

FMI – Functional Mock-up Interface Specification and Applications

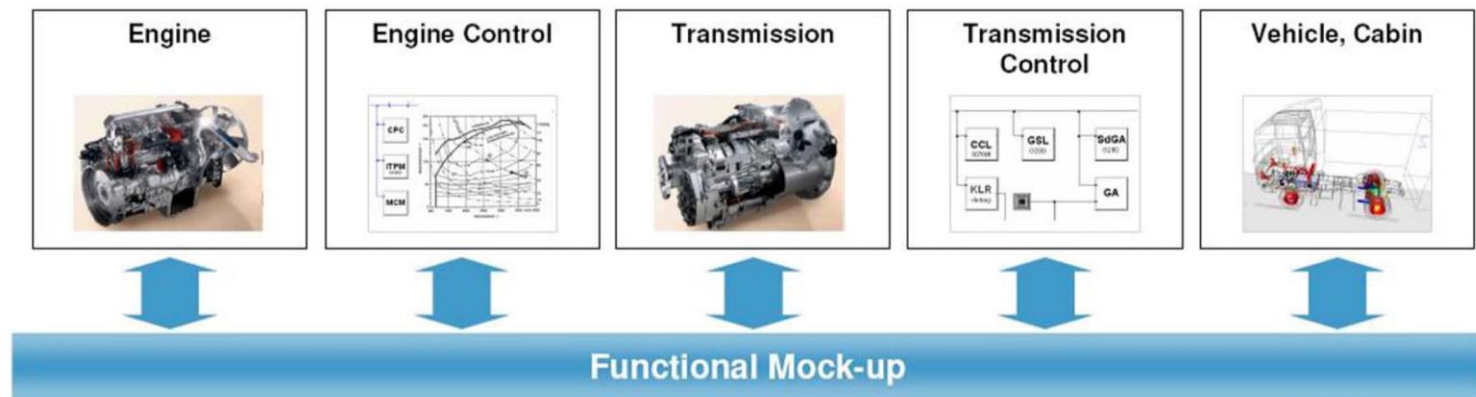
Edmund Widl, AIT Austrian Institute of Technology, Energy Department

ERIGrid JRA2 Workshop

October 5, 2016, Bilbao, Spain

FMI – Functional Mock-up Interface (1/2)

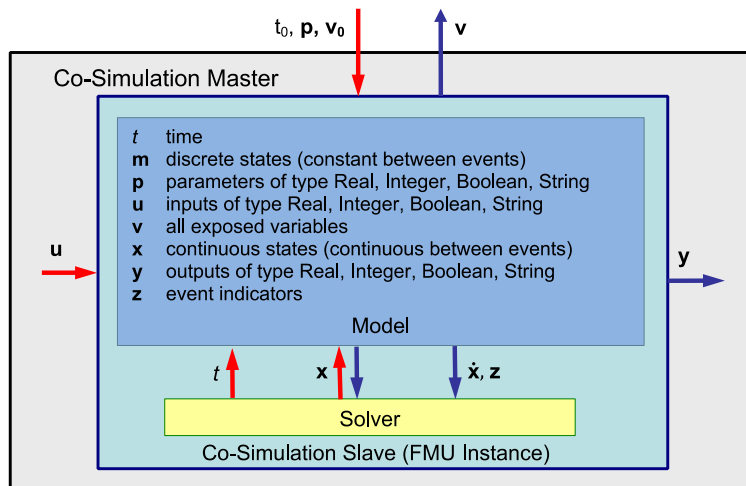
- FMI has been developed to **encapsulate** and **link** models and simulators
 - developed within MODELISAR project
 - driven by a community from *industry and academia*
 - *standardized encapsulation* of *models* and *tools*
 - first version published in 2010, second version published in 2014
 - initially supported by 35 tools, currently *supported by more than 85 tools*
 - see: <https://www.fmi-standard.org/>



