Co-Simulation based Assessment Methods

9th April 2019

Presenters

- Rishabh Bhandia, Arjen van der Meer, Peter Palensky - TU Delft, Netherlands
- Edmund Widl, Thomas Strasser - AIT Austrian Institute of Technology, Austria
- Nabil Akroud - Ormazabal, Spain
- Kai Heussen, Tue Vissing Jensen - DTU, Denmark
- Van Hoa Nguyen - CEA, France

Supported by

- IEEE IES Technical Committee on Smart Grids (TC-SG)
- IEEE SMC Technical Committee Cybernetics for Intelligent Industrial Systems (TC-IIS)

EU H2020 Programme GA No. 654113

Agenda

- Co-Simulation fundamentals
- Introduction and Motivation
- Co-simulation assessment for continuous-time RMS studies (TC-1)
- Combined Hardware and Software Simulation (TC-2)
- Signal-based Synchronization between Simulators (TC-3)
- Discussion

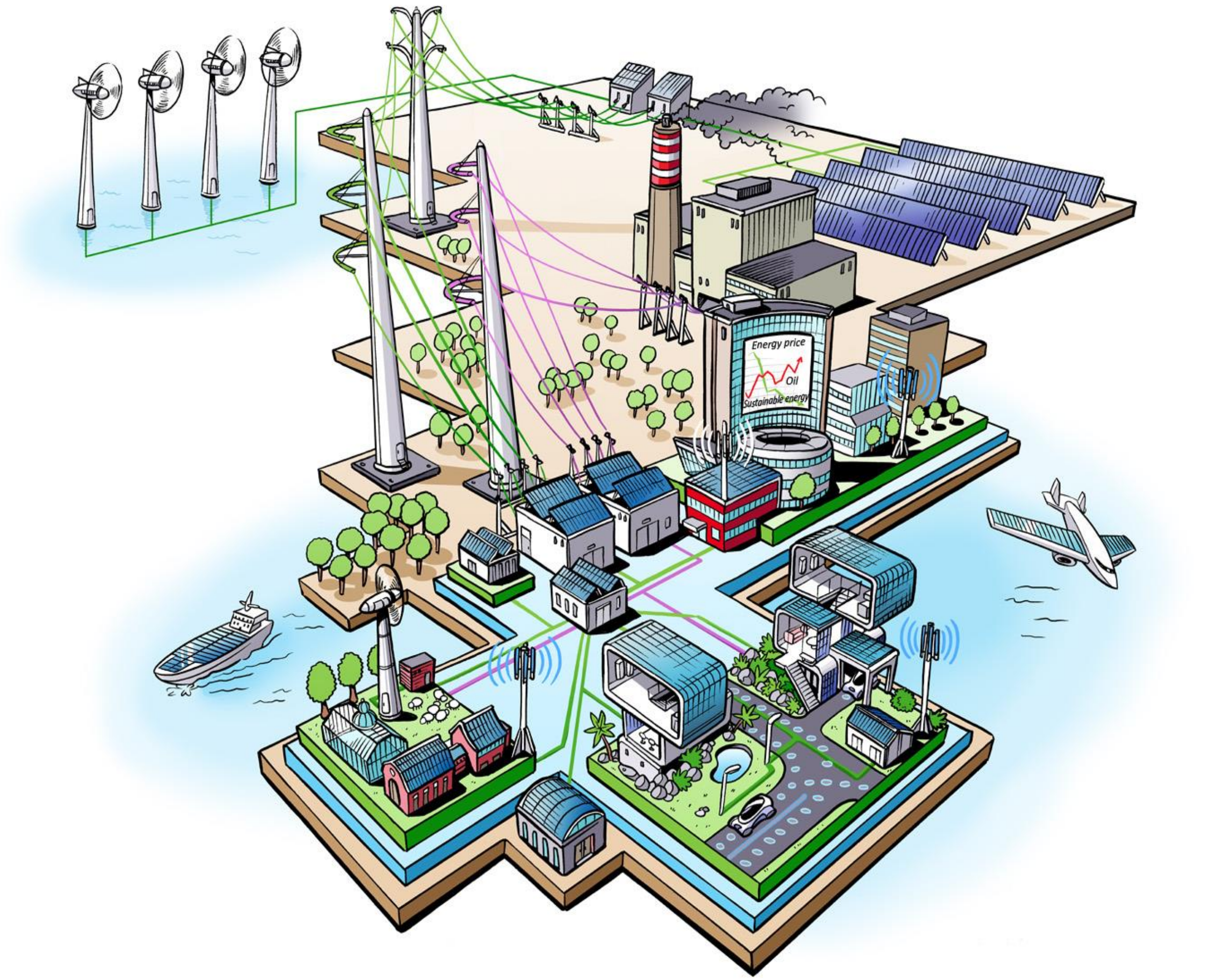
EU H2020 Programme GA No. 654113

Co-Simulation fundamentals

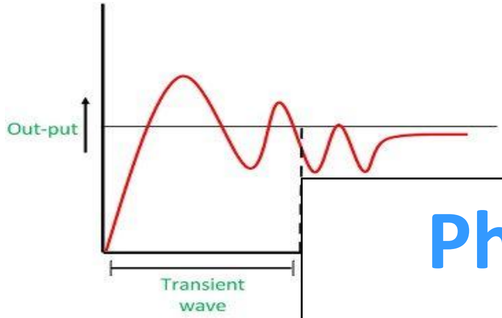
Peter Palensky, TU Delft

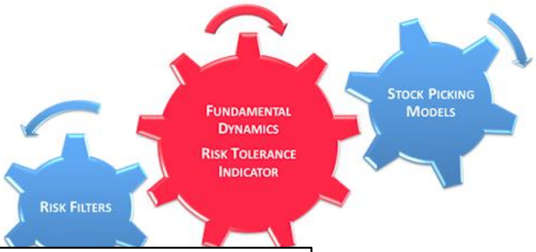
Diversity of the Energy Transition

New applications,
connections,
Dependencies,
markets, mechanisms,
technologies,
constraints,…



EU H2020 Programme GA No. 654113





Physics

continuous process

energy generation, transport, distribution, consumption, etc.

Roles

behavioral process

agents, game theory, market players, etc.

cyber-physical energy system

+ Speed!
+ Size!
+ ...

Information Technology

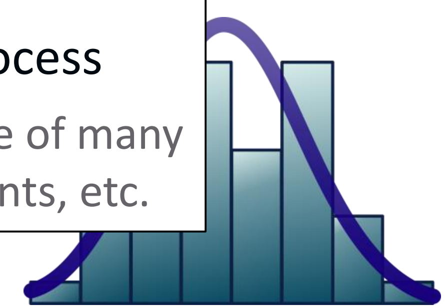
discrete process

controllers, communication infrastructure, software, etc.

Stochastics

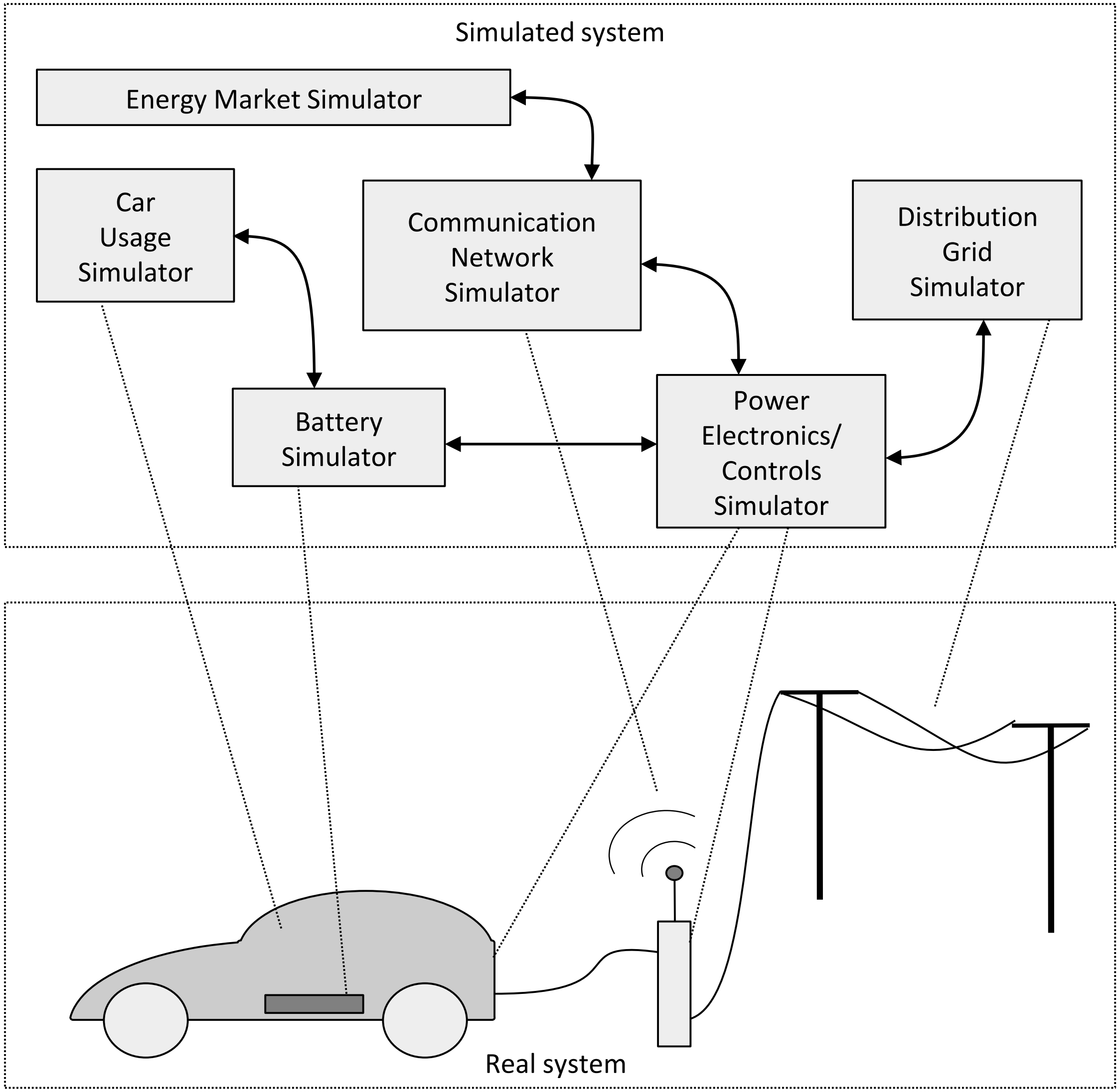
statistical process

weather, aggregate of many individual elements, etc.



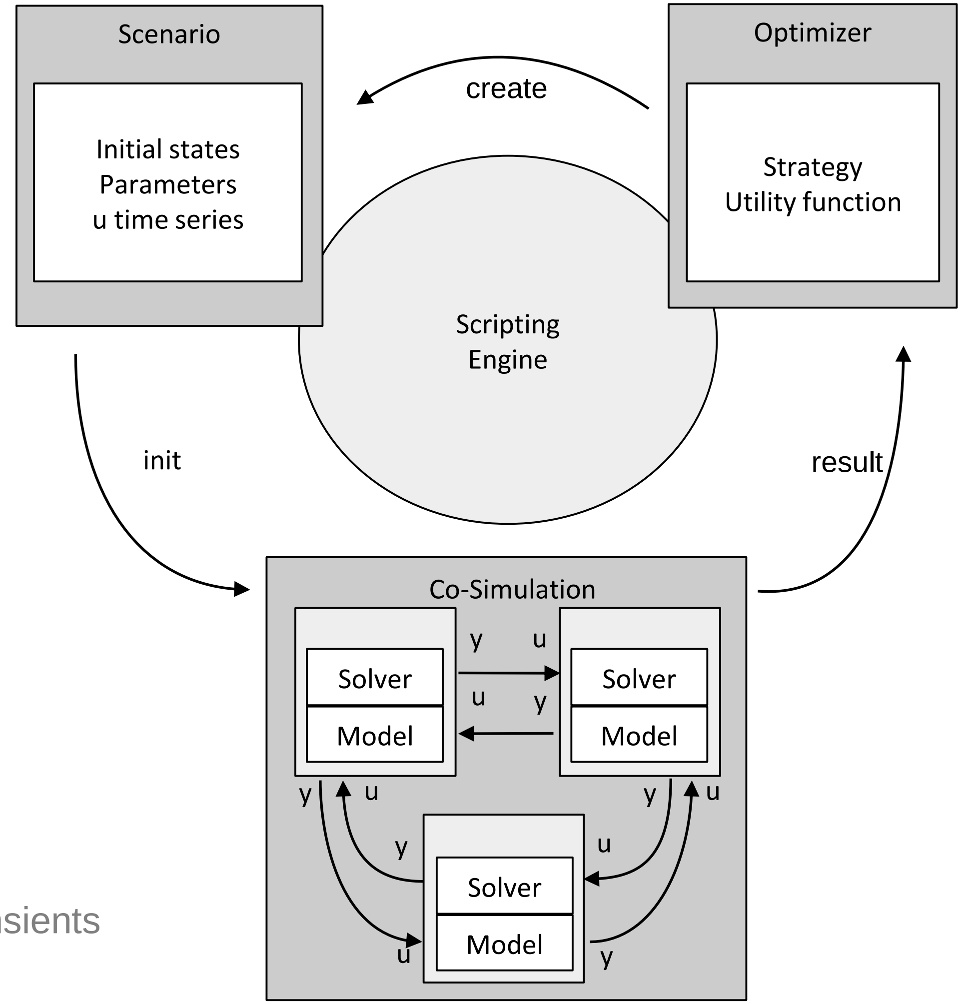
Coupling Simulators…

… for a connected world



Promising but…

- Multiple simulators/models
- Formats, projects?
- How to link?
- Scenario Handling?
- Interfaces?
- Time stepping?
  - EMT vs. TS



