D-JRA2.1 Simulator coupling and Smart Grid libraries

Widl, Edmund; Spiegel, Michael; Moyo, Cyndi; Strasser, Thomas; van der Meer, Arjen; Palensky, Peter; Bhandia, Rishabh; Emhemed, A.; Syed, M.; Gehrke, Oliver

Publication date
2017

Document Version
Final published version

Citation (APA)
Widl, E., Spiegel, M., Moyo, C., Strasser, T., van der Meer, A., Palensky, P., Bhandia, R., Emhemed, A., Syed, M., Gehrke, O., Morales Bondy, D. E., Steinbrink, C., Blank, M., Stathakis, A., Sarris, T., Kotsampopoulos, P., Akroud, N., Sagarduy, I. O., Nguyen, V. H., ... Delaplagne, T. (2017). *D-JRA2.1 Simulator coupling and Smart Grid libraries*. European Commission H2020. https://erigrid.eu/wp-content/uploads/2017/07/DL_D-JRA2.1_Simulator_coupling_and_Smart_Grid_libraries_2017-05-22.pdf

European Research Infrastructure supporting Smart Grid Systems Technology Development, Validation and Roll Out

Work Package 08

JRA2 - Co-Simulation based Assessment Methods

Deliverable D8.1

D-JRA2.1 Simulator coupling and Smart Grid libraries

Grant Agreement No:	**654113**
Funding Instrument:	**Research and Innovation Actions (RIA) – Integrating Activity (IA)**
Funded under:	**INFRAIA-1-2014/2015: Integrating and opening existing national and regional research infrastructures of European interest**
Starting date of project:	**01.11.2015**
Project Duration:	**54 month**

Contractual delivery date:	**30.04.2017**
Actual delivery date:	**22.05.2017**
Name of lead beneficiary for this deliverable:	**1 AIT Austrian Institute of Technology GmbH**
Deliverable Type:	**Other (O)**
Security Class:	**Public (PU)**
Revision / Status:	**released**

Project co-funded by the European Commission within the H2020 Programme (2014-2020)

Document Information

Document Version: 2

Revision / Status: released

All Authors/Partners Edmund Widl, Michael Spiegel, Cyndi Moyo, Thomas Strasser / AIT
Arjen van der Meer, Peter Palensky, Rishabh Bhandia / TUD
Abdullah Emhemed, Mazheruddin Syed / USTRATH
Oliver Gehrke, Daniel Esteban Morales Bondy / DTU
Cornelius Steinbrink, Marita Blank / OFFIS
Aris Stathakis, Theodoros Sarris, Panos Kotsampopoulos / ICCS
Nabil Akroud, Inaki Orue Sagarduy / OCT
Van Hoa Nguyen / GINP
Diana Moneta, Carlo Sandroni, Sergio Corti / RSE
Sanna Uski, Laukkanen Matti / VTT
Przemyslaw Chodura / DNVGL
Ron Brandl / IWES
Tuan Tran, Begonia Lazpita, Tony Delaplagne / CEA

Distribution List ERIGrid consortium members

Document History

Revision	Content / Changes	Resp. Partner	Date
1	First complete draft version	AIT, TUD, USTRATH, DTU, OFFIS, ICCS, OCT, GINP, RSE, VTT, DNVGL, IWES, CEA	14.04.17
2	Version for submission	AIT, TUD	28.04.17

Document Approval

Final Approval	Name	Resp. Partner	Date
Review Task Level	Evangelos Rikos	CRES	20.04.17
Review Task Level	Erik de Jong	DNVGL	25.04.17
Review WP Level	Edmund Widl Abdullah Emhemed	AIT UST	28.04.17
Review WP Level	Arjen van der Meer	TUD	29.04.17
Review Steering Com. Level	Thomas Strasser	AIT	22.05.17

Disclaimer

information is free from risk. Neither the ERIGrid consortium as a whole, nor any single party within the ERIGrid consortium accepts any liability for loss or damage suffered by any person using the information.

This document does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of its content.

Table of contents

Abbreviations

AC	Alternating Current
AMI	Automated Metering Infrastructure (smart metering)
AODV	Ad-hoc On-Demand Distance Vector (routing protocol)
API	Application Programming Interface
BMS	Battery Management System
CHIL	Controller Hardware-in-the-Loop
CS	Co-Simulation
CT	Continuous Time
DC	Direct Current
DER	Distributed Energy Resource
DES	Discrete Event Simulation
DFIG	Double Fed Induction Generator
DLL	Dynamic-Link Library (Windows shared library)
DoE	Design of Experiments
DPSL	Dynamic Power Systems Laboratory
DRTS	Digital Real-Time Simulator
DSO	Distribution System Operator
DT	Discrete Time
EMT	Electro-Magnetic Transient
EV	Electric Vehicle
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
FRT	Fault Ride Through
GNU	GNU's Not Unix
GPL	GNU General Public License
HIL	Hardware-in-the-Loop
HLA	High Level Architecture
HVDC	High-Voltage Direct Current
ICT	Information and Communications Technology
IP	Internet Protocol
IPR	Intellectual Property Rights
JRA	Joint Research Activity
JSON	JavaScript Object Notation
LTE	Long-Term Evaluation (telecommunication standard)
ME	Model Exchange
ML	Model Library
OLSR	Optimized Link State Routing (IP routing protocol)
OLTC	On-Load Tap Changing Controller

OM	OpenModelica
PCC	Point of Common Coupling
PHIL	Power Hardware-in-the-Loop
PID	Proportional-Integral-Derivative (controller concept)
PLC	Power Line Communication
PMSG	Permanent Magnet Synchronous Generator
PV	Photovoltaics
RC	Research Challenge
RF	Radio Frequency
RMS	Root Mean Square
SOC	State of Charge
SA	Sensitivity Analysis
TC	Test Case
TCP	Transmission Control Protocol (networking protocol)
TF	Transfer Function
UQ	Uncertainty Quantification
WP	Work Package
WPP	Wind Power Plant
XML	Extensible Markup Language

Executive Summary

Work package JRA2 focuses on the development of advanced simulation-based methods to check and validate smart grid scenarios, configurations and corresponding applications. The main aim is to employ offline simulation of scenarios where a combination of parallel processing, advanced optimization techniques, and design-of-experiments is used to master the system complexity. Secondary targets include the development of methods for HIL application as well as for the assessment of cyber-security concepts. This assessment will cover the following smart grid properties: system stability, system scalability, component interoperability, and information security. Eventually it is the goal to explore the operational limits and the sensitivity of these system properties towards system parameters.

1 Introduction

1.1 General Overview

1.1.1 Overall Approach of Work Package JRA2: Co-Simulation based Assessment Methods

Work Package (WP) JRA2 will implement a co-simulation approach, using the mosaik platform and standardised interfacing techniques for models and simulation tools, based on the Functional Mock-up Interface (FMI) specification, as much as is technically feasible. Testing its applicability and further developing FMI for smart grid systems is hence a dominant aspect in JRA2.

The overall approach to achieve the abovementioned goals was as follows. First, Test Cases (TC) have been devised based on research challenges relevant for JRA2, which contain various smart grid properties at an elementary level:

- TC1: cyclic dependencies between continuous simulators
- TC2: combined hardware and software simulation
- TC3: signal-based synchronisation between simulators

Secondly, based on these test cases, a collection of requirements has been compiled that defined the necessary functionality to be provided by or implemented for each selected tool and mosaik. Third, smart grid models have been developed along the lines of the domains that have been selected for the three test cases. Finally, each test case implements at least one monolithic simulation (i.e., assessment of the system under test within a single simulator) and one heterogeneous simulation (i.e., co-simulation).

This deliverable reports the overall approach, initial co-simulation developments in the selected tools and mosaik, the proposed smart grid model library, a formal description of the adopted test cases as well as preliminary simulation studies.

1.1.2 Scope of “Simulator Couplings and Interfaces”

Coupling different simulation packages leads to a number of serious challenges: different solvers have to interoperate, various numerical phenomena are not well understood, and fundamentally different models must exchange dynamic state information. An emerging industry standard that touches some of these aspects is the Functional Mock-up Interface (FMI). It allows for coupling simulation packages and to encapsulate models. The individual framework components can be tested individually, due to its modular architecture and well-defined interfaces.

Within this context, the overall goal of this activity was

- An assessment of popular specialized and universal simulation packages for smart grids, and
- A flexible toolset / library to couple these simulation packages.

For the assessment a bottom-up approach has been devised. First, research challenges relevant for JRA2 have been formulated, based on which a set of test cases was devised. Then, a list of corresponding requirements regarding co-simulation functionality was compiled. The best suited tools were then selected accordingly and the development of a flexible toolset started. At the time of writing this report, functional prototypes of all further required developments (i.e., co-simulation interfaces and master algorithms) are available within the ERIGrid consortium or at least conceptually devised.

1.1.3 Scope of “Extended Model-Libraries covering Power System and ICT Components”

Existing domain specific simulation tools do not cover and describe all aspects of modern smart grid technology effectively and sufficiently. For the test and validation of advanced smart grid solu-

tions, more sophisticated and representative models with flexible use are necessary. In response to such requirements, the main objectives were: identification of the gaps in existing smart grid models; extension of existing model libraries across different simulation domains (i.e. power systems, communications networks, and controls); and compliance of the developed models library with the FMI for Model Exchange (FMI-ME) specification. To achieve these objectives, the task was planned and conducted through the following four phases:

- Conduct an FMI appraisal and model gap analysis to understand the state of the art of FMI-ME applications within smart grids and identify the models that need to be reassessed and improved with enhanced performance and usability.
- Model design and Functional Mock-up Units (FMU) development, enabling the developed models to be utilised as tool-independent models.
- FMU validation and integration testing to assess the accuracy and quality of the developed FMUs.
- FMU systems testing to assess and study the interaction between multiple FMUs within a co-simulation framework.

In general, the selection and identification of the developed models were driven by a number of factors, including the requirements of the test cases, the models that would facilitate development and validation of novel control solutions, and the shared research interests of the partners. Each of the models developed is supplemented with detailed documentation to describe the model development and its potential use and applications.

1.1.4 Scope and Structure of the Document

Section 1 presents an executive summary of JRA2 activities and the scope of the tasks relevant for this deliverable. Furthermore, it comprises background information relevant for work package JRA2 by providing a brief introduction of the FMI specification and a statement of the rationale behind the selection of considered simulation domains.

Section 2 covers the work carried out so far concerning simulator couplings and interfaces. First, it covers the assessment and selection process of simulation tools. Then it introduces the domain-specific simulation tools chosen for the further work in JRA2, focusing on co-simulation interfaces, compliance issues with the FMI specification and implementation status. Furthermore, it outlines the rationale behind and the implementation of the devised coupling methods (i.e., off-line co-simulation, virtual lab components).

Similarly, Section 3 covers the work carried out so far concerning smart grid model libraries. First, it covers the issues related to the methodology applied in this task (i.e., model selection, building, testing). Then it gives an overview of the implemented models, focusing on a justification of the model selection and potential applications. Furthermore, the steps taken to ensure the quality of models have been discussed and the experience of utilizing FMI-ME summarized.

Section 4 gives an overview of the test cases developed so far for JRA2. Their preparation and the subsequent derivation of requirements have been a driving factor behind the tool and model selections of both tasks. Furthermore, their implementation will be the basis for the forthcoming tasks as well as for the continuation of the current tasks.

Finally, Section 5 concludes and gives an outlook on upcoming work. Details for further reading are provided in two appendices (formal specification of test cases, model library documentation).

1.2 The Functional Mock-up Interface

The FMI [1] specification is a tool independent standard to support both model exchange and co-simulation of dynamic models using a combination of XML files and C-code (either compiled in

DLL/shared libraries or in source code). With the development pushed by both industry and academia, a growing number of simulation software providers (commercial and community-driven) has adopted and integrated FMI into their products. This allows well-established and specialized simulation tools from a wide spectrum of application domains to be coupled and re-used with only small adaptations in contrast to monolithic approaches.

A simulation component compliant to the FMI specification is called a FMU. An FMU consists of a ZIP file that contains an XML-based model description together with a shared library and/or source code implementing the corresponding interface (as C API). Version 2.0 of the FMI specification supports two distinct types of interfaces:

- *FMI for Co-Simulation* (CS) defines stand-alone black-box simulation components, which can be directly coupled within a co-simulation framework. Data exchange with FMUs for CS is restricted to discrete communication points, in the time between two communication points the encapsulated system model is solved by the FMU's internal solver. An FMU for CS may implement either a self-contained simulation component or call another simulation tool at runtime (tool coupling).
- *FMI for Model Exchange* (ME) provides standardized access to the model equations, enabling simulation environments to exchange models. Models are described by differential, algebraic and discrete equations with time-events, state-events and step-events (unlike the black-box approach pursued with FMI for CS). In case an FMU for ME describes a continuous system, this system has to be solved with integrators provided by the embedding environment.

The number of simulation tools that claim to support FMI is ever growing, but they do so in a variety of different ways. For instance, while some only offer to import FMUs, others are also able to export their models and functionality. Furthermore, the way a simulation tool utilizes the functionality provided by FMUs is strongly related to the simulation paradigm that the tool is based on. For instance, a tool designed for the co-simulation of (physical) models with continuous outputs (compare for instance with [2]) typically focuses on a different subset of FMI's functionality than a tool designed for the deterministic simulation of heterogeneous systems (compare for instance with [3]).

Such common interfaces for a variety of tools and models from different simulation domains will decrease the complexity and efforts related to simulation-based smart grid assessments. A co-simulation component which was developed for one co-simulation setup does not have to be manually ported to another one. Likewise, the effort which is required to adapt a co-simulation setup to include new models or tools will be drastically reduced or eliminated at all. Tedious coupling work which often includes the development of tool specific ad-hoc interfaces is reduced to some configuration actions.

By using the FMI for interfacing in the context of off-line co-simulation and virtual lab components, one step towards the harmonization of modelling and interface methods is taken. Thereby, it is not necessary to develop the coupling strategies for every test-case anew but to utilize existing, well proven algorithms.

1.3 Selected Simulation Domains

As explained above, JRA2 is assigned the task of developing advanced simulation methods to check and validate smart grid applications. In general, smart grid setups that are of interest in the context of JRA2 mainly comprise power system infrastructure, communication systems and control aspects.

The *power system* is obviously the most integral part of any smart grid infrastructure, which is why it also forms the core of all three test cases that have been devised for JRA2. However, smart grid setups typically comprise a complex infrastructure, interacting with various other domains. Hence, robust *automation and control systems* are essential to maintain stability and efficiency. Furthermore, there is a continuous interaction among the various components, devices and domains, which leads to huge amounts of data being exchanged. Hence, also *communication systems* have to be considered as a major simulation domain to be analysed in JRA2.

These are not the only domains relevant to the simulation of smart grids. For example, the integration between the electrical power system and other energy carriers such as district heating and natural gas is a research area of growing importance. However, it was decided between the JRA2 partners to focus on the three aforementioned domains as the core domains, because of their essential importance for smart grid research in general, as opposed to domains only required for special sub-areas.

As the focus of the WP is the development of assessment methods based on co-simulation, the three test cases developed and described later in the document focus on interfacing and coupling at least two of the three simulation domains introduced above. The coupling among these domains helps to create a realistic representation of any smart grid infrastructure and its behaviour. As such, the approach followed in JRA2 serves as a meaningful proof-of-concept for co-simulation-based smart grid assessment.

2 Simulator Coupling and Interfaces

Two specific goals are in the focus of JRA2 concerning simulator coupling and corresponding interfaces: First, an assessment of popular simulations tools in the context of smart grids research has been conducted. Secondly, a flexible toolset to couple these simulation tools has been developed. For the latter, special attention has been paid to comply with the FMI specification as much as possible.

The following section gives an overview of the work done and the targets achieved in this context. It starts with an explanation of the rationale behind the selection of simulation tools for JRA2, continues with a brief description of these tools (focusing on the developments done within JRA2) and concludes with an overview of the coupling methods intended for the further use in JRA2.

2.1 Simulation Domains and Research Challenges

Smart grid technology can be defined as the application of automation to the operation of electrical power systems. A very strong trend in contemporary automation is the replacement of monolithic controller designs with distributed control systems where smart sensors and actuators connect with controllers through a digital communication interface. It is this paradigm shift which is expected to enable the largest benefits of smart grid technology by allowing the system-scale coordination of resources. Consequently, in order to accurately simulate smart grid systems, the interaction between the following three domains is of crucial importance:

- *Electrical power systems*: This domain contains equipment for the generation, consumption and storage of electricity, as well as the technical infrastructure used to interconnect this equipment. This includes static components such as conductors, transformers and bus bars as well as active components such as switchgear or compensation devices. The electrical power system domain represents a time-continuous, nonlinear system which must be in energy balance at all times. Like many other physics simulations, a power system simulation determines the equilibrium point of the entire system in discrete steps of time.
- *Communication*: This domain includes sources and sinks of information, information hubs such as routers and switches, communication media such as cables and wireless connections, as well as auxiliary equipment such as media converters. Most communication simulators focus on packet-based networks; while other types of networks exist (e.g., fieldbuses), these are typically of much lower complexity and less interesting for a detailed simulation. Two main approaches to communication simulation exist: a stochastic approach which determines link utilization and queue lengths for the entire network in discrete steps of time, and an event-based approach which follows the journey of individual packets through the network.
- *Automation and control*: This domain contains control logic and algorithms. Two main categories can be distinguished: Continuous control which can be expressed using classic control theory, and decision-based control which is often easier to represent as a discrete-event system. At the extreme end, the simulation of a controller may simply entail the real-time execution of its control software in a suitable container. At the other end of the spectrum, continuous control systems can be transformed into systems of differential equations which can be solved in discrete steps of time.

For each of these technical domains a broad variety of simulators is available, which made a general assessment unfeasible. However, the actual goal of assessing these simulators in the context of JRA2 was the selection of the most suitable tools for the WPs purpose. Therefore, a bottom-up approach for the assessment was devised, which in the first step required the formulation of the research challenges that will be addressed by JRA2. Based on these challenges, a minimalistic set of test cases was devised (see Section 4 and Appendix A) and a list of corresponding requirements regarding co-simulation functionality was compiled. Finally, these requirements were used to perform a selection of tools further considered for JRA2 (see below).