Cosimulation of the behavioral models and the embedded controller software

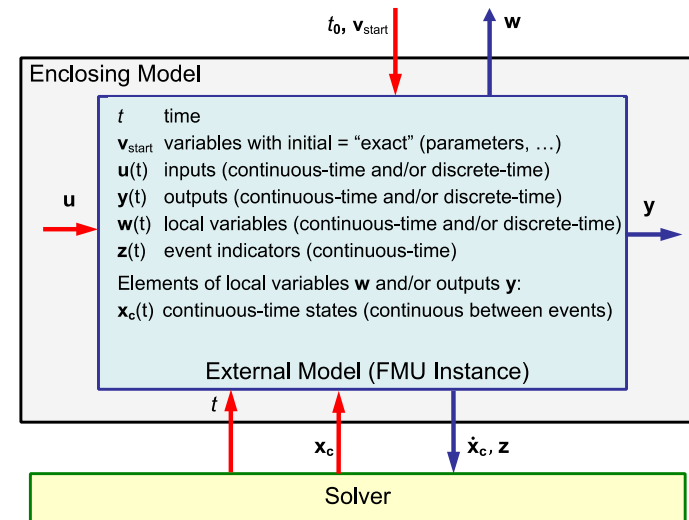
FMI – Functional Mock-up Interface (2/2)

Co-Simulation (CS)



- stand-alone black-box simulation components
- data exchange restricted to discrete communication points
- between two communication points system model is solved by internal solver
- may call another tool at run-time (tool coupling)

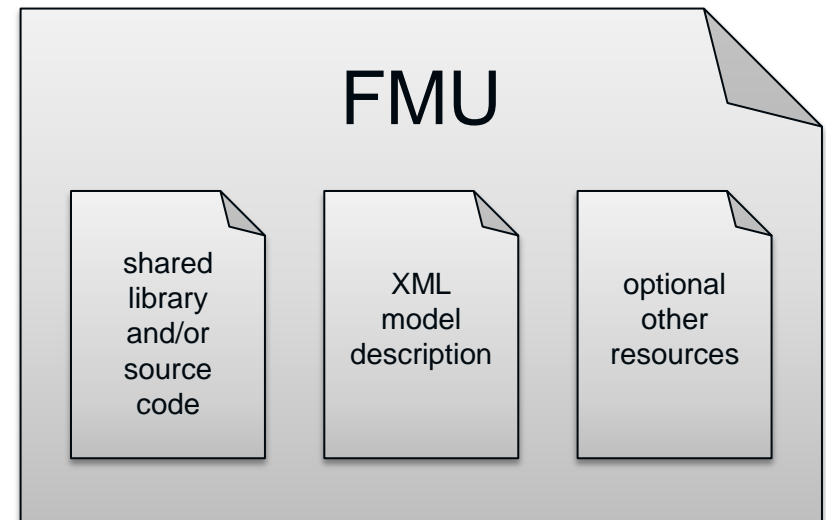
Model Exchange (ME)