EMT: Electro-Magnetic Transients
TS: Transient Stability

Introduction and Motivation

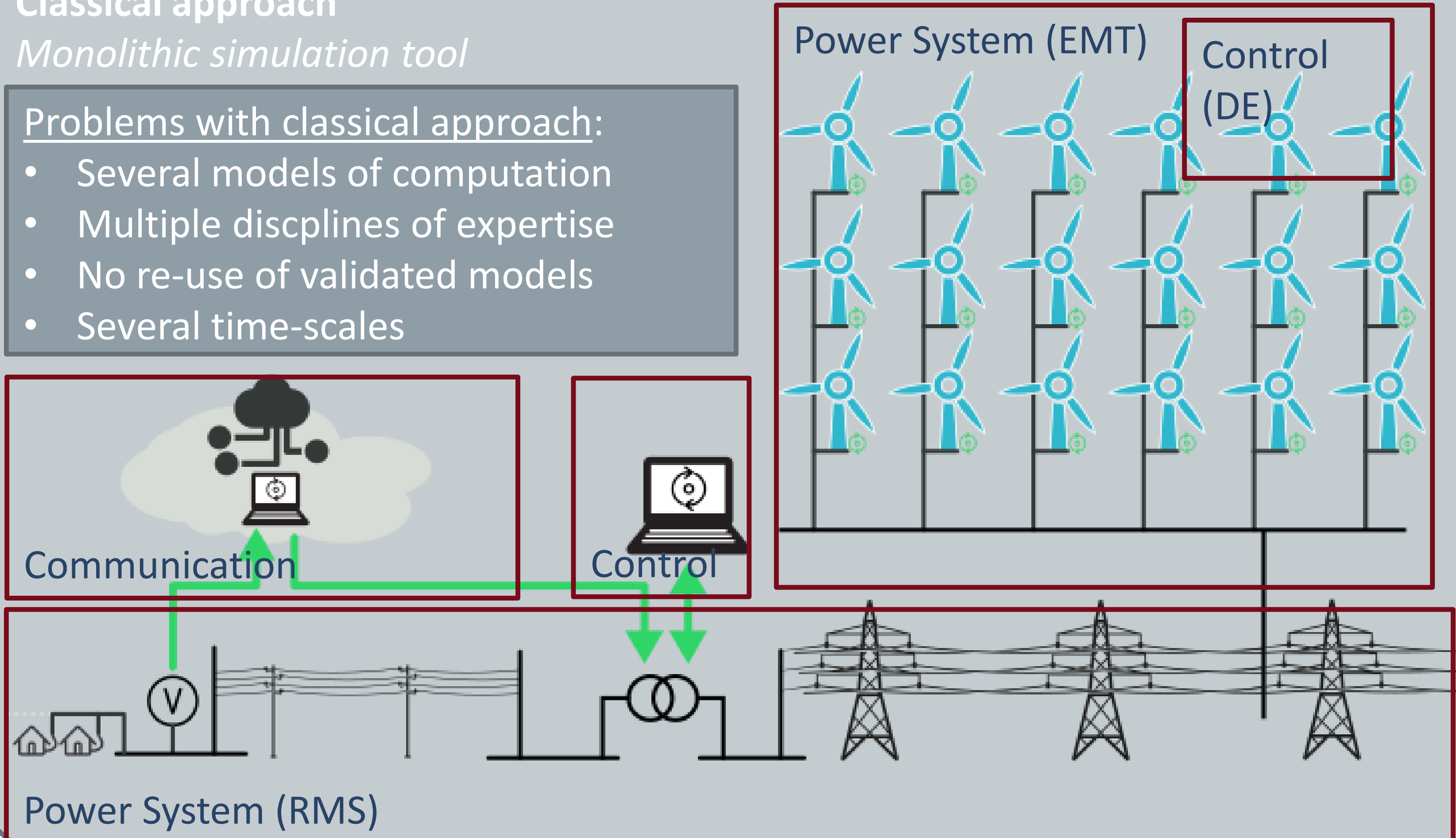
Kai Heussen (DTU)
Van Hoa Nguyen (CEA)

Overview of challenges

Classical approach

Monolithic simulation tool

Problems with classical approach:
- Several models of computation
- Multiple discplines of expertise
- No re-use of validated models
- Several time-scales



Kai Heussen (DTU)

CONCEPT & CONTRIBUTIONS

Vision of ERIGrid JRA2

- **Objective:** Development of co-simulation framework for smart grid assessment.

- **Contributions:**
  - New simulator interfacing:
    - New converters (FMU) for: Matlab, PowerFactory, and ns-3
    - hardware integration in a co-simulation framework
    - Improvements of co-simulation orchestration (mosaik).
  - Synchronization strategies for correction simulation coupling
    - time-shifting relaxation of cyclic dependencies without roll-back
    - Synchronization by state-prediction
    - Message-handling in continuous-time co-simulation using FMI specs
  - Contribution to co-simulation workflows:
    - Development of an easier-to-deploy co-simulation tool chain.
    - Approach to up-scaling of co-simulation scenarios

Today: Benefits and Demonstration

To roll out co-simulation, industry and research are expected to **benefit** from:

- Integration *of black-box components* into validated model environments (as development step prior to hardware deployment)
- *Hardware coupling* against co-simulation models
- Detailed simulation to *realistic depoyment scales*
- Co-simulation *standards* to accommodate industry needs (ICT, automation)

Today we **demonstrate**:

- re-use of *component* and *grid* models, as well as *hardware* across scenarios
- New uses of open source applications of FMI standard:
  automation (4DIAC, IEC61499) & communication simulation (ns-3)
- *Practical scaling-up* of simulation scenarios

“Scaling up” – why?

- A key challenge with real-world smart grid are **many active components**.
  - Co-simulation allows evaluating components in *complex* system context
- Proprietary industry models and controllers only available as “**Blackbox**”
  - *Re-use is better than abstraction*. For validation it is necessary.
- **Large-scale phenomena**: how to assess “real-scale” scenarios?
  - Different phenomena of interest, just *“make it big” is not good enough*. However, strategies for large-scale assessment still under development.
  - Both a question of **tools** & **methodology**.
  - **Pathology classes** defined: the choice of scale and observable will largely influence how the phenomenon shows itself. → *Deliverable D-JRA2.3*

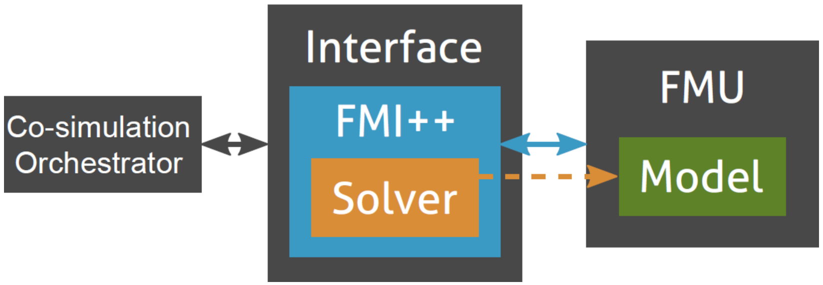
Van Hoa Nguyen (CEA)

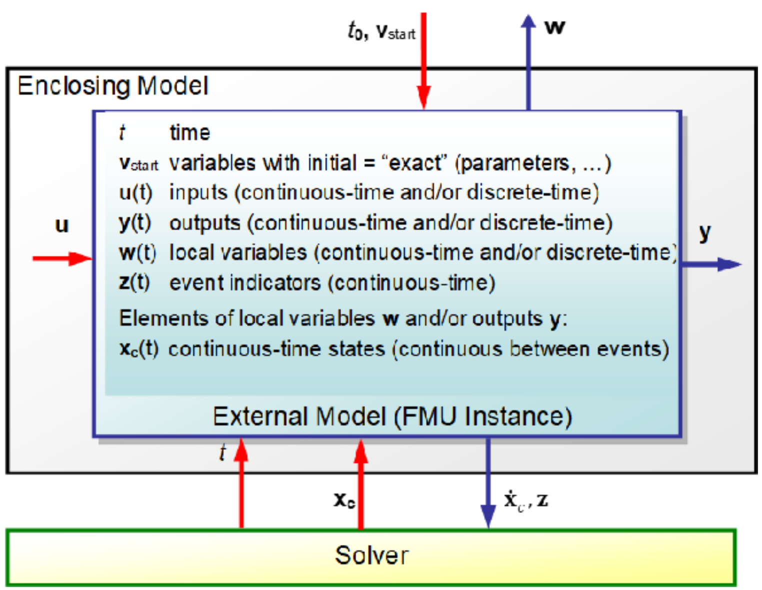
IMPLEMENTATION OVERVIEW

Functional Mock-up Interface (FMI)

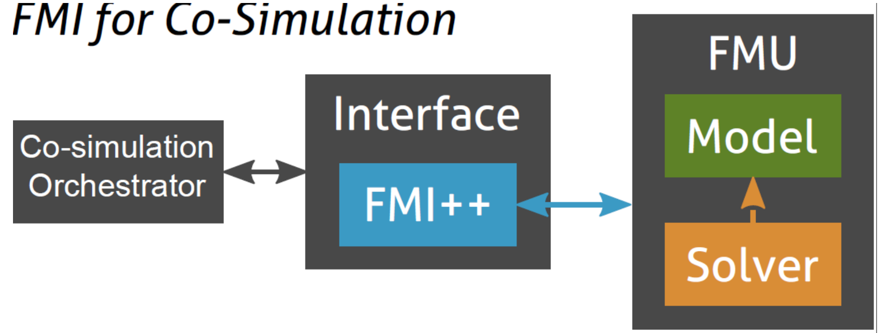
https://fmi-standard.org

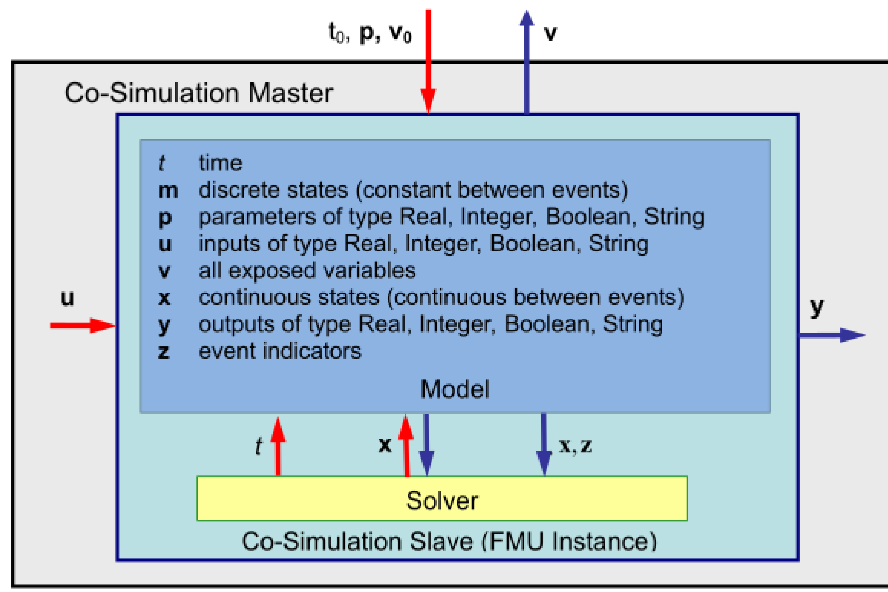
FMI for Model Exchange





FMI for Co-Simulation

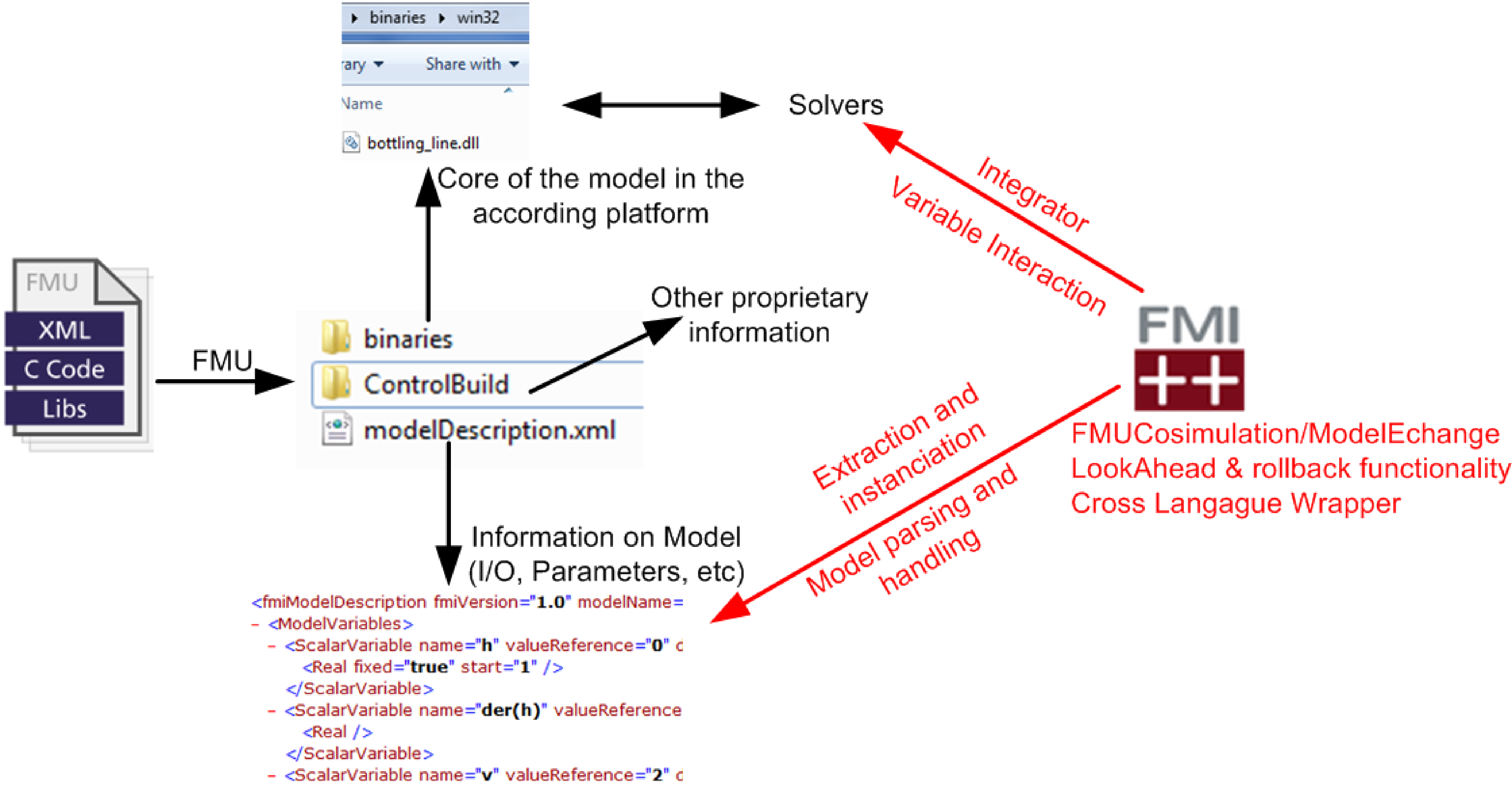




FMI++ Toolbox

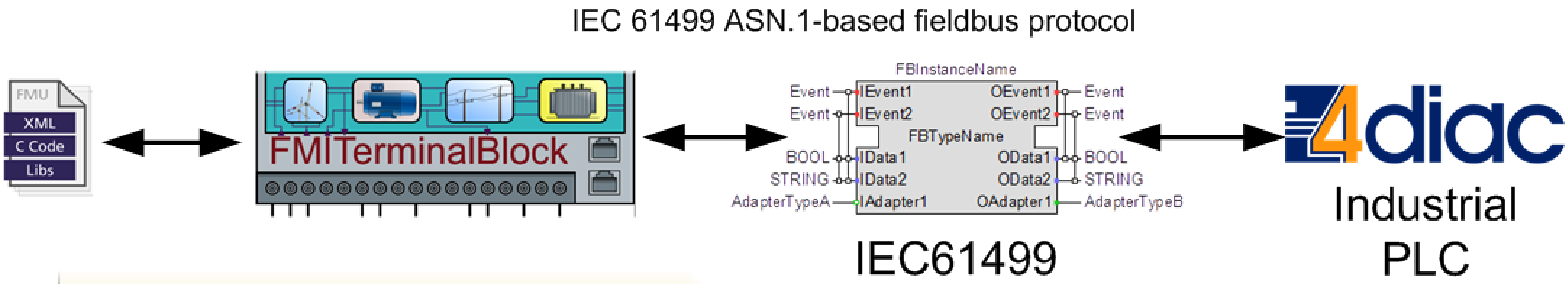
FMI++ Library: https://sourceforge.net/projects/fmipp/