The main Research Challenges (RC) that were identified for JRA2 are:

- *RC1: Handling of cyclic dependencies*. Splitting a physical model in several separate sub-models for co-simulation can often be beneficial. For instance, it allows to re-use already existing models (simplifying interaction between different stakeholders) or to use black-box third-party models (protection of intellectual property). However, the problem arises that the state equations of the individual sub-models are in general interdependent. To resolve these mutual (cyclic) inter-dependencies in a FMI-based co-simulation is challenging, because the interaction between the solvers (integrators) of the individual subsystem is limited.
- *RC2: Coupling with hardware setups*. In certain cases, being able to couple power hardware in a laboratory to a co-simulation platform would provide researchers with a "best of both worlds" testing setup where the flexibility of a simulated setup could be combined with the accuracy of a physical system. Usage examples include the insertion of "virtual" hardware into laboratory experiments, controller hardware-in-the-loop testing and the upscaling of experiments. However, interfaces for coupling power hardware and simulation software are not standardized to the same degree as coupling between simulators is enabled by the FMI standard.
- *RC3: Signal-based synchronization*. Interfacing power system and ICT simulators has become an integral part of simulation-based smart grid research in recent years. Even though challenging, several approaches have been devised that were successful in coupling these very different simulation domains. However, these existing approaches rely on purpose-built coupling solutions which are tied to particular tool combinations. In order to simplify this type of simulation, a standardized and tool-independent approach is required.

2.2 Selection of Domain-Specific Tools

2.2.1 General Selection Criteria

In order to successfully address above research challenges within JRA2, the following list of general selection criteria has been devised:

- *FMI compliance*: Ideally, an FMI-compliant simulation interface should be available for the selected tools. Otherwise, an API (or equivalent mechanism) that allows to control the execution of the tool should be available, such that an FMI-compliant simulation interface can be developed on top of it.
- *State-of-the-art approach*: Each selected tool has to represent the state-of-the-art for its respective domain, such that the final selection provides a representative example that serves as a meaningful proof-of-concept for simulation-based smart grid assessment as a whole.
- *Availability*: Ideally all of the partners should have access to the selected tools in terms of licensing and/or other requirements. This means that open-source solutions would be most preferable, followed by popular tools that represent de-facto industry standards.

Simulations tools from the three domains typically implement very contrasting modelling and simulation paradigms, necessitating domain-specific requirements in addition to the general requirements above. These domain-specific requirements are presented next.

2.2.2 Power System Modelling

Tools for this domain rely on continuous time-based modelling paradigms, typically representing individual components by (sets of) differential algebraic equations. They enable the simulation of the evolution of the system state either with the help of models that depend explicitly on time (RMS and EMT simulation) or by computing a series of subsequent power flow calculations (power flow-based simulation). Within this context, there are two specific selection criteria for power system simulation tools:

- It is required that an (FMI-compliant) interface exists that allows to start and stop RMS simulations at arbitrary times (no restriction to fixed time steps).
- The interface must allow a persistent interaction with the model when stopped, i.e., a modification of the system state must not be neglected when resuming the simulation.

2.2.3 Communication Network Modelling

Simulators for this domain use abstractions of the deployed hardware and software that allows the representation of communication processes as a sequential processing and transmission of (virtual) messages and signals. Hence, communication network simulators commonly implement discrete event-based simulation paradigms, where each event marks a significant step of message processing or transmission. Within this context, there are two specific selection criteria for communication network simulation tools:

- It is required that an (FMI-compliant) interface exists that allows to inquire from the simulator at which point in time the next internal event is scheduled.
- Furthermore, it must be possible to inject new events (i.e., send new messages) between the current co-simulation synchronization point and the next internally scheduled event.

2.2.4 Automation and Control

Tools for automation and control (in power systems and beyond) typically aim at providing easily deployable solutions, which means that apart from the functional aspect (i.e., the control target) also implementation issues have to be considered (e.g., hardware and platform specifications, communication protocols). However, for the purpose of JRA2 basically only the functional context is relevant, as the focus lies on providing an environment for prototyping and assessing control algorithms. Within this context, there are two specific selection criteria for tools for automation and control:

- It is required that an (FMI-compliant) interface exists that allows to issue procedural calls to the tool, i.e., given a certain set of inputs it should immediately return a set of outputs (e.g., computation of set-points as function of measurements).
- The tool should be as flexible as possible in terms of design paradigms and functionality, i.e., it should be a multi-purpose tool that facilitates the implementation of a large range of different algorithms.

2.3 Simulation Tools

2.3.1 Power System Simulation with PowerFactory

2.3.1.1 Overview

DIgSILENT PowerFactory (DIgital SImuLation and Electrical NeTwork calculation program PowerFactory) [4] is a commercial tool for power system design and analyses. It is an engineering tool targeting primarily professional users, enabling the analysis of industrial, utility, and commercial electrical power systems. It has been designed as an advanced integrated and interactive software package dedicated to electrical power system and control analysis in order to achieve the main objectives of planning and operation optimization.

It integrates all required functionality by combining reliable and flexible system modelling capabilities with state-of-the-art algorithms and an advanced database concept. The PowerFactory database environment fully integrates all necessary data, required for defining operation scenarios, variants, single-line graphics, outputs, run conditions, calculation options, graphics and user-defined models of a network section. By saving them within a model (or project in PowerFactory's terminology) the full information required to simulate all defined scenarios at a later stage is available.

2.3.1.2 Compliance to Selection Criteria

PowerFactory does not offer any FMI-compliant model export functionality or co-simulation interface. However, it provides an API that enables basic interactions with a simulation model [5], e.g., setting/retrieving variables and parameters, calculating power flows and starting/stopping RMS simulations. Furthermore, it is possible to issue so-called events during RMS simulations that can change the system state at a specified point in simulation time when executing an RMS simulation. This mechanism can be utilized to change for instance the power consumption of loads or the status of switches, enabling a dynamic interaction at run-time suited for co-simulation. This functionality provided by PowerFactory complies with the domain-specific selection criteria for power system simulation and can be mapped to an interface that is compliant to the FMI CS specification (see below).

Currently there are no state-of-the-art tools for power system simulation available that provide native FMI support. However, given the previous experience and available expertise among a majority of partners within JRA2, PowerFactory complies best with the general selection criteria and has therefore be chosen as the main tool for power system simulation for the purpose of JRA2.

2.3.1.3 FMI Export Functionality

Relying on the functionality explained above (simulation API, event mechanism) the FMI++ PowerFactory FMU Export Utility [6] has been developed. It is a freely available open-source stand-alone tool for exporting FMUs for Co-Simulation (FMI Version 1.0) from PowerFactory models. It uses code from the FMI++ library [7] and the Boost C++ libraries [8] to provide an FMI-compliant interface on top of the functionality provided by PowerFactory. The currently available official release (Version 0.5) only supports steady-state simulations, where the system's evolution with respect to time comprises a series of load flow snapshots. However, a prototype to support RMS simulations has been developed as part of the work carried out for JRA2.

The FMI++ PowerFactory FMU Export Utility provides users with a Python script that creates FMUs from PowerFactory models, including the XML model description and shared libraries. Additional files (e.g., time series files) and start values for exported variables can be specified. A naming convention has been introduced that allows to refer to parameters defined in a PowerFactory model in an FMI-compliant way (e.g., in the model description). This naming convention relies on the concatenation of a parameter's name with its associated object's type and name and similarly for variables related to events.

For the purpose of co-simulation it is necessary to define a notion of time within a simulation model. In PowerFactory there are several ways to do so:

- Models may use externally defined time series (stored in CSV files) and use so-called *triggers* to specify the simulation time. By incrementally advancing time with the help of theses triggers, a series of power flow snapshots can be calculated.
- Models may use scripts written in PowerFactory's scripting language (DPL scripts) to specify the simulation time. By incrementally advancing time with the help of theses scripts, a series of power flow snapshots can be calculated.
- For RMS simulations time is handled directly by PowerFactory. During simulation, external inputs from the co-simulation can be passed to the model as events defined via PowerFactory's DIgSILENT Simulation Language (DSL).

Figure 2.1 shows an example of how a PowerFactory model has to be modified for receiving events from the FMI-compliant interface during co-simulation. The figure depicts a graphical representation of the composite block diagram of the dedicated DSL functions that need to be added to the model. In this example, the blocks on the right (called *ControlledLoad1* and *ControlledLoad2*) represent loads (defined in the network model) whose active power consumption is controlled via the co-simulation interface. The block in the middle (called *FMIEventCall*) receives the values for

the variables associated to the events issued by the FMI interface and feeds them to the loads. This process is triggered by the block to the left (called *FMIEventTrigger*), which fires every time the co-simulation interface send new values.

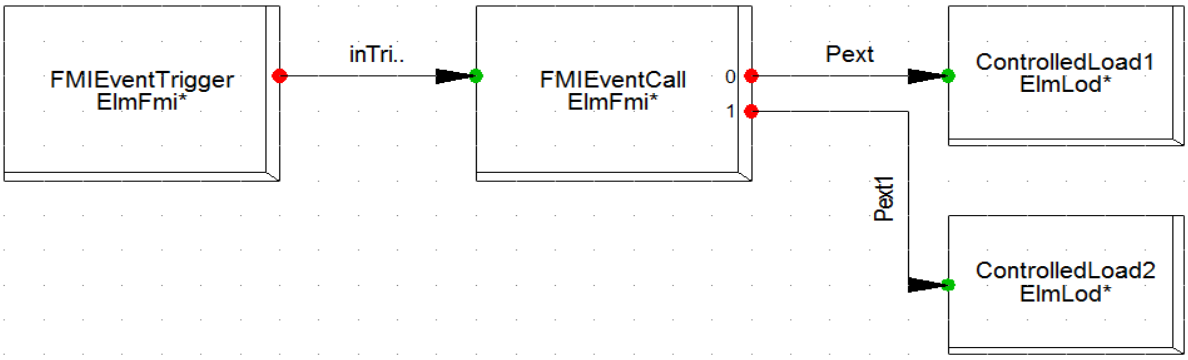


Figure 2.1: Example of a composite DSL model in PowerFactory for receiving events during RMS simulations from the FMI-compliant interface

2.3.2 Electro-Magnetic Transient Simulation with PSCAD

2.3.2.1 Overview

PSCAD/EMTDC [9] is one of the most widely used Electro-Magnetic Transient (EMT) time domain simulation software today. PSCAD is foremost used for modelling and simulations of electrical systems and machines, and their control systems. PSCAD's strengths, in addition to its computational performance and advanced user interface, are in the modelling modularity and ability to model the components using standard library components or user-built model components of desired level of detail. There may arise needs, e.g., to combine PSCAD models of EMT accuracy with larger RMS power system models, power plant process models, or to combine electrical system models with communication simulation models. Co-simulation of PSCAD has been successfully implemented already earlier, e.g., with PSSE [10] by a co-simulation module being available as an extension to the PSSE software suite, and combining wind turbine electrical modelling in PSCAD, with mechanical and aerodynamic simulations in dedicated simulation software via MATLAB/Simulink [11].

2.3.2.2 Compliance to Selection Criteria

Currently, PSCAD does not have any FMI-compliant interfaces available (as of version 4.6.1), but it provides the *Automation Library*, a Python API for interacting with the simulator software[1]. The Automation Library allows full control over PSCAD, such as launching the application, running simulation sets, and accessing/modifying variables and parameters in a simulation model. Additionally, the components in a PSCAD simulation model are programmed using Fortran 95, which enables interoperability with C/C++ languages. Thus, external C libraries can also be used to interact with simulation models.

2.3.2.3 FMI Export Functionality

Based on the above-mentioned Python API and by utilizing the FMI++ Python Interface [12], a tool for exporting FMUs is being developed. Given the PSCAD simulation case file and inputs and outputs as text files, the tool generates an FMI-compliant model description, appropriate Python API function calls and shared libraries, and then wraps them into an FMU to be used with an FMI master algorithm. The implementation will have a similar front-end/back-end structure as was described in the MATLAB section earlier.

[1] For details refer to https://hvdc.ca/knowledge-base/read,article/371/pscad-automation-api-documentation/v:

2.3.3 Communication Network Simulation with ns-3

2.3.3.1 Overview

ns-3 [13] is a discrete-event communication network simulator, targeted primarily for research and educational use. ns-3 is free software, licensed under the GNU GPLv2 license, and is publicly available for research, development, and use. ns-3 is written in C++ and provides Python bindings. Simulation programs are C++ executables or Python scripts.

The ns-3 simulation core supports research on both IP and non-IP based networks. However, the large majority of its users focus on wireless/IP simulations which involve models for Wi-Fi, WiMAX, or LTE for layers 1 and 2 and a variety of static or dynamic routing protocols such as OLSR and AODV for IP-based applications. ns-3 also supports a real-time scheduler that facilitates a number of “simulation-in-the-loop” use cases for interacting with real systems.

2.3.3.2 Compliance to Selection Criteria

Currently, no official FMI support exists for ns-3 (as of version 3.26). However, given the open source nature, ns-3 is highly extensible and can be modified in order to be reused in co-simulation applications in compliance to the FMI specification. ns-3 has already been used in Smart grid co-simulation setups, for instance in FNCS [14], but the coupling between simulators is completed in a custom coordinator specific manner and not via a co-simulation protocol/specification.

ns-3 can be used in order to produce more realistic simulations for the power and control domains by providing realistic delay in the data exchange. Likewise it can also be used by communication network engineers, i.e., in implementing and assessing different communication protocols for smart grid applications. Control engineers can have a more realistic approach of centralized and distributed control algorithms by incorporating the actual network under study and its technological limitations.

ns-3 has a modular structure, and a set of applications can be reused in order to script smart grid co-simulation applications. The simulator engine allows to take control of the simulation execution[2] and new events can be registered to the simulator event scheduler even from the application level. An example of an ns-3 simulation engine takeover is the PyViz live simulation visualizer module[3]. This means that no extra modules need to be developed in order for ns-3 to interact with other simulators. The functionality that ns-3 provides complies with the domain-specific selection criteria for power system simulation and can be mapped to an interface that is compliant to the FMI CS specification.

2.3.3.3 FMI Export Functionality

Due to the open source license of ns-3, the implementation of FMI export is straightforward. New features will be implemented such that the typical workflow of ns-3 stays the same as much as possible.

Currently under development as part of JRA2.1 are:

- A set of ns-3 smart grid application nodes (i.e., smart meters) in order to ease the scripting of smart grid simulation scenarios, called the *smartGridApplicationHelpe*r class.
- A tool to export an FMU from an ns-3 simulation script
- A tool to generate the FMI-compatible binary from the simulation script using the FMI++ library.

In ns-3 the basic abstraction for a node program that generates some activity to be simulated is the application. This abstraction is represented in C++ by the class *Application*. This class provides methods for managing the representations of user-level applications relevant for smart grid applications in simulations. The Application class is specialized in the object-oriented programming

[2] For details refer to https://www.nsnam.org/docs/manual/html/events.html.
[3] For details refer to https://www.nsnam.org/wiki/PyViz.

sense to create new applications. By reusing existing application components such as the *tcpEchoClient*, a node can be scripted to act as a smart grid actor, i.e., a smart meter that when a new event is registered to this node (with a set of parameters) completes a transmission of data over the specified protocol. Moreover, extra nodes generating network traffic that are not directly related to the smart grid application can be added in the communication network model with a set of predefined registered events.

The inputs to and outputs from ns-3 will be no actual messages, but rather message IDs associated to messages. Dummy messages (with configurable size) associated to this message IDs will be used by ns-3 to simulate the processing of the message in the communication network. A dedicated message ID (0) will be used to indicate that no message has been sent from or received at a certain node.

The ns-3 simulation script will be parsed by the FMU export tool written in Python, in order to collect all the smart grid co-simulation nodes and their parameters, and export the appropriate FMU. From the ns-3 simulation script, an intermediate parsing tool in Python will wrap the appropriate C++ functions with the help of FMI++, build the shared library and add it to the FMU, enabling co-simulation with any FMI-compliant master algorithm.

2.3.4 Control Simulation with MATLAB/Simulink

2.3.4.1 Overview

MATLAB [15] is a general purpose numerical computing environment that adheres to no specific modelling paradigm and provides by itself no notion of time. A large variety of toolboxes and extensions is available for MATLAB, which makes it applicable to virtually any mathematical/engineering domain. Maybe the most prominent extension is Simulink, which implements a continuous time-driven simulation environment, utilizing unidirectional block diagrams for a graphical representation of models. Due to its versatility MATLAB/Simulink provides a flexible framework for control algorithm design, as made evident by its popularity and widespread use for this purpose.

2.3.4.2 Compliance to Selection Criteria

Despite the popularity and widespread use of the numerical computing environment MATLAB, there is so far only comparably little support within the context of FMI. The Modelon FMI Toolbox [16] and the FMI Kit for Simulink [17] offer the export of Simulink models as FMUs for Model Exchange, but so far there is no tool available that allows to provide MATLAB's full functionality via an FMI-compliant co-simulation interface. However, MATLAB offers a variety of different ways to interface with it, rendering it is possible to put its full functionality at the disposal of the user via an approach that is compliant with FMI CS (see below).

Due to MATLAB/Simulink's flexibility and widespread use in industry and academia, as well as the availability of interfaces, it complies with all general selection criteria as well as the selection criteria specific to the automation and control domain. This makes the obvious candidate for implementing control algorithms in the context of JRA2.

2.3.4.3 FMI Export Functionality

The MATLAB FMU export has been implemented on top of the FMI++ library (for Windows with 32-bit MATLAB) and is available online. It relies on a concept that comprises two components: A *front-end component* to be used by the simulation master and a *back-end component* to be used by MATLAB. Between these two components a proper data management is established that is responsible for the communication and data exchange between both ends. See Figure 2.2 for a schematic drawing of this concept. The corresponding interfaces are tailored to suit the requirements of the FMI specification and they implement the necessary functionality required for a master-slave concept, i.e., synchronization mechanisms and exchange of data.

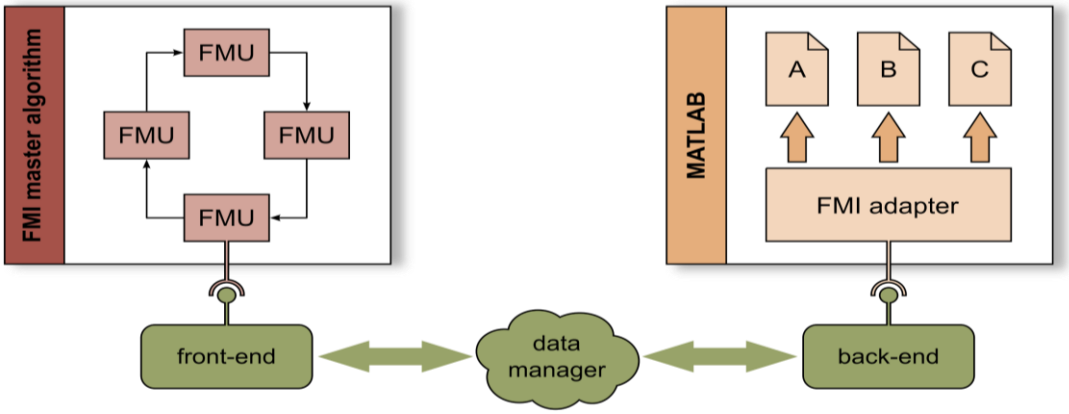


Figure 2.2: Schematic view of the front-end/back-end mechanism used for the MATLAB FMU export

Since MATLAB is a multi-purpose, multi-paradigm computing and programming environment, there are potentially many possible ways to implement an FMI adapter. Here, the SWIG tool has been used to create S-Function bindings to a generic back-end implementation in C++ that can be called from within MATLAB. Even though these bindings can be used directly from MATLAB scripts, it is recommended to utilize their functionality through the dedicated class *FMIAdapter*. In order to utilize its functionality, two abstract methods have to be implemented by a derived class. Since MATLAB defines itself no general notion of time, the current communication point and communication step size are provided as input arguments when calling the FMI adapter at a co-simulation synchronization point. Since the FMI adapter implementation may contain any MATLAB-compliant code, virtually any MATLAB functionality can be made available with the help of this concept.