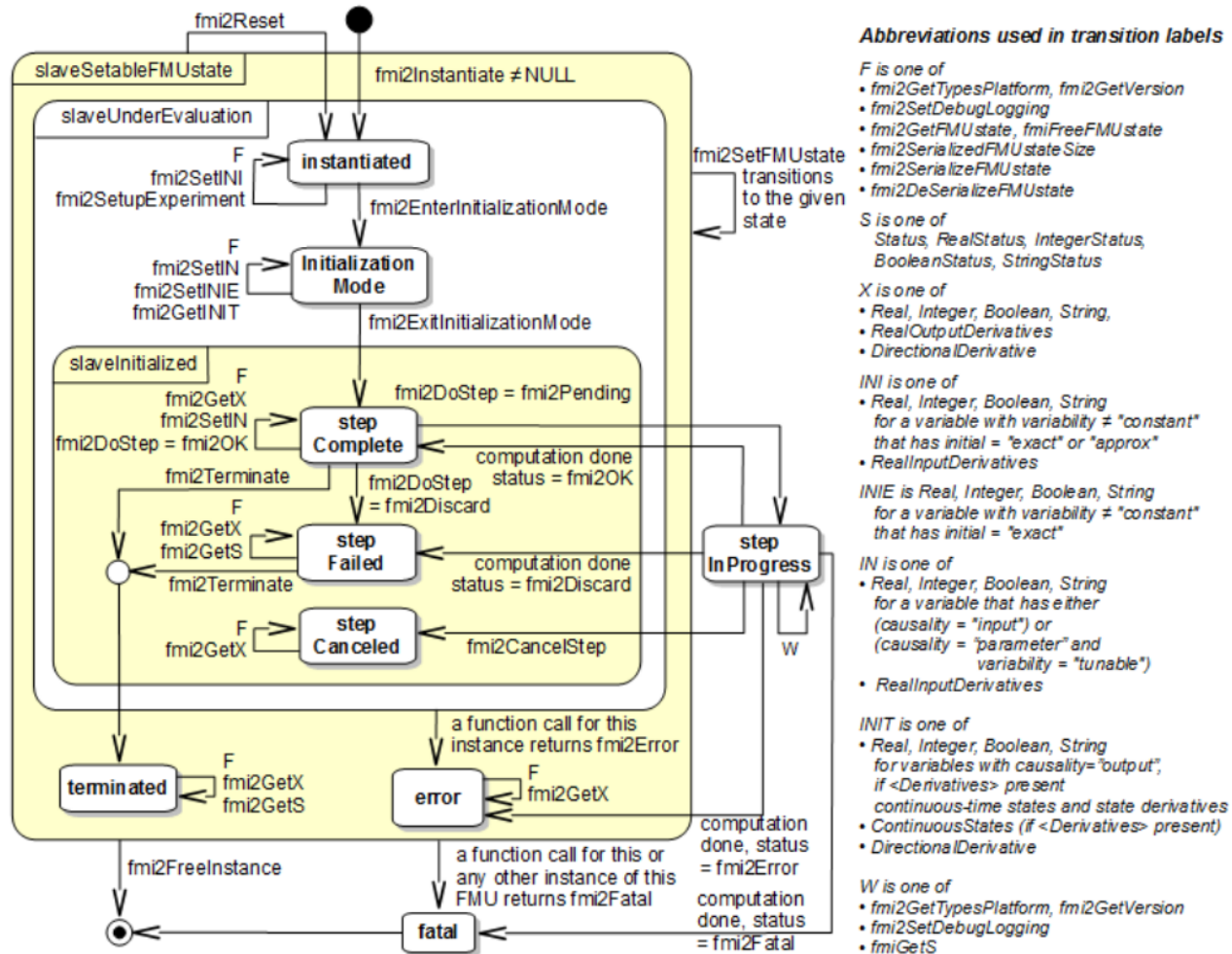
- standardized access to model equations
- models described by differential, algebraic and discrete equations
- time-events, state-events and step-events
- solved with integrators provided by embedding environment.

Functional Mock-up Unit (FMU)

