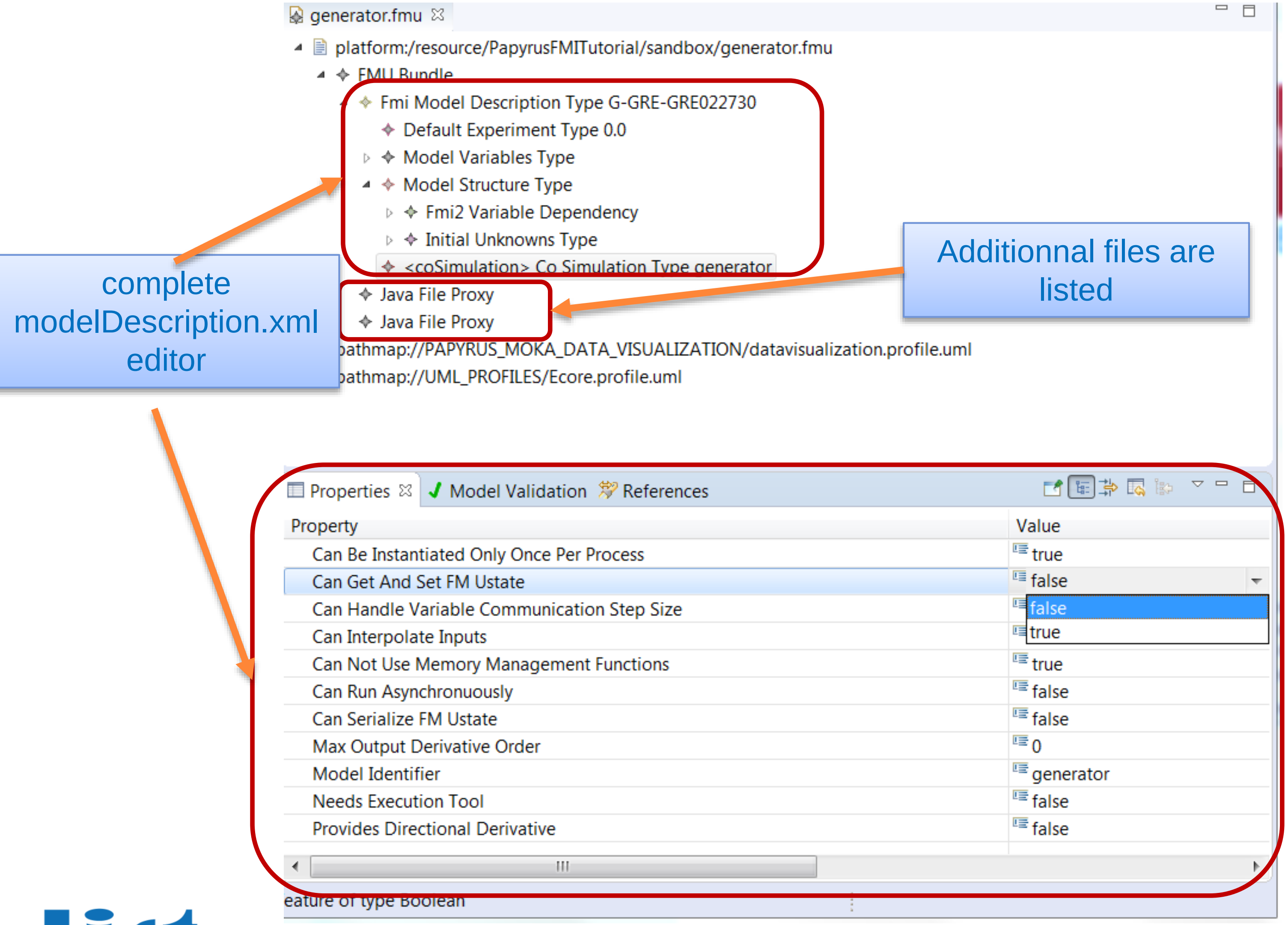
ecore

- Ecore to Ecore Mapping Editor
- Ecore to XML Mapping Editor
- Sample Ecore Model Editor
- Sample Reflective Ecore Model Editor
- Sample Reflective Ecore XML Model Editor
- XSD to Ecore Model Editor

□ Use this editor for all 'generator.fmu' files

□ Use it for all '*.fmu' files

EXERCISE 1: OPEN AN FMU IN ECLIPSE



generator.fmu

- platform:/resource/PapyrusFMITutorial/sandbox/generator.fmu
  - FMU Bundle
    - Fmi Model Description Type G-GRE-GRE022730
      - Default Experiment Type 0.0
      - Model Variables Type
      - Model Structure Type
        - Fmi2 Variable Dependency
        - Initial Unknowns Type
      - <coSimulation> Co Simulation Type generator
    - Java File Proxy
    - Java File Proxy
- pathmap://PAPYRUS_MOKA_DATA_VISUALIZATION/datavisualization.profile.uml
- pathmap://UML_PROFILES/Ecore.profile.uml

Properties | Model Validation | References

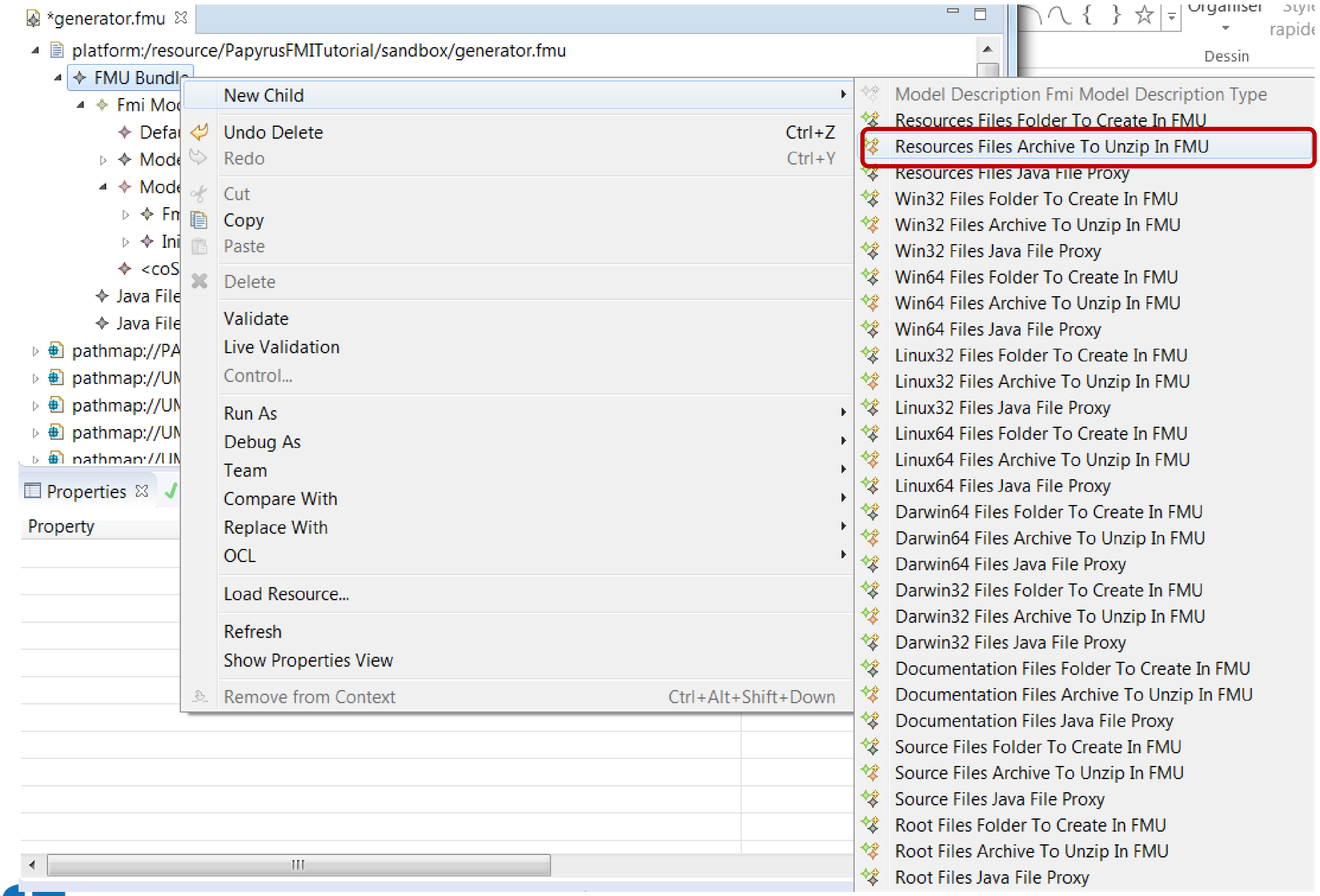
Property	Value
Can Be Instantiated Only Once Per Process	true
Can Get And Set FM Ustate	false false true
Can Handle Variable Communication Step Size	
Can Interpolate Inputs	
Can Not Use Memory Management Functions	true
Can Run Asynchronously	false
Can Serialize FM Ustate	false
Max Output Derivative Order	0
Model Identifier	generator
Needs Execution Tool	false
Provides Directional Derivative	false

eature of type Boolean

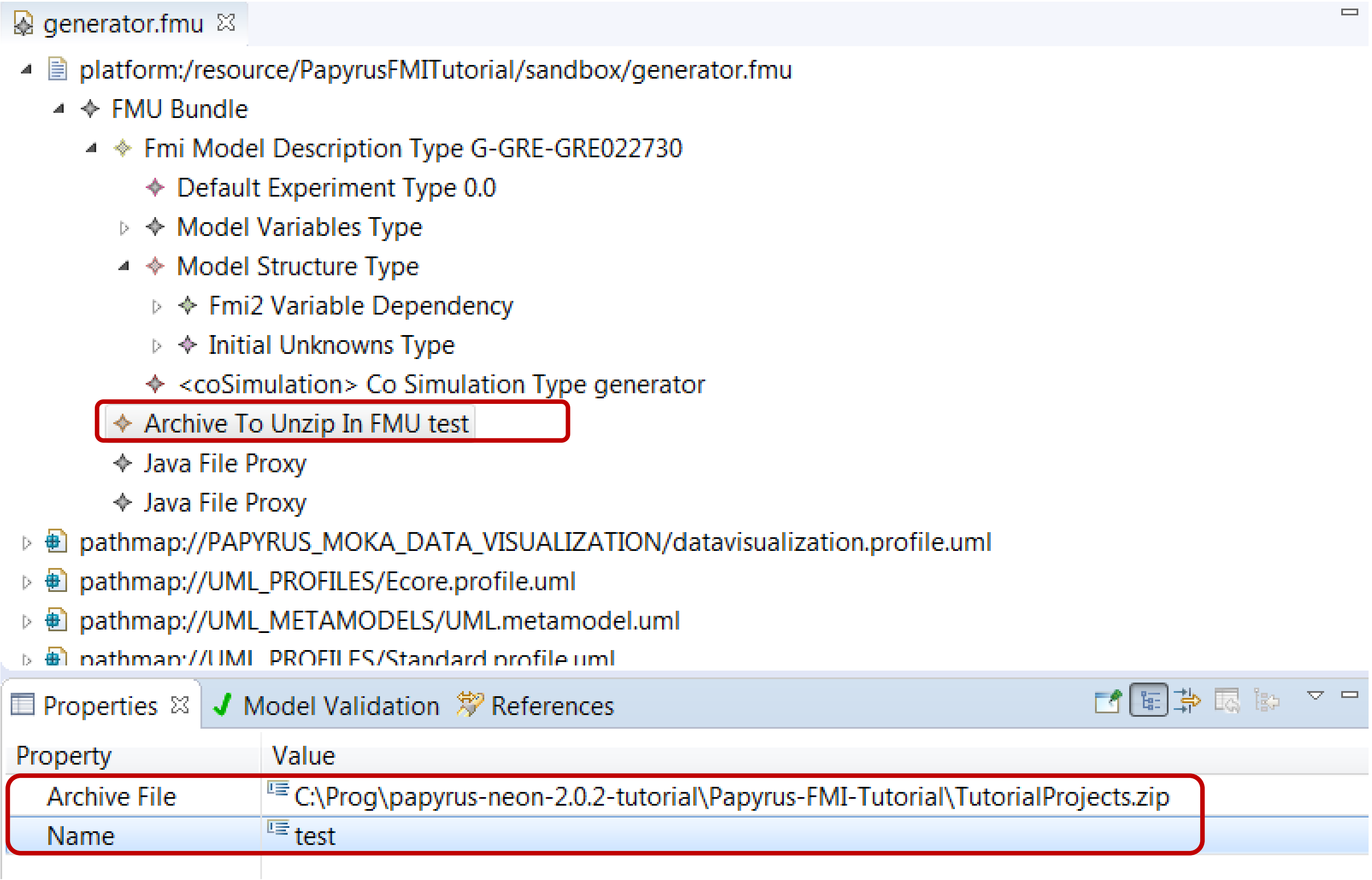
complete modelDescription.xml editor

Additionnal files are listed

EXERCISE 1: ADD AN ARCHIVE IN RESOURCE FOLDER OF FMU…


*generator.fmu
platform:/resource/PapyrusFMITutorial/sandbox/generator.fmu
FMU Bundle
Fmi Mod
Defa
Mode
Mode
Fm
Ini
<coS
Java File
Java File
pathmap://PA
pathmap://UM
pathmap://UM
pathmap://UM
pathmap://UM
Properties
Property
New Child
Undo Delete Ctrl+Z
Redo Ctrl+Y
Cut
Copy
Paste
Delete
Validate
Live Validation
Control...
Run As
Debug As
Team
Compare With
Replace With
OCL
Load Resource...
Refresh
Show Properties View
Remove from Context Ctrl+Alt+Shift+Down
Model Description Fmi Model Description Type
Resources Files Folder To Create In FMU
Resources Files Archive To Unzip In FMU
Resources Files Java File Proxy
Win32 Files Folder To Create In FMU
Win32 Files Archive To Unzip In FMU
Win32 Files Java File Proxy
Win64 Files Folder To Create In FMU
Win64 Files Archive To Unzip In FMU
Win64 Files Java File Proxy
Linux32 Files Folder To Create In FMU
Linux32 Files Archive To Unzip In FMU
Linux32 Files Java File Proxy
Linux64 Files Folder To Create In FMU
Linux64 Files Archive To Unzip In FMU
Linux64 Files Java File Proxy
Darwin64 Files Folder To Create In FMU
Darwin64 Files Archive To Unzip In FMU
Darwin64 Files Java File Proxy
Darwin32 Files Folder To Create In FMU
Darwin32 Files Archive To Unzip In FMU
Darwin32 Files Java File Proxy
Documentation Files Folder To Create In FMU
Documentation Files Archive To Unzip In FMU
Documentation Files Java File Proxy
Source Files Folder To Create In FMU
Source Files Archive To Unzip In FMU
Source Files Java File Proxy
Root Files Folder To Create In FMU
Root Files Archive To Unzip In FMU
Root Files Java File Proxy
Organiser
Style rapide
Dessin

EXERCISE 1: ADD AN ARCHIVE IN RESOURCE FOLDER OF FMU…



generator.fmu

- platform:/resource/PapyrusFMITutorial/sandbox/generator.fmu
  - FMU Bundle
    - Fmi Model Description Type G-GRE-GRE022730
      - Default Experiment Type 0.0
      - Model Variables Type
      - Model Structure Type
        - Fmi2 Variable Dependency
        - Initial Unknowns Type
      - <coSimulation> Co Simulation Type generator
    - Archive To Unzip In FMU test
    - Java File Proxy
    - Java File Proxy
- pathmap://PAPYRUS_MOKA_DATA_VISUALIZATION/datavisualization.profile.uml
- pathmap://UML_PROFILES/Ecore.profile.uml
- pathmap://UML_METAMODELS/UML.metamodel.uml
- pathmap://UML_PROFILES/Standard.profile.uml

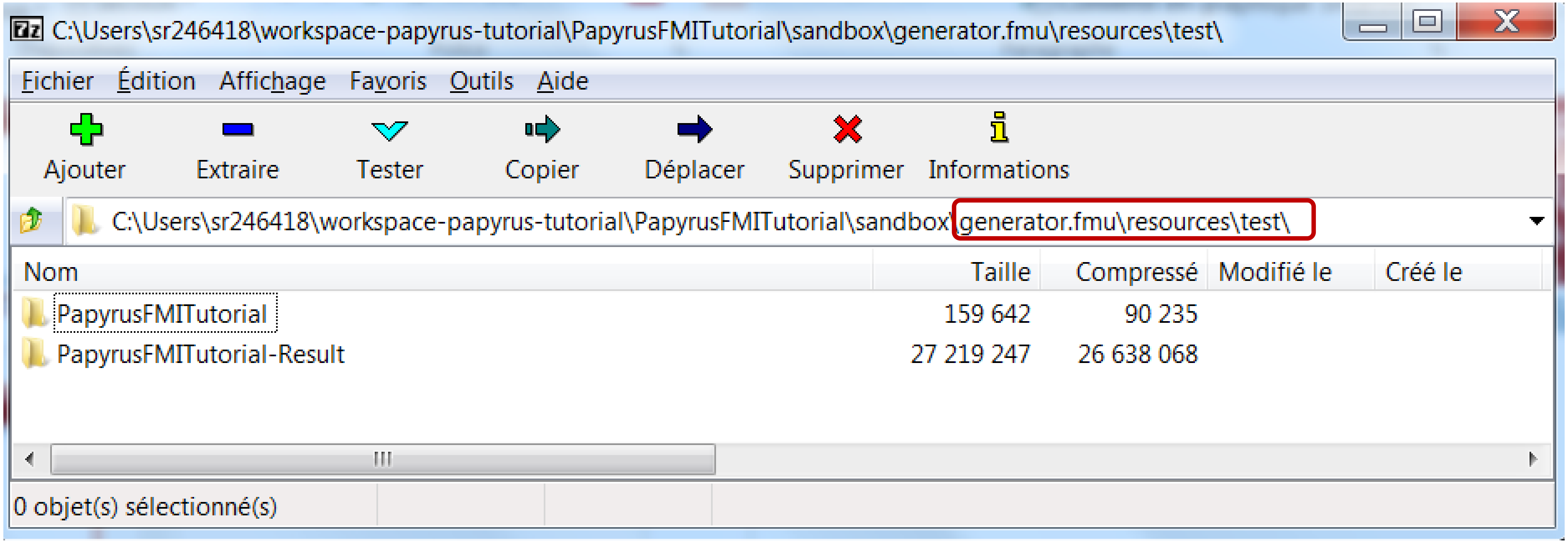
Property	Value
Archive File	C:\Prog\papyrus-neon-2.0.2-tutorial\Papyrus-FMI-Tutorial\TutorialProjects.zip
Name	test