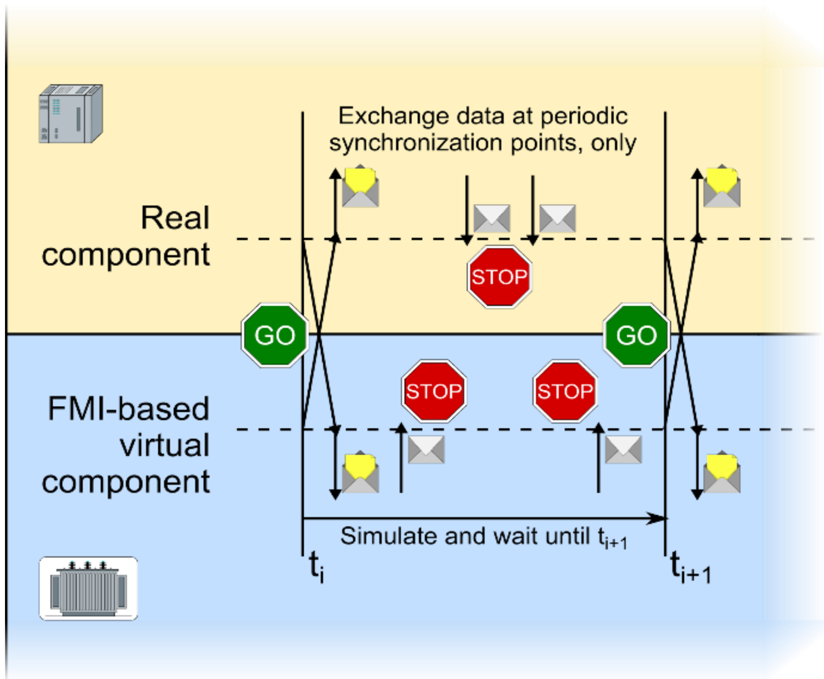
FMI++ Python Wrapper: https://pythonhosted.org/fmipp/

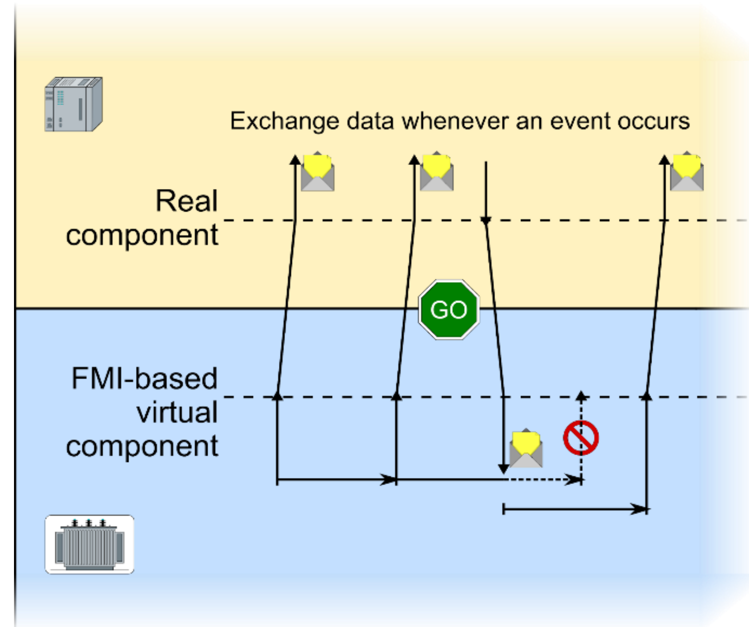


FMITerminalBlock

Available at : https://github.com/AIT-IES/FMITerminalBlock



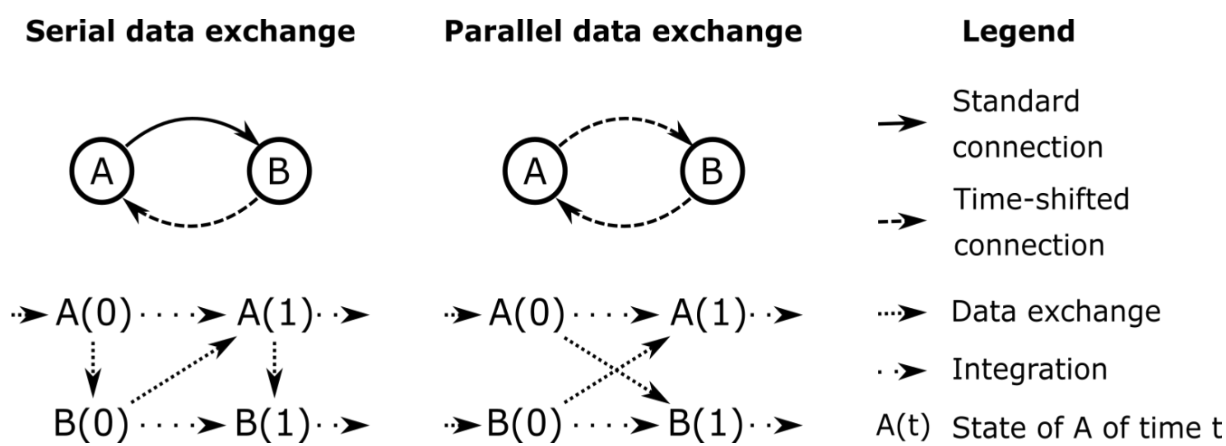




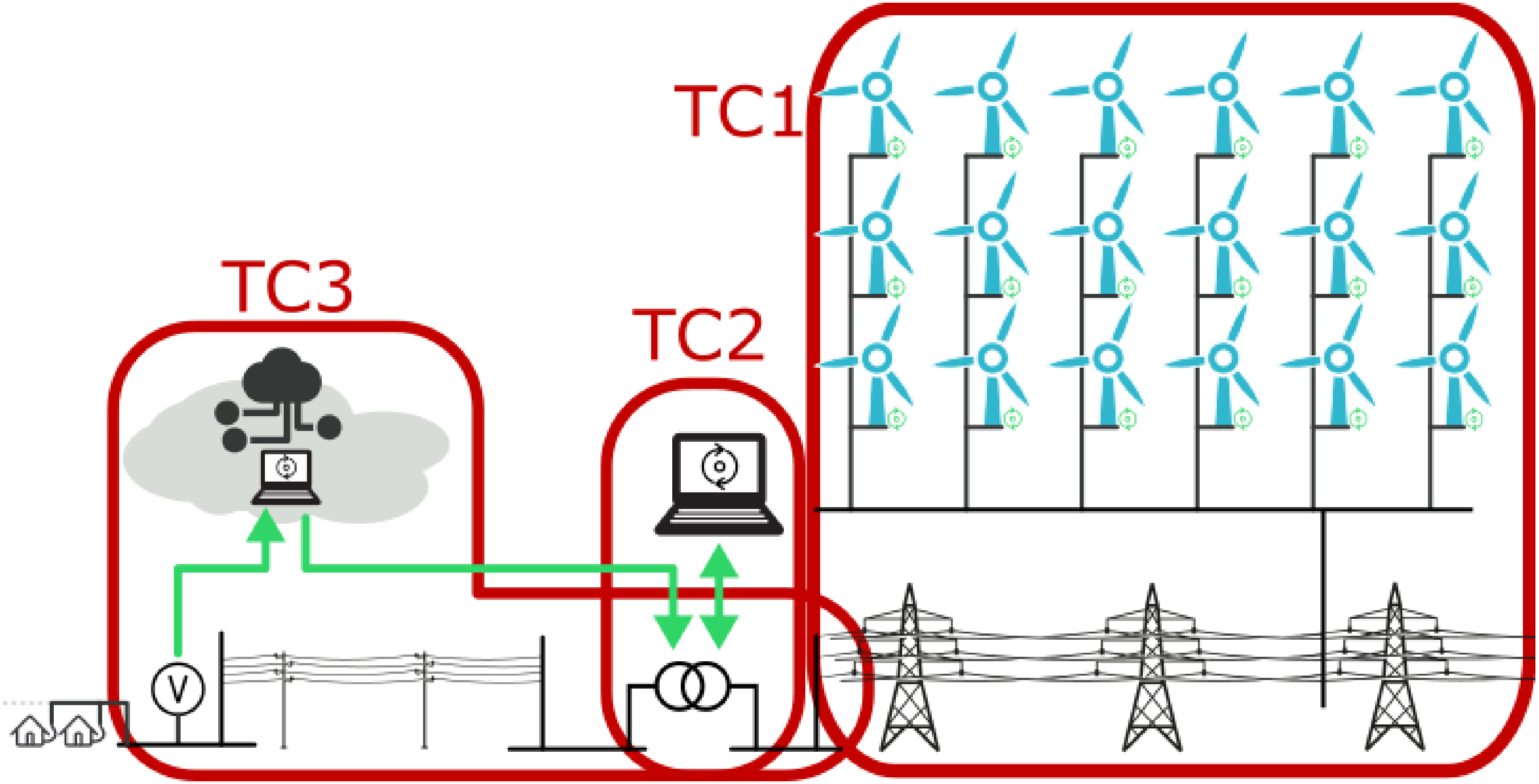
Improvement of Mosaik

Available at : https://mosaik.offis.de

Improvement of Mosaik's capacity to handle cyclic dependency in ERIGrid → Introduction of « Time-shifted connection ».



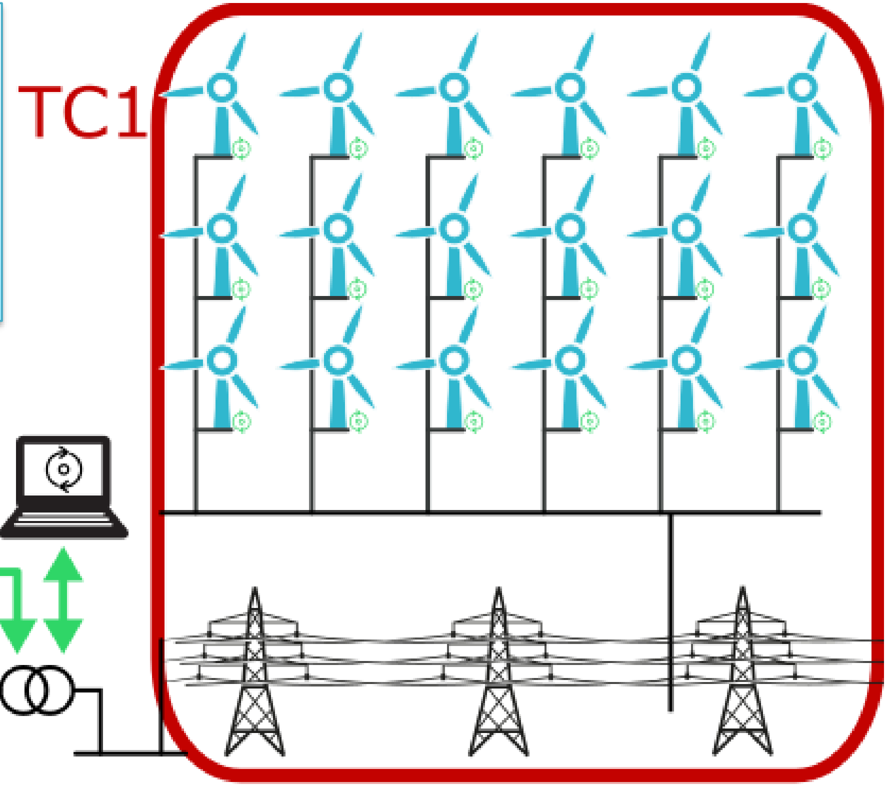
Selected Test-cases

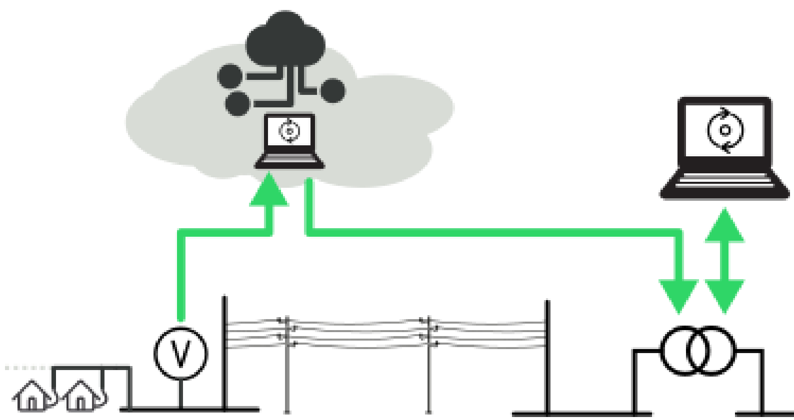


Challenges Addressed: ”TC1”

Grid+Wind farm & LV-FRT

- *Algebraic coupling between grid simulators / cyclic dependency*
- *Re-use of validated models*
- *Scaling-up*



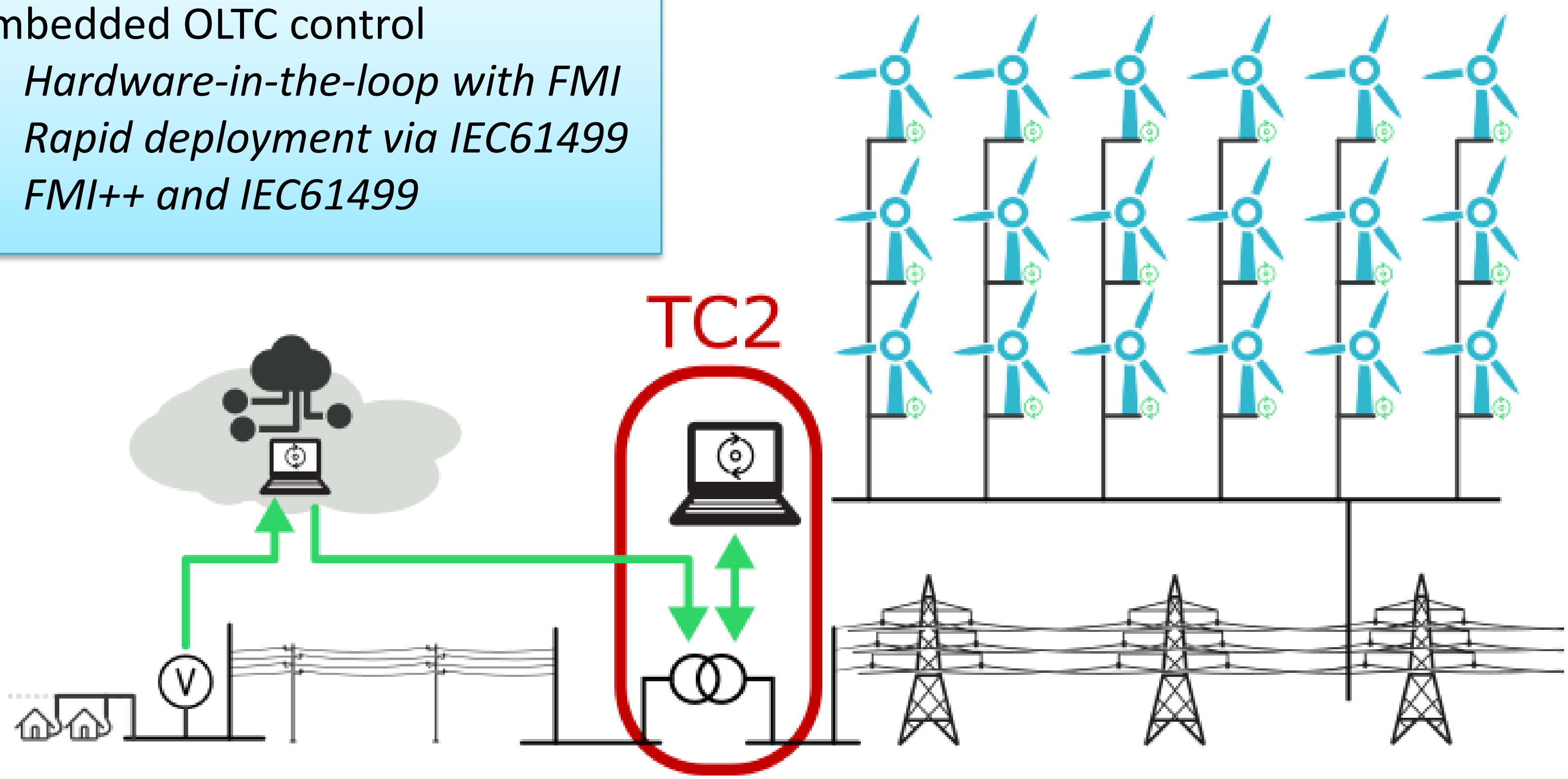


LV: Low Voltage
FRT: Fault-Ride-Through

Challenges Addressed: ”TC2”