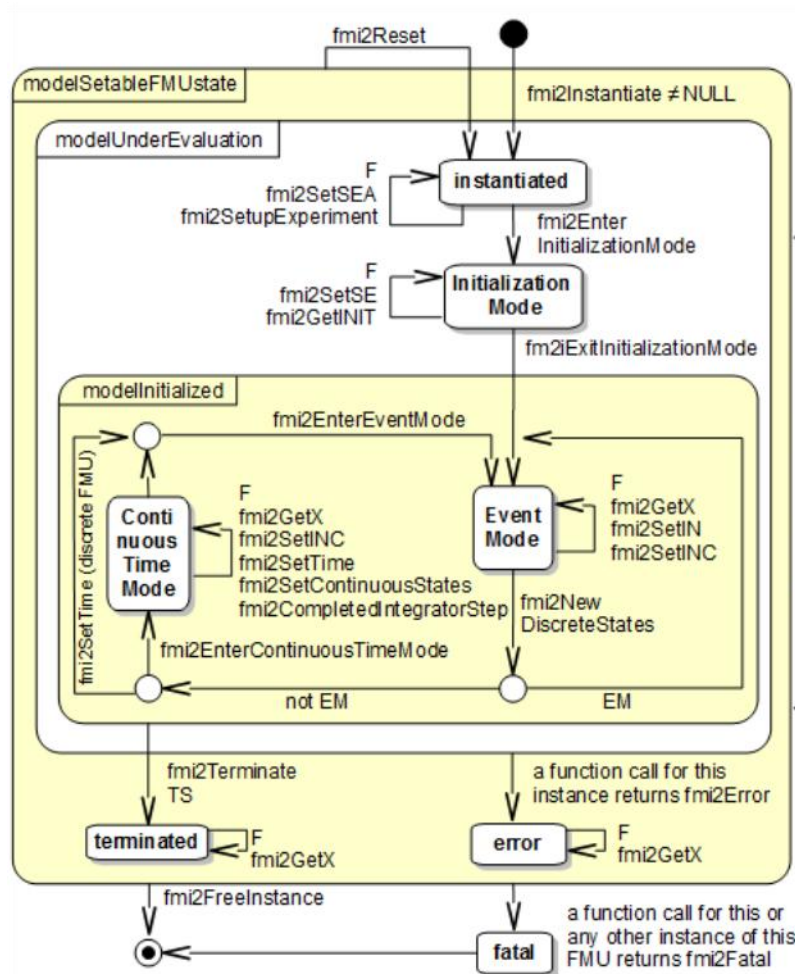
- FMU \equiv **simulation component** compliant with FMI specification
- **ZIP file** that contains:
 - *shared library* and/or source code
 - XML-based *model description*
 - optional other resources (icon, etc.)
- shared library (or source code) implements **FMI API**
- all static information related to an FMU is stored in an XML text file according to the **FMI Description Schema**



API for FMI for Co-Simulation



API for FMI for Model Exchange



Abbreviations used in transition labels

F is one of

- fmi2GetTypesPlatform
- fmi2GetVersion
- fmi2SetDebugLogging
- fmi2GetFMUstate
- fmi2Free FMU state
- fmi2SerializedFMU state Size
- fmi2SerializeFMU state
- fmi2DeSerializeFMUstate
- fmi2GetNominalsOfContinuousStates

X is one of

- Real, Integer, Boolean, String
- Derivatives
- ContinuousStates
- EventIndicators
- DirectionalDerivative

SEA is one of Real, Integer, Boolean, String for a variable with variability ≠ "constant" that has initial = "exact" or "approx"

SE is one of Real, Integer, Boolean, String for a variable with variability ≠ "constant" that has initial = "exact" or causality="input"

INC is Real for a variable with causality="input" and variability="continuous"

IN is one of Real, Integer, Boolean, String for a variable that has either (variability="discrete" and causality="input") or variability="tunable"