EXERCISE 1: ADD AN ARCHIVE IN RESOURCE FOLDER OF FMU…

Result after saving : open the FMU file as an Archive (out of eclipse)

-a new folder named « test » is created inside FMU resource folder

- It contains the contents of the archive



C:\Users\sr246418\workspace-papyrus-tutorial\PapyrusFMITutorial\sandbox\generator.fmu\resources\test\

C:\Users\sr246418\workspace-papyrus-tutorial\PapyrusFMITutorial\sandbox\generator.fmu\resources\test\

Nom	Taille	Compressé	Modifié le	Créé le
PapyrusFMITutorial	159 642	90 235		
PapyrusFMITutorial-Result	27 219 247	26 638 068		

0 objet(s) sélectionné(s)

REPRESENTING FMU IN UML: THE FMI PROFILE

- **Papyrus first class citizen are UML model elements**
  - **We must provide a mechanism to represent FMUs as UML model elements**
  - **This is the purpose of the FMI profile**

- **A profile allows to extend standard UML concepts with domain specific attributes**

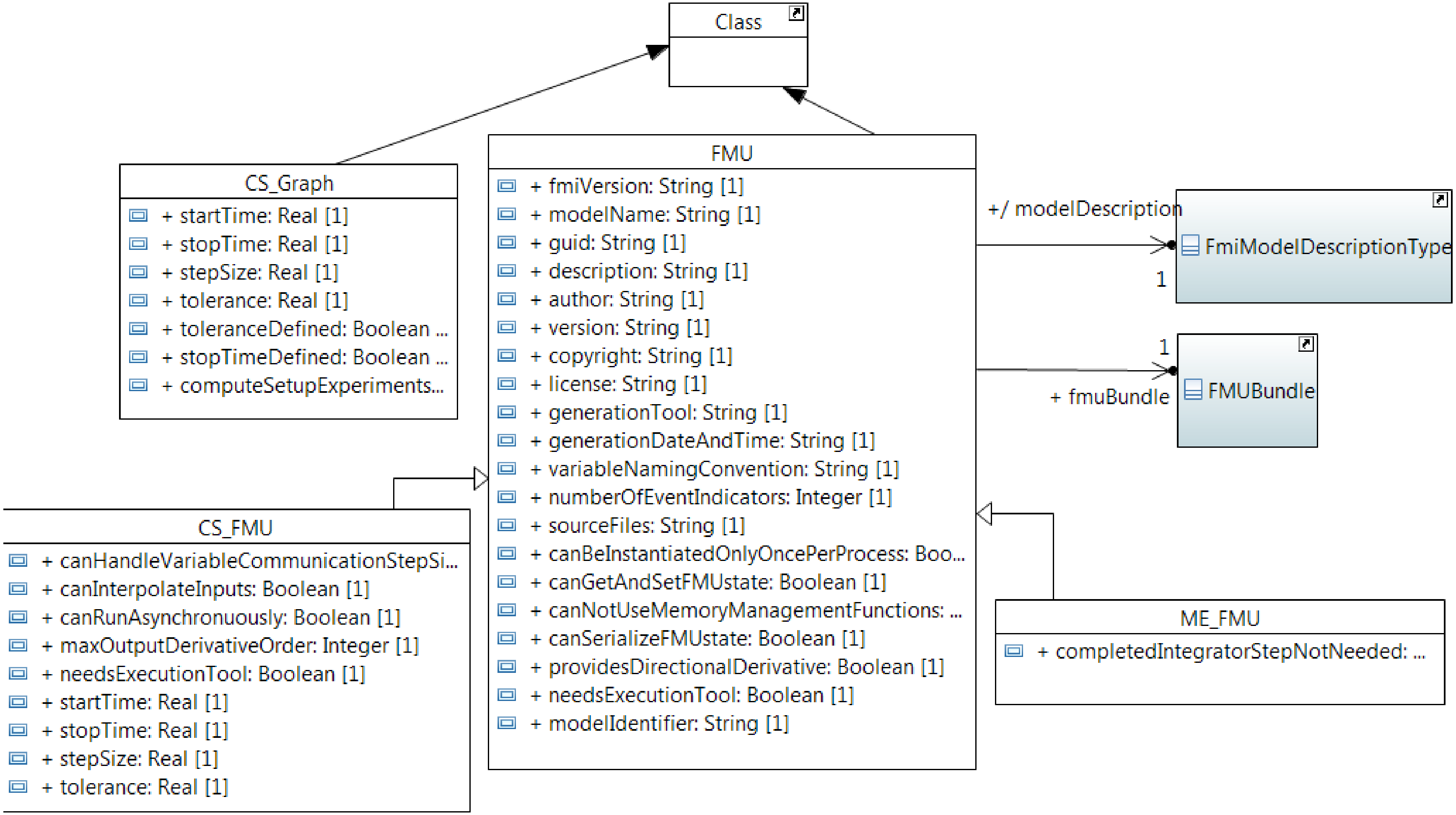
- **FMI profile :**
  - **Adds to UML elements FMI specific concepts**
  - **Not a full one to one translation : only useful concepts for UML display/handling**
  - **Includes a direct link to in-memory original FMU model**

- **FMU import in Papyrus**
  - **model transformation from FMU metamodel to UML + FMI profile**

- **FMU generation**
  - **model transformation from UML + FMI profile to FMU metamodel**
  - **generation according Moka computation mechanism**
    - Ex : only discrete variables

FMI PROFILE OVERVIEW

- FMUs are represented as a special kind of Class



Class

FMU
- + fmiVersion: String [1]
- + modelName: String [1]
- + guid: String [1]
- + description: String [1]
- + author: String [1]
- + version: String [1]
- + copyright: String [1]
- + license: String [1]
- + generationTool: String [1]
- + generationDateAndTime: String [1]
- + variableNamingConvention: String [1]
- + numberOfEventIndicators: Integer [1]
- + sourceFiles: String [1]
- + canBeInstantiatedOnlyOncePerProcess: Boo...
- + canGetAndSetFMUstate: Boolean [1]
- + canNotUseMemoryManagementFunctions: ...
- + canSerializeFMUstate: Boolean [1]
- + providesDirectionalDerivative: Boolean [1]
- + needsExecutionTool: Boolean [1]
- + modelIdentifier: String [1]

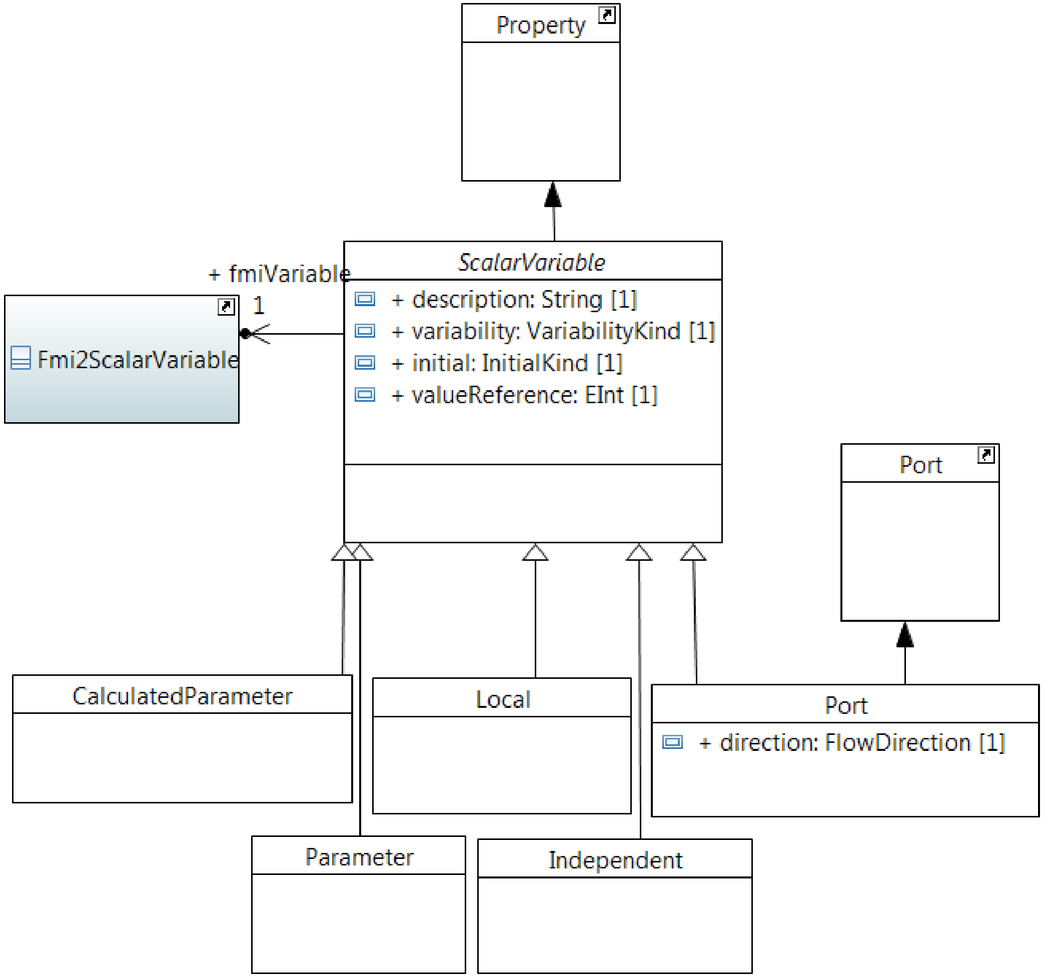
CS_Graph
- + startTime: Real [1]
- + stopTime: Real [1]
- + stepSize: Real [1]
- + tolerance: Real [1]
- + toleranceDefined: Boolean ...
- + stopTimeDefined: Boolean ...
- + computeSetupExperiments...

CS_FMU
- + canHandleVariableCommunicationStepSi...
- + canInterpolateInputs: Boolean [1]
- + canRunAsynchronuously: Boolean [1]
- + maxOutputDerivativeOrder: Integer [1]
- + needsExecutionTool: Boolean [1]
- + startTime: Real [1]
- + stopTime: Real [1]
- + stepSize: Real [1]
- + tolerance: Real [1]

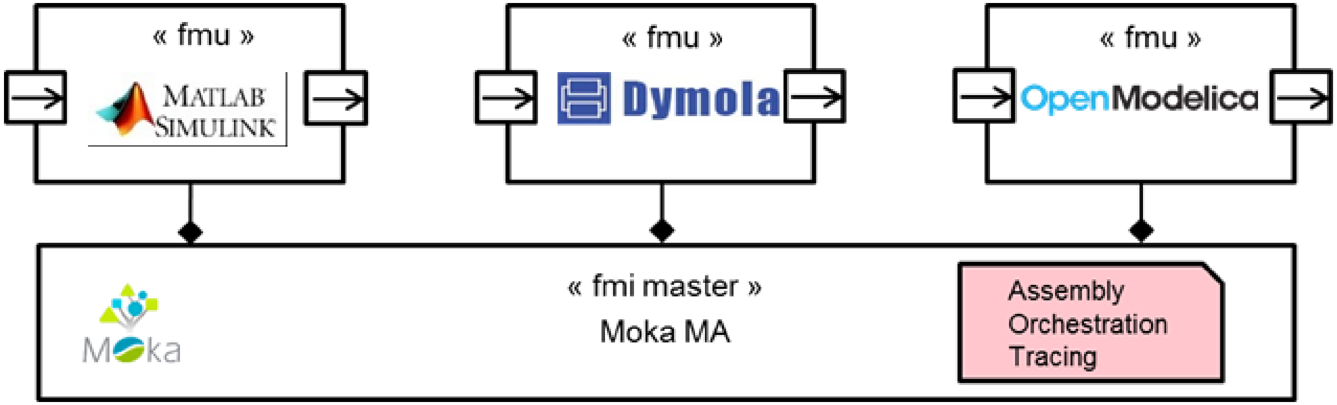
ME_FMU
- + completedIntegratorStepNotNeeded: ...

FMI PROFILE OVERVIEW

- **Scalar Variables are represented as a special kind of Class attributes**


Property
+ fmiVariable
1
Fmi2ScalarVariable
ScalarVariable
+ description: String [1]
+ variability: VariabilityKind [1]
+ initial: InitialKind [1]
+ valueReference: EInt [1]
Port
CalculatedParameter
Local
Port
+ direction: FlowDirection [1]
Parameter
Independent

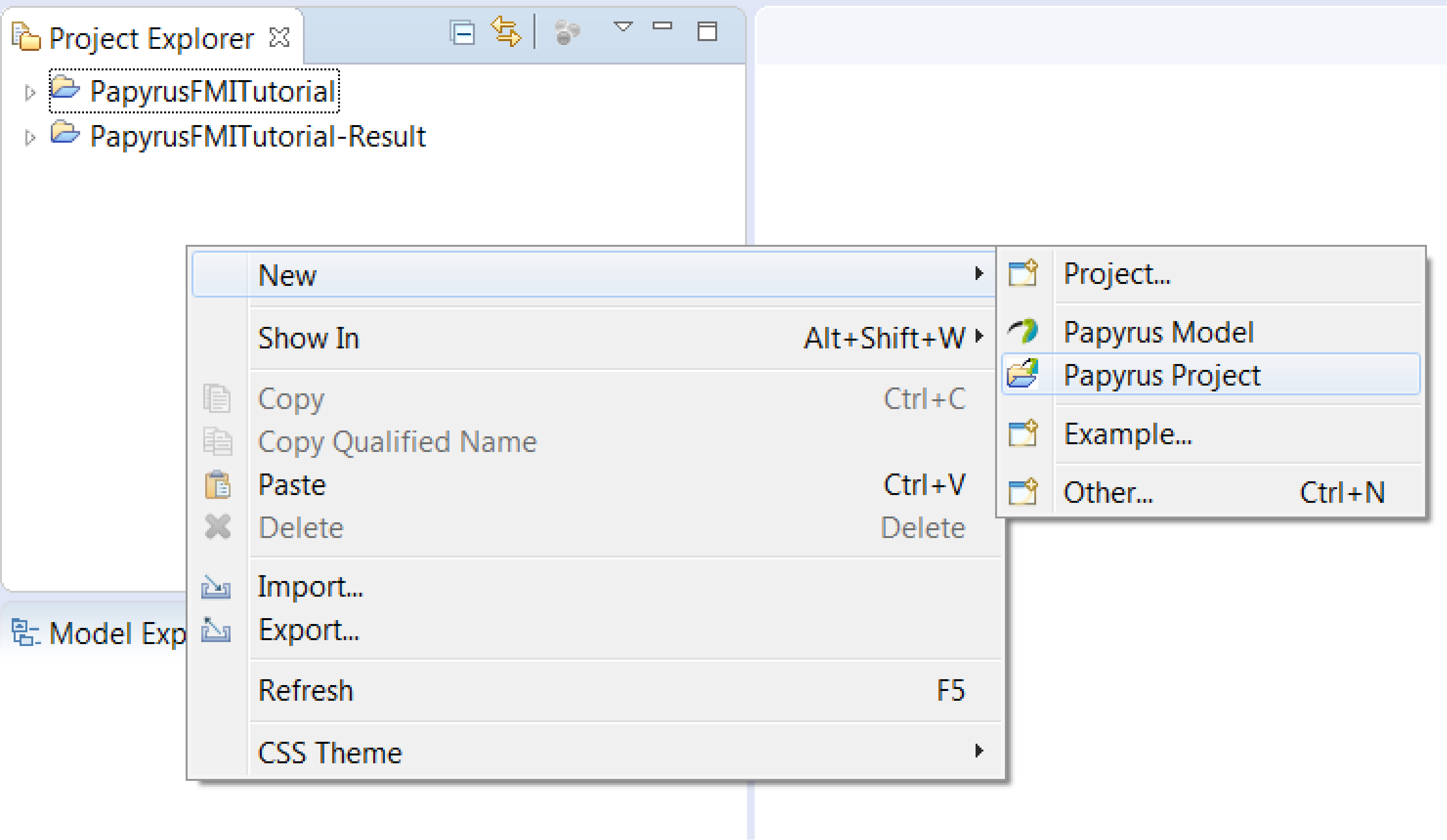
THE FMI MASTER FUNCTIONNALITY



- **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 model, 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**