2.4 Co-Simulation Master Algorithm

2.4.1 The Co-Simulation Framework mosaik

2.4.1.1 Overview

Mosaik [18] is a smart grid co-simulation platform that has been developed at the Oldenburg Institute for Information Technology, OFFIS, in cooperation with the University of Oldenburg [19]. While it originally has been designed as a testbed for multi-agent control strategies, steady development has transformed it into a more versatile framework. A major focus of the mosaik design is high usability. Compared to other co-simulation platforms in the smart grid domain, like HLA or Ptolemy-based systems, the mosaik software is relatively easy to deploy, and facilitates the integration of new simulators as well as the creation of co-simulation experiments. This is achieved via a light-weight software core based purely on Python, a special *Component-API* for simulator integration, and a *Scenario-API* for flexible simulator coupling.

The Component-API is an interface that has to be implemented for each simulator that is to be integrated into the mosaik environment. There are different API versions to make the integration process as straightforward as possible for a large variety of users and simulation tools. The so-called “low-level API” is based on a TCP connection between mosaik and the simulator. It employs message exchange in the JSON format. For greater convenience, several high-level API versions have been established for popular programming languages like Python, Java or MATLAB. They handle the creation of TCP connections and JSON formatting “under the hood” and thus may be employed even by model developers with little general programming knowledge. A number of best practices is provided for integration of simulators whose runtime environments do not correspond to one of mosaik’s APIs. FMU-based components, e.g., are supported via a generic, Python-based interface that utilizes the FMI++ library [7]. Finally, it has to be noted that the Component-API does not only support the integration of simulators, but may also be used as an adapter for components like database and data analysis systems.

Simulators integrated in the mosaik environment may be used by smart grid researchers to set up co-simulation scenarios. This is usually done by writing a scenario script that specifies the parameterization and interconnection of models as well as the simulation time. Executing the script in Python runs the co-simulation. The already mentioned Scenario-API facilitates the scripting process by providing a concise set of commands for scenario design. If the simulator in question supports it, multiple entities may be instantiated from the same simulation model in order to realize large-scale co-simulation experiments. Connection rules may be defined to interconnect such large entity sets in complex setups. Based on simulator interconnections, mosaik automatically establishes a data dependency graph that is used to coordinate data exchange and prevent deadlocks during the simulation process.

Co-simulation execution in mosaik is managed by its scheduler module. It utilizes the previously created data dependency graph to advance integrated simulators in time and organize the data exchange between them. The provided scheduling is discretely timed and does not allow for iterative data exchange within the same time step, i.e., mosaik follows an explicit coupling approach. This design decision has been made in order to support a variety of black-box simulators. There are alternative co-simulation platforms that provide more versatile scheduling, e.g., based on Ptolemy II [20] or the High Level Architecture (HLA) [21]. However, these tools typically come with a large overhead in their handling and the integration of new components. Therefore, mosaik presents an appropriate choice for co-simulation that serves as a proof of concept and a collaboration platform. The framework's high usability is well suited for interdisciplinary research projects with many partners that are not co-simulation experts.

2.4.1.2 Compliance to Selection Criteria

Next to its high usability, mosaik also fulfils all tool selection criteria set up within the project. As mentioned above, FMU-based components may easily be integrated into mosaik co-simulation via the use of the FMI++ library. In fact, the interoperability of mosaik and FMI has already been demonstrated in [22]. In the ERIGrid project, the coupling between the two technologies has been updated for new versions of mosaik and FMI++. Support is now given for both, FMI for Co-Simulation as well as FMI for Model Exchange. The mosaik framework, furthermore, possesses a state-of-the-art character as a relevant co-simulation tool in the smart grid domain. This is reflected by its active development as well as novel research projects that employ the framework. Finally, mosaik is an open-source software product hosted by OFFIS so that it is freely available by all project partners.

2.4.2 Extension of mosaik: Cyclic Data Dependencies

Since the mosaik platform is still under active development, some new features have been introduced based on requirements of the ERIGrid project. One of the most important features in this context is an improved handling of cyclic dependencies between simulators. If two coupled simulators require data from each other within the same time step, this can easily lead to deadlocks that block the whole simulation process. Therefore, a coupling scheme has to be employed that assigns a specific order to such a cyclic data exchange. In general, two types of schemes are possible in explicit simulator coupling. Given two simulators A and B with a cyclic data dependency, the two options are the following (see also Figure 2.3):

- In a serial setup, the simulator A executes a simulation step first. It receives a pre-defined, initial input and is advanced in time. Its output is then provided as input to simulator B that also executes a simulation step. With the output of B, new input can be provided to A so that it may be stepped again, and so forth.
- In a parallel setup, both simulators A and B receive initial input at the beginning of the simulation so that they can execute their simulation steps in parallel. The output each of them provides is then used as input for the next time step of the other simulator.

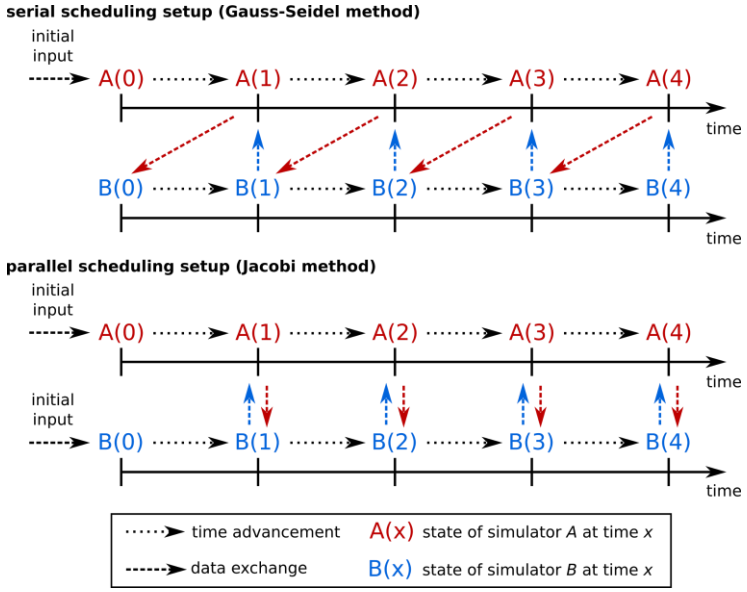


Figure 2.3: Comparison of coupling approaches for simulators with cyclic dependencies

Mosaik has so far followed a rather simplistic approach to cycle handling. In order to prevent deadlocks during the simulation process, mosaik's scheduler detects cyclic dependencies in the scenario script before simulation start and forbids them. Instead, users should employ so-called "asynchronous requests" that essentially realize a serial coupling scheme. In order to utilize such a request, the interface to one of the simulators has to be specifically adjusted. This overhead does not fit with the overall usability-oriented design of mosaik. Furthermore, parallel coupling schemes as well as cyclic dependencies between more than two simulators are not supported in this setup. Therefore, an adjustment of mosaik's scheduler module has been made within the ERGrid project in order to solve these issues. The basic idea behind this adjustment is the notion that cycles are resolved via a shift in time in serial as well as in parallel schemes. In other words, data that is produced in one time step serves as an input for the next time step of another simulator. In a serial setup, one of the connections is "time-shifted" in this sense while in a parallel setup, all connections are. In the adjusted version of mosaik, a second data dependency graph is introduced that organizes all the time-shifted connections (while the original graph still organizes all other connections). Note that it is the duty of the user to specify in the scenario script whether connections require a shift in time. Furthermore, initial input data has to be provided for these connections. Everything else, however, is handled by mosaik. This way, mosaik users may realize serial as well as parallel handling of arbitrary numbers of simulators with cyclic data dependencies. All necessary specifications are part of the scenario creation process so that no simulator interface has to be adjusted.

2.4.3 Extension of mosaik: Discrete-Event and Continuous-Time Simulation

As mentioned before, mosaik has been designed primarily for Discretely Timed (DT) simulation. This may present some challenges for the creation of smart grid co-simulation studies since several types of system components are represented more correctly by other simulation paradigms. As one major example, simulation of physical systems, like power grids, is typically associated with Continuous Time (CT) representation, i.e., for any given point in time, the simulation tools should be able to receive and provide values. Simulation of communication systems, on the other hand, is mostly associated with Discrete Event Simulation (DES) that has input and output being required or provided at varying points in time. Since such simulation models are to be considered in the ERIGrid project, a setup has been developed that allows coordination of CT and DES tools within the scheduling provided by mosaik.

A common simplification of DT simulation is "fixed-step" simulation. This means that the size of a simulation step, i.e., the time period simulated in one step, is always constant. This notion strongly

conflicts with DES since events typically do not occur on a regular basis. CT systems, similarly, tend to present varying rates of state changes that also cannot be expressed properly with fixed simulation steps. In order to facilitate support of DES and CT simulators, mosaik has been designed without a fixed-step constraint. Instead, when a step of a simulator is executed, it informs mosaik about the point in time for its next execution. While this concept improves the handling of a single DES component, a challenge is still given by the coupling of several DES and CT simulators, especially in the context of cyclic data dependencies. This is an issue since different simulators may require data input and provide output at different points in time. Mosaik tries to solve this problem by reusing old data in the exchange between simulators. However, depending on the simulators, this may lead to accumulating coupling errors in the simulation and thus high output uncertainty. Instead, a simple workaround is employed in this project. Simulators exchange time information with each other so that the generation of events in one simulator may also determine the choice of step sizes in another one. As an example, a DES communication simulator is executed in a serial setup before a CT power grid simulator. If the communication simulator can anticipate the next point in time an event occurs, it can provide this information via mosaik to the grid simulator. This component then adjusts the next time step accordingly so that both simulators reach the next synchronization point. This workaround obviously is not an option for any given simulator. However, when applicable, it can be realized purely via adjusting the simulator interfaces and requires no changes to mosaik's scheduler.

A data synchronization issue that typically occurs when coupling CT simulation components with other simulators is the fact that CT systems may require or provide data at any given point in time. Other types of simulators, in contrast, may lack current data values for a given point in time. Reusing old values, as mentioned before, leads to potentially erroneous calculations. On other hand, if no value at all is provided although one is expected, the complete simulation is likely to crash. A workaround for this issue is presented by the introduction of message IDs that are exchanged between simulators. In this case, a special kind of ID is interpreted as a lack of current data values for a given attribute. A simulator interface receiving such information must possess a way to deal with this situation, e.g., by postponing the simulator's next execution. The concept of message IDs also benefits the co-simulation of communication systems. The IDs allow simulators to express not only which data values are transmitted but also how they are transmitted. Obviously, the interfaces of the employed simulators are adjusted in a way that allows them to interpret different ID types.

Finally, issues may occur in the handling of DES systems in mosaik related with the discrepancy between time advancement and input processing. For mosaik, these two actions are always combined. In DES, on the other hand, advancing the system state to the next event and processing other events may be distinct from one another. This problem is circumvented via the usage of FMI for interfacing. FMI's *doStep* function that is normally mapped to mosaik's *step* function can be called with a time step size greater than 0 for time advancement, but also with a step size equal to 0 for input processing without time advancement. These options allow for an adjusted FMI-based interface for mosaik that is more compatible with DES components.

2.5 Virtual Lab Components

2.5.1 Relevance of Hardware-in-the_Loop Simulations

Power Hardware-in-the-Loop (PHIL) and Controller Hardware-in-the-Loop (CHIL) have gained some importance in developing and testing of smart grid components and systems. In such a setup real hardware under test or controller under test are coupled with simulated components. Like pure virtual simulations, HIL setups also suffer from an increased implementation overhead due to various interfaces. Hence, standardized interfaces drastically decrease the implementation effort of PHIL and CHIL setups. In the context of smart grids, standardized virtual lab component interfaces enable the efficient validation of new technologies and products. More often than not, it is not feasible to build the entire context of smart grid components as dedicated test hardware. Efficient interfaces to virtual lab components are one step towards comprehensive test and validation procedures.

Common interfaces to virtual lab components also drastically ease access to laboratory and research infrastructure. A virtual component which was developed for one laboratory environment does not have to be manually ported to another one. Likewise, the effort which is required to adapt the laboratory infrastructure to include new virtual components is drastically reduced or eliminated at all. Tedious coupling work which often includes the development of tool specific interfaces is reduced to some configuration actions.

By using the FMI to interface virtual lab components and hardware, one step towards the harmonization of modelling and interface methods is taken. Thereby, it is not necessary to develop the coupling strategies for every test-case anew but to utilize existing, well proven algorithms. Hence, PHIL and CHIL setups can focus on domain issues without having to mind implementation details in coupling the components.

2.5.2 Coupling the IEC 61499 and the FMI

The interaction of real laboratory equipment, automation controllers and virtual components imposes several major challenges. Hardware components naturally operate in real-time only. It is usually not feasible to control the notion of time of external laboratory equipment without altering its dynamic behaviour. For instances, a controller may be artificially halted to synchronize with simulation time but the controlled physical process may not. Some approaches introduce the notion of scaled real-time but these techniques also require virtual components to synchronize with a scaled variant of real-time. Hence, time management in virtual lab components is different from synchronization of pure virtual simulations but also requires careful consideration to gain accurate and reliable results.

It is often distinguished between soft and hard real-time operation. In hard real-time operation, it must be guaranteed that certain timings such as the response time of a controller are met. Any timing violation may have catastrophic consequences such as injuries or the loss of property. In soft real-time operation, timing violations may cause inaccurate results but the system is engineered in a way such that no catastrophic consequences can happen. Clearly, hard real-time operation requires engineering practices of all involved components which are not feasible in the development of most simulation software tools. Hence, within the context of work package JRA2 a soft real-time operation of virtual lab components is targeted and the physical laboratory equipment must be capable of tolerating timing violations. Since there are no hard real-time guarantees, interface software must monitor the timing and any timing violation of virtual lab components to assess the quality of the result.

The difference in representing varying data between the FMI and the IEC 61499 [23] (distributed control approach used also as laboratory automation system – e.g., in case of AIT's SmartEST lab) introduces another challenge which needs to be considered properly. In the IEC 61499, data is linked to events which trigger the execution of a control algorithm. Whenever an event occurs, the corresponding control algorithms may be executed and data may change. The primary purpose of events in the IEC 61499 domain is to trigger control action. The FMI introduces a different data representation and notion of events. Events in the FMI domain mark discontinuities of otherwise continuous but not necessarily constant variables. An approach which couples IEC 61499-based lab components and FMI-based models must translate the data representations while maintaining the postulated accuracy.

Several feasible approaches were identified to couple virtual and physical lab components. Each approach has its own merits and drawbacks which need consideration. In theory, an IEC 61499-based controller may passively provide an FMI, either for model exchange or for co-simulations. Such an interface could be included into a simulation which uses the functionality of the controller in a more complex setup. Since the progress of simulation time is controlled by the external solver or master algorithm, the runtime environment which executes the control program would have to be modified to synchronize its event queue with the external masters. Real-time operation, in general, is problematic because the timing is controlled by an external entity which is usually not destined for real-time operation. One can see that the integration of external hardware which does not allow con-

trolling timing aspects externally, leads to inaccurate solution on using the FMI passively. Hence, the passive approach is expected to lead to suboptimal results and will not be further discussed.

Another way is the inclusion of FMI-based components into IEC 61499-based systems. Such an inclusion directly leads to virtual lab components which are represented by connected FMUs. Virtual lab components may either actively use FMUs for CS or ME. An interface component must provide a master algorithm or solver to handle connected FMUs. In contrast to passive usage, an interface component may utilize algorithms which feature a soft real-time operation. Still, the timing of included models or simulation tools may not be entirely predictable but the algorithms may be optimized to fulfil the real-time requirements according to best effort strategy. In case of real-time operation, the IEC 61499 runtime environment may not have to be modified which drastically decreases the implementation costs.

One of the simplest and most widely supported coupling approaches is a strictly periodic operation of included FMUs. The basic algorithm may be applied to both, co-simulation and model exchange. Data between the FMUs and the controllers is exchanged at discrete, periodic points in time only. Whenever an event occurs, it will be delayed until the next scheduled synchronization point. Hence, any discontinuity in a signal meant to be exchanged between the controller and the FMU is delayed. Especially, if one event triggers immediate action, the response will be delayed by one step. Similarly, intermittent states of continuous variables will not be transferred to the controller, until the next synchronization point. In order to keep numerical errors which are introduced by the algorithm within a reasonable bound, it is essential to carefully choose the step size according to the needs of the particular experiment.

The periodic operation integrates well in many closed-loop control systems. These systems often sample the sensors of a plant and output results to the actuators at fixed intervals. Nevertheless, a periodic operation does not fully utilize the capabilities of IEC 61499-based controllers. IEC 61499 features an asynchronous operation which is controlled by events which are not necessarily tied to fixed instances of time. Hence, prediction-based approaches were developed for work package JRA2 that feature both an immediate processing of events and a strictly periodic operation.

The basic idea of all prediction-based approaches is to calculate expected outputs of a model or simulation in advance. On detecting an event or a significant change in the calculated outputs, these changes can be issued to the IEC 61499-based controller in time or with minimal delay. The prediction is made under the assumption that the inputs which are issued by the controller and connected hardware stay constant. If the IEC 61499 or any connected hardware issues an event to the model or simulation, pre-calculated results become invalid. In such a case a rollback operation until the time of the issued event is performed. Predictions beyond the time of the most recent event need to be nullified and recalculated.

Frequent recalculations increase the computational effort but allow reducing the delay of events to a minimum. Only if the time between two events is too short such that necessary computations cannot be fully performed between the events, real-time performance is degraded. As in the periodic approach, monitoring is used to detect these delays and to judge the temporal quality of the outcome.

One critical aspect in all prediction-based approaches is the ability of the simulation models to roll back to a certain point in time. The roll back mechanisms, if any, differ between the FMI for co-simulation and the FMI for model exchange. One approach which targets the FMI for model exchange stores predicted states of the model until an event is detected or a certain prediction horizon is reached. On receiving an external event, the state at the time of the event is interpolated and the event is processed. Another approach targeting the FMI for co-simulation relies on resetting the very last communication step. Since the prediction-based approach for co-simulation highly depends on some optional FMI features, it may not be supported by all connected tools. A periodic approach may be used as fall-back strategy instead.