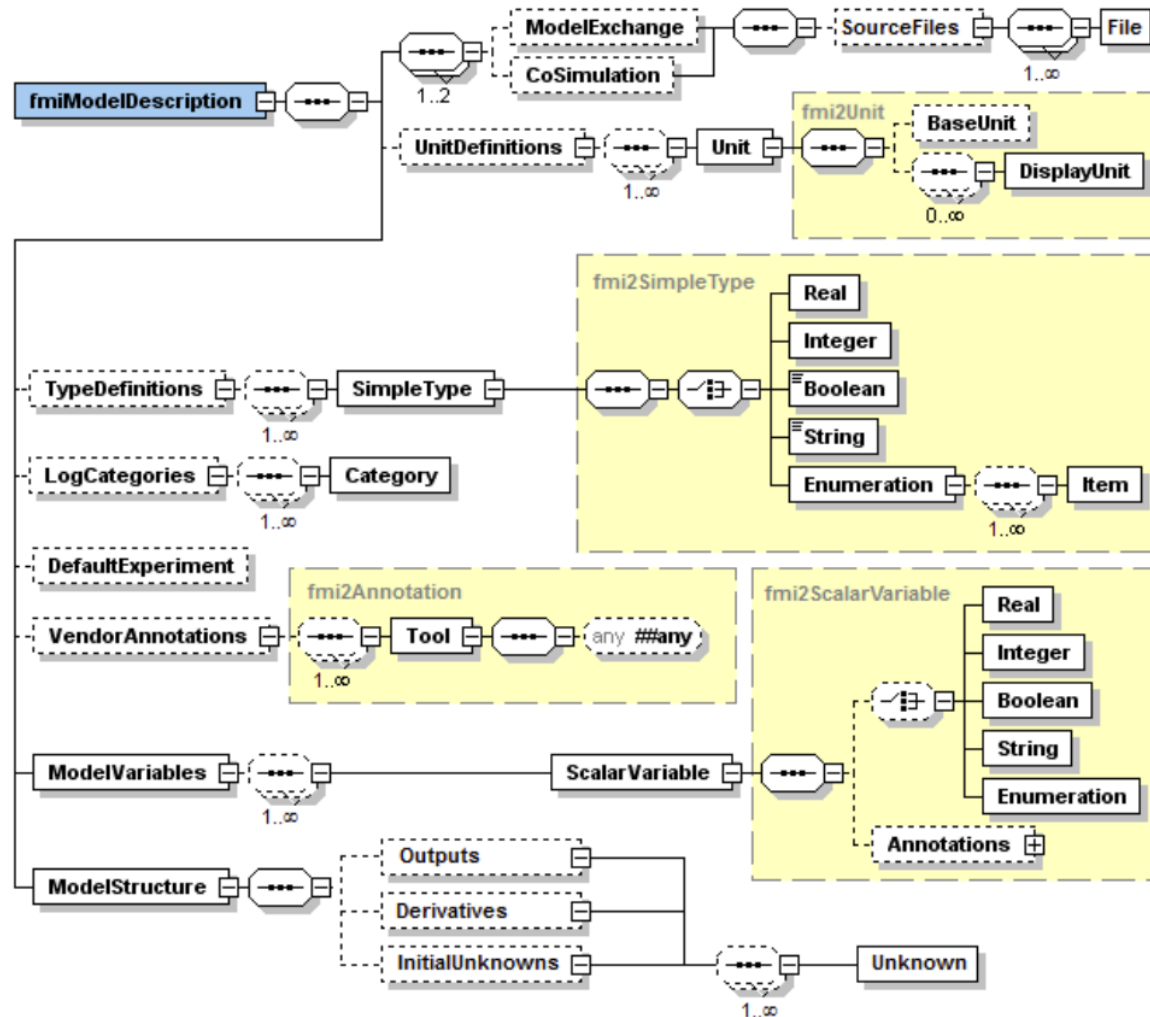
INIT is one of

- Real, Integer, Boolean, String for variables with causality="output", continuous-time states and state derivatives
- ContinuousStates
- DirectionalDerivative

TS: terminateSimulation is returned by at least one FMU or the simulation run ended successfully

EM: newDiscreteStatesNeeded is returned by at least one FMU and no FMU returns terminateSimulation

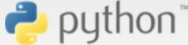
FMI Description Schema



The FMI++ Python Interface (for Windows)

- An easy-to-use Python interface for handling FMUs
 - based on the FMI++ Library
 - uses SWIG, SUNDIALS and BOOST
 - open-source
 - available at <https://pypi.python.org/pypi/fmipp>
- Features:
 - high-level *access* to FMUs
 - model initialization, get/set variable by name, etc.
 - high-level simulation *functionalities*
 - integrators, event handling, rollback mechanism, look-ahead predictions, etc.