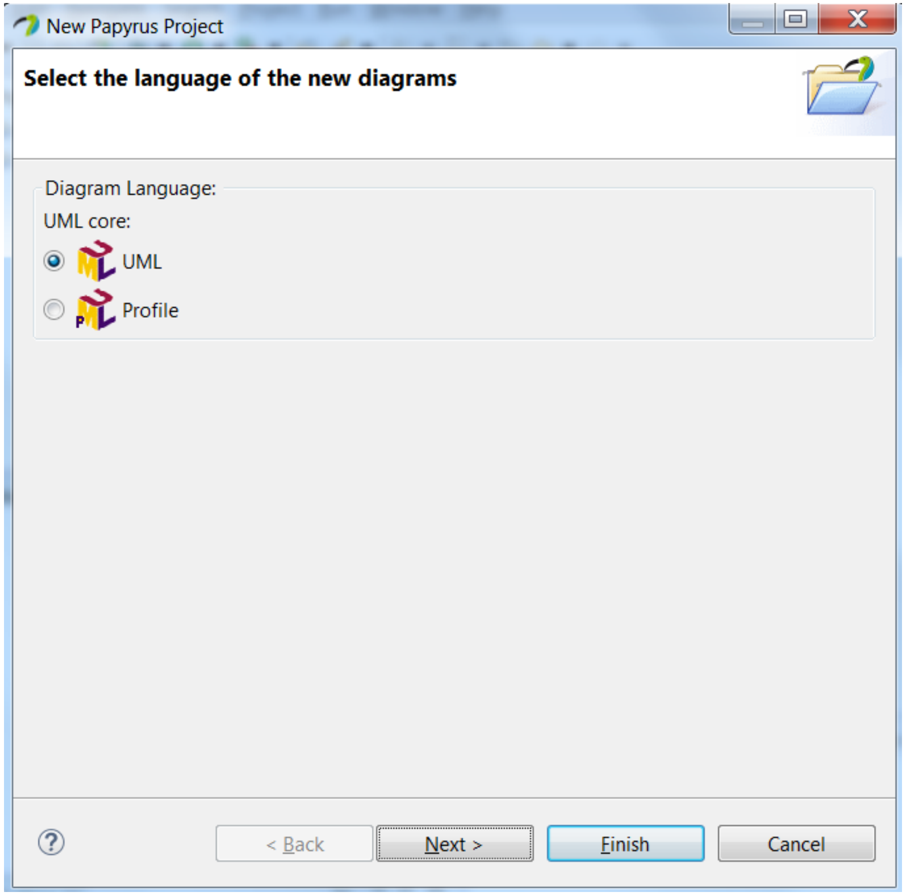
EXERCISE 2 : PAPYRUS FMU IMPORT

Create a new Papyrus project



EXERCISE 2 : PAPYRUS FMU IMPORT

Select UML -> next -> name the poject and the model file



New Papyrus Project

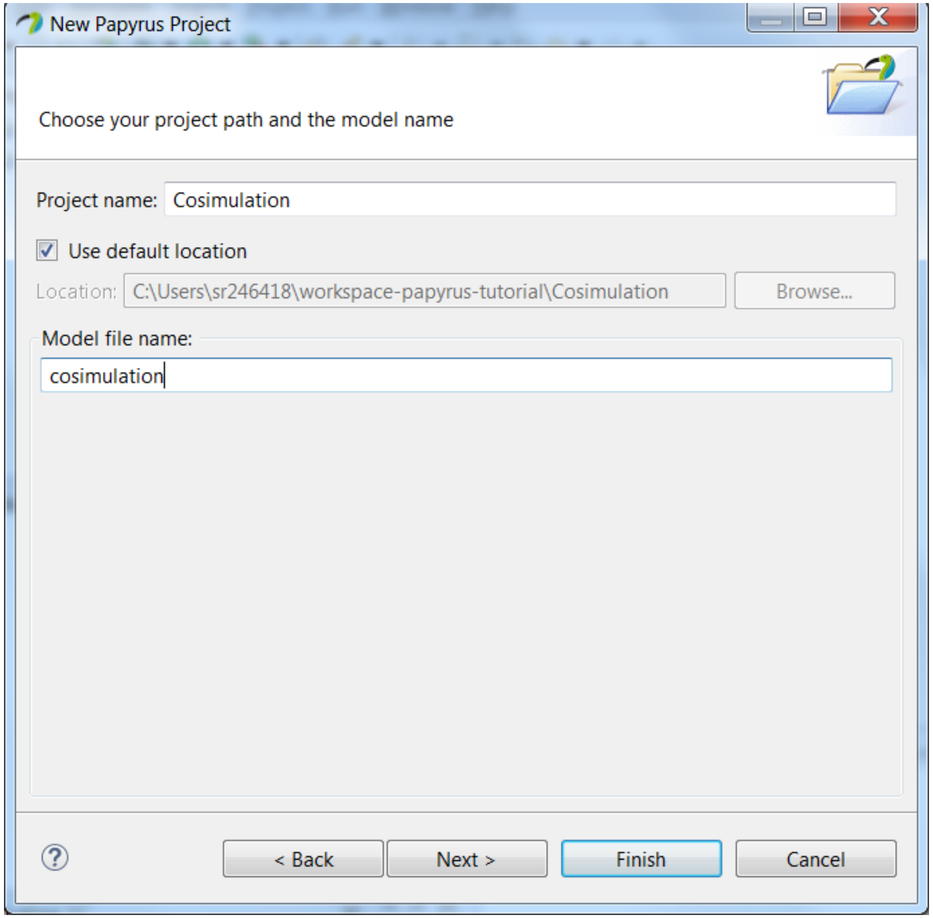
Select the language of the new diagrams

Diagram Language:

UML core:

● UML

○ Profile



New Papyrus Project

Choose your project path and the model name

Project name: Cosimulation

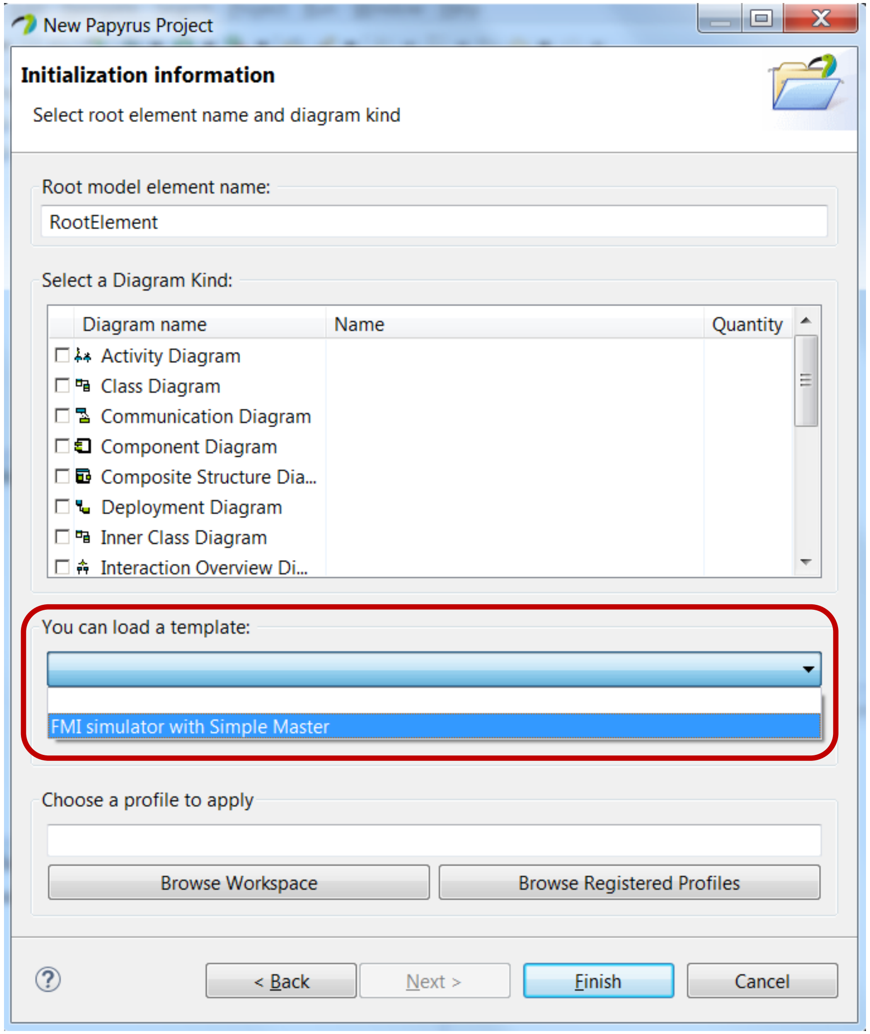
☑ Use default location

Location: C:\Users\sr246418\workspace-papyrus-tutorial\Cosimulation

Model file name:

cosimulation

EXERCISE 2 : PAPYRUS FMU IMPORT



New Papyrus Project

Initialization information

Select root element name and diagram kind

Root model element name:

RootElement

Select a Diagram Kind:

Diagram name	Name	Quantity
□ Activity Diagram		
□ Class Diagram		
□ Communication Diagram		
□ Component Diagram		
□ Composite Structure Dia...		
□ Deployment Diagram		
□ Inner Class Diagram		
□ Interaction Overview Di...		

You can load a template:

FMI simulator with Simple Master

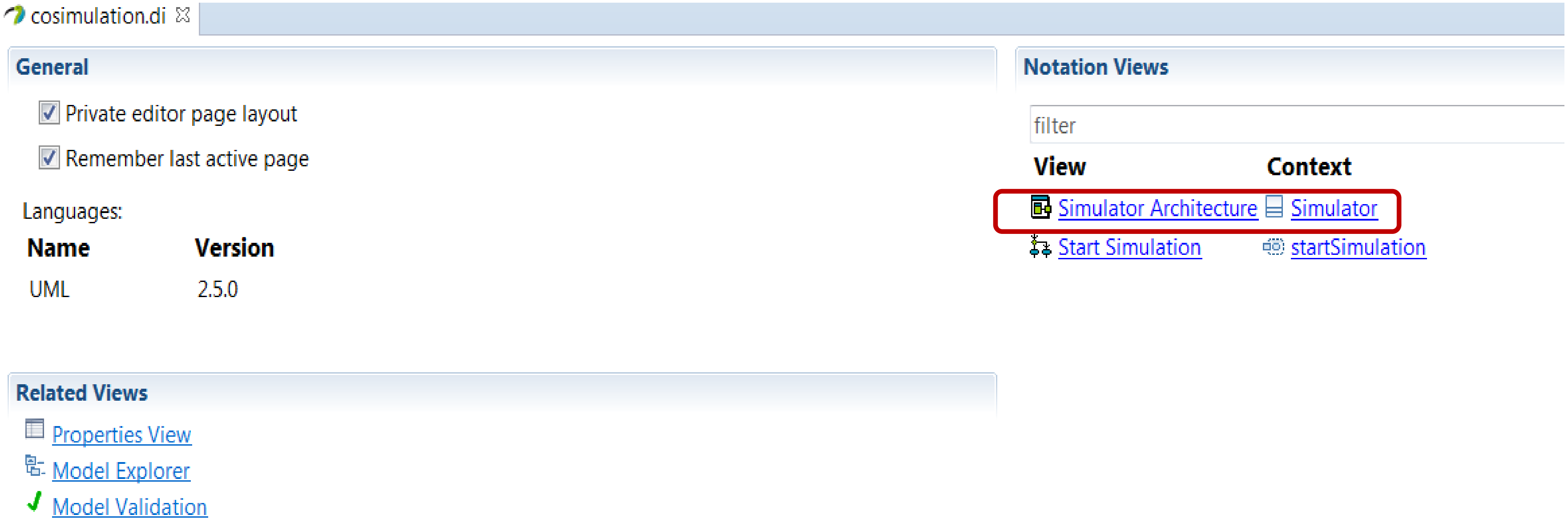
Choose a profile to apply

Select FMI simulator model template and finish

→predefined « ready to run » Simulator model

EXERCISE 2 : PAPYRUS FMU IMPORT

Open Simulator architecure diagram



cosimulation.di

General

☑ Private editor page layout

☑ Remember last active page

Languages:

Name	Version
UML	2.5.0

Related Views

Properties View

Model Explorer

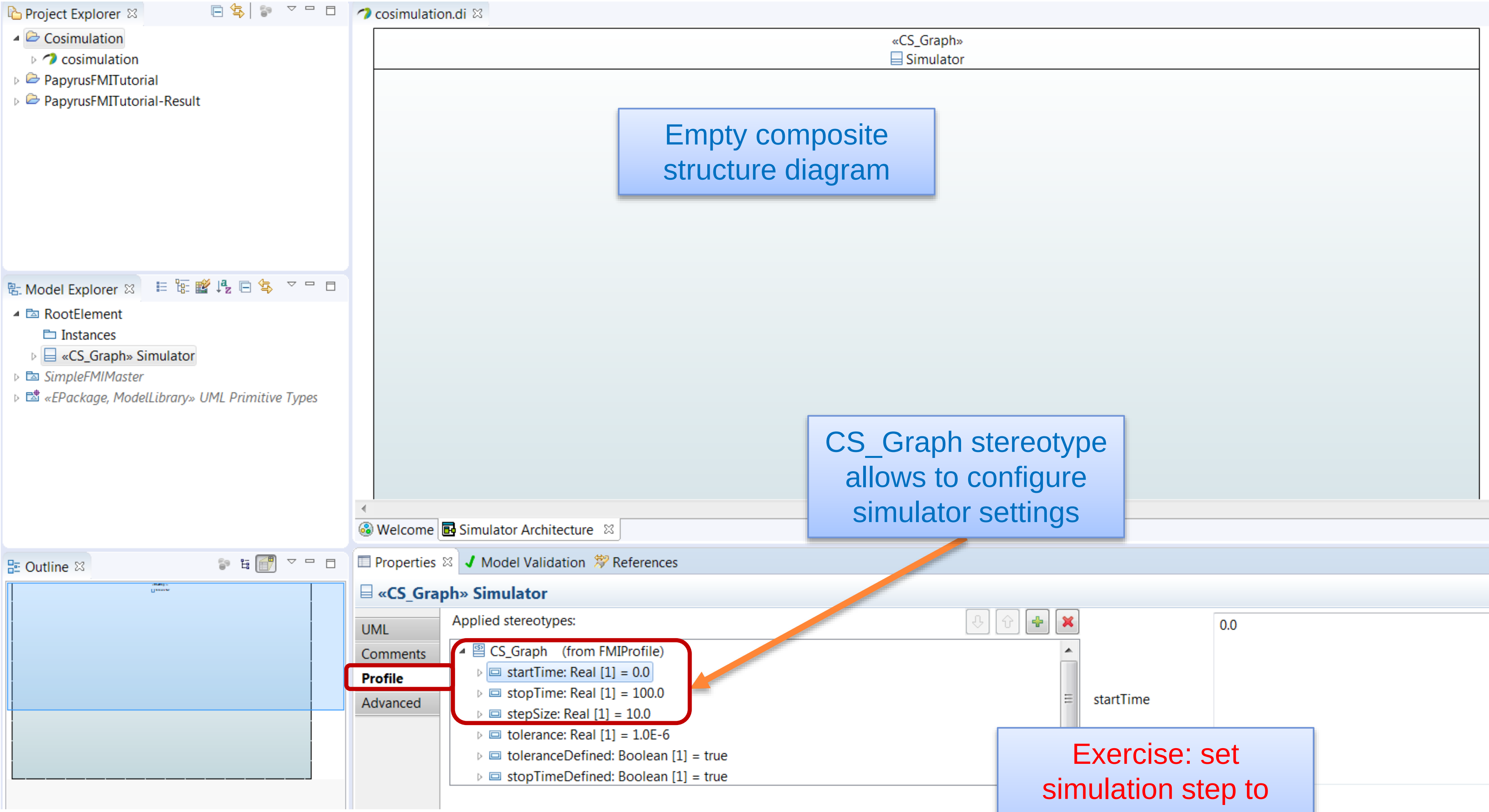
✓ Model Validation

Notation Views

filter

View	Context
Simulator Architecture	Simulator
Start Simulation	startSimulation

EXERCISE 2 : PAPYRUS FMU IMPORT



«CS_Graph» Simulator

Applied stereotypes:

- CS_Graph (from FMIProfile)
  - startTime: Real [1] = 0.0
  - stopTime: Real [1] = 100.0
  - stepSize: Real [1] = 10.0
  - tolerance: Real [1] = 1.0E-6
  - toleranceDefined: Boolean [1] = true
  - stopTimeDefined: Boolean [1] = true

startTime 0.0

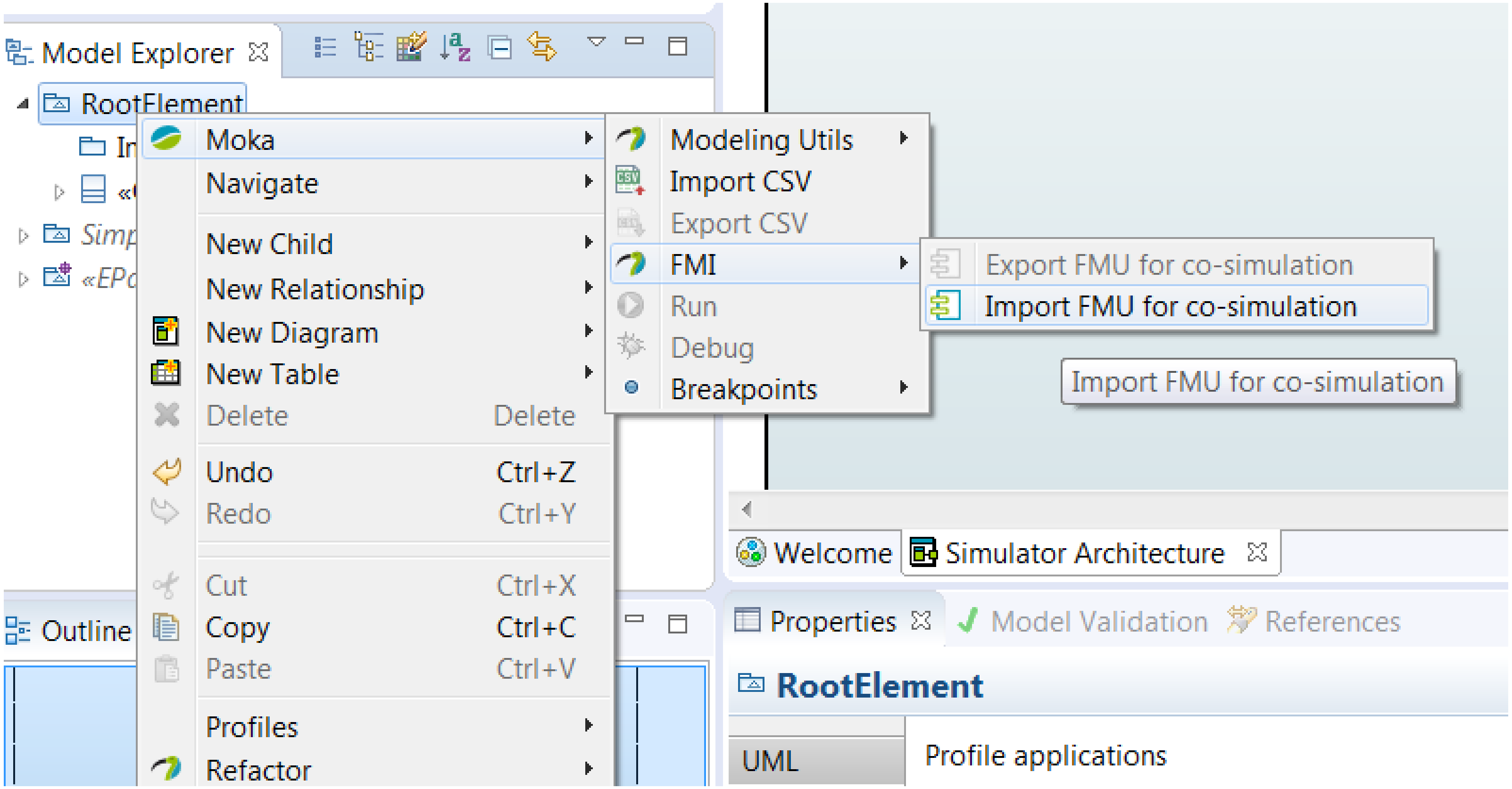
Empty composite structure diagram

CS_Graph stereotype allows to configure simulator settings

Exercise: set simulation step to 0.01

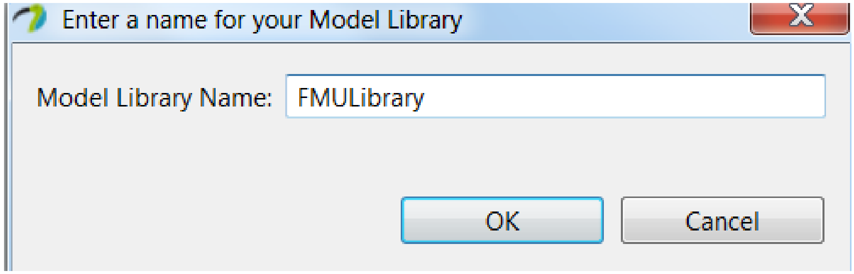
EXERCISE 2 : PAPYRUS FMU IMPORT

From model explorer root : right click, Moka, FMI, Import FMU for co-simulation



EXERCISE 2 : PAPYRUS FMU IMPORT

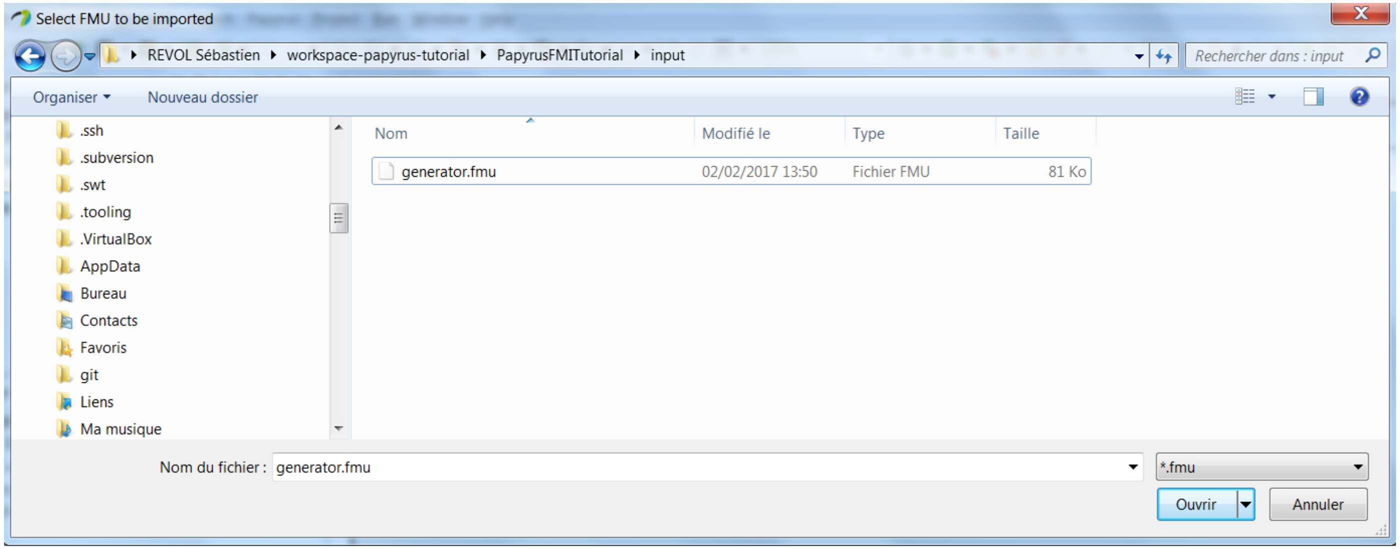
Model Library is useful to group several FMUs



Enter a name for your Model Library

Model Library Name: FMULibrary

Select generator.fmu from workspace/PapyrusFMITutorial/input



Select FMU to be imported

REVOL Sébastien ▸ workspace-papyrus-tutorial ▸ PapyrusFMITutorial ▸ input

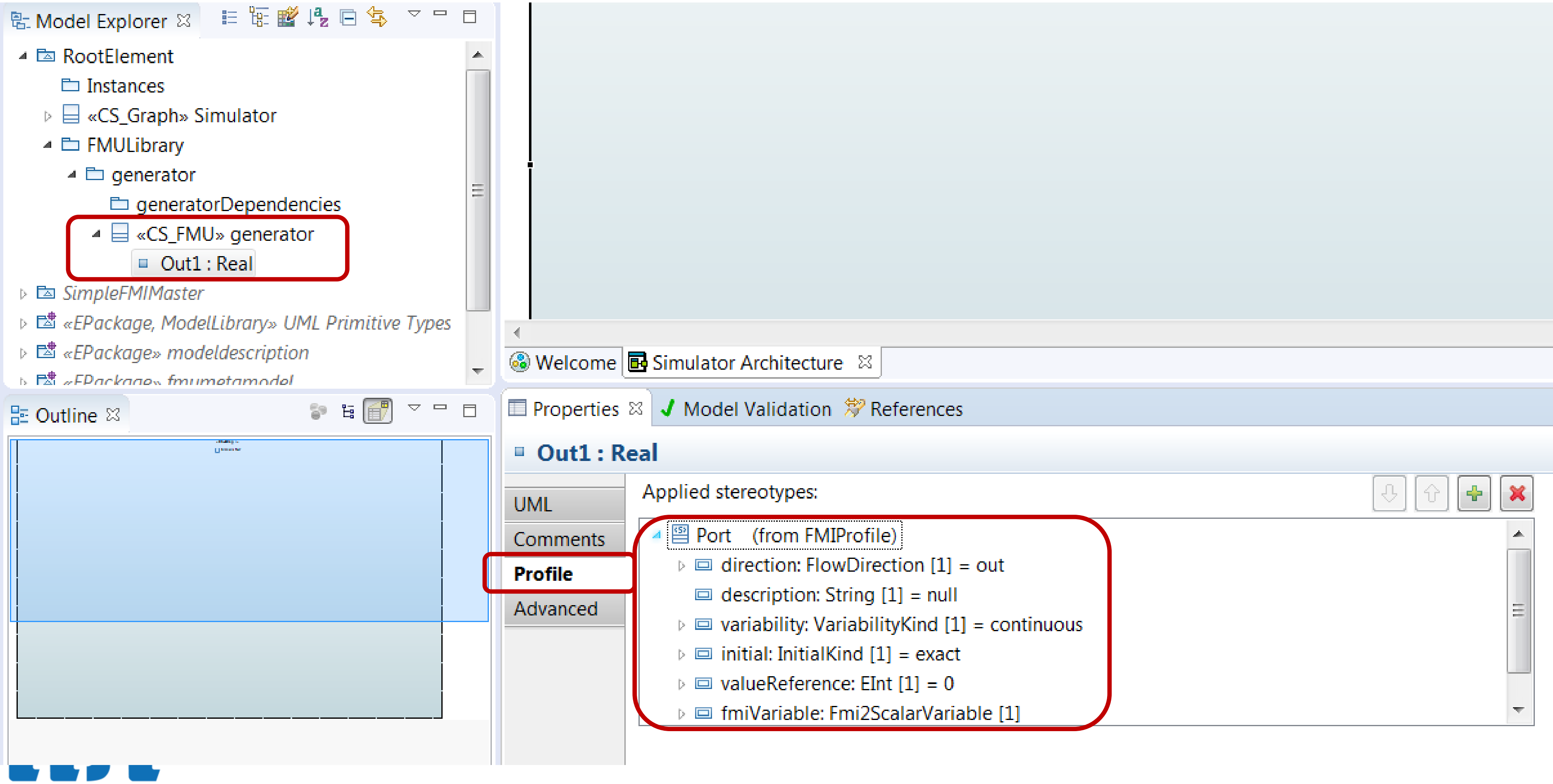
Nom	Modifié le	Type	Taille
generator.fmu	02/02/2017 13:50	Fichier FMU	81 Ko

Nom du fichier : generator.fmu

*.fmu

EXERCISE 2 : PAPYRUS FMU IMPORT

We obtain a new Class named « generator » with an ouput port called « Out1 »



Model Explorer

- RootElement
  - Instances
  - «CS_Graph» Simulator
  - FMULibrary
    - generator
      - generatorDependencies
      - «CS_FMU» generator
        - Out1 : Real
- *SimpleFMIMaster*
- *«EPackage, ModelLibrary» UML Primitive Types*
- *«EPackage» modeldescription*
- *«EPackage» fmumetamodel*

Outline

Welcome | Simulator Architecture

Properties | Model Validation | References

Out1 : Real

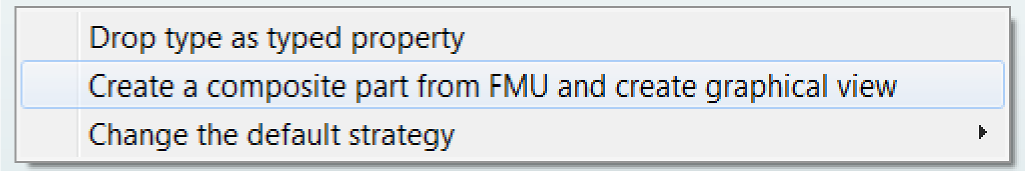
UML | Comments | **Profile** | Advanced

Applied stereotypes:

- Port (from FMIProfile)
  - direction: FlowDirection [1] = out
  - description: String [1] = null
  - variability: VariabilityKind [1] = continuous
  - initial: InitialKind [1] = exact
  - valueReference: EInt [1] = 0
  - fmiVariable: Fmi2ScalarVariable [1]

EXERCISE 2 : PAPYRUS FMU IMPORT

Drag and drop the class into the diagram, and select FMU-specific Papyrus drop strategy

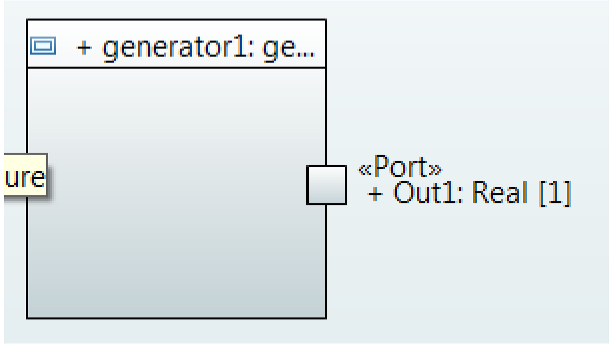


Drop type as typed property

Create a composite part from FMU and create graphical view

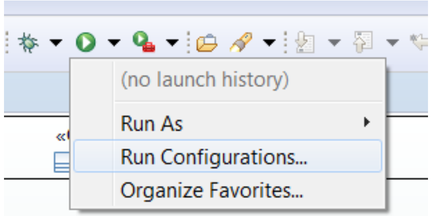
Change the default strategy

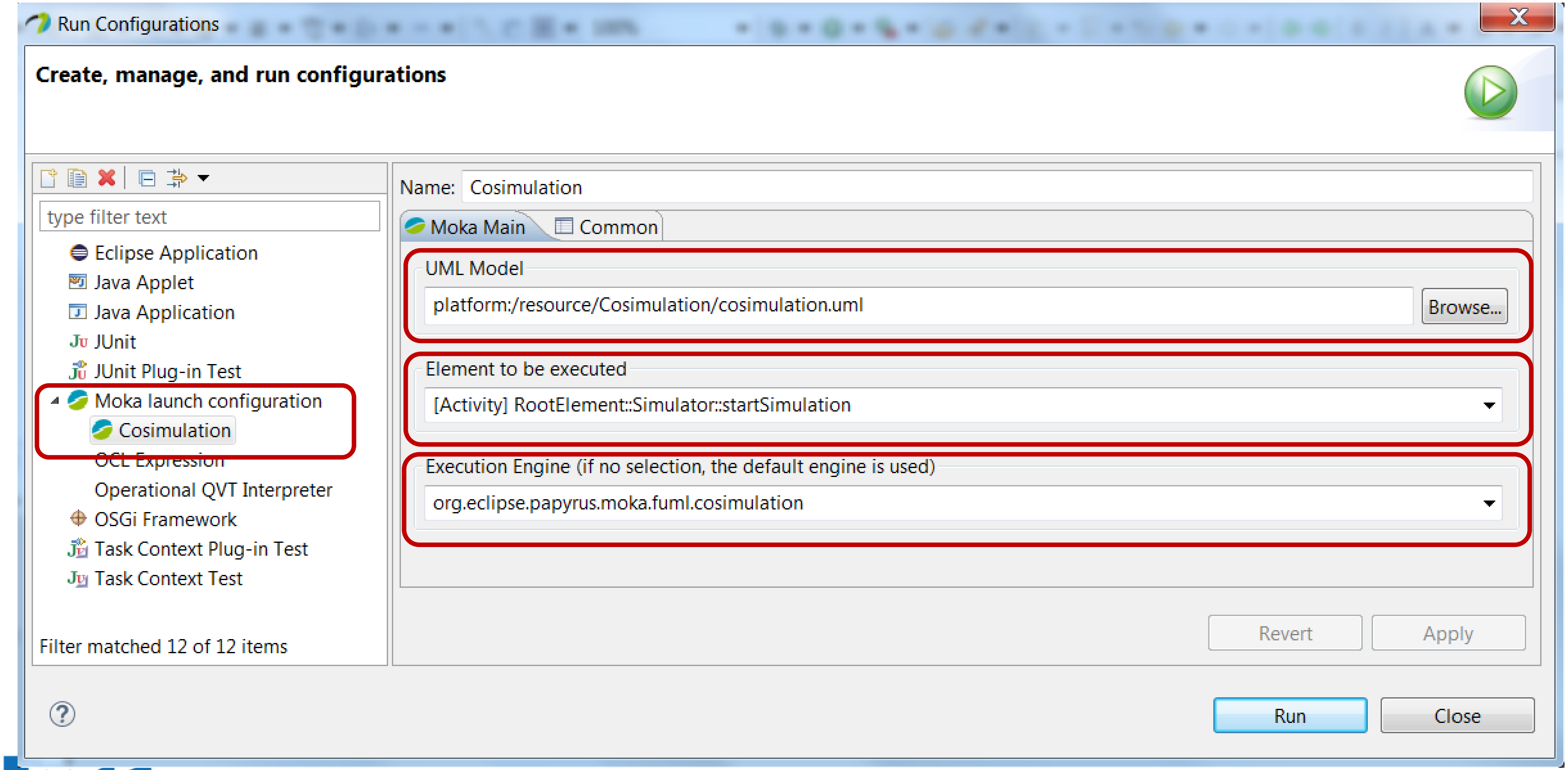
We get a new FMU instance



EXERCISE 2 : PAPYRUS FMU IMPORT

Create a new Moka Run configuration





Create, manage, and run configurations

Name: Cosimulation

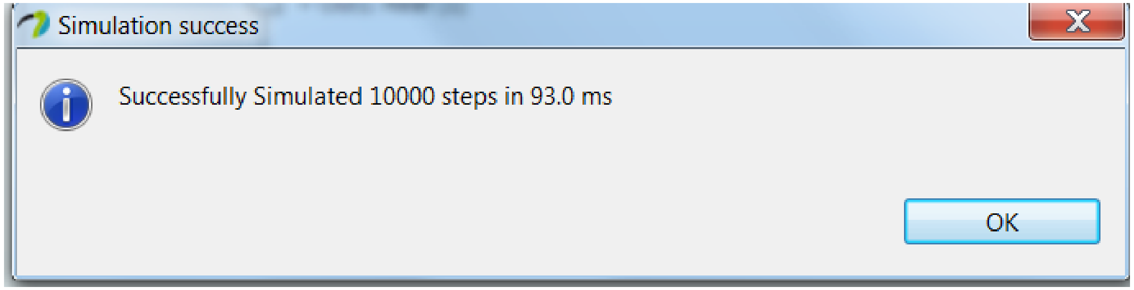
UML Model: platform:/resource/Cosimulation/cosimulation.uml

Element to be executed: [Activity] RootElement::Simulator::startSimulation

Execution Engine (if no selection, the default engine is used): org.eclipse.papyrus.moka.fuml.cosimulation

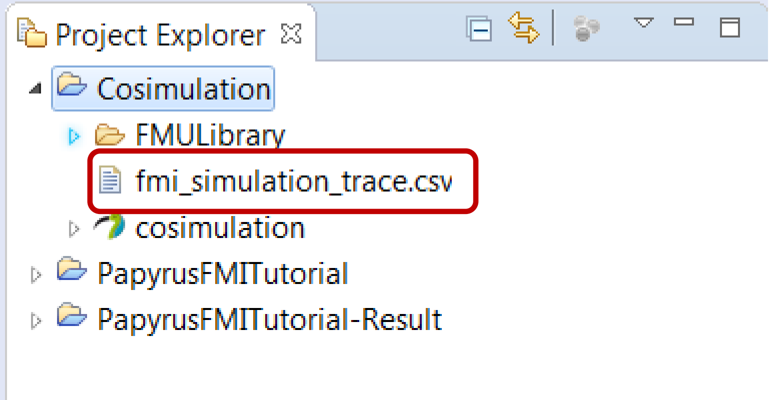
EXERCISE 2 : PAPYRUS FMU IMPORT

If everything is ok….