Embedded OLTC control

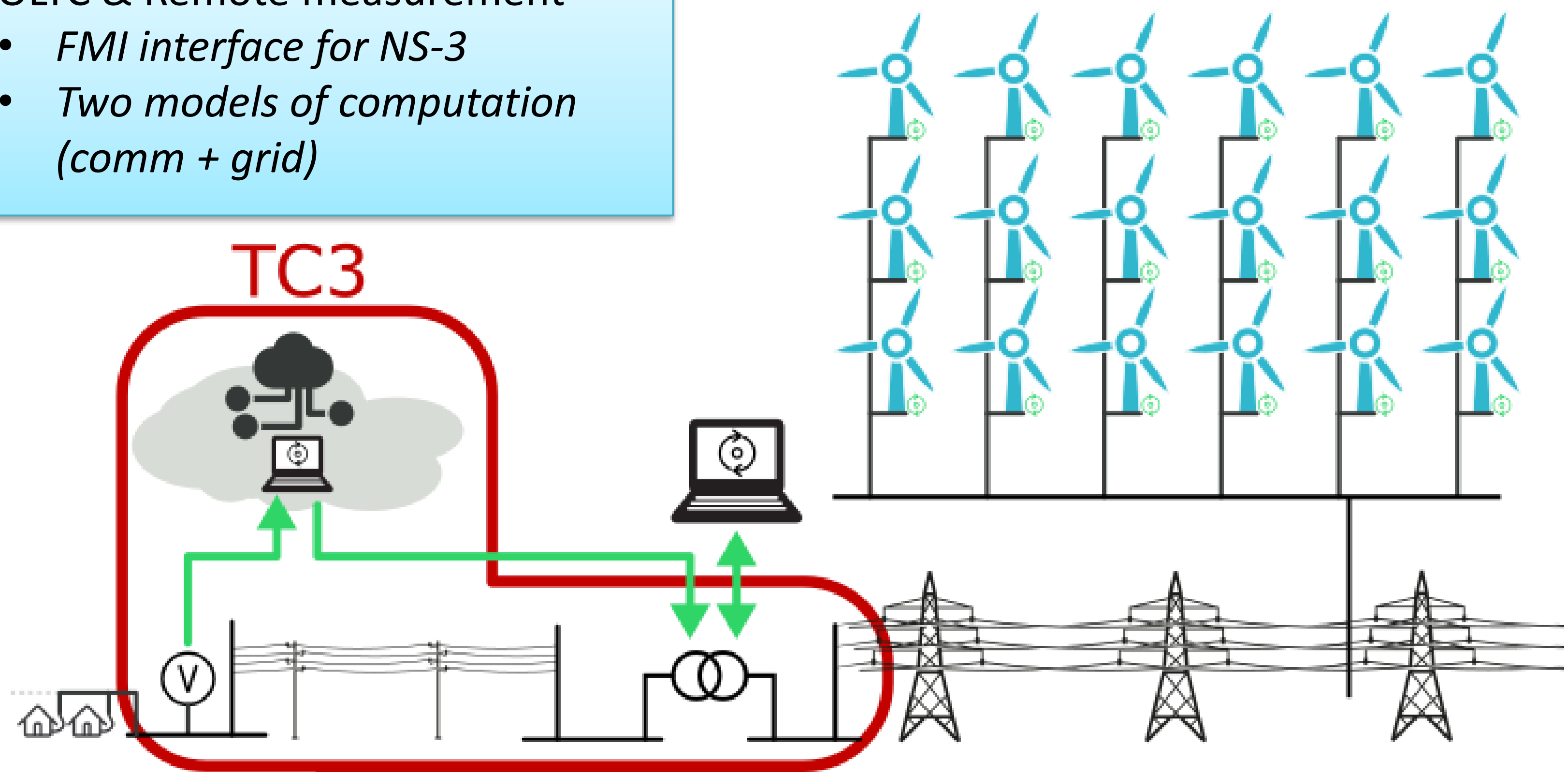
- *Hardware-in-the-loop with FMI*
- *Rapid deployment via IEC61499*
- *FMI++ and IEC61499*



Challenges Addressed: ”TC3”

OLTC & Remote measurement

- *FMI interface for NS-3*
- *Two models of computation (comm + grid)*



OLTC: On-Load-Tap-Changer

Closer look at the test-cases

Resources

Downloadable deliverables/reports, results, publications, press information and newsletters from the ERIGrid consortium:

Deliverables | Publications | Open Access Tools | Transnational Access | Press Area | Newsletters

Research Infrastructure (RI) database schema	SchemaSpy description of the RI database defined in Deliverable D-NA5.2 Partner profiles. The file includes information on the table structure, data formats, primary keys etc. and can be used to view the details of the DB.
JaNDER Level 1 access	Small interfacing layer between the openIEC61850 library and Redis database as defined in Deliverable D-JRA4.1
Local JaNDER database to claud replication	Small command line program to replicate remotely the commands sent to a local Redis instance as defined in Deliverable D-JRA4.1
JRA2-TC1	JRA2 Test Case TC1 mosaik implementation according to ERIGrid Deliverable D-JRA2.3
JRA2-TC2	JRA2 Test Case TC2 implementation according to ERIGrid Deliverable D-JRA2.3
JRA2-TC3	JRA2 Test Case TC3 mosaik implementation according to ERIGrid Deliverable D-JRA2.3
JRA2-LSS2	JRA2 Test Case LSS2 mosaik implementation according to ERIGrid Deliverable D-JRA2.3
ns-3 FMI Export Module	Module fmi-export enables the FMI-compliant simulation coupling with ns-3 scripts, i.e., ns-3 script are launched and executed through an FMI-compliant co-simulation interface. In terms of FMI terminology, ns-3 is the slave application,

Source code of the test-cases are available on ERIGrid's Github and website.
https://github.com/ERIGrid
https://erigrid.eu/dissemination/

Deeper presentations on the test-cases presented by:

- TC1: Arjen Van der Meer, Rishabh Bhandia (TUDelft)
- TC2: Nabil Akroud (Ormazabal)
- TC3: Edmund Widl (AIT)

Further Reading https://erigrid.eu/dissemination/

- ERIGrid Deliverables

Simulator coupling and Smart Grid libraries	Deliverable 8.1 – D-JRA2.1
Smart Grid ICT assessment method	Deliverable 8.2 – D-JRA2.2
Smart Grid simulation environment	Deliverable 8.3 – D-JRA2.3

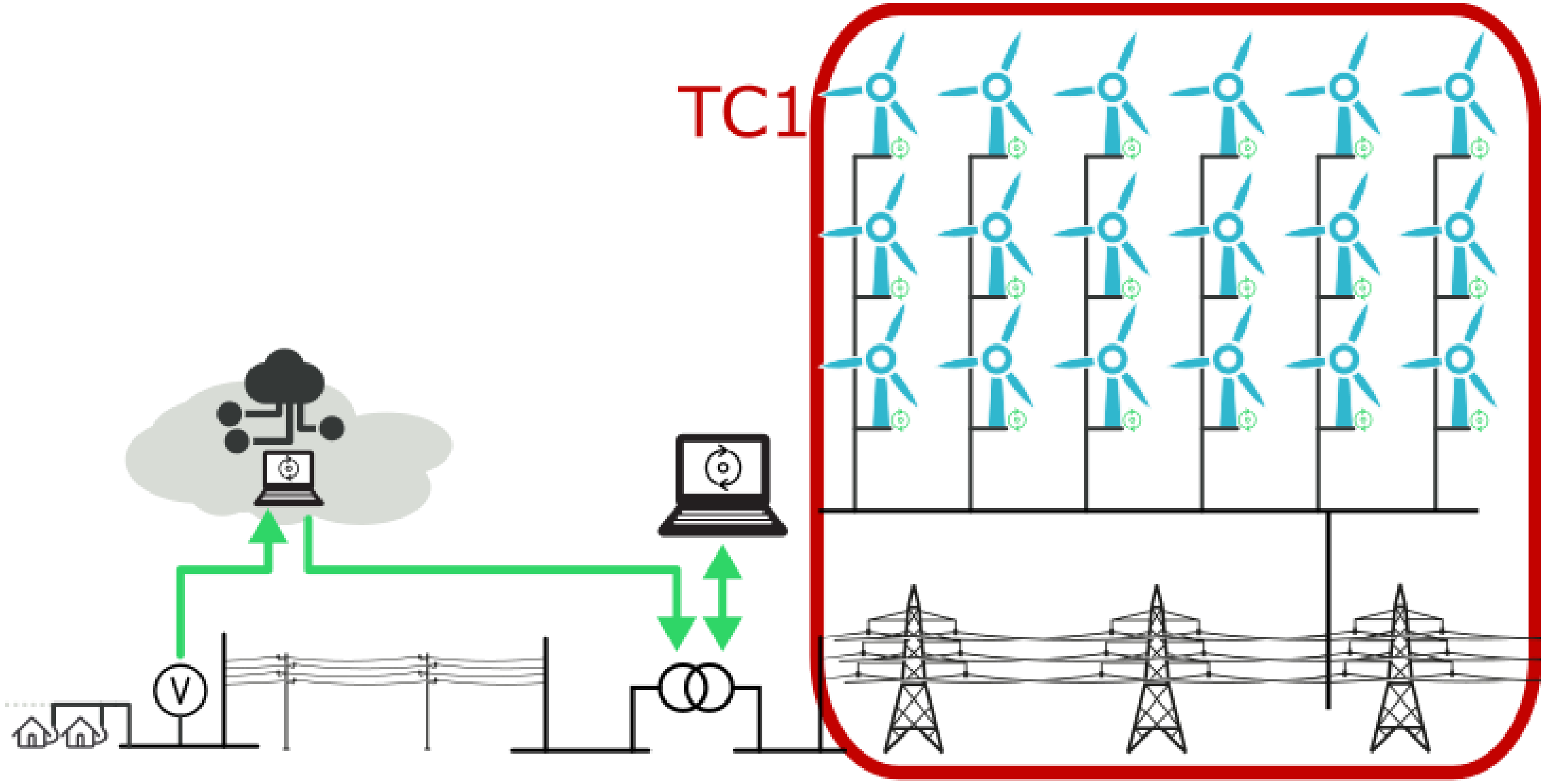
- Publications *(extract)*

A co-simulation approach using PowerFactory and Matlab/Simulink to enable validation of distributed control concepts within future power systems	K. Johnstone, S. M. Blair, M. H. Syed, A. Emhemed, G. M. Burt, T. Strasser CIRED 2017 – 24th International Conference on Electricity Distribution, Jun. 12-15, Glasgow (UK), 2017 (Golden Open Access).
Cyberphysical system modeling, test specification and co-simulation based testing	A. A. van der Meer, P. Palensky, K. Heussen, et al. 2017 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems, Apr. 21, Pittsburgh, PA (USA), 2017 (Green Open Access).
Simulation-based Validation of Smart Grids – Status Quo and Future Research Trends	C. Steinbrink, S. Lehnhoff, S. Rohjans, T. I. Strasser, et al. 8th International Conference on Industrial Applications of Holonic and Multi-Agent Systems (HoloMAS 2017), Aug. 28-30, Lyon (FR), 2017
On Conceptual Structuration and Coupling Methods of Co-Simulation Frameworks in Cyber-Physical Energy System Validation	V.H. Nguyen, Y. Besanger, Q.T. Tran, T. L. Nguyen Energies, vol. 10, no. 12:1977, 2017, doi: 10.3390/en10121977 (Golden Open Access).

Co-simulation assessment for continuous-time RMS studies (TC-1)

Arjen van der Meer (TU Delft)
Rishabh Bhandia (TU Delft)

Test case 1



Workflow

Specification

- define co-simulation design criteria
- specify system configuration and experiments
- assign focal tools
- specify dynamic model properties

Development

- build and test models in sub-systems
- build functional mockup interface wrappers, exporters, and API scripts

Results

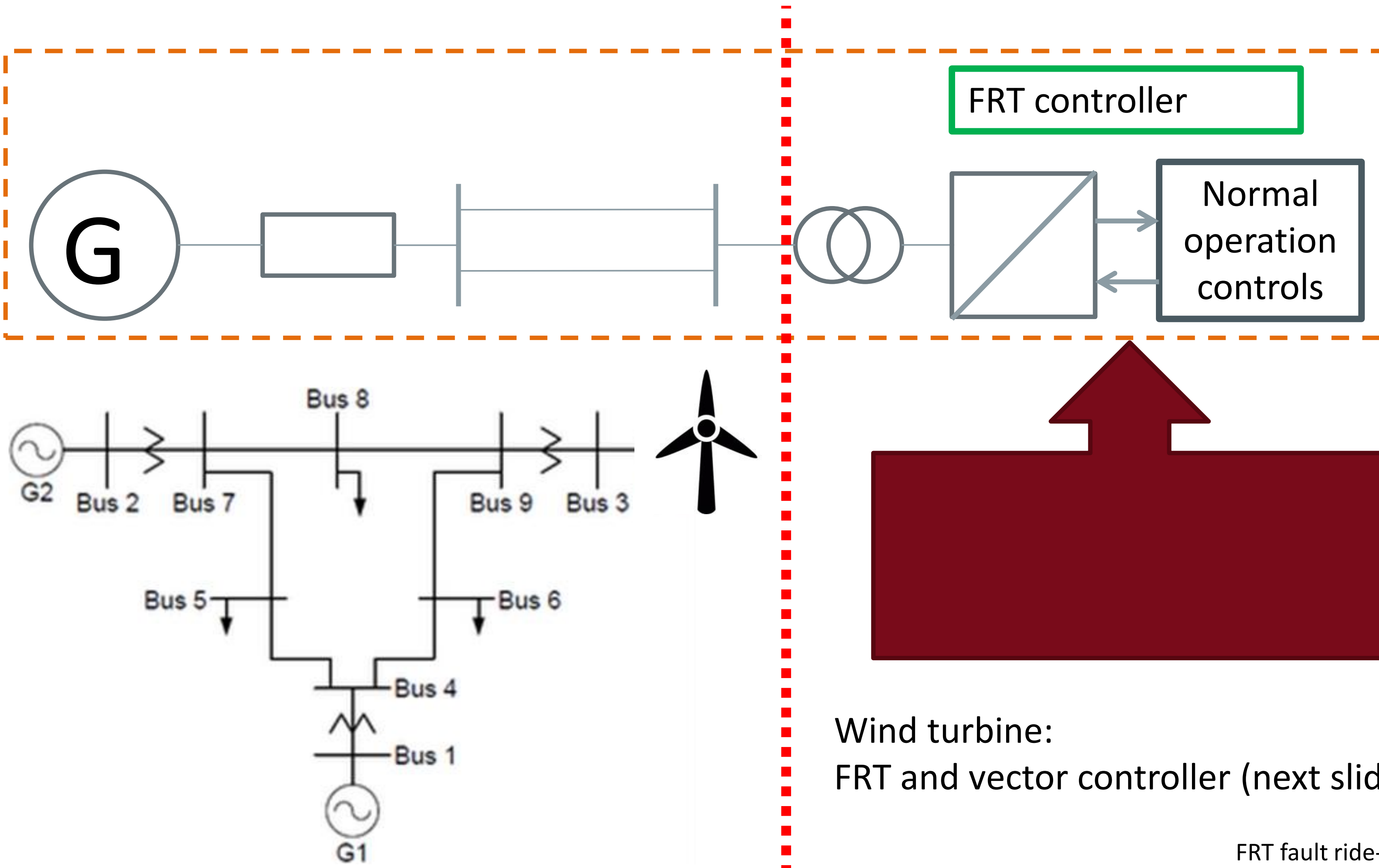
- run experiments
- test validity
- test scalability