The FMI++ Python Interface for Windows



» Package Index > fmipp > 1.1

PACKAGE INDEX »

- Browse packages
- Package submission
- List trove classifiers
- List packages
- RSS (latest 40 updates)
- RSS (newest 40 packages)
- Python 3 Packages
- PyPI Tutorial
- PyPI Security
- PyPI Support
- PyPI Bug Reports
- PyPI Discussion
- PyPI Developer Info

ABOUT »
NEWS »
DOCUMENTATION »
DOWNLOAD »
COMMUNITY »
FOUNDATION »
CORE DEVELOPMENT »

fmipp 1.1

FMI++ Python Interface for Windows

[Package Documentation](#)

[About](#)

The **Functional Mock-up Interface** (FMI) specification intentionally provides only the most essential and fundamental functionalities in the form of a C interface. On the one hand, this increases flexibility in use and portability to virtually any platform. On the other hand, such a low-level approach implies several prerequisites a simulation tool has to fulfil in order to be able to utilize such an FMI component.

The **FMI++ Python Interface** is a Python wrapper for the **FMI++ Library**, which intends to bridge the gap between the basic functionality provided by the FMI specification and the typical requirements of simulation tools. The FMI++ Library provides high-level functionalities that ease the handling and manipulation of FMUs, such as numerical integration, advanced event-handling or state predictions. This allows FMUs to be integrated more easily, e.g., into fixed time step or discrete event simulations.

This package provides a stand-alone version of the Python interface for the **FMI++ Library** for Windows. For other operating systems, this package can be built from [source](#).

Dependencies

In order to provide a reliable, stable and portable solution, the FMI++ Library relies on other state-of-the-art tools where necessary. Especially, it depends upon

- the **Boost library** (especially the **ODEINT library**) and
- the **SUNDIALS** numerical integrator package.

Details on the licenses of the FMI++ Library, Boost and SUNDIALS can be retrieved via:



```
import fmipp
fmipp.licenseInfo()
```

Documentation

The FMI++ Python Interface provides several classes that allow to manipulate FMUs for ModelExchange and for Co-Simulation. An overview on how to use it can be found [here](#). More extensive background information can be found in the documentation of the **FMI++ Library**.

| File | Type | Py Version | Uploaded on | Size |
|--------------------------------|----------------------|------------|-------------|-------|
| fmipp-1.1.win32.exe (md5) | MS Windows installer | 2.7 | 2016-02-08 | 614KB |
| fmipp-1.1.win32.py35.exe (md5) | MS Windows installer | 3.5 | 2016-02-19 | 545KB |

Not Logged In

- [Login](#)
- [Register](#)
- [Lost Login?](#)
- Use [OpenID](#) 
- [Login with Google](#) 