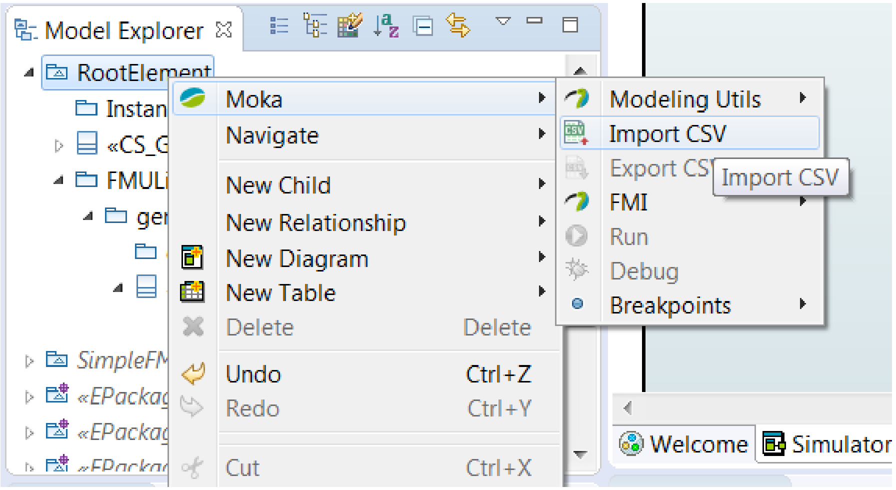
Successfully Simulated 10000 steps in 93.0 ms

A new simulation trace appears in project explorer (after refresh, press F5)

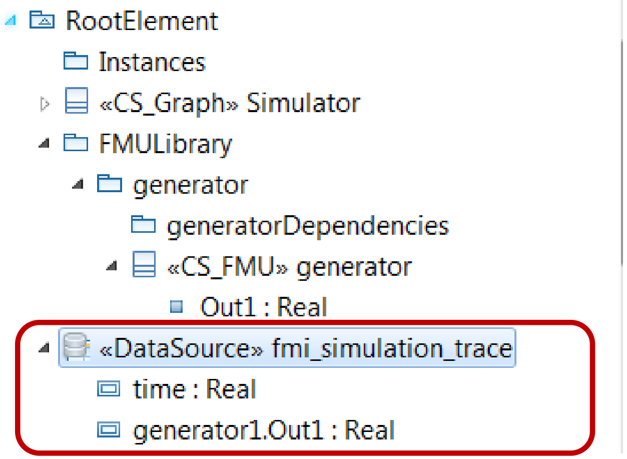


EXERCISE 2 : PAPYRUS FMU IMPORT

Import CSV into Papyrus Model



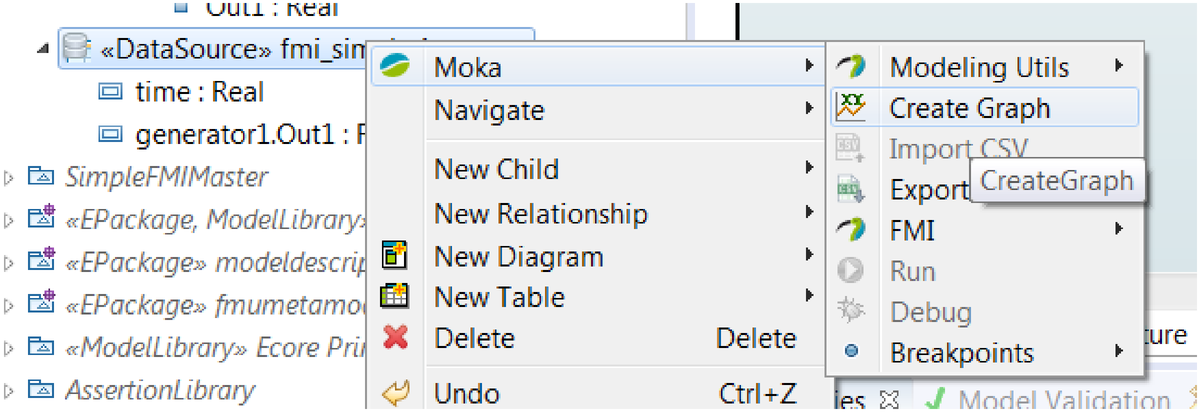
A new « DataSource » appears in model explorer



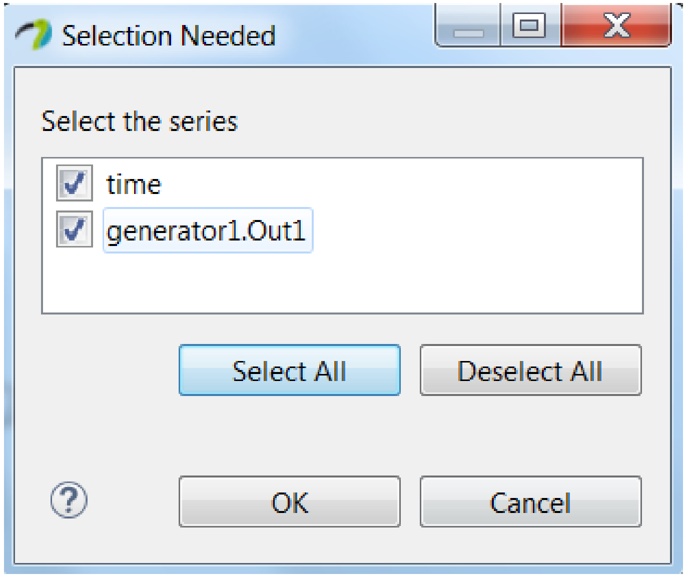
- RootElement
  - Instances
  - «CS_Graph» Simulator
  - FMULibrary
    - generator
      - generatorDependencies
      - «CS_FMU» generator
        - Out1 : Real
  - «DataSource» fmi_simulation_trace
    - time : Real
    - generator1.Out1 : Real

EXERCISE 2 : PAPYRUS FMU IMPORT

Create a new graph from the data source



Select the traces to display

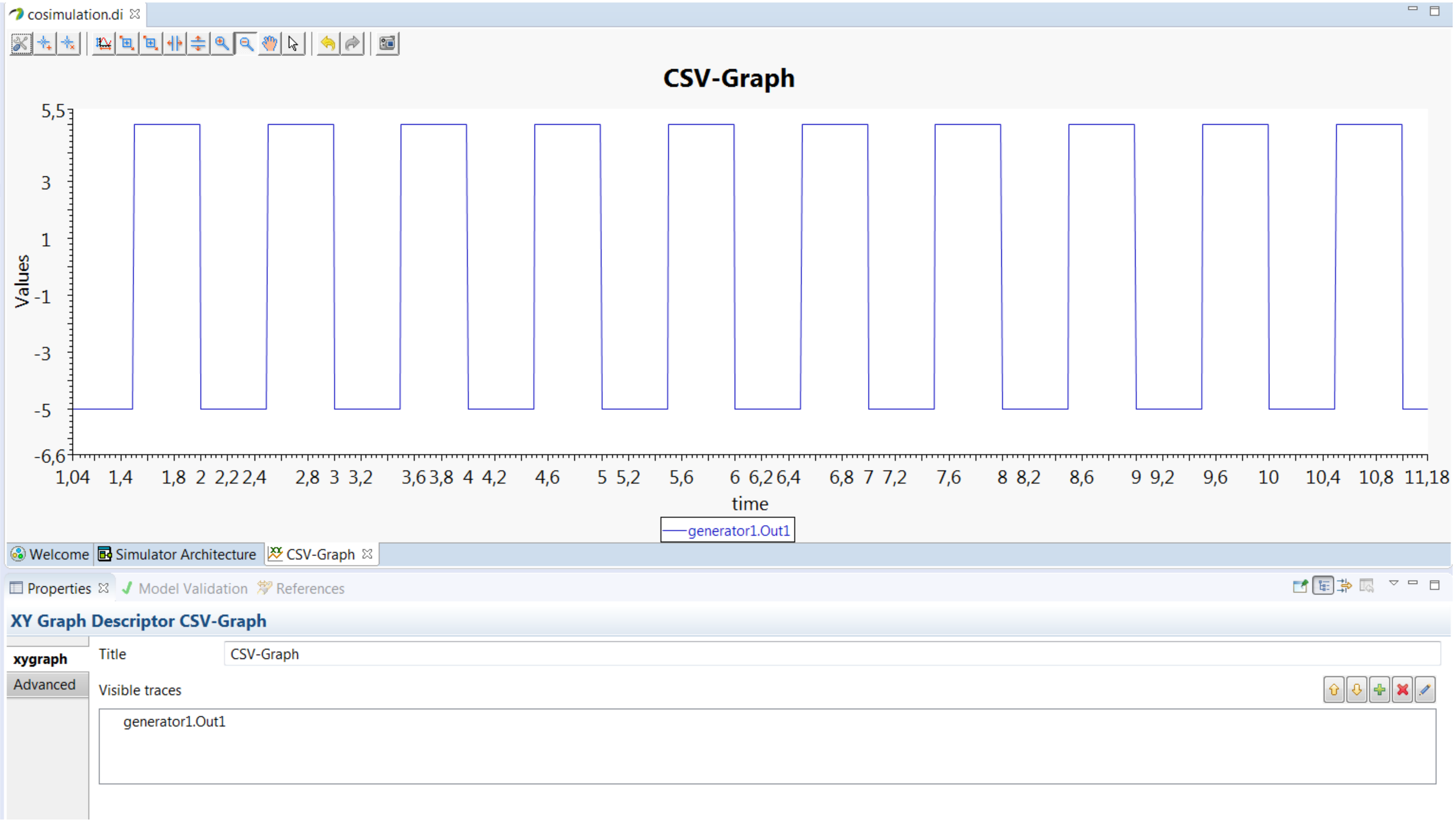


Select the series

☑ time
☑ generator1.Out1

EXERCISE 2 : PAPYRUS FMU IMPORT

XY graphes are new kinds of Papyrus Diagrams



XY Graph Descriptor CSV-Graph

xygraph

Advanced

Title: CSV-Graph

Visible traces

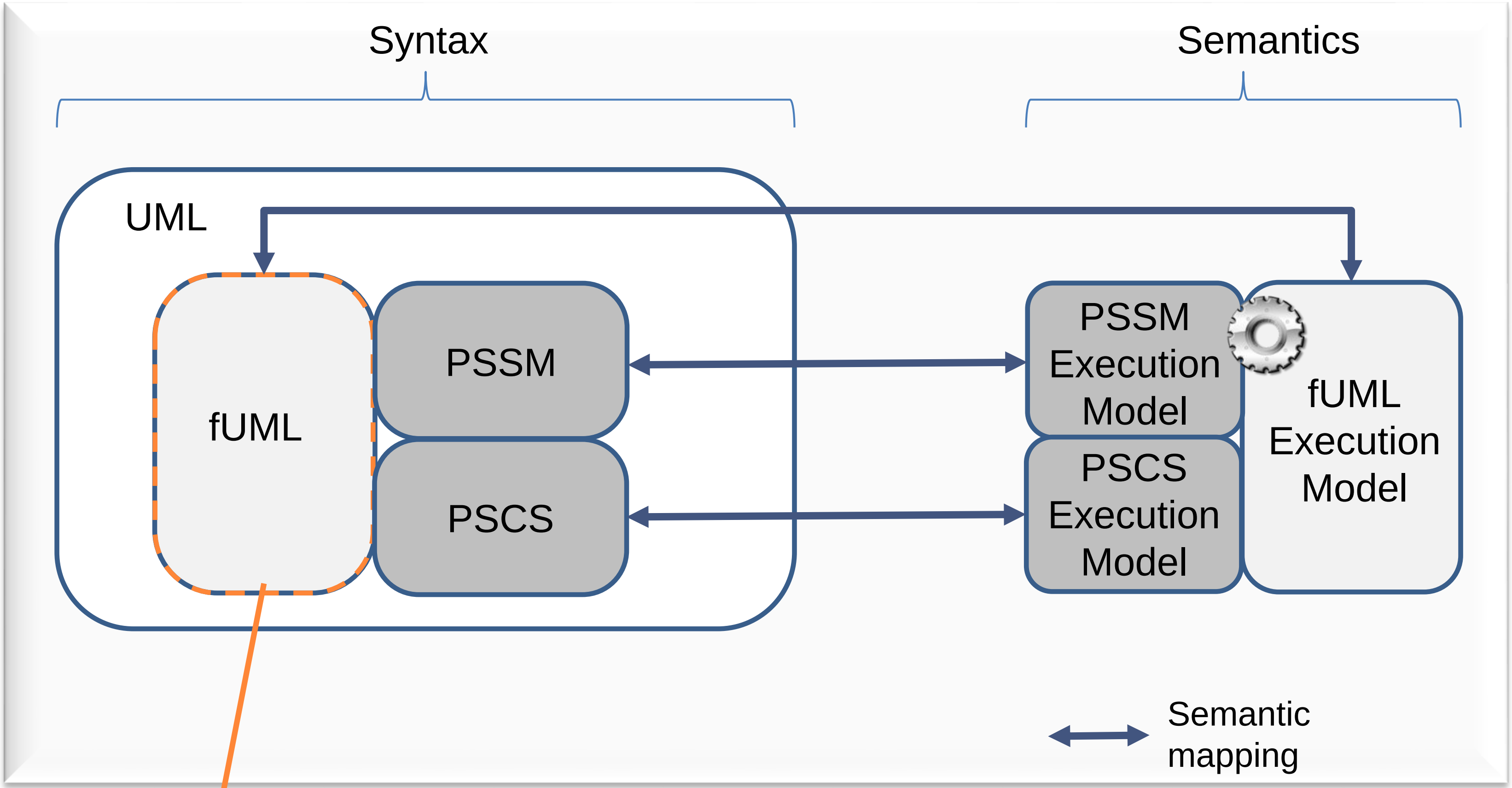
generator1.Out1

PART III

–

PAPYRUS AS FMU DESIGNER

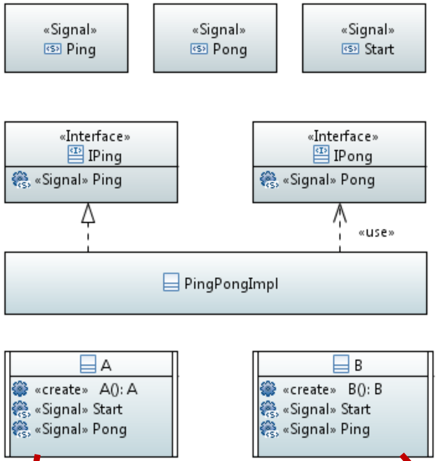
EXECUTABLE UML OMG SPECIFICATIONS



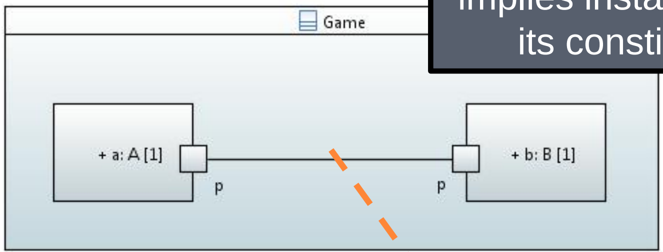
Alf (Action Language for fUML):
- Textual surface notation for the fUML subset