Design criteria and focal tools

- Mosaik: issue with running simulations that mutually depend on each other over time → cyclic dependencies
- Cyclic dependencies commonly occur in *physical* models → dynamic simulation of a power system **containing converter-interfaced generation. Tools:**
  - Matlab/Simulink with SimPowerSystem toolbox
  - Powerfactory for RMS simulation
- Application of both FMI for model exchange and FMI for co-simulation → general simulation tool with FMU exporter
  - OpenModelica // Simulink
  - FMI for co-simulation exporter for Powerfactory

RMS: root mean square
FMI: functional mockup interface
FMU: functional mockup unit

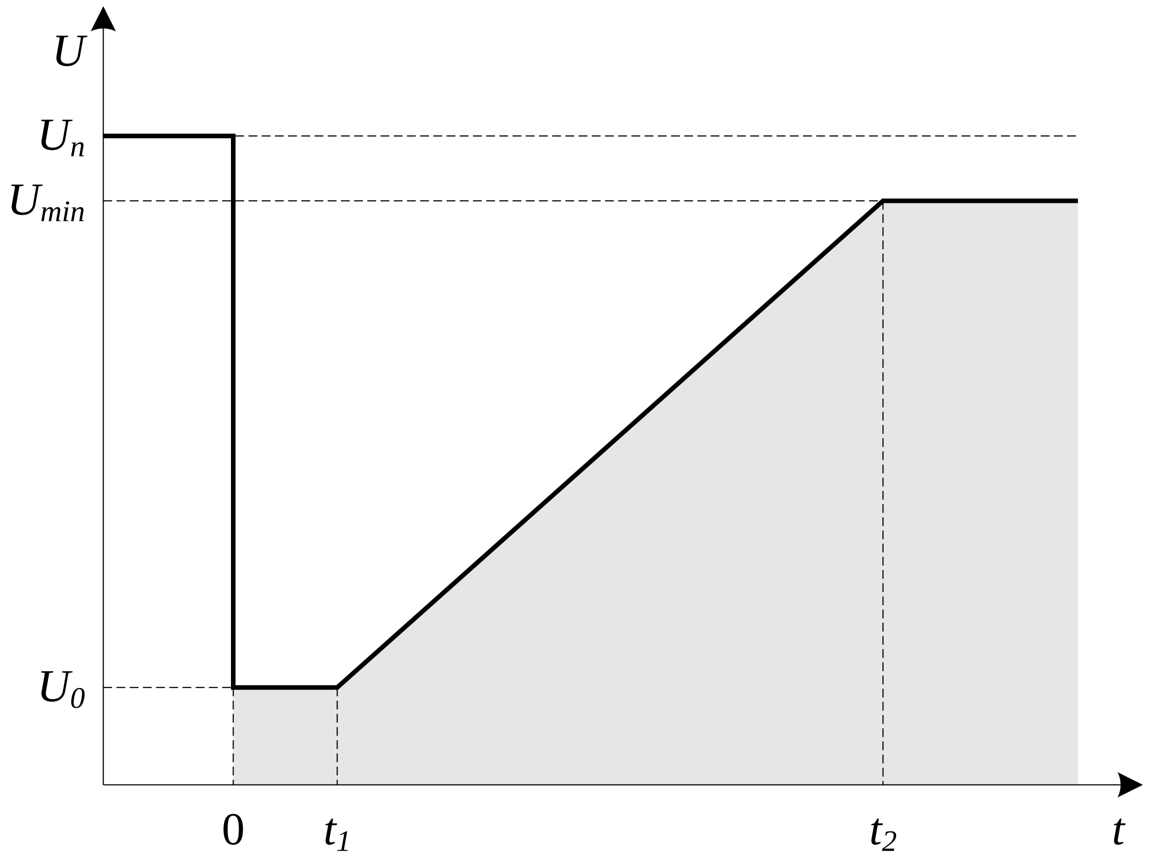
System under Test



Wind turbine:
FRT and vector controller (next slides)

FRT fault ride-trough

Fault ride through requirement

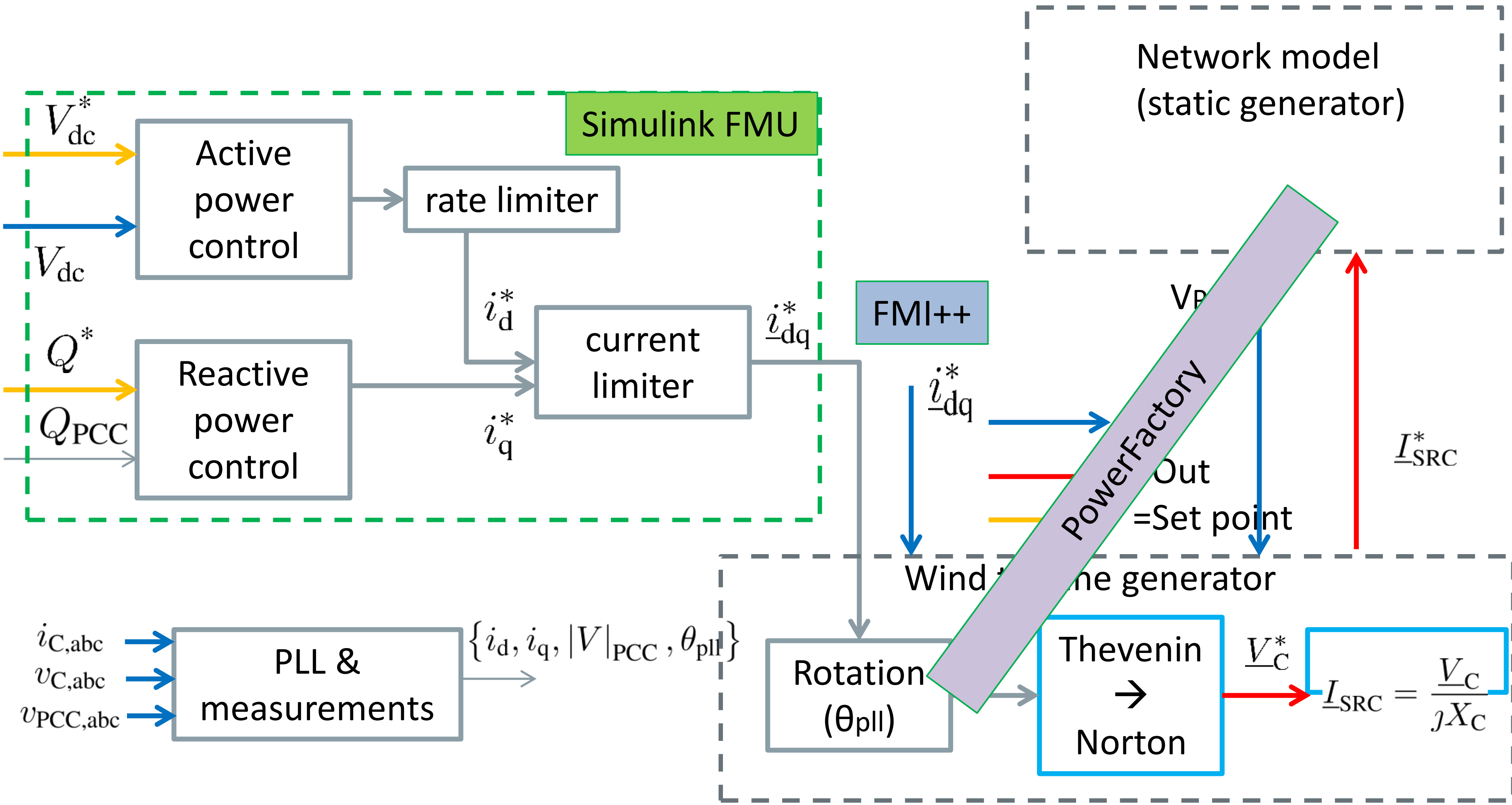


- Wind and PV power plants must comply to the **low-voltage ride through time profile**
- Disconnection allowed when the voltage measured at the terminals enters the grey area
- Parameters differ per TSO
- Converter requires additional overvoltage protection, blocking mechanisms, current limiting schemes, power recovery mechanism

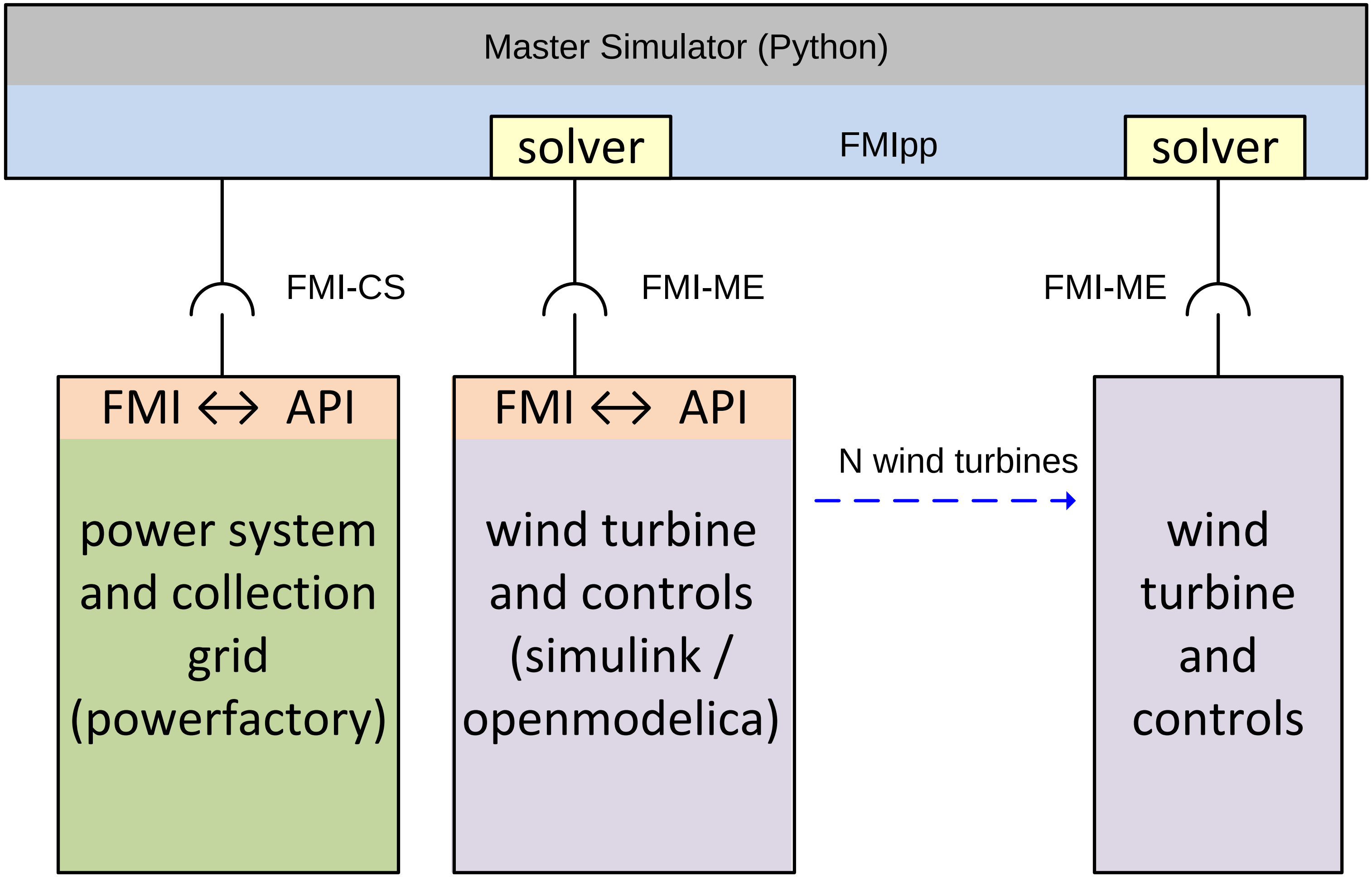
FRT mechanism added on top of wind turbine controller (next slides)

PV: photovoltaic
TSO: transmission system operator

Wind turbine controls for normal operating conditions (RMS-mode)



Co-simulation experiment setup



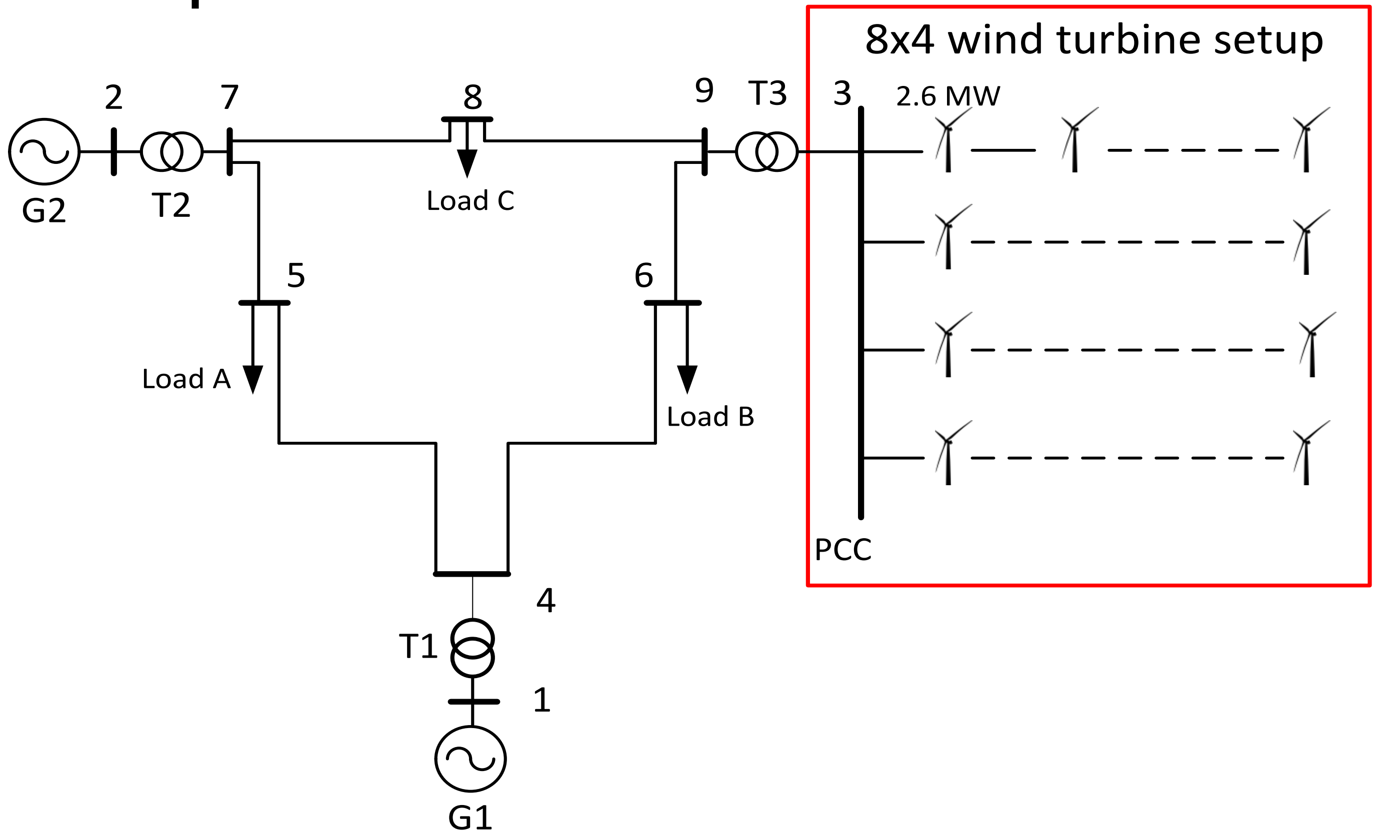
Co-simulation Testing

Test Name	Platform	Purpose	Modifications
Monolithic	PowerFactory	Reference simulation	Gen. G3 in IEEE 9 Bus replaced by WPP
Small Scale Co-Simulation	PF+Matlab+FMI++	Simple co-sim for assessment	No model modifications
Large Scale Co-Simulation	PF+Matlab+FMI++	Co-sim performance check for complex situations and numerically bigger systems	WPP divided in 32 smaller WTGs to have realistic representation. Similarly 32 added converter and FRT controllers.