2.5.3 Implementation

An interface component which integrates FMUs into IEC 61499-based controllers could either be implemented by extending an existing runtime environment or by implementing a standalone application which communicates with IEC 61499-based controllers. Extending an existing runtime environment allows configuring the coupling component via the configuration interfaces of the environment itself. Since the number of inputs and outputs is not constant across all FMUs, the configuration of the controller must be highly flexible to adapt to various models. Additionally, extending a particular runtime environment would hinder a migration from one environment to another. Updates in the source code structure of the extended runtime environment will most probably also affect the interface component. In order to gain flexibility and maintainability, it was chosen to implement the interface component into an external application. The external application is interfaced by means of standard communication function blocks as defined in the IEC 61499.

The application entirely relies on standard-based communication which increases the range of connected controllers. Although the interface component is designed to fully leverage the event processing capabilities of IEC 61499-based controllers, also lab infrastructure which does not fully conform to IEC 61499 may be coupled. The application is designed in a way such that multiple communication protocols can be handled in parallel. Hence, the communication is not restricted to the ASN.1-based communication protocol which is included in the IEC 61499.

A central message queue-like storage structure manages external events received by connected controllers as well as events which are generated by the included model or tool. As soon as new external events are received, the event will be distributed and any pending predicted event will be invalidated. A dedicated timing interface was created for the purpose of JRA2, which logs the timing parameters of each event in detail. The interface allows detecting any late event and may be used to assess the reason of any deviation. A central configuration interface allows a user to parametrize the application in a uniform way. Besides the mapping of FMI variables and communication channels, also data types and data conversions may be specified in detail.

Currently, the integration of FMUs for model exchange into the interface application and the timing interface are fully functional. The interface application is able to perform data type conversions and reliably send simulation results to an IEC 61499-based controller. Closing the loop still requires some attention to the networking stack.

3 Smart Grids Model Library

3.1 Introduction

To complement and expand the functionality provided by the FMI-based co-simulation, a dedicated Smart Grid library of different models has been developed within the ERIGrid project and presented in this section. The developed Model Library (ML) provides a set of models that have been carefully selected and developed for validating and accelerating the adoption of new smart grid solutions. To increase the potential use of the library within a wider smart grid community and meet the ERIGrid key objectives, three simulation domains have been considered: Power Systems, Communications Network, and Controls. The choice of the models across these three domains is driven by their importance, supported by the shared interest and widespread expertise of the ERIGrid partners.

The selected models have been developed and exported as FMU compliant to the FMI-ME specification to allow their tool-independent implementation and facilitate their reusability within different simulation tools (without rebuilding the models for each specific tool). All the library models were originally built either in MATLAB/Simulink or OpenModelica (the tools that allow to export models as FMUs for ME).

The ERIGrid ML has demonstrated the added value of FMI-ME for extending the functionality and capability of simulation tools to accept models that have been originally built in different simulation domains and by different tools. This will potentially accelerate the adoption of FMI-ME within the power industry, saving time and resources for model development and testing.

The details and description of the models development methodology, model justification and implementation, and models quality and validation are presented in the following subsections.

3.2 Methodology

The four staged methodology, which was adopted within this task is presented in Figure 3.1.

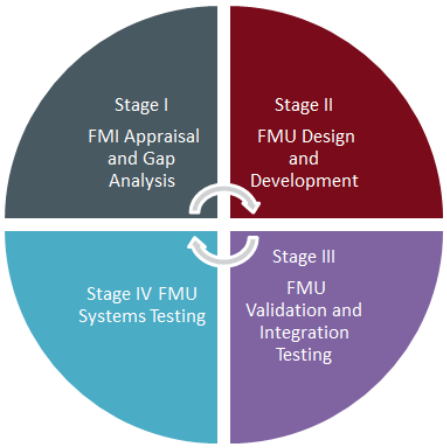


Figure 3.1: JRA 2.2 Methodology

3.2.1 FMI Appraisal and Gap Analysis

The first stage of the ML development involved two parallel sub-tasks with two key objectives: *(i)* to appraise the use of FMI-ME within smart grids and *(ii)* to identify the gaps in existing smart grid models.

The first sub-task involved understanding the state-of-the-art FMI-ME applications and practices within smart grid literature. From a comprehensive literature review, two power system simulation tools that support FMI-ME were identified: MATLAB/Simulink and OpenModelica. Exporting models

as FMUs is an inherent feature of OpenModelica, however, an additional toolbox is required for exporting FMUs from MATLAB/Simulink. Two toolboxes for MATLAB/Simulink (toolbox from Modelon and toolkit from Dassault Systems) were evaluated. The performance of FMUs from both the toolboxes was identical. As the FMI Toolkit from Dassault Systems was not available within the consortium, the toolbox from Dymola was chosen for use within the task. A more detailed analysis on the performance of the two toolboxes can be found in Appendix B.

The second sub-task involved identifying the models that would be developed within the task. The identification of models was driven by *(a)* the requirements of the test cases within JRA 2.1, *(b)* the models that would facilitate development and validation of novel control solutions, and *(c)* the shared research interests of the partners. In total, fourteen models were identified: seven for power systems, two for communications and five for the controls domain.

3.2.2 FMU Design and Development

The selected set of models across the aforementioned three domains was developed within the second stage of the methodology. The models have been broadly classified into two categories: *(i)* existing models that need to be extended/adapted as tool-independent models to allow their easy utilization by the wider community (for example, the converter control model) and *(ii)* the models that were identified to be missing and their development is required to facilitate the validation of smart grid solutions (for example, accurate measurement and communications models). It was of utmost importance to develop models as generically as feasible to allow their widespread use. This involved incorporating simple steps within the design stage such as adopting per unit (p.u.) convention, adding parameters that presented flexibility etc. Each of the models was developed by one or more partner based on their common research interests.

3.2.3 FMU Validation and Integration Testing

All the developed models have been thoroughly tested for their representative behaviour by the model developers. The interoperability of the exported FMUs was tested by means of integration tests as shown in Figure 3.2 below. The FMU exported from MATLAB/Simulink was imported within OpenModelica and vice-versa for testing the model performance. Furthermore, each FMU was also tested within mosaik. The imported FMU performance is verified by means of a predefined input-output relationship (obtained within simulation tool utilized for model development).

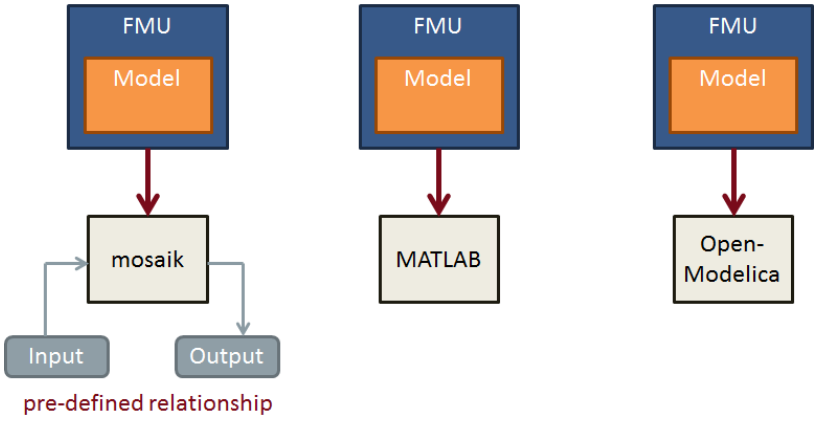


Figure 3.2: Integration testing

3.2.4 FMU Systems Testing

The interaction between multiple FMUs by means of system tests is out of the scope of this report, and with the recently enhanced capability of mosaik to import FMUs, such tests will only be conducted within the mosaik framework as part of future work within the ERIGrid project. The differ-

ence between integration tests and systems tests have been presented in Figure 3.3. Such tests will enable the definition of the envelope within which the developed models can be utilized for the purpose of smart grid solutions validation by the wider community with confidence.

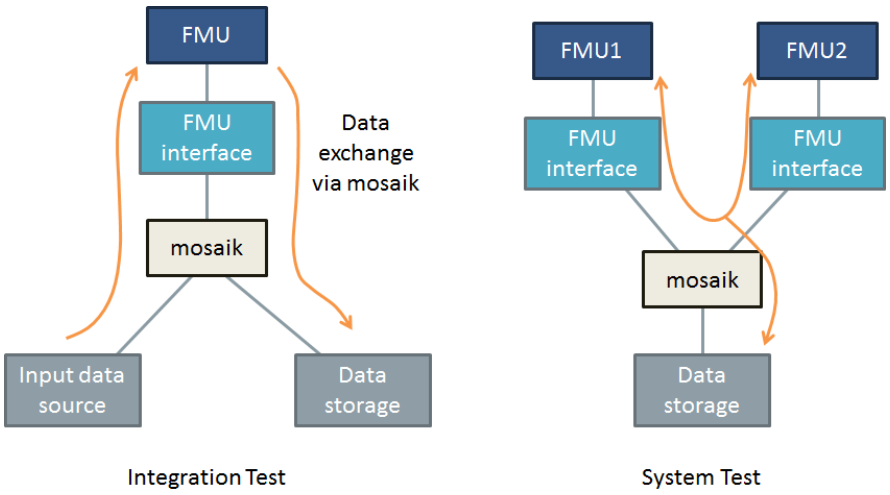


Figure 3.3: Integration test vs. system test

Each of the models developed is supplemented with detailed documentation. The documentation serves as a guide for future utilization of these models, explaining the mathematical basis of the model, the inputs, outputs and parameters of the models, its integration within simulation tools and its potential applications. Exported FMUs do carry certain dependencies on the platform within which they are developed and therefore FMUs for Windows 32-bit and 64-bit are made available.

3.2.5 Model Selection and Implementation

Modern power systems are continuously evolving and adapting to ensure stable operation under a myriad of required transitions such as the incorporation of high penetration of renewables and advanced energy resources, the necessity to supply more heat and transport demands, and the deployment of decentralised and demand side controls. Understanding the impact of such changes on power systems and quantifying the interaction between new advanced controls and active devices will require more sophisticated and representative models that allow for such an analysis to be undertaken.

The selection of the developed models is governed by the ERIGrid JRA2 test case requirements and the need for the provision of a variety of advanced and flexible models that can be utilised to enable testing and validation of a range of new smart grids technical solutions. The selected models cover different electrical, measurement, control, and communication aspects. These include the models of interface devices such as power converters and their associated controls (for interfacing PVs and wind), dynamic load models (e.g., dynamic heat demand), different distributed energy resources (such as PV, Wind turbine, battery storages), and a representative distribution network. Such a powerful combination of FMI-ME compliant models will allow addressing the limitation of testing systems incorporating different technologies from different technical domains on a common platform.

The description and potential implementation of different models across the three selected technical domains; power systems, communications, and control are presented as follows.

3.3 Power Systems Simulation Domain Models

A number of models specific to power systems have been developed as FMUs compliant to FMI-ME, including a reduced-order distribution dynamic equivalent model, FlexHouse thermal model, PV and battery models, and a wind turbine mechanical model. This set of tool-independent models can potentially be used for different simulation studies and proof of concepts.

3.3.1 Reduced Dynamic Equivalent Model of Dynamic Power System Laboratory (DPSL)

Modelling of detailed power networks is a complex task and it is often time consuming with limited flexibility to run a wide range of studies. Development of larger network models is also limited by the simulation tools' ability to compute a solution in an acceptable amount of time. Therefore, reduced equivalent models are of practical importance when large scale simulations need to be undertaken.

For this purpose, within ERIGrid, a reduced dynamic equivalent model of a selected configuration of the state-of-the-art smart grid laboratory at University of Strathclyde, Dynamic Power System Laboratory, has been developed. The model has been developed based on an accurate load measurement modelling technique.

Such models can be readily utilized within voltage and small-signal stability studies where large scale simulations are necessary. Furthermore, for studies that require Monte Carlo simulations to be undertaken, such load model helps reduce the simulation time.

3.3.2 Simplified Sodium Sulphur Battery Model

When the amount of intermittent renewable generation increases, the operation of the electrical system changes significantly. Energy storages are one means to balance the consumption and production in the system and are, therefore, one important part of smart grids. Different energy storages operate on different time scales and have different characteristics and limitations. Being able to simulate their operation in an adequate detail is vital for smart grid testing.

Within the ERIGrid project, a model that can realistically emulate one type of energy storage, sodium sulphur (NaS) battery, has been developed. NaS batteries are typically applied for larger scale storage applications, and their models are not currently available in the usual electrical system simulation programs (e.g., PSCAD and MATLAB/Simulink).

NaS batteries have some characteristics that need to be taken into account in the model. NaS internal resistances depend strongly on temperature and battery State of Charge (SOC). In addition, the internal resistance is likely to increase with the number of charge-discharge cycles. Internal resistances mainly define the output power of the battery. Battery temperature and input voltage coming from the DC/DC-converter controlling the battery are given as inputs to the model and based on these values, the developed model calculates the internal resistances, output DC current and battery SOC. The battery rated capacity, initial SOC and number of use cycles are given as constant input parameters to the model. The developed NaS model can be utilized as a part of an electrical system simulation when detailed modelling of storage characteristics is needed.

3.3.3 Detailed Steady State Battery Model

This model is developed as an additional battery model with more details in comparison to the previous NaS model. The electrochemical characteristics of a battery are represented by an equivalent electrical circuit model incorporating aging characteristics and few basic Battery Management System (BMS) limitations such as SOC and voltage. The model also takes into account the power lost in the auxiliary systems of the battery like for temperature control with ventilation, pumps to circulate the electrolyte etc. It also gives the maximum charging and discharging power available in the battery for a given operating condition. This is a steady-state model originally developed in MATLAB/Simulink.

This model can be potentially used to study the electrical behaviour (current, voltage, SOC, power etc.) of the battery with consideration of calendar and cycle aging characteristics. Along with this, the basic safety limit indicators as that of BMS are also signalled to the user if and when the battery safety limits are surpassed.

3.3.4 Dynamic Thermal PowerFlexHouse Model

Demand response will be a key element in future smart grid systems. Although demand response will come from many consumption sources, given that more than 50% of the global final energy consumption is used for heat production [24], it is relevant to model units that consume electricity to produce heat.

Within ERIGrid, a linear and deterministic model of the heat dynamics of the PowerFlexHouse has been implemented. The PowerFlexHouse is an experimental building within the Technical University of Denmark, and the original model has been derived in [25]. It represents the heat dynamics of the building through an equivalent electrical circuit model, and the parameters of the model have been identified through experiments.

The model can be used for analysis of demand response potential in households. Specifically, it can be used to test/simulate control algorithms for smart heating in houses (individual control) based upon the interior temperature (user comfort). Also, by instantiating several house models, and slightly varying the model parameters, it can be used to test aggregation algorithms.

3.3.5 Photovoltaic Model

The developed PV model within ERIGrid is based on the interpolated model of PVs and represents the operation of one single PV module. The basic advantages of the selected model is that it makes use of a set of input parameters known by manufacturers' datasheets and is scalable, allowing for modelling of large-scale PV plants.

The developed PV model provides accurate representation of instantaneous voltage and current, enabling it to be coupled with transient inverter models for assessment of electromagnetic transients and controller behaviours (e.g., MPPT, droop control and virtual inertia). The developed model is appropriate for use within LV and MV distribution grids. The developed model is flexible and can be utilized within simulation tools that assess environmental conditions such as solar potential of geographical areas. Furthermore, with the extensive use of PV panels as building elements for the energy efficient and sustainable buildings it is possible to use the FMU in combination with tools that analyse the energy performance of smart buildings. It is evident that all the above mentioned applications are multidisciplinary and may require the use of different simulation tools, therefore, an FMU emulating the operation and performance of PVs is a valuable addition to the ML.

3.3.6 Inverter Model for PV

PV inverter is a critical component to interface and synchronise the variable DC outputs of the PV to the AC grid. Important functions such as maximum power point (MPP) tracking and protection can be implemented by the inverter. JRA2.2 developed a simplified inverter model represented as an injected current source. The model takes into account important factors of a real inverter such as: the delay to start the inverter, the conversion efficiency, and the protection parameters. The advantages of such considerations allow the modelling and simulation of power generation of large-scale PV plants.

The developed PV inverter model can be used for simulation studies to analyse voltage deviation within local distribution networks which can be caused by the connection of PVs and to evaluate frequency stability issues in micro-grids with a high share of PV generation. In addition to this, the developed model is scalable and can be considered for small and large scale PV systems. The model is also valid for the grid interface of other sources with constant/variable DC outputs.

3.3.7 Wind Turbine

The developed mechanical model of a wind turbine is scalable, modular and enables the user to connect, test and validate different variable speed generators such as a Permanent Magnet Syn-

chronous Generator (PMSG) or a Double Fed Induction Generator (DFIG). Comparison between different generators could provide valuable insight into the operation of the overall Wind Turbine. Furthermore, the model could be potentially used for testing different control algorithms for pitch angle control in order to address the criteria for optimal operation of the wind turbine. In addition, dynamic, steady state phenomena and extreme conditions for the wind turbine could be examined. Generally, the model (connected to a proper electrical model) provides the user with the capability to study the impact of wind turbine penetration on the grid and the respective issues that could occur, as well as, to study the provision of ancillary services by wind turbines.

3.4 Controls Simulation Domain Models

Smart controls with accurate measurements and intelligent ICT play fundamental roles in automation and optimal power system operations. Within this simulation domain, JRA2.2 has developed the following models: an intelligent control model which can be applicable to transformer On-Load Tap Changers (OLTC) to address voltage saturations that can potentially be caused by the connection of large volume of DERs; a scalable and adaptable wind generation grid-side converter controller to control and optimise the extracted power from the wind during normal operation; and a discrete flexible Fault Ride Through (FRT) controller to control the wind generation converter under faulty grid conditions and provide internal protection. In addition, two models of different type of Phasor Measurement Units (PMUs) to provide reliable and accurate measurements are developed. The following subsections provide more information on these models.

3.4.1 Convertor Controller Model

A large number of DERs are nowadays interfaced by power converters as the main generation units work at a variable frequency or inherently generate DC. Regardless of the specific power source, the interface with the grid requires DC/AC conversion with a common control strategy of the converter. DC/AC converters can significantly enhance the controllability of the system and address a wide range of technical challenges in smart grid applications. The convertor control is one of the main components enabling the converters to meet these requirements.

Within JRA2, a DC/AC converter controller is selected and developed. The implemented control method is based on vector control in d-q axes reference frame with Sinusoidal Pulse Width Modulation (SPWM) technique. The vector control allows decoupling of the active and reactive power control in order to provide various grid services. In particular, the provided controller model manages the active power to control the DC bus voltage and the reactive power to control the AC bus voltage at the Point of Common Coupling (PCC).