KEY SEMANTIC ASPECTS

Structure



1. Class diagram (~ BDD)

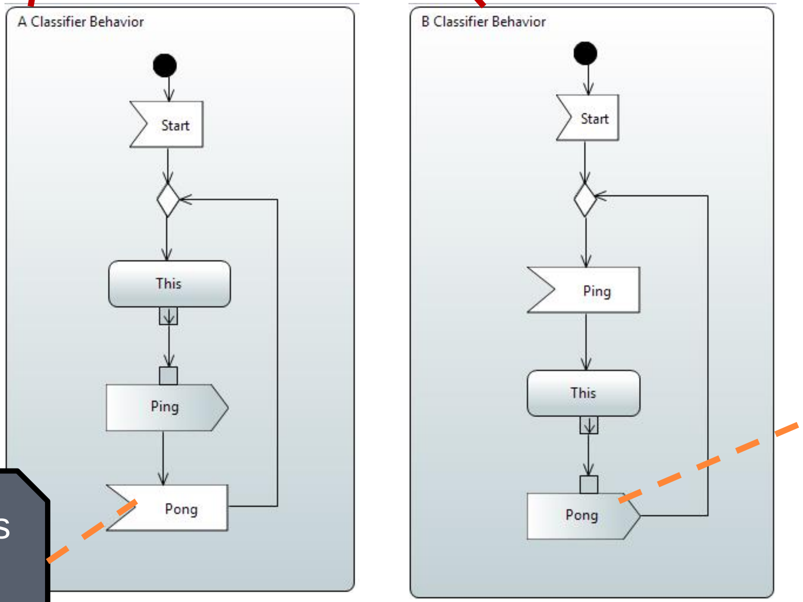


2. Composite structure diagram (~ IBD)

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

THE FMU EXPORT FUNCTIONNALITY

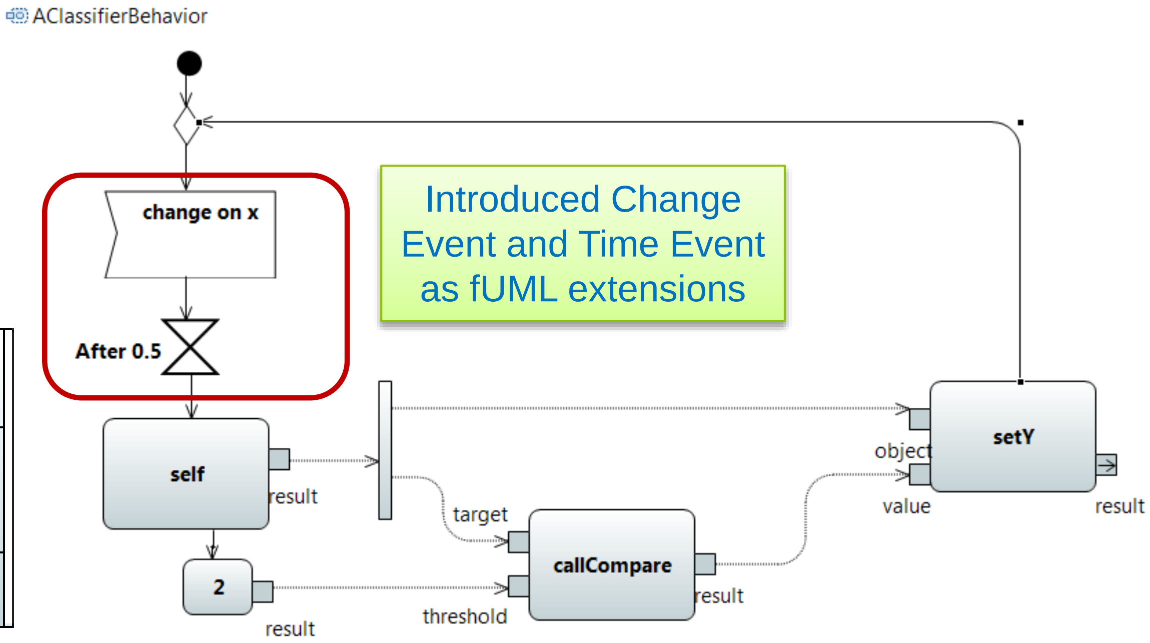
MODELING AND SIMULATION OF REACTIVE ASPECTS

An FMU is an fUML Active Object with a classifier behavior described with and Activity diagram
(state-machine support on-going)

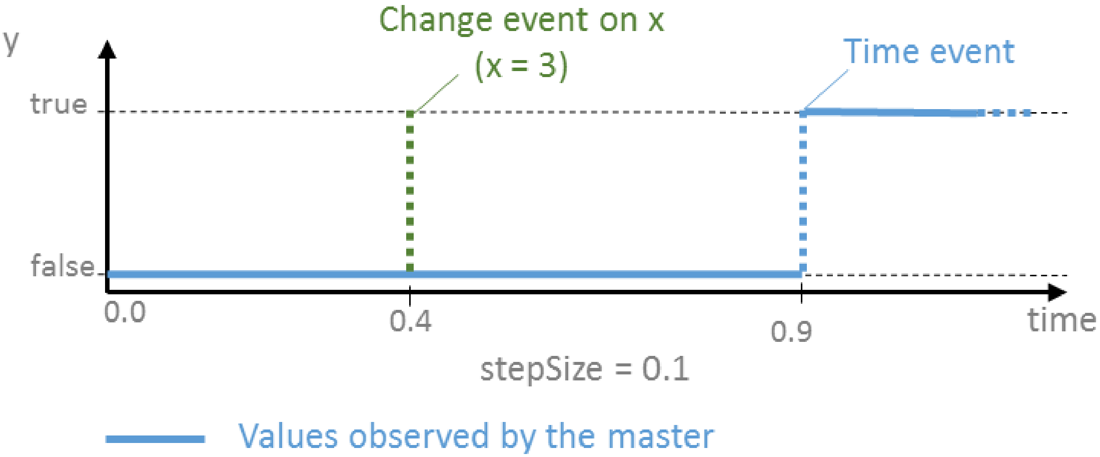
«CS_FMU»
FMU

«Port» {direction=in } x: Integer
«Port» {direction=out } y: Boolean

compare(in threshold: Integer): Boolean

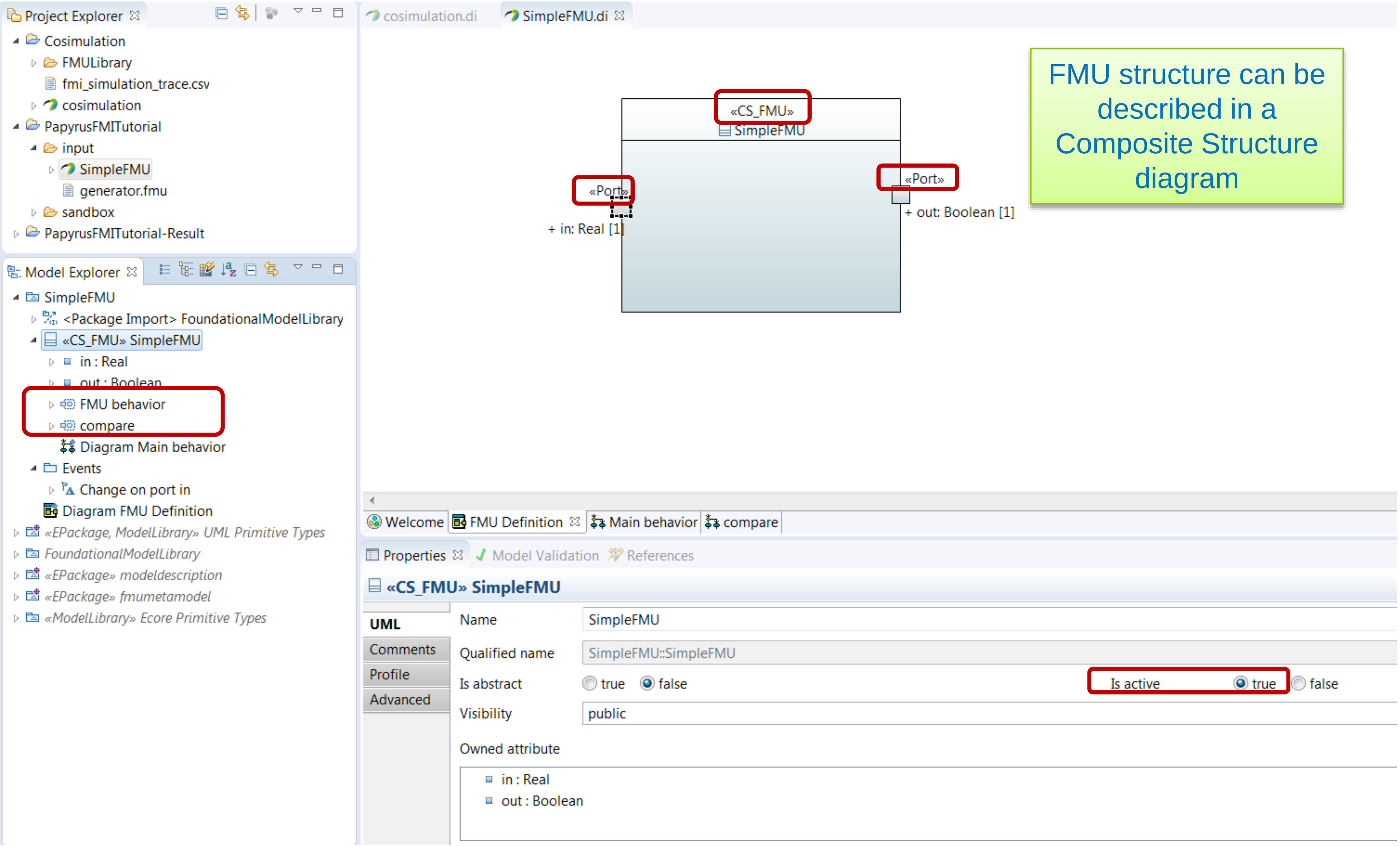


Introduced Change Event and Time Event as fUML extensions



EXERCISE 3 : : PAPYRUS FMU DESIGN

Open PapyrusFMITutorial/input/SimpleFMU UML model



FMU structure can be described in a Composite Structure diagram

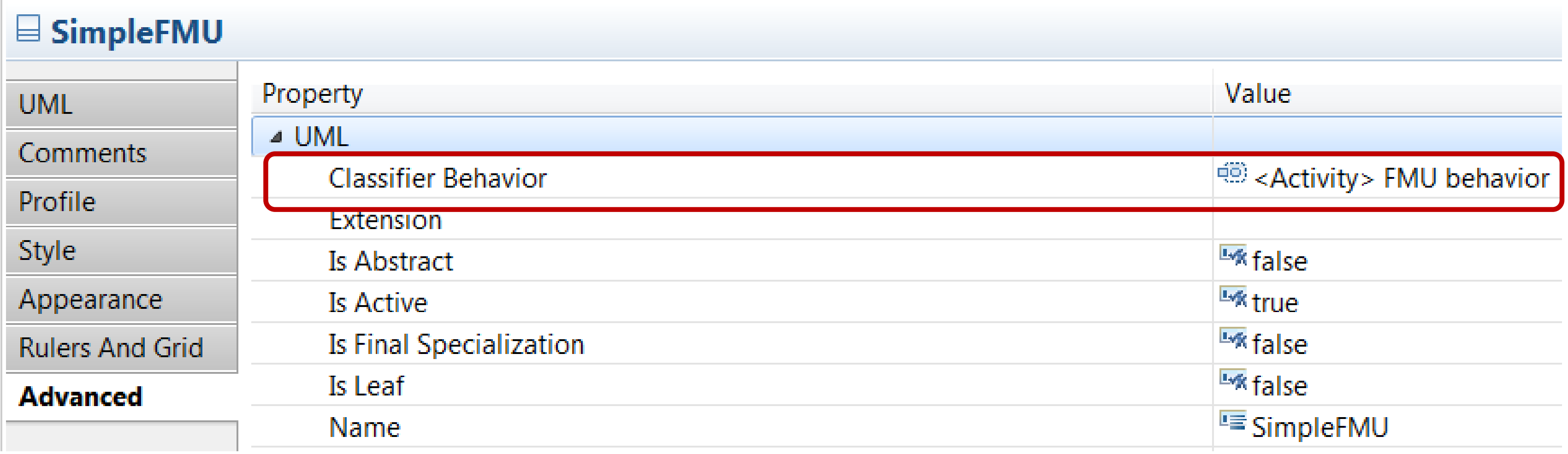
Name	SimpleFMU
Qualified name	SimpleFMU::SimpleFMU
Is abstract	○ true ● false
Is active	● true ○ false
Visibility	public

Owned attribute

- in : Real
- out : Boolean

EXERCISE 3 : : PAPYRUS FMU DESIGN

- **FMU Class key aspects :**
  - **Should be an active class**
  - **Should be stereotyped with FMIProfile::CS_FMU stereotype**
    - No need to feel stereotypes attributes, they will be filled by Moka at export time
  - **Can own several behaviors**
    - Only one should be referenced as Classifier behavior
    - Other behaviors can be called from the Classifier behavior



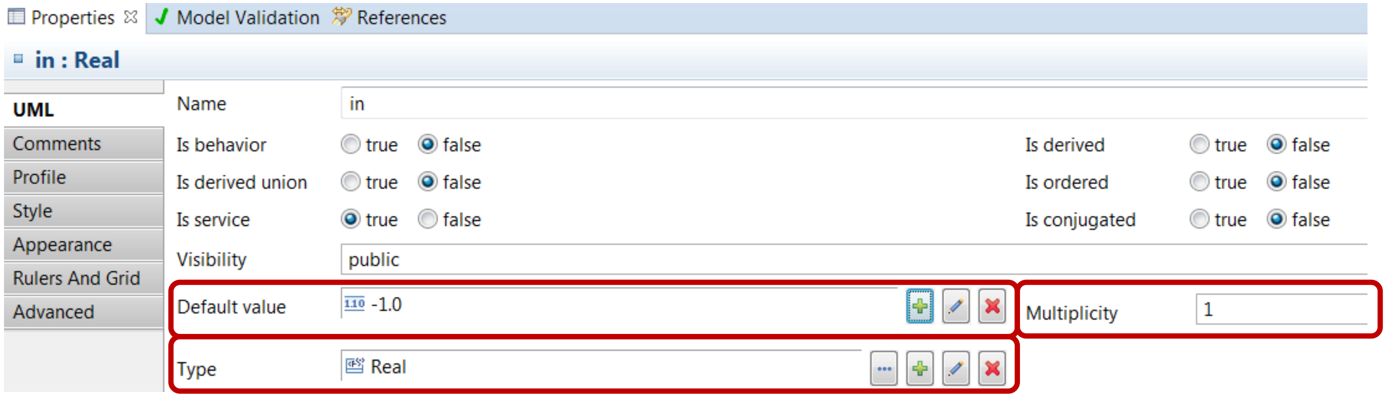
SimpleFMU

UML
Comments
Profile
Style
Appearance
Rulers And Grid
Advanced

Property	Value
UML	
Classifier Behavior	<Activity> FMU behavior
Extension	
Is Abstract	false
Is Active	true
Is Final Specialization	false
Is Leaf	false
Name	SimpleFMU

EXERCISE 3 : : PAPYRUS FMU DESIGN

- **FMU Port key aspects**
  - **Should be stereotyped with FMIProfile::Port stereotype**
    - **direction** (in/out) and **valueReference** (unique ID) should be specified
    - Other attributes will be generated at FMU export
  - **Ports should have a type**
    - Only UML standard primitive types (Integer, Boolean, String, Real)
  - **Ports should have a default value**
    - Only UML primitive types values (LiteralInteger, LiteralBoolean, LiteralString, LiteralReal)
  - **Multplicity must be set to 1**



in : Real

Name: in

Is behavior ○ true ● false

Is derived union ○ true ● false

Is service ● true ○ false

Is derived ○ true ● false

Is ordered ○ true ● false

Is conjugated ○ true ● false

Visibility: public

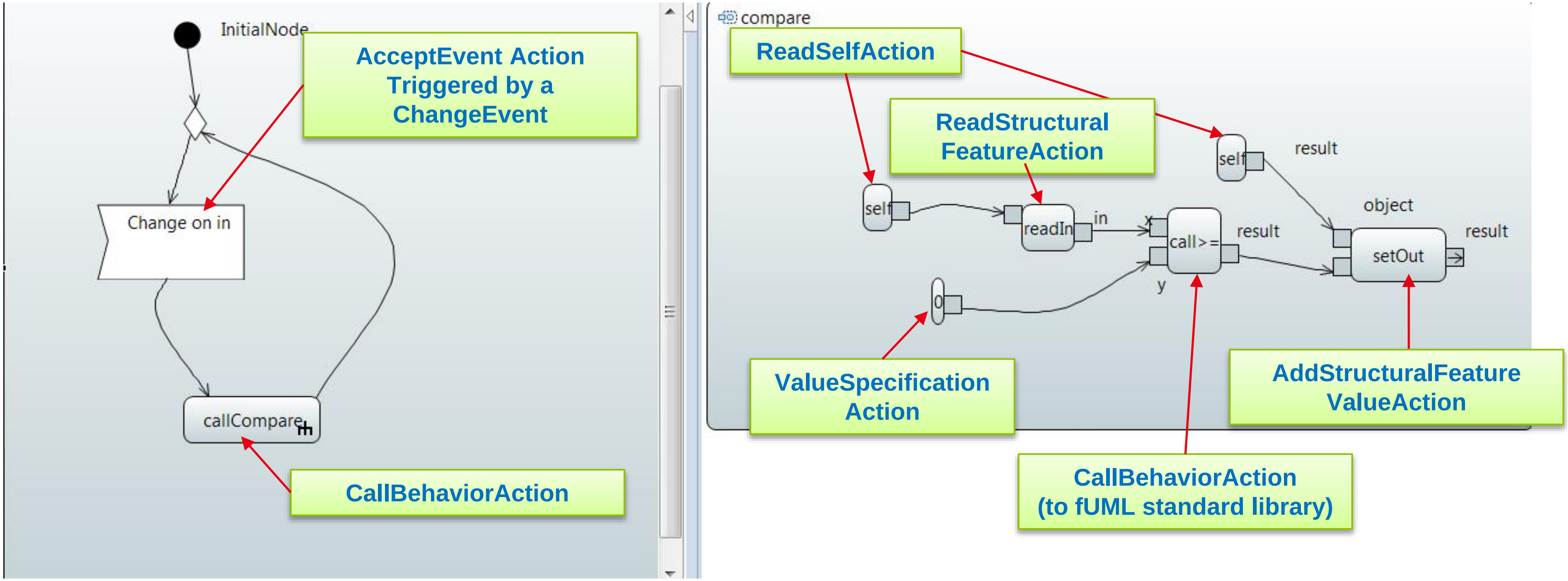
Default value: -1.0

Multiplicity: 1

Type: Real

EXERCISE 3 : : PAPYRUS FMU DESIGN

- **Simple FMU behavior**
  - Unfinite loop
  - Waiting on input changes
  - And comparing the input versus 0
  - Write true on output if input greater or equals to 0