PF: powerfactory
WPP: wind power plant
WTG: wind turbine generator

Upscaled TC1 experiment

- Goal: test validity and applicability of co-simulation approach
- Split aggregated wind park into 32 wind turbines
- Cable array added in PowerFactory
- 65 FMUs in total



Demonstration Video TC1

Combined Hardware and Software Simulation (TC-2)

Nabil Akroud (Ormazabal)

Definitions

FMI (Functional Mock-up Interface):

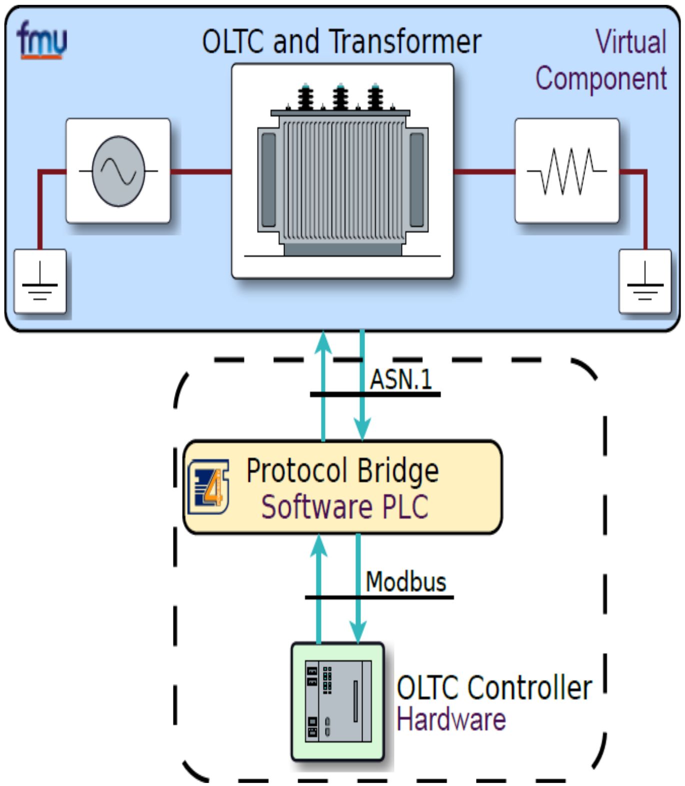
Open and tool-independent standard for exchanging dynamical simulation models between different tools in a standardized format.

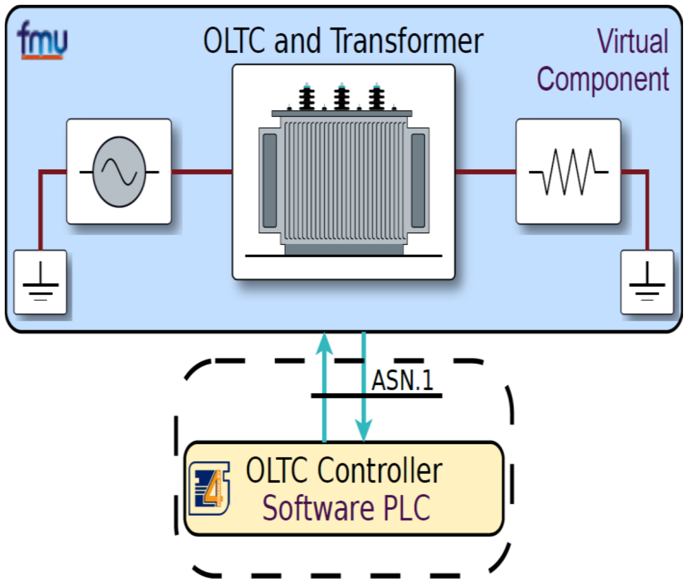
FMITerminalBlock:

Ad hoc FMI Orchestrator + Software PLC IEC-61499

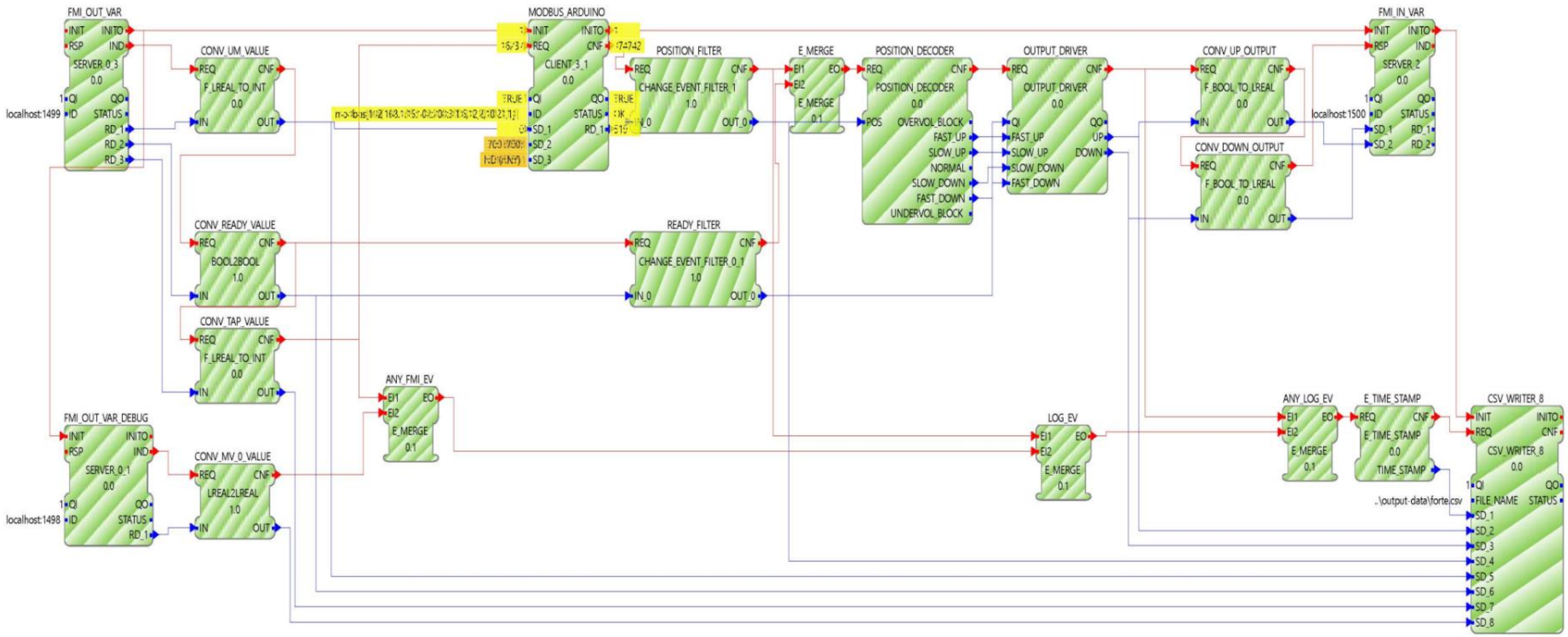
OLATC (On Load Automatic Tap Changer):

Electromechanical device mounted on MV distribution transformer to automatically handle the voltage ratio on the secondary winding in order to maintain the voltage within the accepted level.

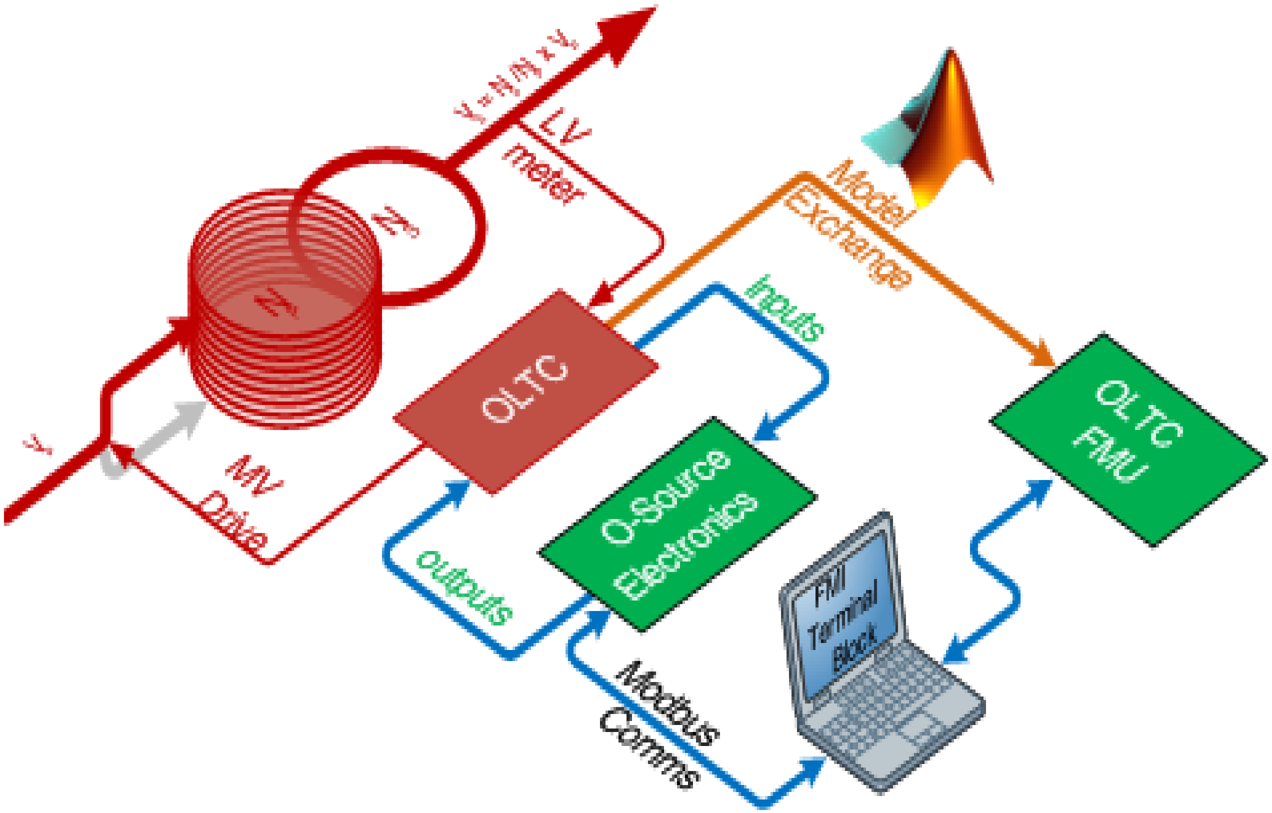

fmu
OLTC and Transformer
Virtual Component
ASN.1
Protocol Bridge
Software PLC
Modbus
OLTC Controller
Hardware