The converter model is further provisioned to receive signals from the FRT controller developed within the project, to provide a better response during grid fault events.

3.4.2 Fault Ride Through (FRT) Controller Model

Converters commonly exhibit non-linear and/or discontinuous behaviour during faults. The interaction between the electricity network and converter-interfaced devices is one of the main driving forces behind scrutinizing the cyclic-dependency issues in the ERIGrid Test Case (TC1). To fully utilize the cascaded nature of vector controlled converters, such as commonly applied in wind turbines and HVDC stations, it is considered practical to separate the implementation of the grid code requirements and the converter’s controls as much as feasible. Hence, in JRA2.2, it was decided to develop separate converter and FRT controller FMUs. The FRT Controller is a discrete controller, implemented as a finite state machine, which very flexibly enables the detection of grid fault conditions, engages internal protection mechanisms, and adjusts control switches and parameters accordingly. The FRT Controller is designed to operate on top of the converter controller, which is developed for TC1 and is also available as an FMU. However, the underlying functions, states, and their transitions are very easily tuneable for other intelligent electrical devices that employ a similar, cascaded control scheme.

3.4.3 On-Load Tap Changer Controller (OLTC)

Within ERIGrid, an OLTC controller used for testing voltage control of a distribution network within an industrial research centre, namely the Ormazabal Corporate Technology Center (OCT), has been modelled and made available as an FMU. The OLTC controller model was chosen to investigate the capabilities of FMI to support hardware in the loop simulations (compare with TC2). With the availability of a smart transformer at Ormazabal Corporate Technology, modelling the smart transformer controller as an FMU for ME allows for exploring the limitations of such hardware in the loop simulation setups.

The controller model can also be utilized for offline simulation studies that assesses the impact of intelligent controls on the degradation of power transformers or be utilized to benchmark the performance of novel coordinated voltage control algorithms (for algorithms that utilize OLTC).

3.4.4 Phasor Measurement Unit (PMU)

A large number of smart grid applications increasingly depend upon reliable measurements being obtained from within the network. Recently, network critical applications, such as protection, are dependent upon utilizing accurate measurements from a large number of PMUs. Although PMUs are not widely installed within the network today, they are expected to be widely deployed in the near future.

Within ERIGrid, both P-Class (for protection applications) and M-Class (for measurement applications) PMU models compliant with IEEE C37.118.1a have been developed as in [26]. The filtering within the model is based around the use of cascaded filter sections, each of which is adaptive to the measured fundamental frequency. The model performance is robust against the effects of off-nominal frequency and harmonic contamination. The model is streamlined for execution speed in real-time.

These models can be utilized by the wider community for development of novel applications that rely on data from PMUs. Examples include any wide area monitoring and control applications such as protection, equivalent load modelling, and power system stability studies.

3.5 Communications Simulation Domain Models

The development of novel intelligent control solutions for power systems has evolved the domain into a cyber-physical system. These intelligent control solutions often rely implicitly on communications. Although the proposed novel control solutions offer promising results for the applications they have been developed for, the implications of the performance of the assumed communications infrastructure is not well understood. This is often due to the difficulty of incorporating the effects of communications within a power system simulation tool (due to the discrete, event-based nature of packet-based communications). Therefore, often simplistic fixed-time delay models are utilized. To capture the realistic behaviour of communications and to study its impact on the performance of proposed control solutions, it is often required to co-simulate. However, co-simulation requires expertise for two domains (power systems and communications), and increases simulation times and the overall complexity of the setup.

3.5.1 Latency Calculation

The latency calculation FMU aims to provide a more realistic delay calculation model, which can be readily integrated into power system simulations than the classical simplistic fixed-time delay. The model takes into account several factors that contribute to the time delay: the distance between sender and receiver, the employed protocol and the physical transmission speed. Due to various random factors that affect the final latency, some assumptions were made to facilitate the calculation (detailed in the model description). In this first version, the model requires users to configure manually the characteristic inputs of the protocol and the topology of the network. In a future version, it is expected that there would be an integrated library of protocols and a self-configured topology block. The Latency calculation block also provides the rate of packet loss as outputs.

The FMU aims to be used in offline co-simulation for the communication in power system control. Some potential applications can be considered: evaluating the quality of control with respect to quality of communication (i.e., delay, packet loss), cyber-security testing, study impact of communication and cyber-attack to the system, evaluation the sensitivity of HIL testing with communication delay.

3.5.2 Latency Emulation

Within the ERIGrid project, a model that can realistically emulate communications latency within power system simulation tools has been developed. The developed model takes minimum and maximum delay (and optionally mean and standard deviation) values as inputs, in order to delay an input signal based on a random value chosen from a Gaussian distribution. To maintain the original order of data within the queue, the delay applied to the input data is forced to be greater than that already in the queue (this is representative of the level of service that can be achieved using modern packet-based Wide-Area Networks (WANs)). The model will be updated to allow for choosing if the order of data is to be maintained. This will enhance the flexibility of the model to represent a wider variety of communication infrastructures. This will enable power system studies to incorporate a simple yet more realistic, representative delay.

This model can be readily utilized within power system simulations where it is of importance to understand the impact of communication latencies. Such analysis will bolster the confidence in proposed control solutions. Such a model is best-suited to be utilized for time-critical applications such as protection and fast frequency control, and can enable new algorithms to be demonstrated with increased confidence.

3.6 Model Quality

The evaluation of the quality of a simulation model requires different concepts such as performance, accuracy, and fidelity to be employed. These concepts are associated with the model's capability to reproduce the behaviour of the real-world system it represents. Typically the quality of a model is assessed indirectly, by determining the uncertainty of the model. To do so, well documented procedures such as Uncertainty Quantification (UQ) and Sensitivity Analysis (SA) can be employed.

The UQ process is used to estimate the adequacy of simulation output to make predictions about the real world. It is conducted by calculating the output uncertainty based on the given initial uncertainty. This may be uncertainty in input data like parameter values, uncertainty lying within the model structure itself, or, more realistically, a mixture of several uncertainty sources [27]. SA, on the other hand, provides information about the response in model output to changes in the model input [28]. In other words, SA points out how strongly the output uncertainty will change if changes in a given source of initial uncertainty occur. Initial uncertainties are thus only handled hypothetically in SA whereas in UQ they are assumed to be correctly estimated.

A model is generally developed for a certain application, with necessary assumptions to aid simplification yet not impact its performance. Such decisions on necessary assumptions are usually taken by the experts developing the models. To determine the quality of the model, first, it is a necessity to understand the scope of the model. For example, the reduced dynamic equivalent model of DPSL has been developed based on measurements obtained for $\pm$10% voltage disturbances (from nominal voltage of 400V). Therefore, the model is valid for use in simulations where the voltage would be within $\pm$10% of the nominal voltage. For the purpose of fault analysis, where the voltage is likely to be beyond $\pm$ 10% of the nominal voltage, the developed model is not representative and should not be utilized. Determining the scope of the model provides the necessary boundary within which the quality of the model should be assessed. Furthermore, when a model is validated by the developers, it is often against a limited set of real-world data. Extensive validation is limited either due to the costs involved in obtaining the data by experiments or more often the unavailability of real-world system to conduct the experiments. Therefore, it is important to provide information about the model's scope and validity together with the model library. The uncertainty

information gained from validation studies should then be described in a standardized format that can easily be employed for systematic UQ. One example of such a format has been suggested by [29]. However, further analysis is required to guarantee its practicability or point out alternatives.

As mentioned earlier, the quality of the model within the defined boundary needs to be evaluated. Therefore, a list of parameters within the set boundary impacting the performance of the models needs to be identified. These parameters are not necessarily all associated with the model, but also with the simulation tool and process of simulation itself (for example, the response of the model for different time steps and integrator modules). SA can be conducted to provide sensitivity information about a model's input parameters. This way, UQ analysts employing the model can focus on data uncertainty sources with high impact while excluding less impactful parameters – a procedure that commonly has to be applied in UQ due to the curse of dimensionality affecting the process performance [30]. Conducting thorough SA for all models of the library should, thus, be the next step in the library refinement. Due to the low computational cost of all involved models, a sampling-based SA approach would be a favourable choice. Design of Experiments (DoE) methods may be used in such a setup for increased efficiency and reduced variance [31]. All models are given as FMUs and may thus be integrated into mosaik using the same generic interface. Accordingly, mosaik may serve as framework for comparable SA of the whole library.

3.7 Summary, Lessons Learnt and Recommendations

This section has presented the developed ERIGrid JRA2 Smart Grids ML in detail. It explains the methodology that has been adopted for development of ML, discusses the importance and applications of the selected models and the steps taken to ensure quality of the developed models. The ML covers a wide range of models across different simulation domains, including important DERs such as PV, battery, wind; power electronics converters with their associated controls; measurements and communication models. All the models have been tested and validated for their representative behaviour, and developed for tool-independent use and are FMI-ME compliant. The models will be further refined and subjected to testing at systems level for their use within the project.

The use of FMI-ME within the project has proven to be informative and the following advantages have been noted:

- *Enhanced simulation tools capabilities and model reusability:* With the FMI-ME conversion tools available, it is quite easy to export a model as FMU from one simulation tool and to import it within another simulation tool. This enables the reusability of models and offers an effective way of extending the capabilities of a single domain simulation tool to incorporate the behaviour of another domain (for example, incorporating the communications latency behaviour within power systems simulation tool).
- *Protection of model Intellectual Property (IP):* The FMI-ME conversion tools provide the ability to export black-box FMUs that emulate the model functionality without providing any information of its implementation (no parameters of the model are visible and hence cannot be modified). This will allow the sharing of models more comfortably when the disclosure of sensitive IPR can be an issue.

Although these advantages have been reported in literature, the hands-on experience gained by the consortium has raised individual partner awareness on reusability of models and their confidence in model sharing. It is hoped that the experiences and results from the project will draw the interest of researchers around the world and serve as an impetus for more proprietary simulation tools across domains to support FMI standard.

Although FMI-ME does offer promising potential for tool-independent modelling, the following shortcomings were recognized:

- The API definition and model description scheme provided by the FMI-ME specification are independent of hardware, operating systems and compilers. However, in practice FMUs for ME are tied to a specific platform (e.g., 32-bit Windows or 64-bit Linux) by the available implementations of the shared library it contains. Tools creating FMUs for ME typically do not support to create FMUs containing more than one implementation, which could in theory solve issues like 32/64-bit support or Windows/Linux cross-compilation. The alternative suggested by the FMI-ME specification, i.e., to provide source code instead of binary implementations, is typically either not an available option during FMU generation (due to IPR protection) or unfeasible for non-experts due to often very complex third-party library dependencies in the compilation process.
- Apart from specifics regarding platform and compilers, FMU implementations may often have additional dependencies at run-time. Some of these dependencies may be by design, e.g., the requirement to connect to a license server for a successful instantiation. Others may be unintentional and caused by poor modelling, e.g., an FMU may require access to a specific file with a specific hard-coded path.
- Dedicated hardware setups (and in particular closed-source proprietary setups) can also pose a challenge. For instance, an FMU exported from a Windows PC cannot be run on Digital Real-Time Simulator (DRTS) with essentially different processor architecture.
- MATLAB does not allow to export certain kinds of connectors as inputs or outputs for FMUs for ME. Whereas typical causal Simulink connectors (e.g., used for control signals) can be exported, acausal Simscape connectors cannot (e.g., used for electrical signals). This limits the effectiveness of electrical component models (such as synchronous generators) being exported as FMUs. There are workarounds that can be utilized such as measuring the voltages of the generator and reproducing them within the simulation with controlled voltage source and providing measured currents as feedback. This is method is similar to one adopted for power hardware in the loop simulations. However, this can be made simpler by means of providing “adapters” that would allow seamless conversion of electrical to control signals and vice-versa. This should further be explored and the capabilities expanded.

4 Proof-of-Concept Applications

Smart grid system configurations comprise a complex and distributed infrastructure with a lot of components and embedded functions. There are various domains interlinked, which are interdependent amongst themselves. The technical and ICT aspects of a smart grid presents us with both a conceptual and a technical challenge to model our system architecture, use-case descriptions and test-case specifications. This cyber-physical system is very complex and various approaches are used for its holistic evaluation. Heterogeneous modelling (multiple domains and tools) is generally used for the holistic evaluation. Heterogeneous modelling is mostly realised by co-simulation. In the context of JRA2, application mostly utilise the mosaik framework and FMI, which provides numerous advantages regarding co-simulation but also increases the development and implementation challenges. In this spirit, three main test cases have been formulated, of which each contains a co-simulation experiment targeting one specific and relevant development need. As such, this selection of test cases provides a representative set of examples that serve as a meaningful proof-of-concept set of applications for simulation-based smart grid assessment.

4.1 Power System Simulation with Cyclic Dependent Models (TC1)

4.1.1 Motivation

The power system of today has a multitude of components (DERs, ICT infrastructure, control systems), all with diverse time constants and possibly mixed discrete and continuous nature. Such a power system architecture leads to models with cyclic dependencies between the electrical and control domains, requiring control mechanisms to maintain the synchronism. The functional requirement of the simulator(s) to be applied is to enable the inclusion of these causal dependencies. TC1 deals with the cyclic dependencies between different simulators. It models a Wind Power Plant (WPP) connected to the grid at transmission level. The power electronic devices and their controls are required to be compatible with the power system dynamics. This mutual influence is predominantly exhibited during and after short circuits. In terms of co-simulations this also leads to cyclic dependencies among the implemented co-simulation components. Hence, this is considered an excellent opportunity to test synchronisation mechanisms between continuous simulators.

The modelling and simulation approach adopted here is to first model the relevant components into a monolithic simulator, MATLAB/Simulink in this case, export the wind turbine part of the system under test as FMUs complying to FMI-ME, and apply them later on for a co-simulation experiment.

4.1.2 State-of-the-Art on Coupling Continuous Simulations

4.1.2.1 Use Case and Test Case Relevance

A common problem when performing dynamic power system studies is the lack of models for various controllers and their level of detail. The rising significance of new approaches in the field of low-carbon energy supply, which are increasing the complexity of power system components, respective controllers, and their penetration in the system. This is of special importance if the performed studies are equipment-specific, e.g., grid code compliance related studies, power system stability studies or various interaction studies; as such studies have often legal and financial implications for asset owners and developers. In these cases, accuracy of the used models is crucial and their verification desired to ensure that the simulation corresponds to reality as much as possible. Development and verification of models for each power system simulation software is not feasible, which leads to usage of translated, simplified, unverified models. Such simplified models are fit mainly for specific purposes of a selected case study, typically standard transient situations within normal operating range and conditions of the controlled assets. In non-standard situations and different case studies, the performance cannot be relied upon, due to the specific simplifications made. Furthermore, in case of disputes the question whether simulated behaviour is performance of only the model or the actual system arises.

The grid code compliance of a wind power plant is a typical use case that exhibits the abovementioned challenges. Many (control) functions, which are usually assumed to act independent from the remainder of the grid configuration, now cause mutual interaction. An example in this respect is the low-voltage ride-through of wind turbines and reactive (or voltage) control.

Co-simulation expanding with standardized FMI approaches allows the use of verified complex control system models in multiple power system simulation tools and in this way, solves these challenges. Therefore, models of complex power system components and detailed controllers can be implemented, validated and compiled to a sharable, easy-to-access model to expand other simulation platforms. To demonstrate and validate the application of FMI combined co-simulations, the example of a power system simulation in PowerFactory and wind turbine controller in Simulink was selected.

4.1.2.2 Co-Simulation Examples from Literature

Presently, most of the work in co-simulation in the field of smart grids is related to the interconnection of power system (continuous) and communication network simulators (discrete events). The surveys in [32], [33], [34], [35] presented the comprehensive reviews of existing work involved in co-simulation of smart grids. However, only few studies involve co-simulation of different power simulation tools [36], which represent a complex power system or deal with the composition of existing models into an overall simulation (continuous-continuous). In this respect, two major standards exist: FMI and HLA [37], as well as quasi-standard master algorithm tools such as mosaic [18] or Simantics [38].

In general the data exchange between the master algorithm and the simulators can take place by means of memory sharing or higher level inter-process communication (like TCP/IP). As for the general case of simulator coupling, the main challenge of the master algorithm is the handling of the synchronization between the simulators. Two classical approaches are available for the master algorithm to synchronise simulators: the conservative [39], [40] and the optimistic [41], [42] methods, respectively. The conservative approach allows a simulator to proceed only for a time period in which it can be proven that no events sent out from other simulators are expected. This lockstep synchronization can be considered as the specification of a global real-time so that all participating simulators may each proceed their local time up to the global time [43], [44]. Various simulator couplings are registered using variations on the conservative method [45], [46], [47], [48]. In the optimistic case, every simulator processes its internal events until no more activity can be determined [49]. However, when a synchronization problem occurs, a roll-back is required. Application of this method with HLA standard is found in [50].

A comparative study of conservative and optimistic methods is established in [51] and another study on deciding between conservative and optimistic approaches on massively parallel platforms is found in [52]. While popularly used to co-simulation of continuous processes, it is equally possible to apply these methods to co-simulation of continuous/discrete and discrete/discrete simulators.

4.1.2.3 Overview on FMI based Interfaces for Similar Experimental Setups

HLA and FMI are two major standards for distributed simulation and co-simulation. While HLA aims at defining an architecture for the complete simulation setup, FMI is focused exclusively on the interfacing between the simulators and a (non-specified) master algorithm. Thus, several multi-domain energy system simulations may be found that utilize FMI. Some work has been done in creating a hybrid co-simulation platform using HLA and FMI, simulating, e.g., feedback between energy pricing and household heating [53]. Another hybrid simulation case employs PowerFactory for power grid simulation and GridLAB-D for simulator coupling in the validation of a flexible-demand EV charging environment. In this case with more complex simulators, FMI is only used for interfacing of simple OpenModelica models [54]. More strongly FMI-based interfacing is provided in [55] where commercial energy system simulators are used and Ptolemy II is employed for scheduling. The consideration of cyclic simulator feedback is similar to the one in the presented test case while the simulation itself is oriented towards heating systems rather than power systems. Several other works related to simulation of electrical or thermal energy systems employ FMI mostly for integrating additional com-

ponent models into complex energy system simulation tools, providing for more strongly controllable but less flexible co-simulation than presented here [56], [57].

4.1.2.4 Overview on Coupling Power System Simulators

There are numerous power system simulators used by different stakeholders in the field. Often the network companies make long-term decisions on the selections of the software to be used for modelling and simulations of their network system. Power system device and machine models are being made for several levels of detail, ranging from aggregated models for market studies, power flow models for N-1 and protection coordination, and dynamic models for transient and stability analysis. For the latter there may arise specific needs, e.g., to combine more accurate modelling of individual power plant models with large-scale electrical network models in order to be able to represent the interactions correctly.