AcceptEvent Action Triggered by a ChangeEvent

CallBehaviorAction

ReadSelfAction

ReadStructural FeatureAction

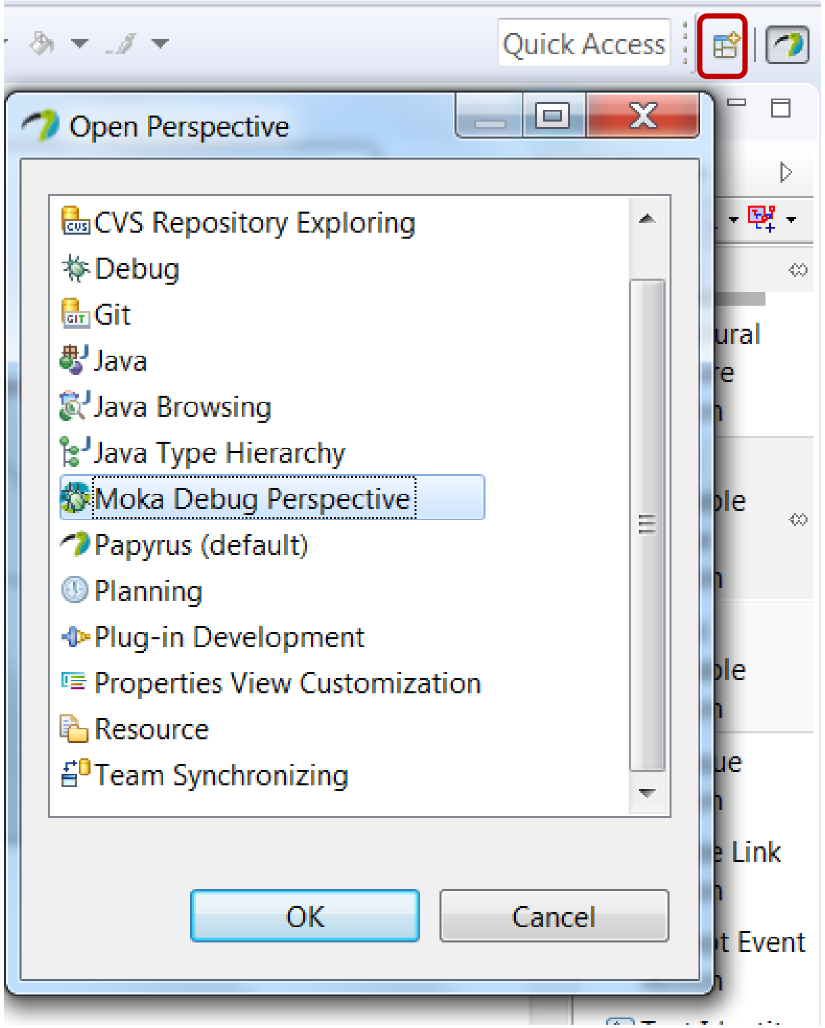
ValueSpecification Action

AddStructuralFeature ValueAction

CallBehaviorAction (to fUML standard library)

EXERCISE 3 : : PAPYRUS FMU DESIGN

- **Switch to Moka Debug Perspective**

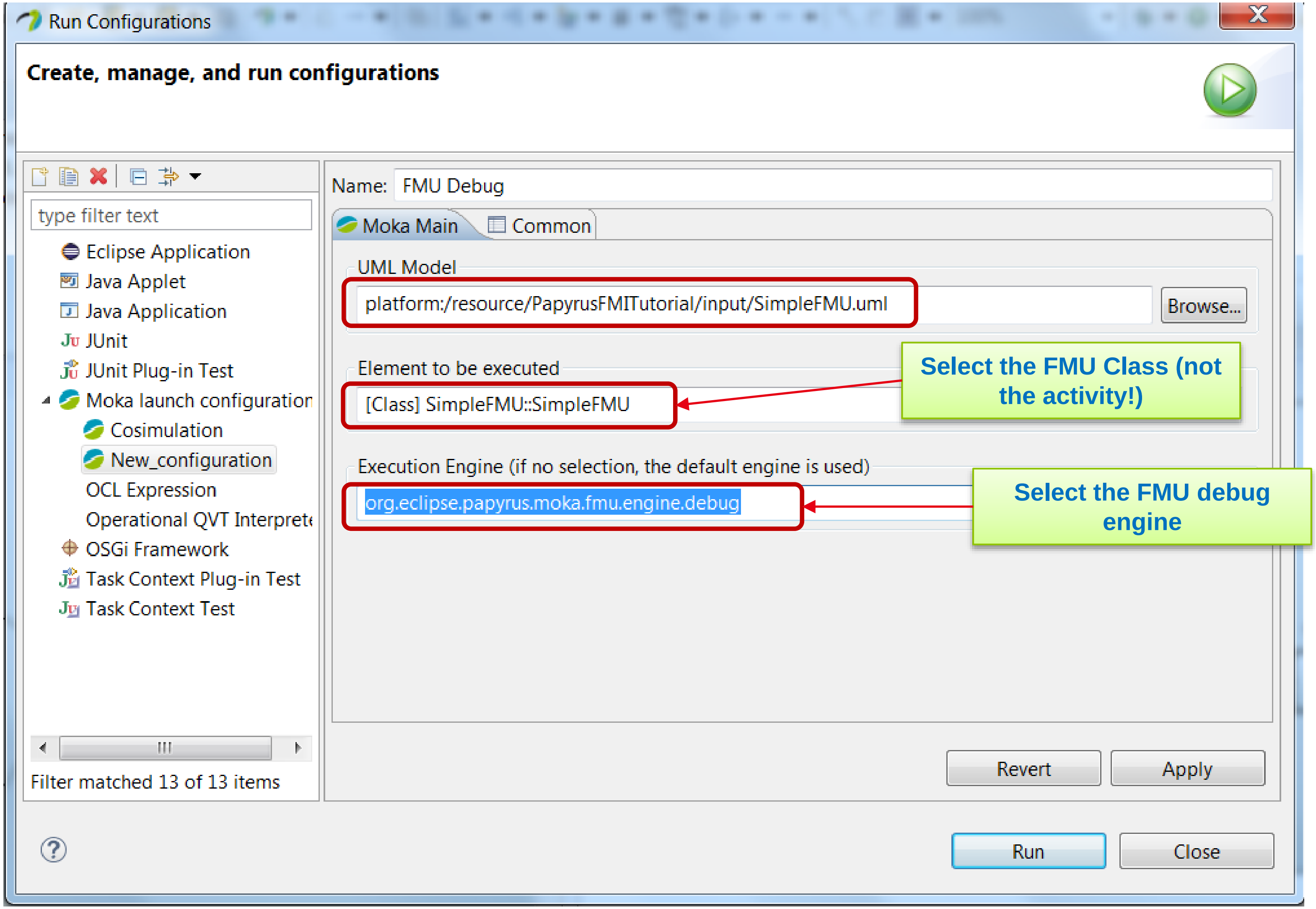


Open Perspective

- CVS Repository Exploring
- Debug
- Git
- Java
- Java Browsing
- Java Type Hierarchy
- Moka Debug Perspective
- Papyrus (default)
- Planning
- Plug-in Development
- Properties View Customization
- Resource
- Team Synchronizing

EXERCISE 3 : PAPYRUS FMU EXPORT

- **Create new Moka run configuration**



Create, manage, and run configurations

Filter matched 13 of 13 items

Name: FMU Debug

UML Model: platform:/resource/PapyrusFMITutorial/input/SimpleFMU.uml

Element to be executed: [Class] SimpleFMU::SimpleFMU

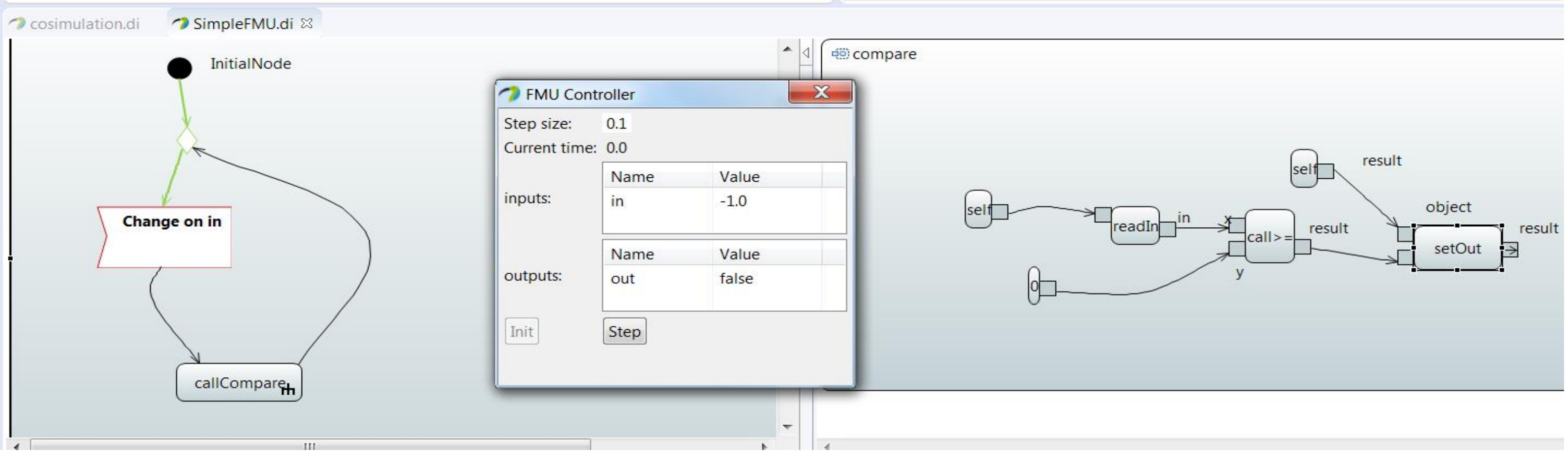
Select the FMU Class (not the activity!)

Execution Engine (if no selection, the default engine is used): org.eclipse.papyrus.moka.fmu.engine.debug

Select the FMU debug engine

EXERCISE 3 : PAPYRUS FMU DESIGN

- **FMU controller allows to :**
  - Change inputs
  - Configure size and run FMI steps



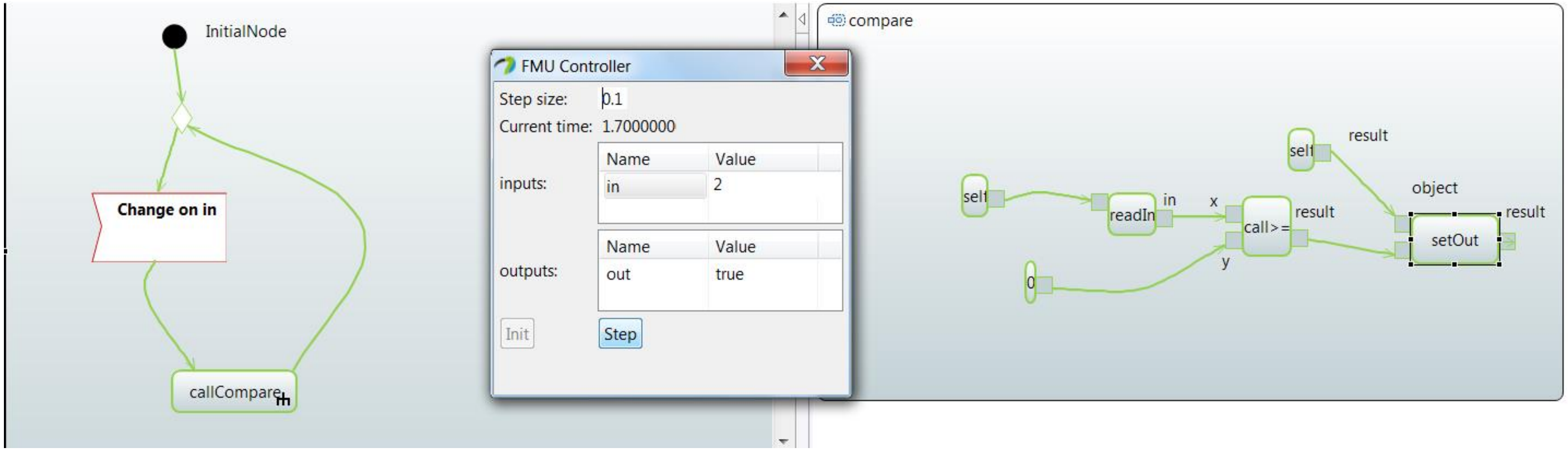
Step size: 0.1
Current time: 0.0

inputs:

Name	Value
in	-1.0

outputs:

Name	Value
out	false



Step size: 0.1
Current time: 1.7000000

inputs:

Name	Value
in	2

outputs:

Name	Value
out	true

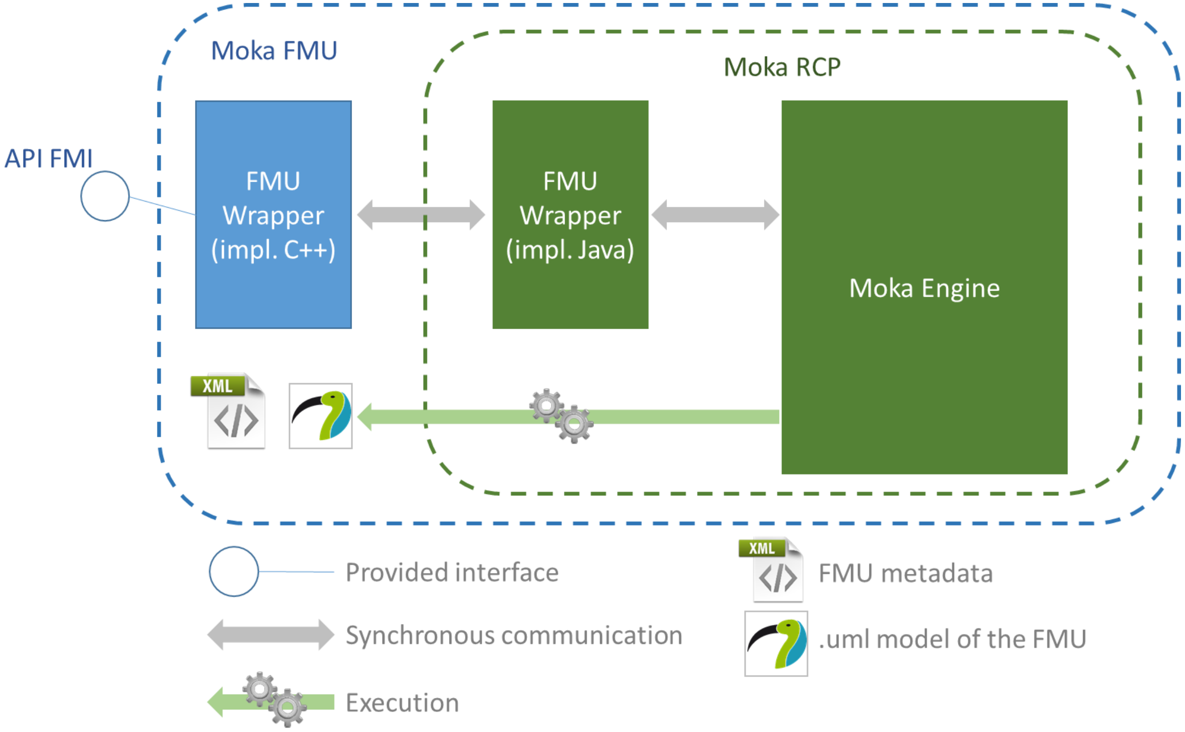
PART IV

–

FMU GENERATION

THE FMU EXPORT FUNCTIONNALITY

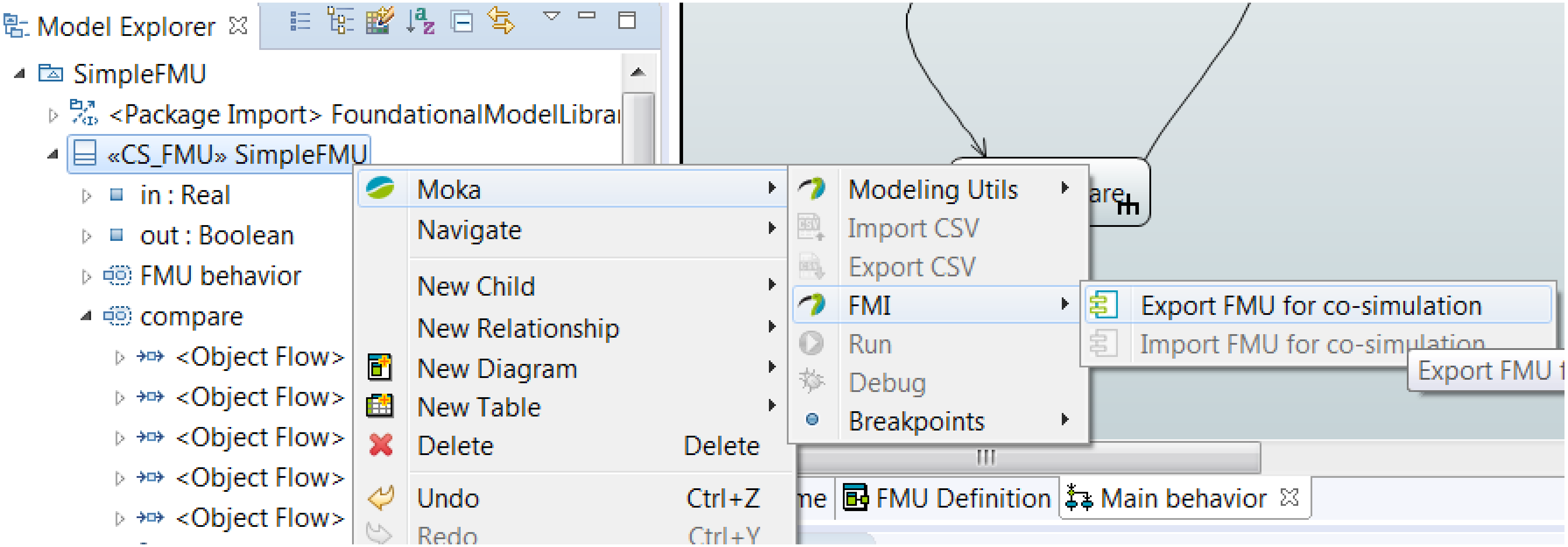
- **Architecture of exported FMUs**



- **No code generation**
  - Only the modelDescription.xml is generated
  - The generated FMU includes the UML model a minimal Moka interpreter
  - And a generic DLL implementing the FMI interface and interacting with Moka

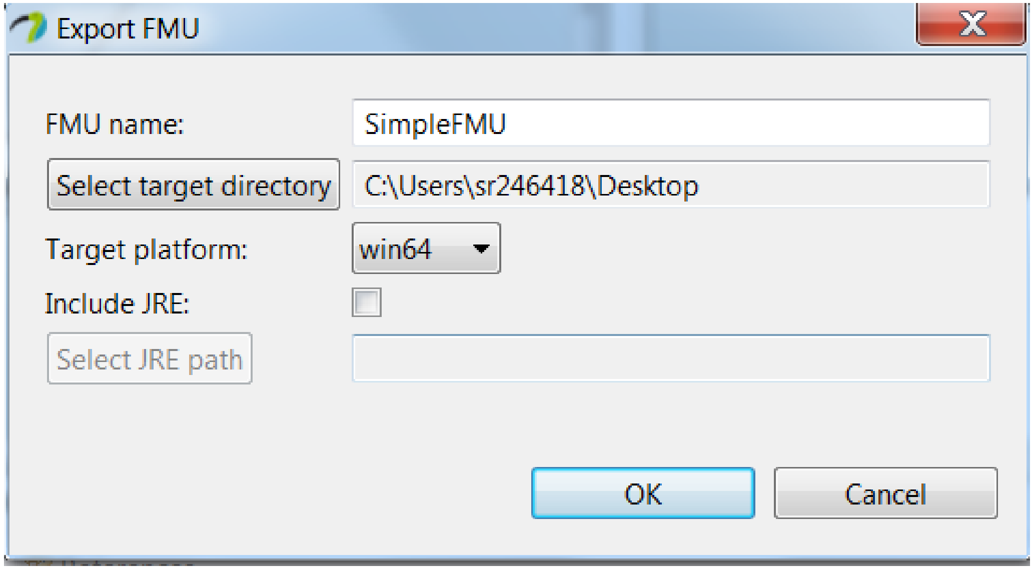
EXERCISE 4 : PAPYRUS FMU GENERATION

- **From the FMU Class : right click, Moka, FMI, Export FMU for co-simulation**



EXERCISE 4 : PAPYRUS FMU GENERATION

- **Provide an FMU name (FMI model identifier)**
- **Select the target directory**
- **Select the target platform**
  - **Currently only win64, Linux32 and Linux64 are supported**
  - **Other platforms can be supported on demand**
- **Optionally : a JRE can be embedded in the FMU**
  - **Can be a minimal JRE (example Linux Embedded)**
  - **Useful if target platform doesn't have a JRE installed**



Export FMU

FMU name: SimpleFMU

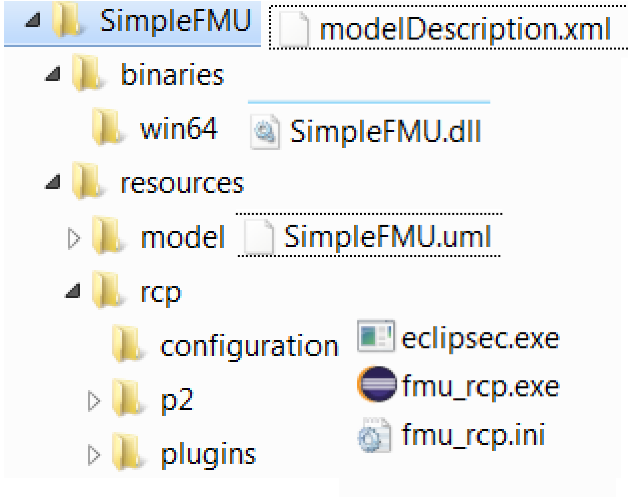
Select target directory: C:\Users\sr246418\Desktop

Target platform: win64

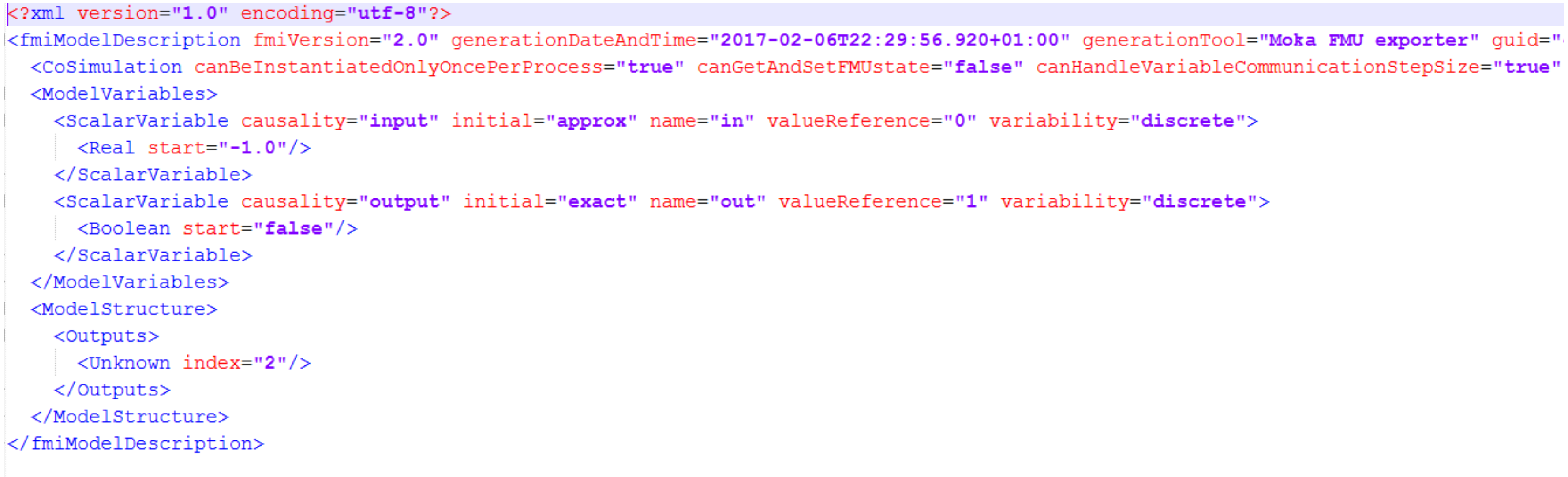
Include JRE: □

EXERCISE 4 : PAPYRUS FMU GENERATION

- **FMU structure and modelDescription.xml**



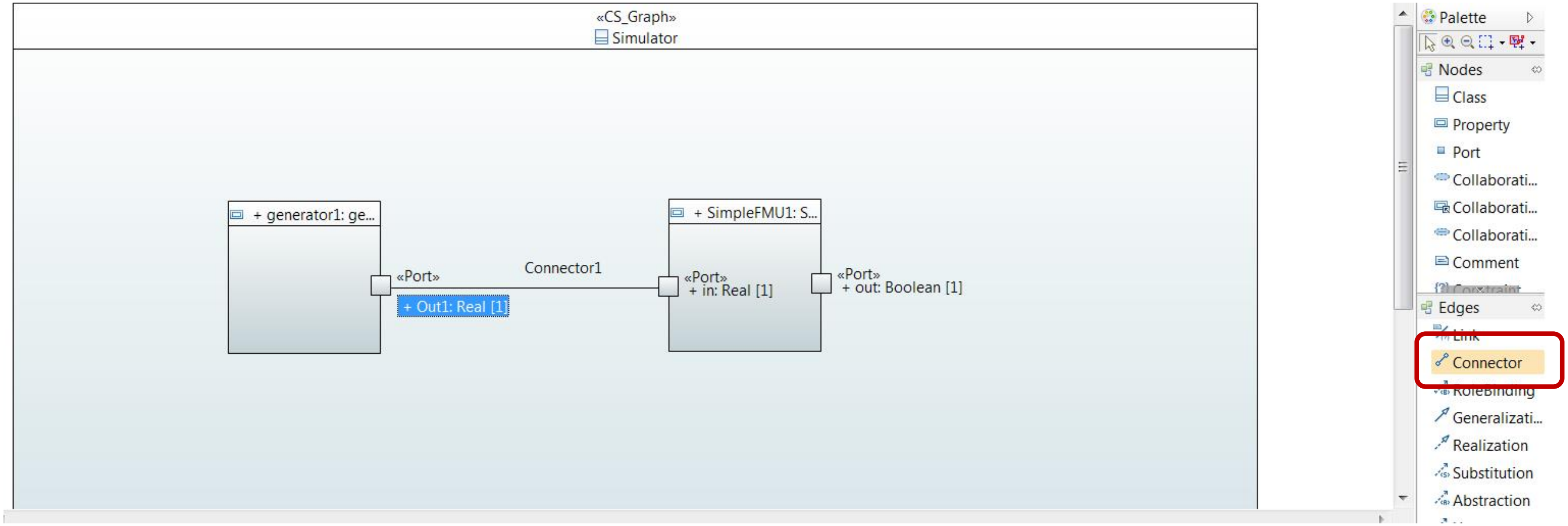
```
SimpleFMU                modelDescription.xml
  binaries
    win64                SimpleFMU.dll
  resources
    model                SimpleFMU.uml
    rcp
      configuration      eclipsec.exe
      p2                 fmu_rcp.exe
      plugins            fmu_rcp.ini
```



```xml
<?xml version="1.0" encoding="utf-8"?>
<fmiModelDescription fmiVersion="2.0" generationDateAndTime="2017-02-06T22:29:56.920+01:00" generationTool="Moka FMU exporter" guid="
  <CoSimulation canBeInstantiatedOnlyOncePerProcess="true" canGetAndSetFMUstate="false" canHandleVariableCommunicationStepSize="true"
  <ModelVariables>
    <ScalarVariable causality="input" initial="approx" name="in" valueReference="0" variability="discrete">
      <Real start="-1.0"/>
    </ScalarVariable>
    <ScalarVariable causality="output" initial="exact" name="out" valueReference="1" variability="discrete">
      <Boolean start="false"/>
    </ScalarVariable>
  </ModelVariables>
  <ModelStructure>
    <Outputs>
      <Unknown index="2"/>
    </Outputs>
  </ModelStructure>
</fmiModelDescription>
```

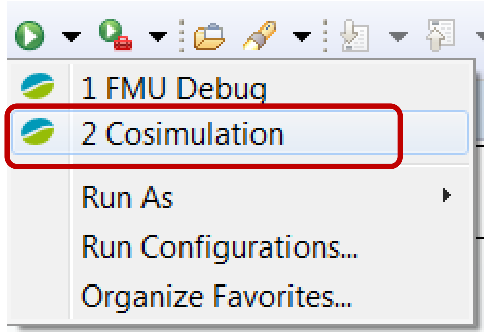
EXERCISE 4 : PAPYRUS FMU GENERATION

- **Import generated FMU in first co-simulation model (cf exercise 2)**
  - **Connect generator output to SimpleFMU input**

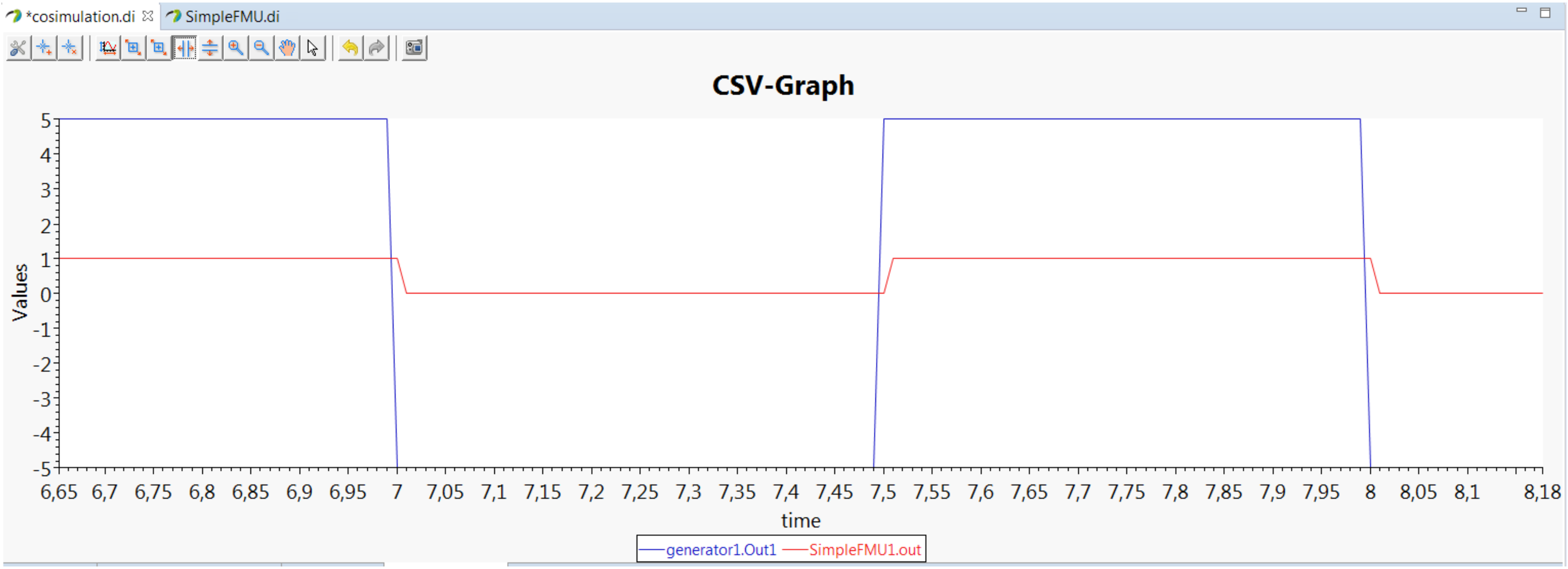


EXERCISE 4 : PAPYRUS FMU GENERATION

- **Re-run simulation**



- **Re-import new CSV**



REMAINING WORK

- **On Master Side :**
  - FMU parameters configuration (almost there!)
  - Simulation debug (breakpoints at time, at port value, step by step simulation, runtime values visualization…)
  - Delegation to external master (Cosim or Model exchange)
  - Improve logging interface (select values to be logged, direct graph generation without CSV import)
  - .mat file simulation trace support

- **On Slave Side :**
  - State machine support (almost there!)
  - Rollback support
  - Performance improvement
  - New target platform support

- **Part of these features will be developed in OpenCPS ITEA project**

THANK YOU

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

papyrus

www.eclipse.org/papyrus

GETTING STARTED WITH MOKA:

HTTPS://WIKI.ECLIPSE.ORG/PAPYRUS/USERGUIDE/MODELEXECUTION

VIDEO TUTORIALS :

HTTPS://WWW.YOUTUBE.COM/CHANNEL/UCXYPOBLZC_RKLS7_K2DTWYA