fmu
OLTC and Transformer
Virtual Component
ASN.1
OLTC Controller
Software PLC

SW PLC Controller Diagram IEC-61499 based

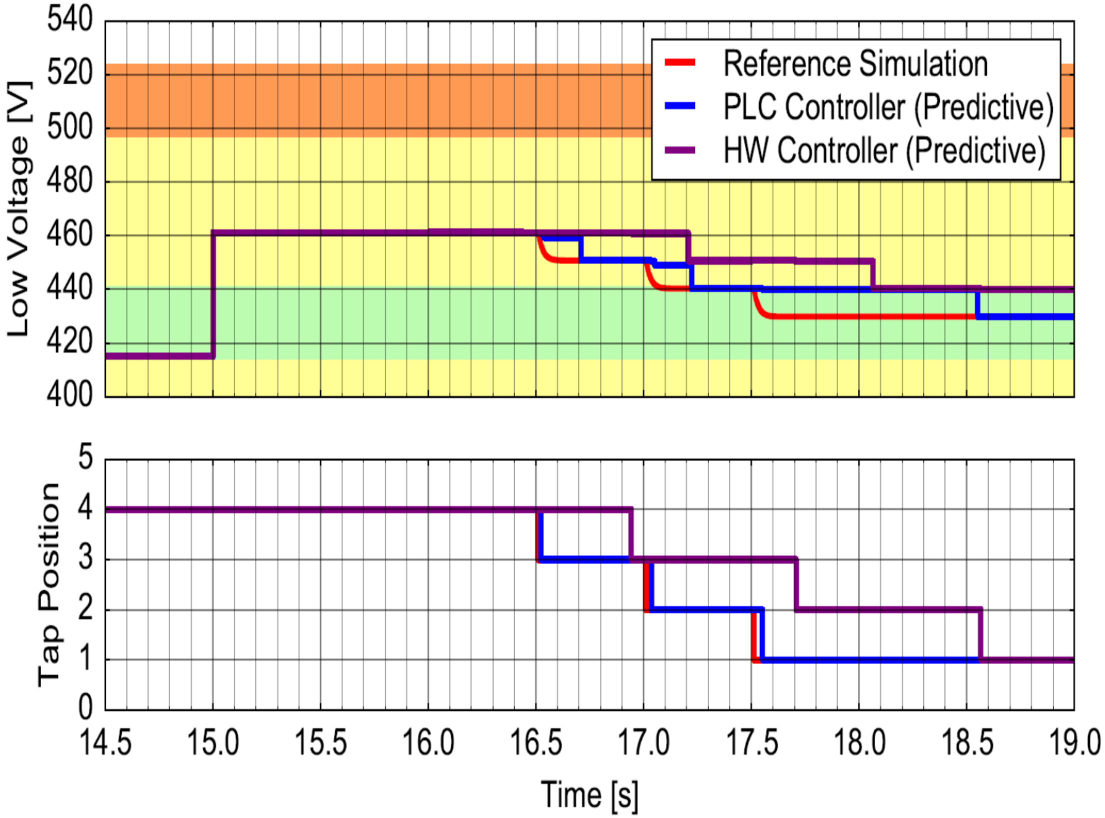


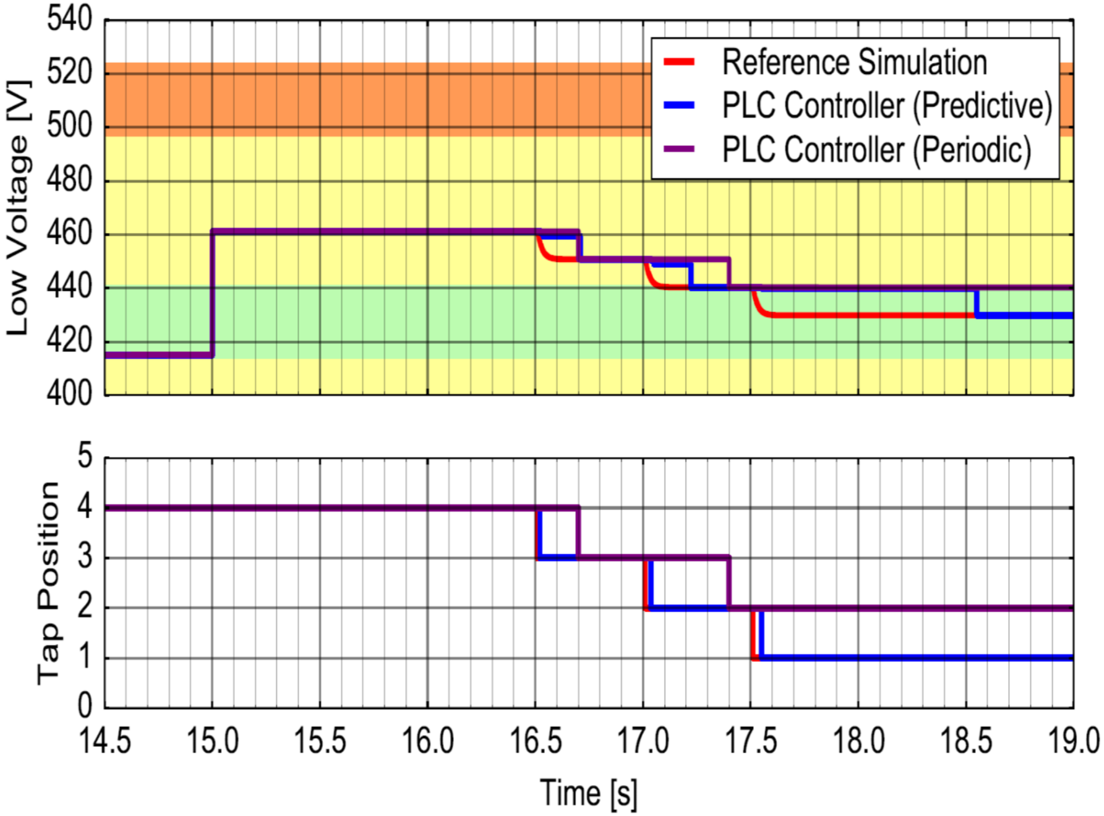
HW Controller Diagram



Demo

Results - Details





Low Voltage [V]
540
520
500
480
460
440
420
400
Reference Simulation
PLC Controller (Predictive)
PLC Controller (Periodic)
Tap Position
5
4
3
2
1
0
14.5
15.0
15.5
16.0
16.5
17.0
17.5
18.0
18.5
19.0
Time [s]

Signal-based Synchronization between Simulators (TC-3)

Edmund Widl (AIT)

Introduction

- *Growing trend*: access information and actuate controllers in Smart Grid applications through *communication networks*
- *Question*: What is the effect of the properties and physical limitations of communication channels on the system?
  - stability of a closed-loop control systems
  - handling of communication errors (loss of information, reordering of message sequence, bit errors, etc.)
  - intentional injection, inhibition or manipulation of data in transit as part of a cyberattack
- *Co-simulation* has become a popular approach to assess the impact of these phenomena on Smart Grid applications
  - *advantage*: use the most appropriate tool for each of the involved domains
  - *challenge*: it is hard to re-use existing work and to exchange models between the existing approaches (lack of openly available simulator and interface implementation)
- ERIGrid approach
  - based on the open interface specification *Functional Mock-up Interface* (FMI)
  - open-source prototype implementation using the communication *network simulator ns-3*

Challenges of FMI-based co-simulation of communication network models

- co-simulation of *physical systems*:
  - exchange of information that corresponds directly to *physical properties* (voltages levels, temperatures, etc.)
  - send *values* of associated model variables from one simulator to another
- *communication systems*:
  - do not just exchange values, but *messages*
  - *transmission* with the help of *protocols* (metadata, data formats)
  - communication network simulators provide *dedicated functionality* to handle the details of data
    transmission protocols
- challenges regarding FMI
  - provide no functionality regarding message transmission
    → details have to be hidden behind FMI-compliant co-simulation interface of the simulator
  - limited support for event-based co-simulation
    → no support for event detection or event prediction
    → no notion of an input or output being absent

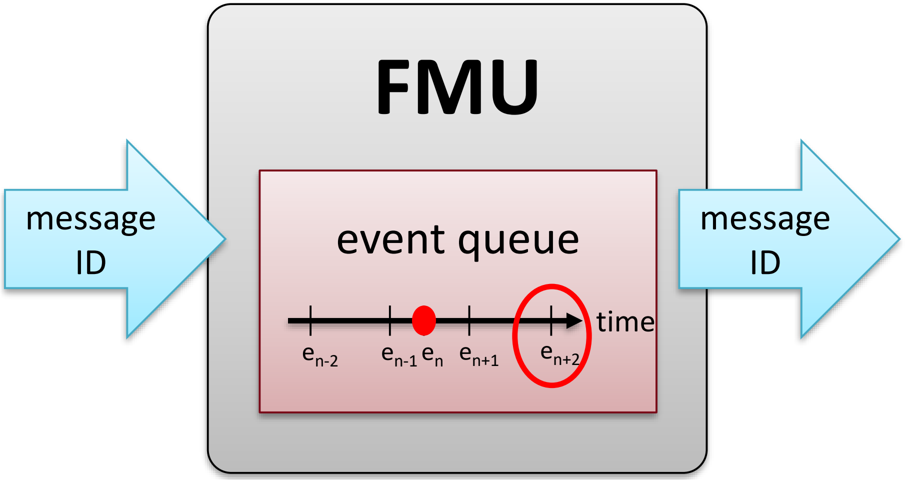
Proposed FMI-compliant approach: Data exchange with message-based simulators

Details of data transmission protocols must be *hidden behind the FMI-compliant interface*:

- *message IDs*
  - transmitted data is associated with a unique message ID
  - message ID is being forwarded to the simulator
- *mock-up messages*
  - simulator generates an internal mock-up message associated with the message ID
  - network model is executed with the mock-up message as stand-in replacement for the original data
  - no need to consider the translation of the original data into a proper format for transmission
  - once the mock-up message has propagated through the network model, its message ID is passed back to the co-simulation framework
- *absence of messages*
  - based on the concept of unique message IDs, a special value represents the absence of input and output messages

Proposed FMI-compliant approach: Event handling for FMUs for Co-Simulation (1/2)

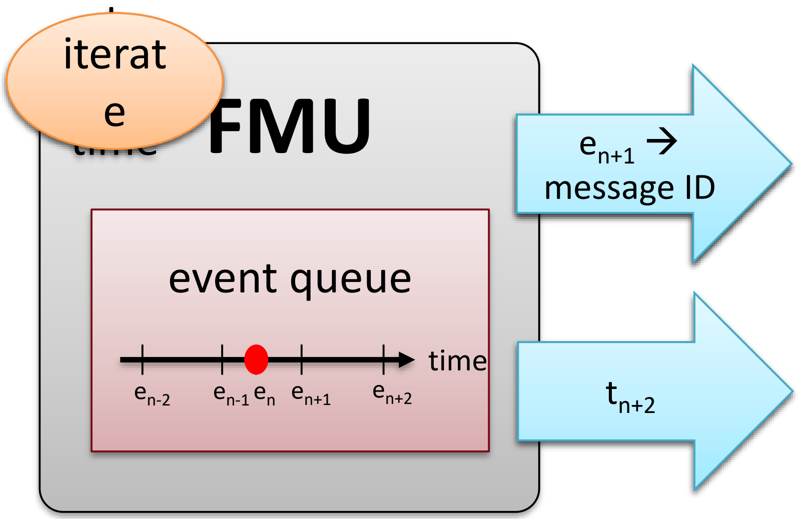
- two types of events are of special interest:
  - *input events*
    - mark the arrival of new messages at an input of the simulated communication network
    - value of an associated FMU input variable changes from 0 to the corresponding message ID
  - *output events*
    - marks the arrival of a message at an end node in the communication network simulator
    - corresponding output message ID as the value of an associated FMU output variable



FMU: Functional Mock-up Unit

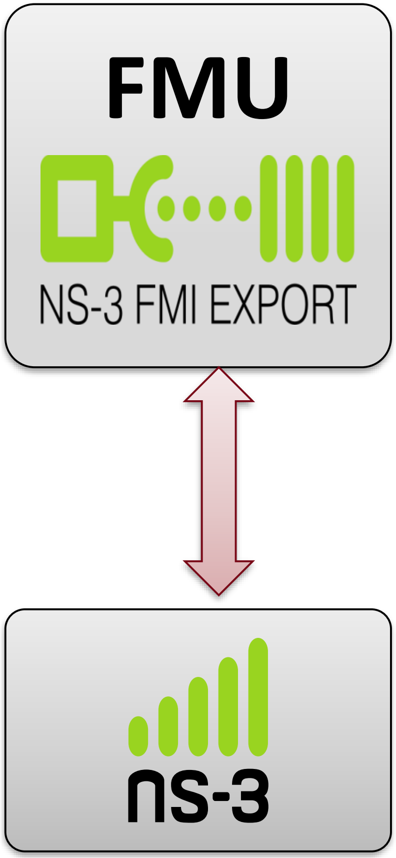
Proposed FMI-compliant approach: Event handling for FMUs for Co-Simulation (2/2)

- FMI specification does not (yet) support the handling of (internal) events for FMUs for Co-Simulation
- ”quick-and-dirty” solution → demonstrate feasibility of approach, but do not put too much focus on specific proposal for FMI extension
  - *internal event prediction*
    - FMUs have to define a dedicated output variable for event prediction
    - value always corresponds to the time of the next internal event
  - *event processing*
    - use *iterations* (simulation steps with step size equal to zero) to trigger the FMU to process events



FMI-support for the ns-3 network simulator

- ns-3 module *fmi-export*
  - creates an FMU for Co-Simulation from a user-defined ns-3 script
  - implements a tool coupling mechanism
    - control the execution of the ns-3 simulator
    - establish a connection for data exchange during run-time
  - interaction with ns-3 is limited to the repeated execution of the same ns-3 script
    - call the FMU’s step method → ns-3 executes the same model
    - use different random seeds each time → produce different outputs
- user has to implement a dedicated class → class *SimpleEventQueueFMUBase*
  - provides functions for declaring input and output variables
  - provides functions for adding events to internal event queue
- open source, available at https://erigrid.github.io/ns3-fmi-export/



EU H2020 Programme GA No. 654113

open source, available at:

https://erigrid.github.io/ns3-fmi-export/

ns3-fmi-export

View On GitHub

Project maintained by ERIGrid

Hosted on GitHub Pages — Theme by mattgraham

DOI 10.5281/zenodo.1934876

The ns-3 FMI Export Module

About

Module **fmi-export** enables the FMI-compliant simulation coupling with ns-3 scripts, i.e., ns-3 script are launched and executed through an FMI-compliant co-simulation interface. In terms of FMI terminology, ns-3 is the slave application, and generated FMUs launch ns-3 and synchronize its execution during runtime (tool coupling).

Module **fmu-examples** provides examples for using the **fmi-export** module. The module comprises dedicated models (clients and servers), helpers and simulation scripts implementing example applications, whose functionality is then exported as FMU for Co-Simulation. Furthermore, test applications (written in Python) show how the resulting FMUs can be used in a simulation.

Prerequisites and installation on Linux

In addition to ns-3, the following tools/libraries need to be installed:

- **Cmake**
- **Boost**: all header files plus compiled *date_time*, *system* and *filesystem* libraries

Follow these instructions to install the **fmi-export** module:

1. This module relies on a lot of functionality provided by the FMI++ library. Hence, in order to install this module, the latest version of the FMI++ library should be cloned from its repository:

```
$ git clone https://git.code.sf.net/p/fmipp/code fmipp
```