Status

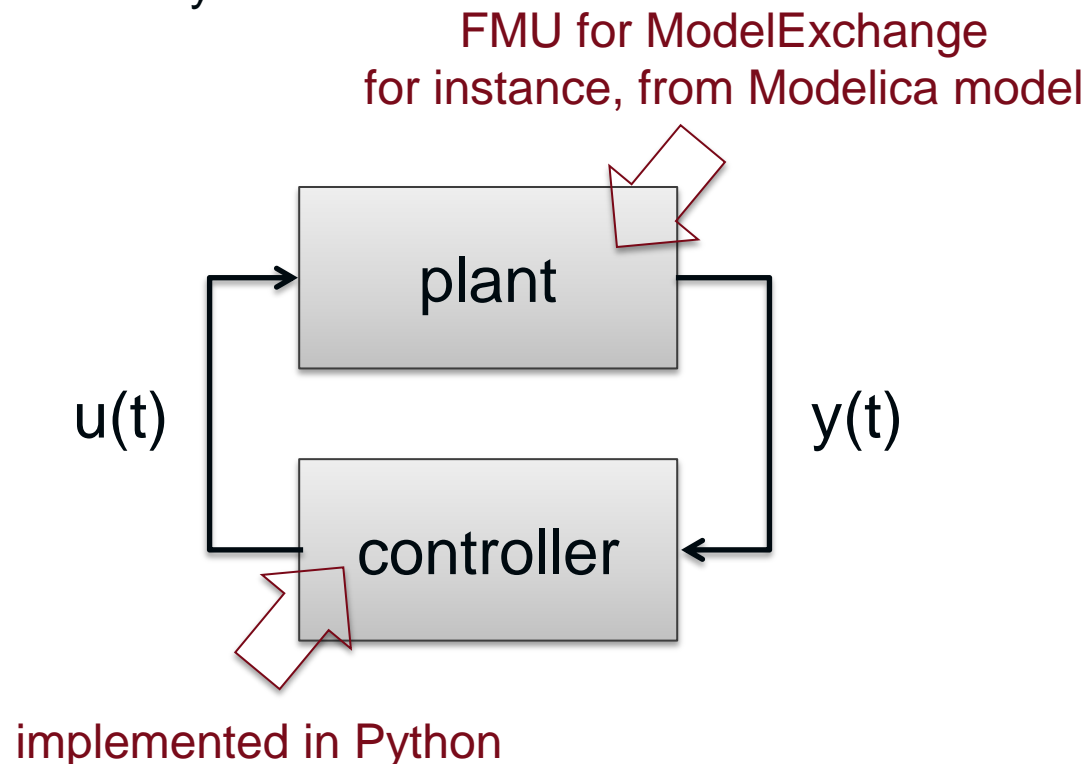
- [Nothing to report](#)