For the power system domain, a common situation is the coupling between EMT and transient stability simulations (referred to as hybrid simulations). The system under test is then split into a detailed system, which is included into the EMT-type simulator, and an external system being modelled into the transient stability simulator. This is also shown in Figure 4.1. The split is commonly realised at the connection point of the device or plant that needs more detail (i.e., the interface node), and the mutual interactions are modelled by time-varying equivalent source representations. The interface variables are the active and reactive exchange between the two subsystems. The master duties are commonly orchestrated by the simulator that has the largest time step-size, in this case the stability simulator. Figure 4.2 shows an overall flowchart of the simulation, implementing the interaction protocol defined in the master algorithm. In this case the conservative approach has been applied.

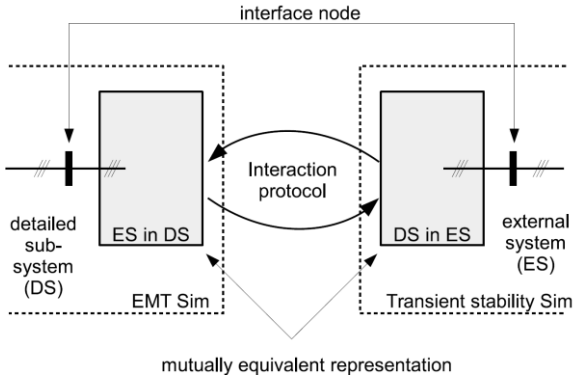


Figure 4.1: General concepts of EMT and transient stability simulation coupling

Various implementations of hybrid EMT and transient stability simulations have been proposed over the past decade. An overview of existing interfacing techniques can be found in [58], VSC-HVDC specific issues have been treated in [59], further generalisation steps have been conducted in [60], [61], and FMI-based hybrid EMT-TS simulations have been researched in [62]. A non-FMI based co-simulation of the EMT-type software PSCAD/EMTDC with PSS®E has been implemented already earlier and is commercially available[4].

4.1.3 Test Case Overview

This test case is focused on the interaction between a WPP and the main grid to which it is connected, in particular during faults in the external system. The specific case is about the capability of the WPP to stay connected during a three phase short circuit inside the transmission system, referred to as FRT. The grid codes often require conformance at the PCC in terms of connection stability and contribution to the fault mitigation.

[4] For further details see http://w3.siemens.com/smartgrid/global/en/products-systems-solutions/software-solutions/planning-data-management-software/planning-simulation/pages/pss-e.aspx.

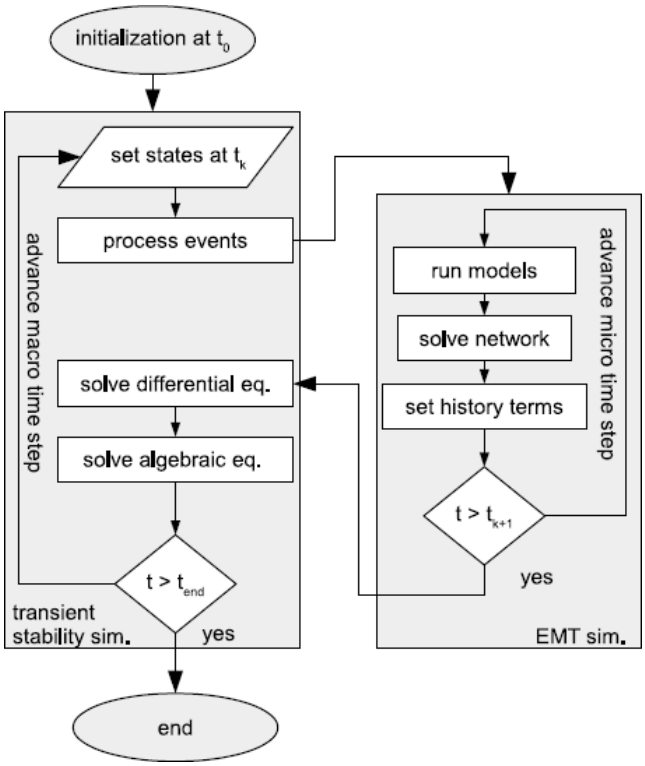


Figure 4.2: Example of an interaction protocol based on conservative coupling between transient stability and EMT-type simulators

The PCC is a key point of the entire test case: it is the legal boundary between the power park module owner and the transmission system operator, and it is the point where the compliance to the grid codes is required. The transmission grid and the WPP have different modelling needs and thus can have different levels of detail and can be modelled with different simulators. Owing to the mutual interaction, reactive power and/or voltage control are essential for FRT. Thus, the functions under test are the FRT capability and the reactive power control of the WPP, in order to validate the compliance of the WPP as a whole to the voltage against time profile during voltage sags and against a voltage-reactive power curve for normal operating conditions.

4.1.3.1 Description of Electrical System

The electrical system is mainly composed by the following components:

1. Transmission Grid
2. Wind Power Plant
   a. Wind farm interface, composed by the HV/MV transformer, the collecting MV grid and the MV/LV transformer
   b. Wind turbine generator

Transmission Grid

The wind power plant is connected to IEEE 9-bus test system at bus 9 as a replacement of generator G3, transformer T3 and Bus 3 (see Figure 4.3). Furthermore one third of the loads in the test system were replaced by dynamic loads (asynchronous machines). In order to ensure transient stability of the system generator G1 was equipped with a governor (1981 IEEE type 2 governor & turbine model) and an automatic voltage regulator (1992 IEEE type DC1A excitation system model).

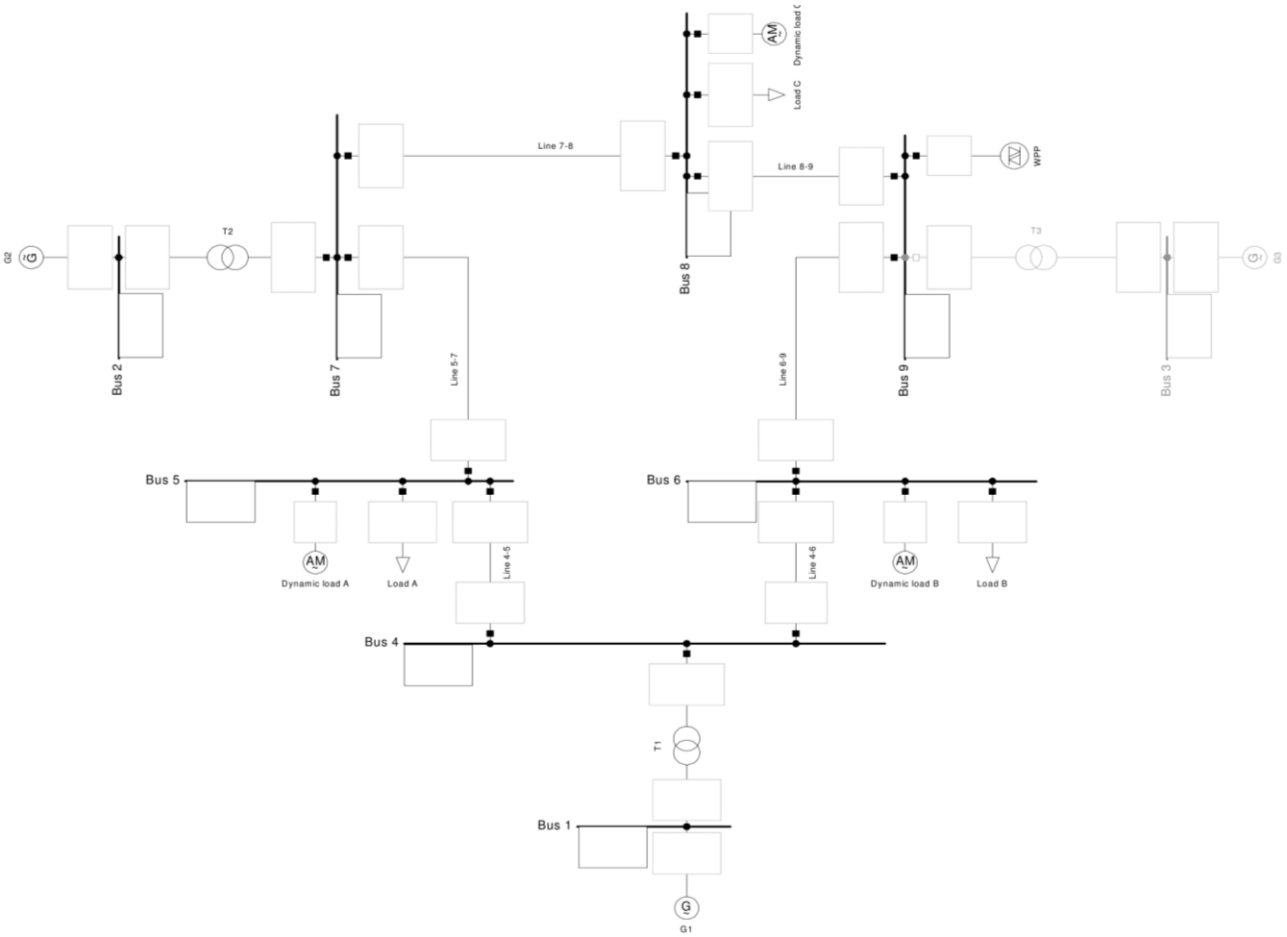


Figure 4.3: Modified IEEE 9-bus system as applied in the grid configuration of TC1

Collecting Grid

The selected wind turbine has a rated electrical power of 2.6 MW. The WPP collector grid is assumed to have a radial connection scheme (i.e., array layout). It consists of 4 cable collection strings, with 8 wind turbines in each string. The wind turbines are assumed to have a typical/ideal wind turbine spacing, which in turn determines approximately the cable lengths within the WPP. The typical wind turbine spacing currently is 7 times the turbine rotor diameter (i.e., 7D). The rotor diameter for a 2.6 MW turbine would be approximately 100 m. Thus the constant cable length between the turbines of 700 m is assumed. In addition the same distance of 700 m is assumed between the power plant main transformer and the first wind turbines on each collection string.

Cable data collected for and used in [63] are also applied here (i.e., see Table 4.1). The collector grid is over-dimensioned according to 125 % current of the actual rated currents on nominal voltage.

The calculations for the equivalent collection network impedance are based on [64] and [65]. The test case aggregated wind power plant collection network equivalent impedance parameters are R = 0.08739 ohm, X = 0.06714 ohm and B = 0.00124 S.

Table 4.1: Parameters selection for 33 kV cables

Irat [A]	r [ohm/km]	l [mH/km]	c [uF/km]
270	0.342	0.46	0.155
320	0.247	0.437	0.16
360	0.196	0.4	0.17
410	0.159	0.38	0.18
530	0.098	0.36	0.22
690	0.063	0.33	0.26
850	0.042	0.31	0.32
1010	0.03	0.29	0.38
1188	0.023	0.481	0.455

Wind Turbine Generator Converter

The Wind turbine generator is composed by the following components:

- Grid-side converter: DC to AC
- DC Bus
- Generator-side converter
- Electrical machine: PMSG
- Wind Turbine: it includes the mechanical rotor, the gearbox, the blades and the pitch and yaw controllers

For this test case the dynamics of generator side components are assumed to be damped by internal controllers and are hence negligible for grid-side interactions. Consequently, the elements 3, 4 and 5 of the list above can be simplified as a constant power injection on the DC bus along the period of interest.

4.1.3.2 Description of Controllers

The main controllers involved in this test case are the controller of the grid-side converter and the FRT controller. As explained above, the controllers of the wind turbine and of the generator-side converter are neglected. Also the transmission grid includes governing mechanisms: the synchronous machines of generator G1 is equipped with an Automatic Voltage Regulator (AVR) and a Governor for frequency control. For this purpose, standard controllers from IEEE with default parameters have been used: DC1A Excitation System as AVR and Type 2 Governor.

Controller Converter

The grid-side converter controller is composed by two main subsystems. The first one is the measurement and signal processing: it acquires and filters the measurements, calculates the frequency and angle through a PLL, and applies the Park transformation to the relevant variables.

The second subsystem is the vector controller itself: it calculates the reference voltage signal for the converter in order to control the voltages on both AC and DC side. Some of its parameters are changed by the FRT controller described next.

FRT Controller

The FRT controller measures the AC and DC bus voltages in order to detect fault conditions and engage corresponding control modes and protection mechanisms. In the context of this test case, its main purpose is to increase the reactive power contribution of the converter in case of fault, engage the DC-side braking resistor, vary current limiting schemes, and ensure minimum active power recovery rates.

This is all done by implementing a state machine on top of the vector control scheme for normal operating conditions (see Figure 4.4). It applies the following states: start, normal, FRT, and post-FRT, each employing a particular set of recovery rates, control gains, etc. State transitions are triggered by violation of voltage thresholds, AC and DC alike.

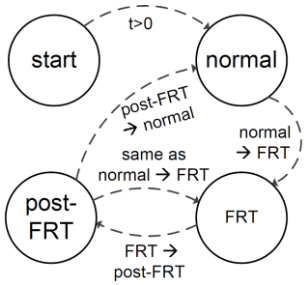


Figure 4.4: Finite state machine implementation of the FRT controller

4.1.3.3 Test Specification

The test objective is to verify the behaviour of the WPP at the PCC during and after a voltage dip that occurs at the PCC. In order to reproduce this situation, a 3-phase-to-ground fault is triggered in a transmission grid node. Two various fault types are tested, both with the FRT controller enabled or disabled, starting from the same grid conditions. The test is successful in case the WPP remains connected and no further overcurrents, voltages, and overfrequency occur.

4.1.3.4 Experimental Setup for Reference Simulation

The reference simulation setup involves a monolithic model, entirely run in the same simulation environment. The simulator chosen is MATLAB/Simulink, using the Simscape Power Systems Toolbox for the simulation of the electrical components. The simulation is entirely run as a continuous simulation in EMT mode with a fixed time step of 20 µs.

The electrical components have been selected and adopted from the Simscape Power System library [66]. As for the WPP, the grid-side converter uses an average valued model, composed of three controllable voltage sources and an output filter, neglecting the power electronic switches. The DC Bus is composed of a capacitor and two programmable DC sources, representing the grid- and the generator-side converters. The grid-side DC source is driven by the grid-side converter, while the generator side DC source injects a constant power, emulating a wind turbine generator with a constant mechanical power at the shaft.

The electrical model is initialized to a steady state condition using the Load Flow tool integrated in Simscape Power Systems. A proper separate Simulink model has been built for this purpose. Also the controller states are initialized with a MATLAB script in order to avoid large initial transients.

4.1.3.5 Conceptual FMI-compliant Co-Simulation Setup

Figure 4.5 shows a schematic view of the conceptual co-simulation setup for TC1. The overall system configuration will be split in separate parts, each represented by an individual FMU and connected to mosaik, which orchestrates the co-simulation execution. During the execution, mosaik

uses the functionality provided by the FMI++ Library to interact with the FMUs (both CS and ME) and to integrate the models contained in FMUs for ME (compare with Section 2).

The AC grid part of the test system configuration is simulated in PowerFactory and comprises the IEEE 9-bus system, the collection grid, and the AC grid interface of the WTG. The WTG is modelled as a variable quasi-stationary current source, with the set point of this current source being a co-simulation input controlled by mosaik (through the FMI-CS interface). The wind turbine, its controls, and the FRT controller are each implemented in Simulink and subsequently exported as FMUs for ME.

As such, this setup demonstrates the benefits of the co-simulation approach adopted in work package JRA2:

1. It allows to design and validate models in a multi-domain simulation environment.
2. It allows couple these models in a standardized way (FMI-ME), interfacing them with a commercial, domain-specific simulation tool (FMI-CS).

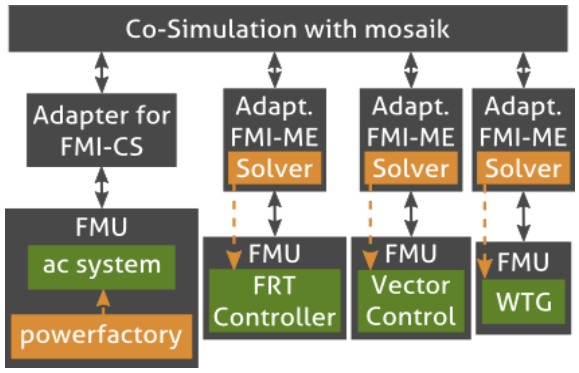


Figure 4.5: Schematic of the conceptual co-simulation setup of TC1

4.1.4 Simulation Results

The starting operating condition is largely equal to the original test system operating point [67], and is assumed equal for every sub-experiment. For the sake of completeness, the steady state conditions are reported in Table 4.2.

Starting from this condition, 4 different sub-experiments have been defined and tested, varying the fault conditions as described in table. The fault is always occurring at node 4 of the transmission grid, starting from at 0.4 s and cleared 180 ms later.