2. Get the source code from GitHub.

```
$ git clone https://github.com/ERIGrid/ns3-fmi-export.git
```

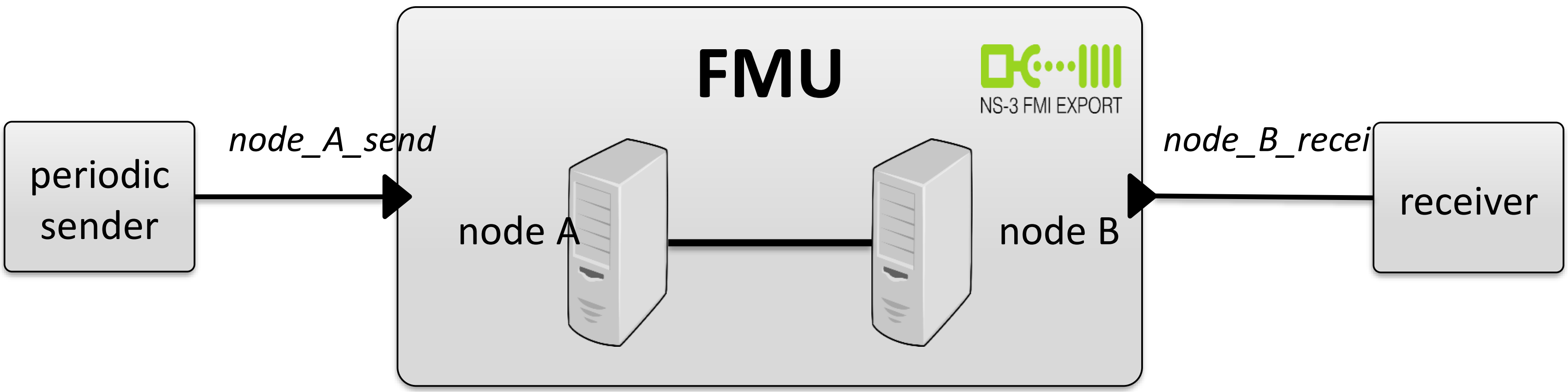
Dedicated ns-3 Application Layer Models

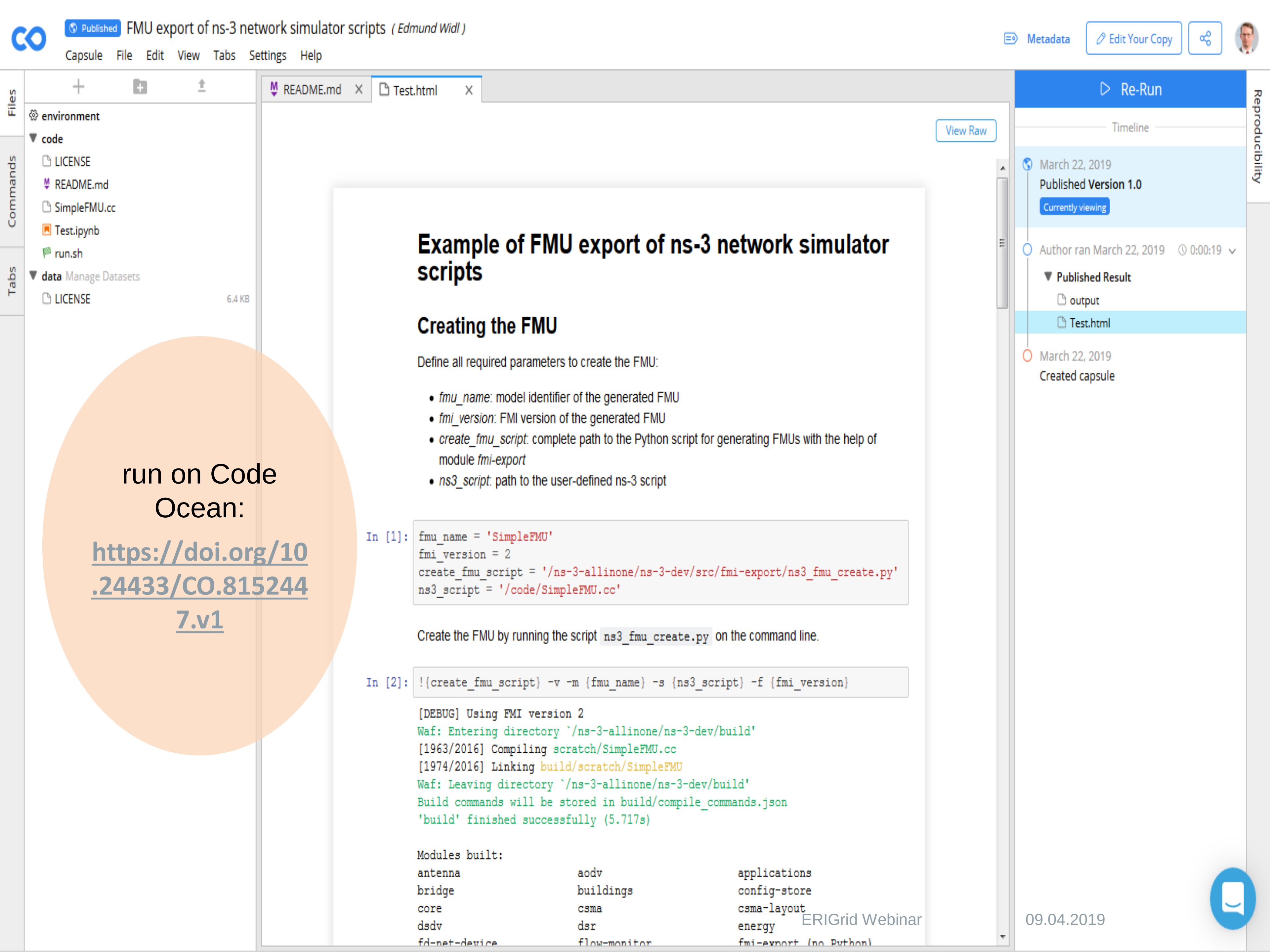
- to utilize the event queue of module fmi-export, dedicated application layer models (ALM) need to be used in ns-3 scripts
- specific functionality of ALMs depends on considered application, but there are in general two distinct types:
  - *clients*
    - ALMs of client applications wait for new events (i.e., incoming message IDs at the FMU's inputs)
    - they send mock-up messages accordingly
    - for subsequent calculation of the end-to-end delay, the time of the packet creation is added as part of the message header
  - *servers*
    - upon receiving a packet, they extract the packet's header to calculate and store the end-to-end delay of the transmission
    - used to calculate a corresponding timestamp and add an event to the event queue
- otherwise, standard ns-3 component models can be used

EU H2020 Programme GA No. 654113

A simple example

- run a simple simulation:
  - 1 periodic sender → sends messages through FMU input variable *node_A_send*
  - 1 receiver → receive messages through FMU output variable *node_B_receive*
- example available online: https://doi.org/10.24433/CO.8152447.v1

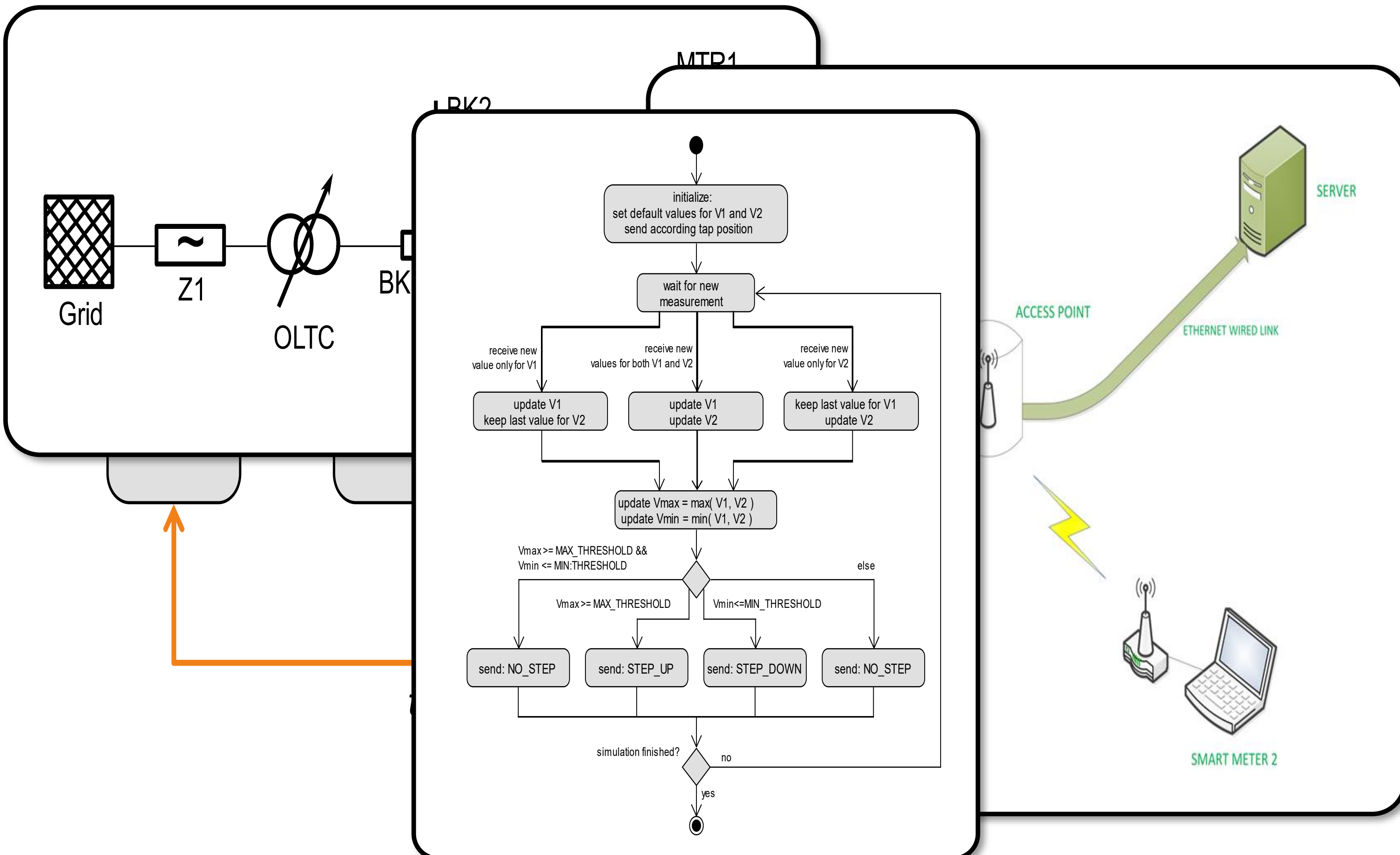




run on Code Ocean:

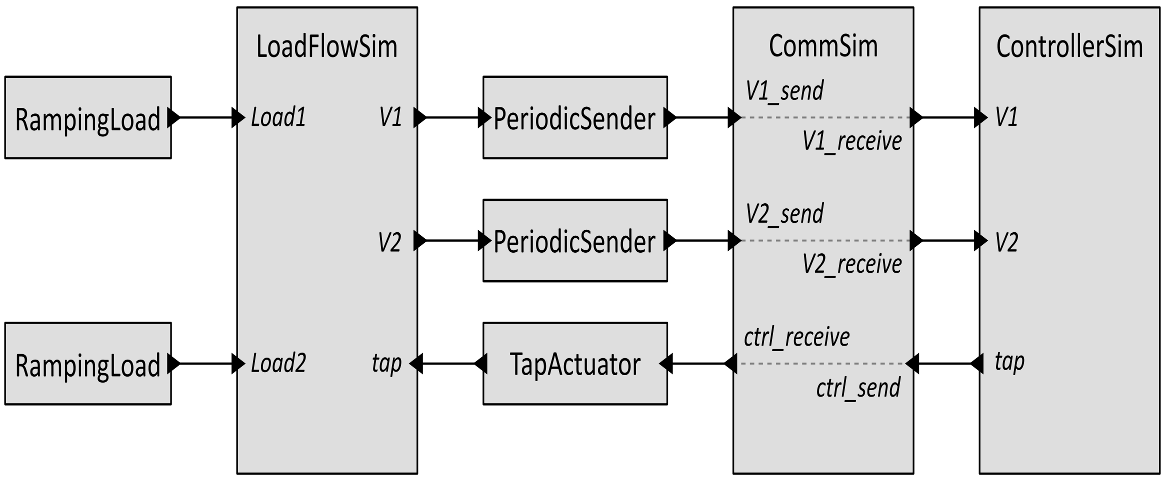
https://doi.org/10.24433/CO.8152447.v1

More advanced test case (1/3)



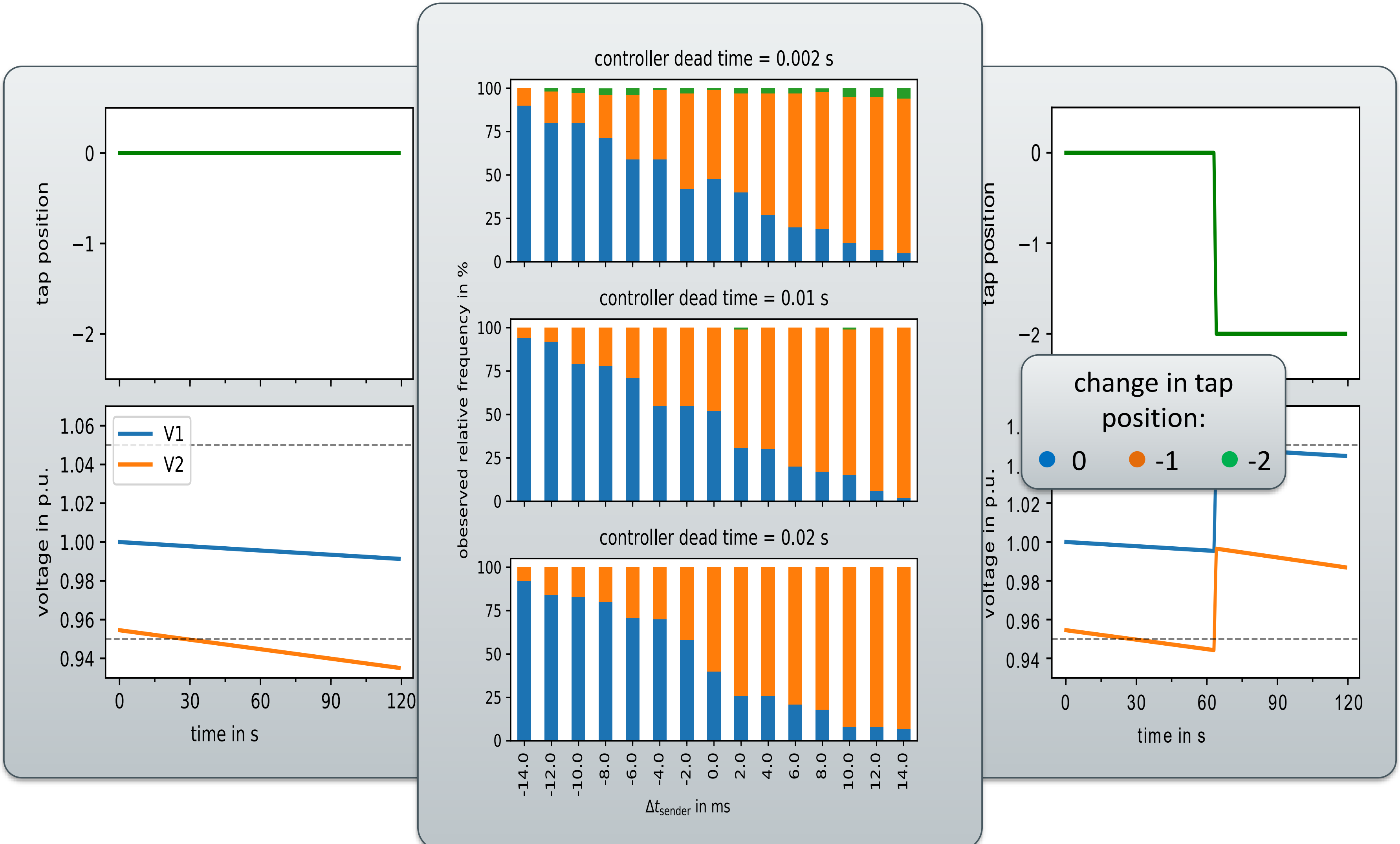
neter
nger

More advanced test case (2/3)



- implemented in the co-simulation environment *mosaik*
- *FMUs* for all domain-specific models
  - communication network → *ns-3*
  - power system → *PowerFactory*
  - controller → *MATLAB*
- implementation available online: https://github.com/ERIGrid/JRA2-TC3

More advanced test case (3/3)



Conclusion and outlook

- ns-3 module *fmi-export* is a prototype of an FMI-based co-simulation interface for the ns-3 communication network simulator
  - open-source
  - available at https://erigrid.github.io/ns3-fmi-export/
- based on a semantically clear mapping of the requirements for message-based simulations to the FMI specification
- where such a mapping was not possible, simple workarounds have been implemented that are expected to be compatible with future extensions of the FMI standard
- future developments will aim at a more dynamic coupling
  - synchronizations at run-time not restricted to individual simulation runs
  - instead, the simulation within ns-3 itself should be synchronized to the master algorithm

Co-simulation Models available as Open Source

- The work was done as a part of European Commission funded programme Horizon 2020 under the project European Research Infrastructure supporting Smart Grid Systems Technology Development, Validation and Roll Out (ERIGrid) [https://erigrid.eu].
- Interfaces and test systems developed available as Open Source models in github.

Links:

- Co-simulation assessment for continuous-time RMS studies (TC-1): https://github.com/ERIGrid/JRA2-TC1
- Combined Hardware and Software Simulation (TC-2): https://github.com/AIT-IES/FMITerminalBlock https://github.com/NabilAKROUD/OLTC_Arduino
- Signal-based Synchronization between Simulators (TC-3): https://github.com/ERIGrid/JRA2-TC3

Discussion (QnA)

Moderated by: Thomas Strasser (AIT)