Downloads ↓

installers for Python 2.7
and Python 3.5

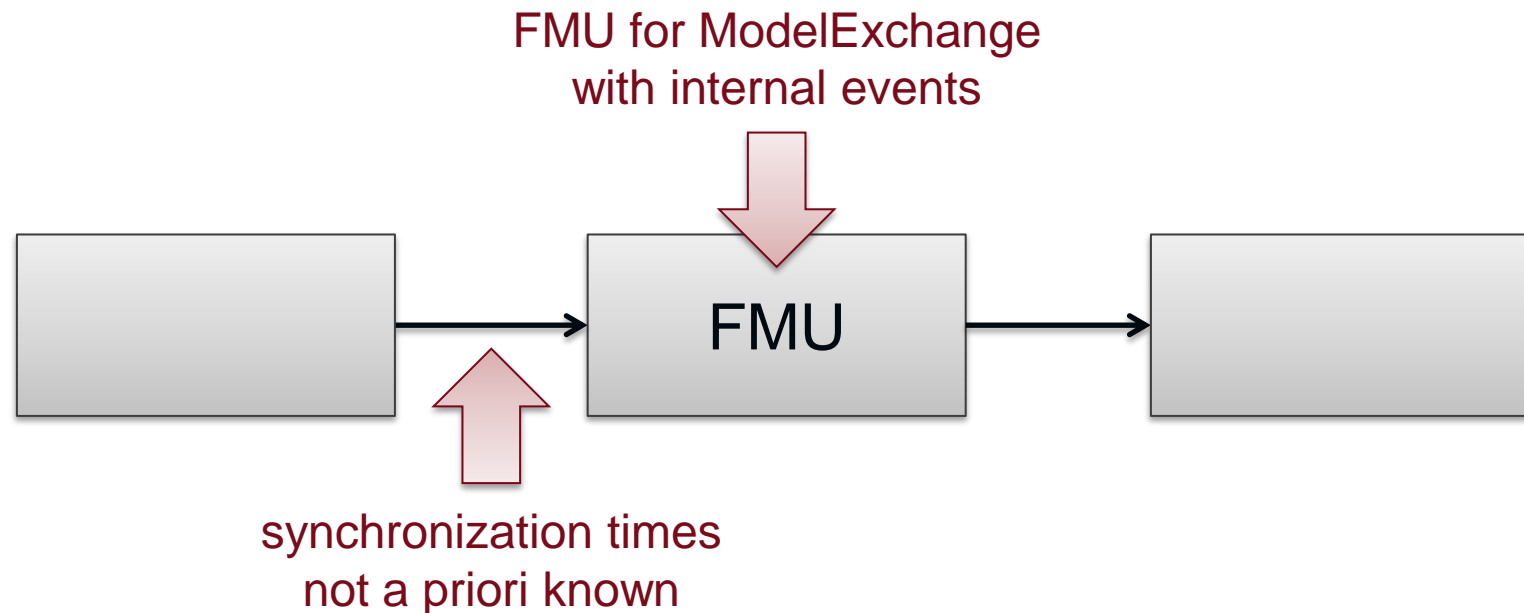
Example: Rapid prototyping of controls

- provide plant model as FMU for Model Exchange
- develop controller in Python



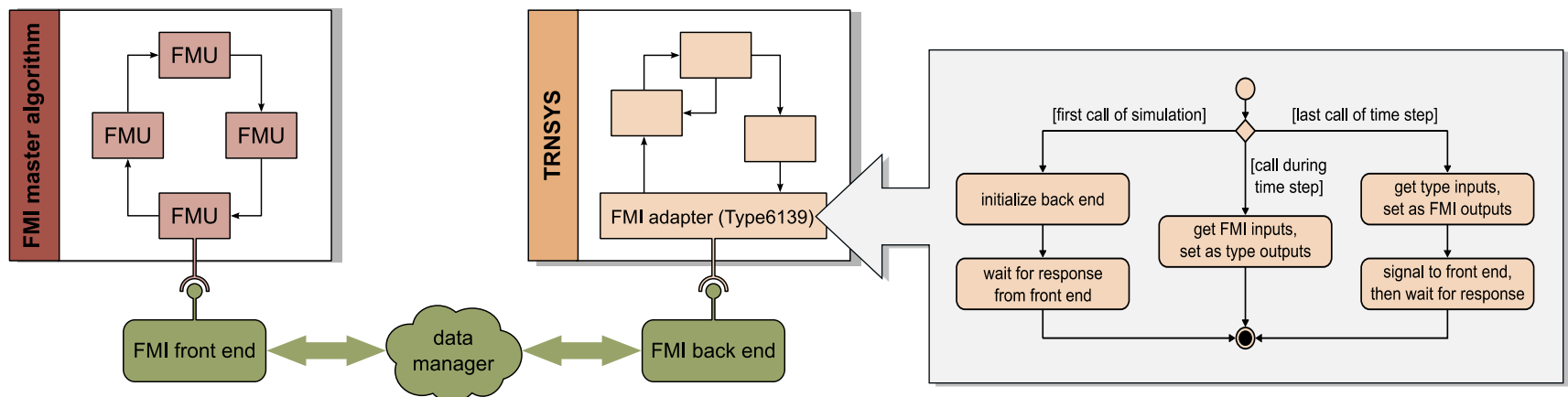
Example: Include FMU in co-simulation setup

- use an FMU in the context of a discrete event-driven co-simulation setup



The FMI++ export utility

- FMI++ library provides **generic tools** for **FMI-based tool coupling**
- introduces concept of *front end*, *back end* and *adapter*.
 - **front end**: implements *fmiComponent*, to be used by the master algorithm
 - **back end**: generic gateway between simulation tool and front end
 - **adapter**: component in simulation tool that utilizes the back end



Links

- Official FMI homepage: <https://www.fmi-standard.org/>
- FMI++ Library: <http://fmipp.sourceforge.net>
- FMI++ Python Interface (for Windows): <https://pypi.python.org/pypi/fmipp>
- FMI++ MATLAB Toolbox (for Windows): <http://matlab-fmu.sourceforge.net>
- FMI++ PowerFactory Export Utility: <http://powerfactory-fmu.sourceforge.net>
- FMI++ TRNSYS Export Utility: <http://trnsys-fmu.sourceforge.net>