Table 4.2: IEEE 9-bus Initial conditions for TC1

	Active Power [MW]	Reactive Power [MVAr]
Load A – Static	-77.2	-23
Load A – Motor	-44.8	-28.5
Load B – Static	-56.0	-13.9
Load B – Motor	-32.3	-20.5
Load C – Static	-59.3	-15.5
Load C – Motor	-36.0	-22.6
Generator 1	147.7	73.6
Generator 2	78.0	6.7
Generator 3 (WPP)	85.0	-10.9

	Voltage [p.u. RMS]	Phase [deg]
Node 1 (G1, Slack)	1.04	0
Node 4 (PCC G1)	1.003	25.33
Node 7 (PCC G2)	0.964	24.57
Node 9 (PCC WPP)	0.946	26.15

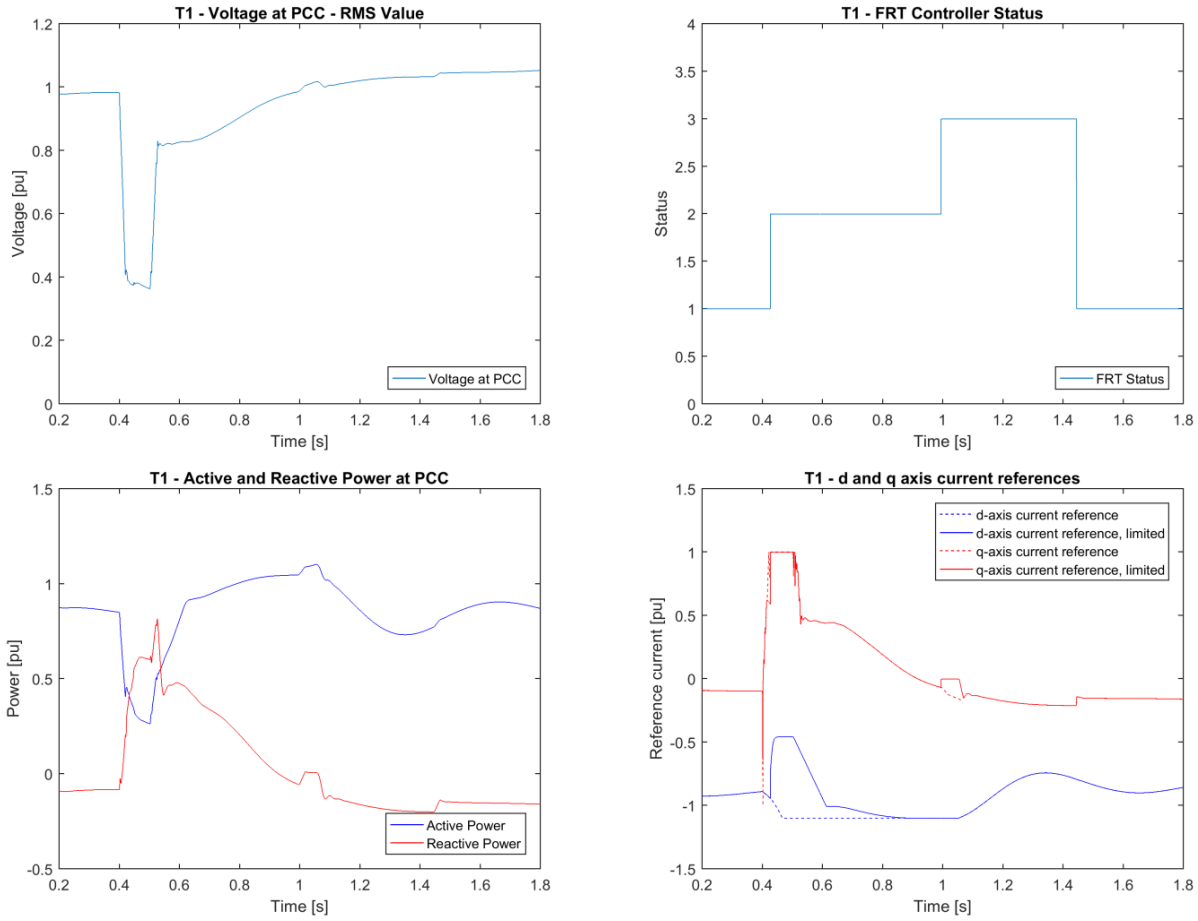
	Fault resistance	Fault duration	FRT mode
Test 1	5 ohm	100 ms	enabled
Test 2	5 ohm	100 ms	disabled
Test 3	40 ohm	200 ms	enabled
Test 4	40 ohm	200 ms	disabled

4.1.4.1 FRT Simulation of a Wind Power Plant using Simulink

The first test is analysed in details, while for the others only the main results are reported.

Test 1

In the first test the fault in the transmission grid causes a deep voltage dip at the PCC below 0.4 p.u. This condition activates the FRT controller that switches the controller priority to the AC voltage control, increasing the iq current reference and thus the reactive power injected. The DC voltage remains below 130% of the rated voltage and the normal conditions are restored after about 1 second from the fault clearance.



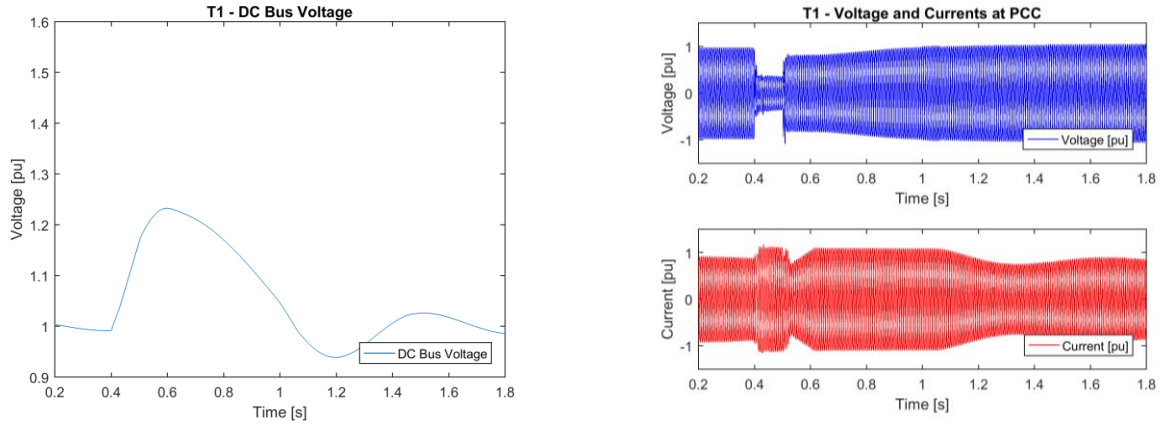


Figure 4.6: Results for Test 1

Test 2

The second test runs in the same conditions of the first one, with the FRT disabled. The lack of a prompt reactive power injection results in a deeper voltage sag that cannot be recovered, with the consequence of subsequent opening of undervoltage relays.

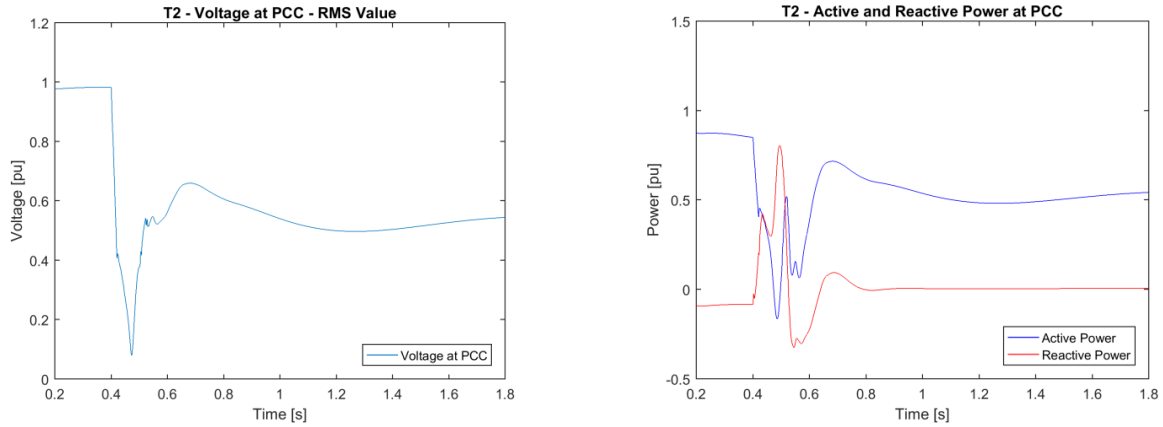


Figure 4.7: Results for Test 2

Test 3

Here the voltage sag at the PCC is shallower, with a recovery transient similar to the one of Test 1. The WPP successfully rides through the voltage dip.

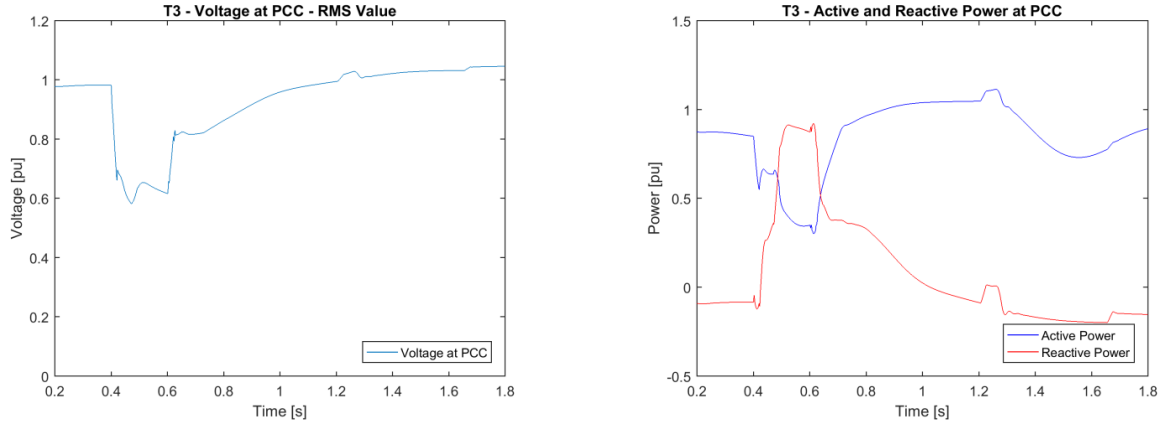


Figure 4.8: Results for Test 3

Test 4

As in test 2, also during this test 4 the lack of reactive power injection doesn't permit to recover from the low voltage fault.

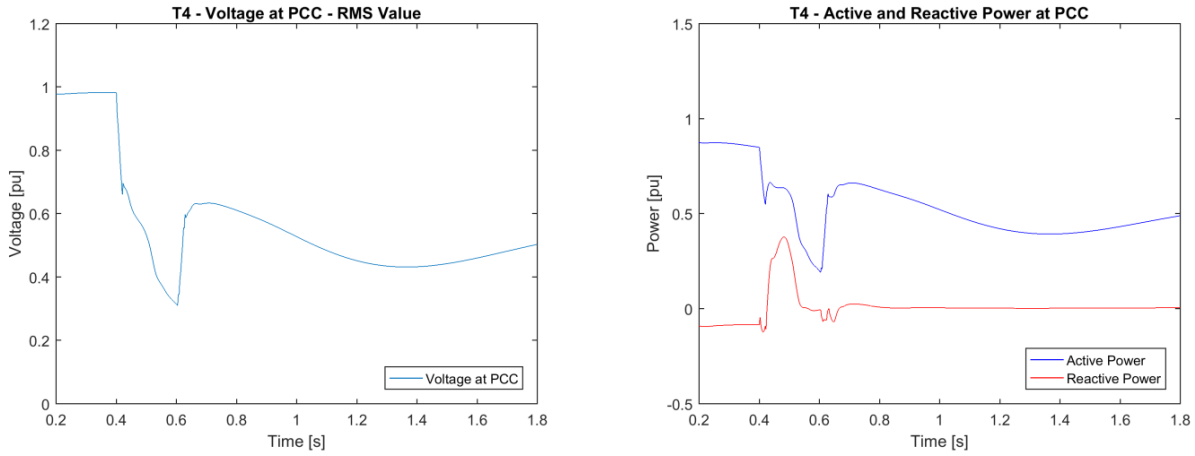
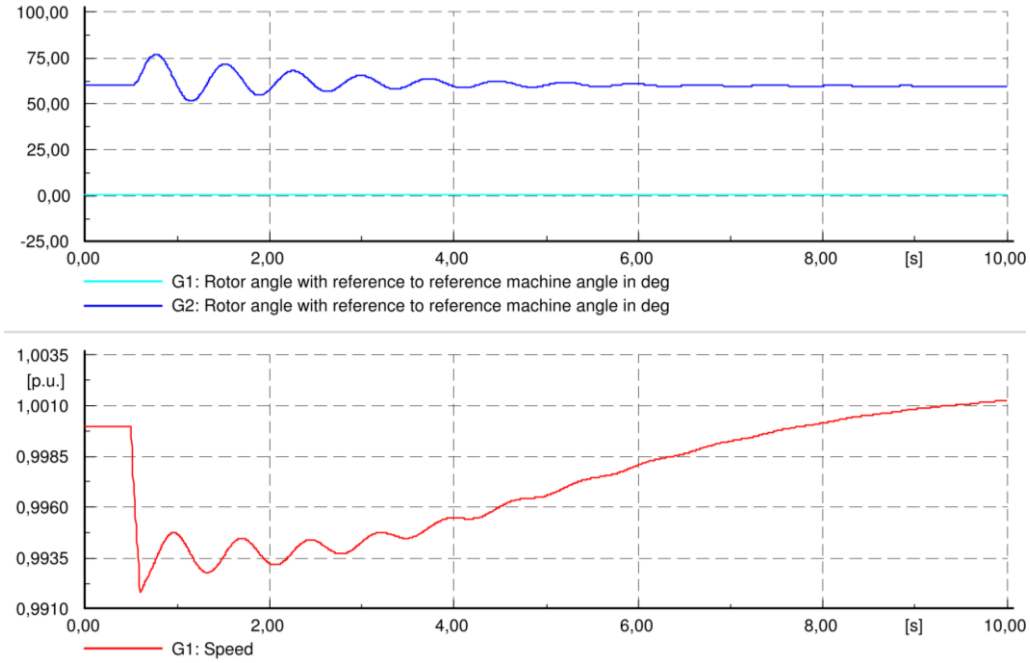


Figure 4.9: Results for Test 4

4.1.4.2 IEEE 9-Bus System Response in PowerFactory

The modified IEEE 9 bus model was prepared in the RMS partition of PowerFactory, with the wind park equivalent turbine as an FMI interface. The stability of the external grid was tested by a simulation of the system without an FMI-CS link (i.e. uncontrolled wind park current injection, which remains fixed during the experiment) subjected to a fault to Bus 4 at t = 0.5 s, which is self-cleared after 100 ms. The dynamic response of the system is shown in Figure 4.10 and can be considered as acceptable.



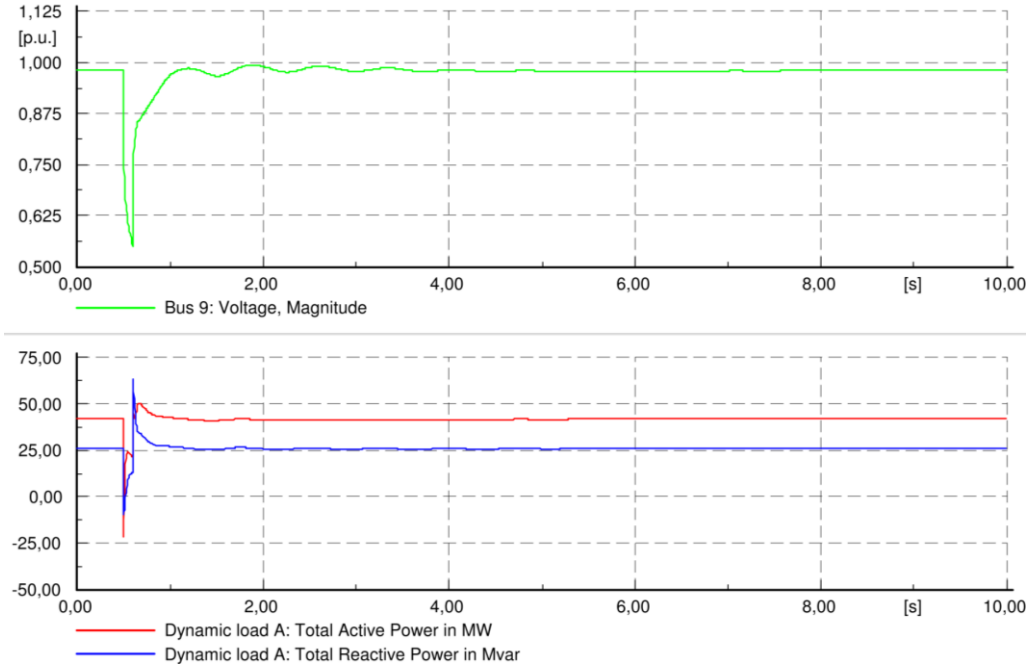


Figure 4.10: Simulation results of a test disturbance at bus 4 in the IEEE 9-bus system. The WPP is regarded as a fixed current injection.

4.2 Combined Hardware and Software Simulation (TC2)

4.2.1 Motivation

Simulation and laboratory testing are the two primary methods used for the functional verification of newly developed technology in an electrical power research context. Each of these methods has specific advantages and limitations. The main limitation of laboratory testing is the lack of flexibility regarding possible system configurations: The scale of the largest system that can be represented, and the physical properties of its components, are given by the installed equipment and can only be changed with significant effort and cost.

Simulation, on the other hand, is much less restricted in scale or configuration, and new types of components can often be added with limited effort - an attractive feature when testing early prototypes. The main disadvantage of simulation is a loss of accuracy due to the discretization of interactions between parts of the system. This is of particular relevance where complex interactions between the cyber and physical domains must be represented (see also TC3).

In certain cases, being able to couple power hardware in a laboratory to a co-simulation platform would provide researchers with a “best of both worlds” testing setup where the flexibility of a simulated setup could be combined with the accuracy of a physical system.

Usage examples include:

- If a laboratory experiment requires a particular type of component which is not physically available at the laboratory, it is often possible to save significant cost and effort by simulating the component in real-time and inserting it into the laboratory by means of a coupling device. One of the main benefits of using the JRA2 co-simulation platform for this process is the possibility for reusing existing component models, either those already included in the domain simulators' standard libraries, or those developed as part of JRA2.2.
- The scale of typical power system testing laboratories is of the order of 10-20 busbars and tens of components. On the other hand, many smart grid applications, which for instance aim at providing

power system services by harvesting the flexibility of DER units, require tests with a larger population of units in order to obtain a test system with relevant statistical properties. Hardware/Software coupling would allow the creation of a large-scale experiment in which the laboratory setup represents e.g., a single distribution feeder which operates as part of a larger, simulated power system in which physical and simulated DER units are controlled by the same control algorithm.

- As part of the development of physical controller hardware, testing of the controller against a simulated power system through a CHIL setup is an efficient verification method. The development of a co-simulation platform with CHIL capabilities provides additional degrees of freedom; specifically, the operation of a hardware controller against a simulated power system which interacts with simulated controllers in a multi-domain simulation is of great interest in the context of smart grids.

The above examples rely on the availability of suitable controllable hardware device which can be used to link simulation and physical system. In the first two cases, this could be an amplifier, back-to-back converter, controllable load or storage device. The third case may require e.g., a fieldbus "stub". Unfortunately, neither PHIL nor CHIL interfaces have converged to a common standard; part of the reason for the lack of standardization can be assumed to be the market situation in the highly specialized, competitive and low-volume market for commercial HIL solutions.

The approach chosen by ERIGrid works around the lack of standardization by developing adapters between laboratory hardware and the existing FMI standard.

4.2.2 State-of-the-Art on Coupling Hardware and Software

4.2.2.1 Real-Time Simulation and Hardware Coupling for Diverse Time Scales

Real-time simulation of power system aims to reproduce the output waveforms (voltage/currents), with the desired accuracy, which is representative of the behaviour of the real power system being modelled. It is a challenging task that requires the computation system to be strictly deterministic (i.e., to solve the model equations for one time-step within the same time in real-world clock). A power system can be simulated entirely inside the simulator and does not involve external interfacing or inputs/outputs (often in study of behaviour of a system under certain circumstances resulting from external or internal dynamic influences. Another approach (HIL) involves replacing parts of the system with actual physical components connected through input-output interfaces e.g., filters, digital-to-analogue and analogue-to-digital converters and signal conditioners.

This approach is utilized to minimize the risk of investment through the use of a prototype and to test the hardware under test in extreme conditions. Coupling with hardware can be found in testing of converters, fault-current limiters, protection devices or any other electrical equipment and may require voltage or current amplifiers. The hardware/software interface is however specific for each application and is often configured according to the user's requirements, with or without amplifiers and power exchange. This practice prevents interoperability as well as the reusability of the models or the hardware. Application of FMI standard to this coupling interface possesses therefore a great interest.

4.2.2.2 Challenges

Real-Time simulators provide the potential of calculating and exchanging information according to their internal real-time clock. Nevertheless, including additional hardware includes diverse real-time clocks leading to challenges such as: initialization of the simulated and coupled models, synchronization of data exchange between those devices according to different time-steps and possible unequal simulation methods, like EMT or RMS simulation. This leads to delays and non-synchronized time-steps of the holistic simulation environment, causing inaccuracy of the combined simulation results. In the end, decisions need to be made to be aware of feasible results of a combined simulation using several software tools and/or hardware devices:

- *Computation step size*: According to different step sizes, different results can be obtained. In general, the highest step size prioritizes the results that can be investigated. For instance, run-

ning a co-simulation with 1 s time steps combined with 1 ms time steps produces only feasible results of phenomena that can be investigated only in the time scale of 1 s.

- *Signal conversion*: According to possible different simulation methods, different signals will occur in the simulation environment. These require conversions to be readable in the requested simulation tool and leading to additional delays between several devices.
- *Real-Time capability and synchronization*: In general, the coupled devices using their own computation clock, which needs to be synchronized (e.g., hard real-time coupling) or additional delays needs to be taken into account, since clock flags and signal exchange won't match (e.g., soft real-time coupling).

4.2.2.3 FMI-based Applications in this Direction

Several tools which target real-time HIL simulation claim to support some variants of the FMI[5]. Most of these tools focus on the automotive domain, although some more generic tools such as AVL Model.CONNECT™ exist[6]. AVL Model.CONNECT™, for instance, aims at connecting real and virtual components in a co-simulation approach via various interfaces such as the FMI. The FMI-capable HIL system SCALEXIO® and a corresponding use-case was presented by dSPACE[7]. The system is able to execute real-time capable C-code FMUs in a HIL experiment. In the presented automotive use-case, a sample time of 1 ms was achieved.

Although some tool support is available, little academic research has been undertaken to couple FMI-based models or tools and real hardware. Pang et al. coupled some building energy models via FMI with building automation infrastructure in order to evaluate the building performance in real-time [68]. Since the targeted domain does not require quick response times, little attention was put in the temporal properties of the setup. First steps in coupling automation infrastructure and FMI-based applications were done by Spiegel et al. who presented a tool which transfers the outputs of a model to an IEC 61499-based controller [69]. They presented some theoretical aspects, but a closed-loop operation of the coupled systems was not demonstrated.

Zehetner et al. stated some challenges in coupling real hardware and simulation models which are not optimized for real-time operation [70]. They proposed an approach which is based on polynomial extrapolation to reduce the delay in the hardware-software coupling and to increase stability. The approach was briefly demonstrated at an engine test-stand and a controller HIL setup.

4.2.3 Test Case Overview

The purpose of this test case is a first step forward in the implementation of a control block using functionality provided by an FMU for ME. In addition, this test case aims to build an interface between a simulated part of the power system and a physical device (OLTC). The real OLTC controller can be designed to perform more complicated functions such as monitoring, supervision and statistics among other advantages that the transformer can bring to a power system in terms of electrical power efficiency. But in order to make this model reached by this test case, only the voltage-regulation closed-loop will be applied.

4.2.3.1 Hardware and Software

The test case is separated in two parts; the software part includes the simulated model built in the MATLAB/Simulink environment. The control block is aimed to be implemented using an FMU for ME. The hardware part is represented by the controller of the medium voltage transformer embedded in an open-source electronics platform along with the OLTC transformer itself. Many electronic devices can be used for this objective such as an Arduino card, which is an open-source electron-

[5] For further details see https://www.fmi-standard.org/tools.
[6] For further details see https://www.avl.com/en/iodp/-/asset_publisher/MQahPiTr3eTp/content/model-connect-.
[7] For further details see https://www.dspace.com/en/pub/home/products/hw/simulator_hardware/scalexio/scalexio_fmi/hil_fmi.cfm.

ics platform based on easy-to-use hardware and software. The Arduino device can be programmed directly using models developed and compiled in the Simulink environment.

4.2.3.2 Description of Hardware and System Dimensioning

The setup comprises an electro-mechanical OLTC that is able to change the tap position automatically under load, by means of a combination of fixed and movable contacts, along with a set of vacuum interrupters (two per phase) on the MV side of the transformer. The vacuum interrupters guarantee that the tap changing is performed safely because the arc, promoted by the switching process, is located inside the vacuum bottle. This prevents contamination of the transformer dielectric fluid.

The OLTC changes the ratio of the transformer by adding or subtracting turns from the MV winding. A transition impedance bridges adjacent taps for the purpose of transferring load from one tap to the other without interruption or appreciable change in the load current. Besides, they limit the circulating current allowing continuous loading in the case of reactors.

As functional testing of new solutions for voltage control is complicated in real networks, the research to assess the operation of the smart distribution transformers with OLTC is done in the UDEX (Demonstration and Experimentation Network) [71], [72]. This is done with the help of a highly configurable MV network able to reproduce both normal and anomalous working conditions of typical distribution grids, designed as a platform for the research of new products and systems for smart grids.

This concept of a highly configurable MV network for development and testing of new technologies not only contemplates equipment testing, as with standard laboratory testing, but also solutions and applications for network infrastructures and systems, including the smart distribution transformers.

The smart distribution transformers are placed in the UDEX test bay and the test program is controlled from the local control room of the UDEX allowing close observation of the sample performance, as well as fully monitoring all the relevant parameters of every operation in real-time.

The UDEX is connected to a 36kV utility grid and is adjusted, by means of an autotransformer, to the required voltage level. Different loads along the MV Network are connected and disconnected to promote LV drops and rises to assess the performance of the system during the produced voltage fluctuations.

4.2.3.3 Description of Controller

The OLTC is associated with a control system (controller), which governs the changing of taps inside the transformer on the MV side based on the variation of its low voltage output. The controller has four symmetrical areas of actuation, which are characterized by the voltage level ($BLQ^{\pm}$, $ACC^{\pm}$, $TOL^{\pm}$, SetP) and their durations. These areas are mirrored by the voltage set point axis, see Figure 4.11.

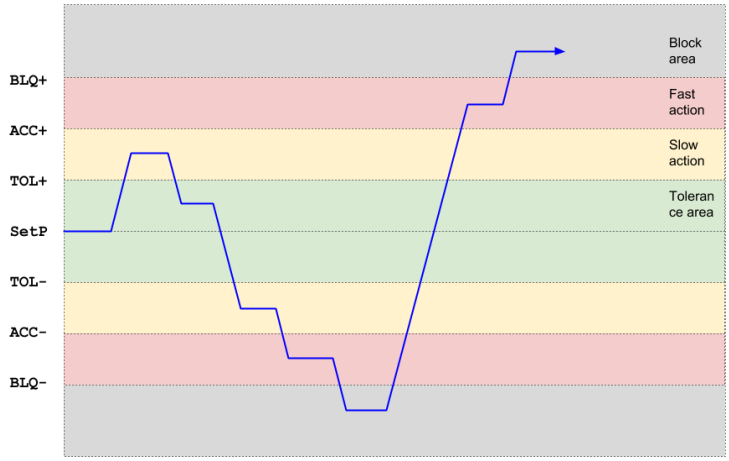


Figure 4.11: Actuation diagram of the OLTC control

In the following, a brief description of the controller logic is provided (based on the areas depicted in Figure 4.11):

- *Block area:* The system only enters this state in case the voltage has anomalous values (e.g., overvoltage). Then the controller must block the device and no actuation must be performed.
- *Fast action:* When the output voltage is less than the block area but in excess of the tolerance area, the controller must seek the suitable tap for the desired voltage within a specified time (*delay1*).
- *Slow action:* The output voltage is near but not in the tolerance during a time longer than *delay2*, the controller must change a tap position.
- *Tolerance area:* When the output signal of the transformer is within the set point tolerance, the controller will supervise the output voltage without any action.

4.2.3.4 Experimental Setups for the Reference Simulation

The reference simulation is created as a monolithic MATLAB/Simulink model. The model covers the OLTC control algorithm as well as all relevant electrical components of the setup. A monolithic simulation approach was chosen to avoid any inaccuracies and flaws which may be introduced by coupling multiple components. Hence, it is justifiable to use the results of this monolithic simulation as a baseline for further investigations. The monolithic simulation was also used to quickly test the controller and electric vicinity (plant) without having to deploy the controller to the actual hardware. The controller model has been built in Simulink and linked to a model of the power grid built using the SimPowerSystems toolbox, see Figure 4.12.

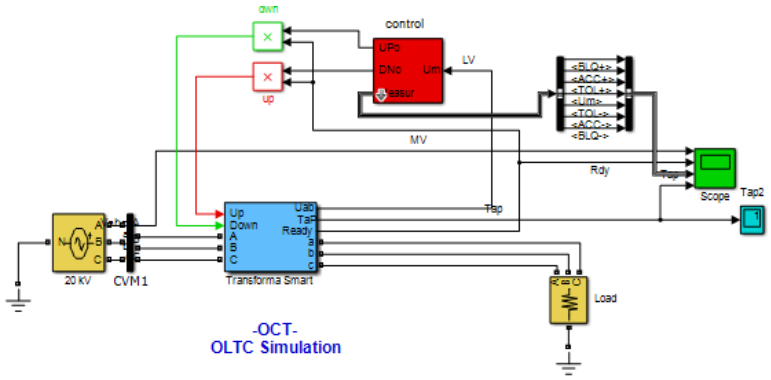


Figure 4.12: Graphical representation of the monolithic simulation model

The controller is located inside the red coloured subsystem, which provides two control inputs for the transformer motor gear depending on secondary transformer winding voltage. In this setup, the controller is continuously supervising the output voltage of the transformer.

4.2.3.5 Experimental Setup for Simulink/Arduino Experiment

Since the real-time behaviour of an embedded controller is very complex, it is usually not modelled in full detail. Instead, a logical abstraction is used to represent the control algorithm in MATLAB/Simulink. In the first experiment, the logical abstraction of the controller is transformed to an actual control program which will then be uploaded into an embedded controller. In order to test the generated control program in a virtual environment which covers the electric vicinity, a hardware/software co-simulation will be performed. The control program will be executed on its designated hardware and the plant will be simulated by a MATLAB/Simulink model. The simulated outputs of the plant (mainly the low voltage readings) will be transferred to the controller and control actions of the controller will be used to update the state of the simulation.

As a controller hardware, a low-cost low-power, 8-bit microcontroller will be used, which is commonly used in productive environments. More specifically, an ATMega2560 chip which is clocked with 16 MHz is deployed in an Arduino development board. Since the computational power of the

chip is very low in comparison to a standard PC executing the MATLAB/Simulink model, significant insights are expected in executing the control program on the embedded controller. The communication between the MATLAB/Simulink model and the controller hardware will be performed via a proprietary protocol. The CHIL simulation will be compared to the reference simulation of the first experiment in order to mark significant deviations.

Figure 4.13 shows the controller model, which determines the actuation area for the output voltage at the secondary winding of the transformer and outputs this results as a physical IO to the Arduino.

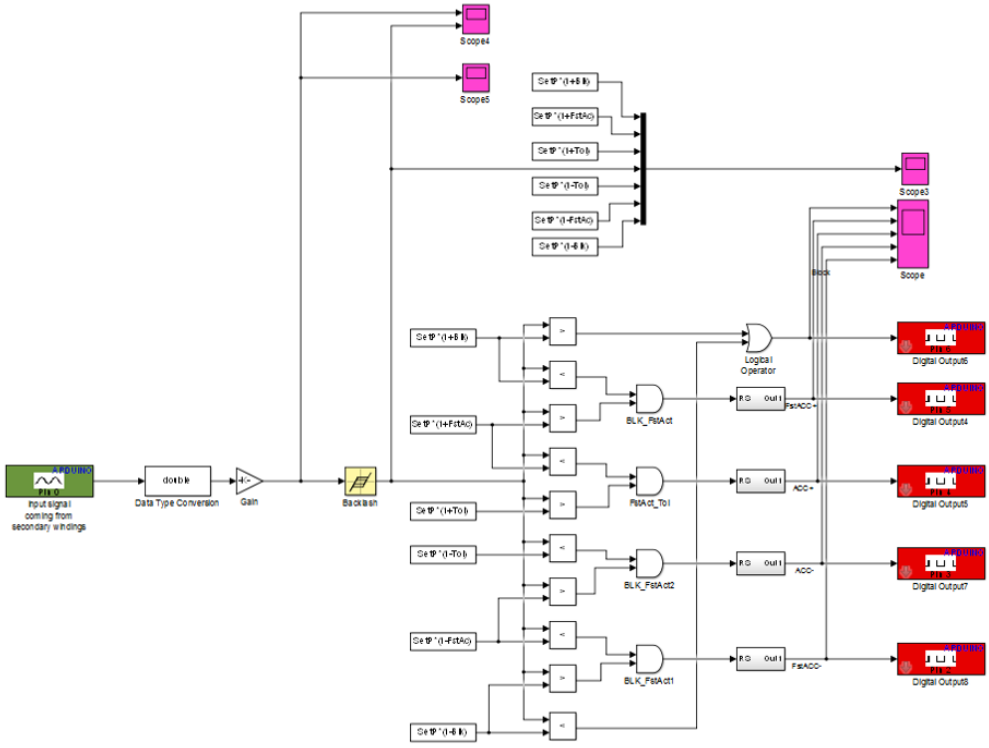


Figure 4.13: Graphical representation of the control model

4.2.3.6 Conceptual Experimental Setup for the FMI-compliant Co-Simulation

The next two experiments will utilize a FMI-based HIL simulation. The third experiment will cover a prototype stage that integrates the physical plant (i.e., the OLTC transformer and the measurement equipment) with a simulated controller. Since the embedded controller on the Arduino platform is not capable of directly executing an FMU, an industrial communication protocol will be used to connect the simulation host which executes the controller, to the IO interfaces driving the plant. Figure 4.14 illustrates the experimental setup.

By interfacing the model via the FMI and the plant via industrial communication protocols, a significant gain in flexibility and multiplicity is expected. Simulation models and tools may be easily varied without having to develop new interface logic. Likewise, automation and test setups may be dynamically adapted to the needs of the domain-specific development process.

The measurements and simulation results of the experiment will again be compared to the baseline simulation. In case of integrating a physical plant and a virtual controller, notable deviations to the baseline-simulation are expected. Such deviations may result from parameter inaccuracies and uncontrollable parameters such as voltage drops in the grid supply. Nevertheless, a correlation within the bounds of the parameter inaccuracies is expected. Another focus of the experiment is the stability of the test setup. Although some unavoidable communication delay is introduced, the virtual controller must still be able to control the low voltage level of the OLTC transformer.

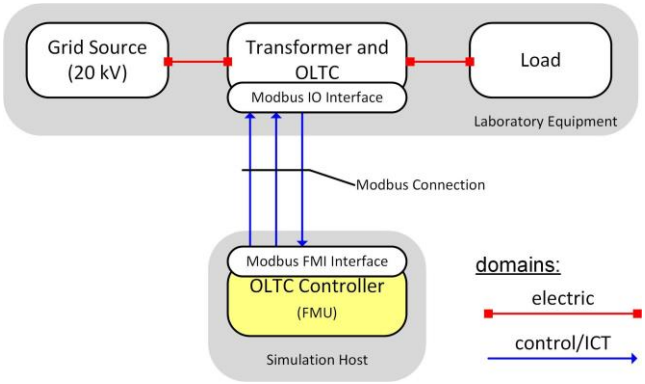


Figure 4.14: HIL setup including a physical plant

As a communication protocol, Modbus is used to interface the hardware. The simulation host which executes the model of the controller as an FMU will also host some interface components which couple the model and the automation infrastructure. The interface components are implemented in a generic way such that they are not limited to the targeted setups. These options cover a wide range of FMI-based models as well as various industrial communication protocols. The flexibility of the interface components will also be demonstrated in the fourth experiment which will use the same interface components in a different setup. In contrast to the third experiment, the fourth experiment focuses on testing the OLTC controller hardware without having to implement the actual plant. A testbed emulates the dynamics of the plant and directly drives the IO lines of the controller. Hence, the controller may not have to be extended by a model-controller interface and may use its dedicated IO subsystem. Figure 4.15 shows the main components of the experimental setup.

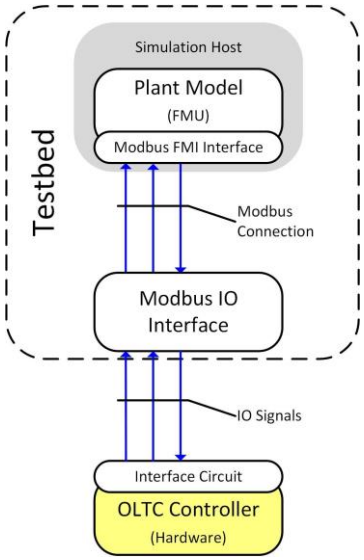


Figure 4.15: Controller HIL setup

The testbed will consist of industrial IO interfaces which drive the IO lines of the embedded controller. As in the previous experiment, the model and the IO interfaces will communicate via Modbus communication protocol. The plant model of the first experiment will now be exported into an FMU. The interface components will solve the FMU and couple it as a virtual lab component to the controller under test. In order to increase the comparability of the results and to ease the implementation of the controller, it will be exported from the baseline simulation and uploaded into the embedded platform. The measurement and simulation result will be compared to the baseline simulation as well. Special attention will be put to the stability of the system and the accuracy of the coupling methodology. It is expected that the measured results closely resemble the baseline simulation but that the additional delay which will be introduced by the testbed will be observable as well.

4.2.4 Experimental Results

4.2.4.1 Results from Simulink Experiment

As a reference case, a voltage drop has been injected into the monolithic model, in order to verify the behaviour of the OLTC and its controller. Figure 4.16 shows the corresponding simulation results. The simulation starts with the grid in its steady state, with a distribution feeder supplying a nominal voltage of 20 kV to the transformer and the load. At the 12 second point, a tap position increment is triggered with an undervoltage event, and the process continues until the output voltage returns to the tolerance area. At the 16 second point the grid returns to its steady state (20kV). However, due to these changes in tap position, the transformer effectively has an overvoltage event and the tap position is decreased to compensate.

4.2.4.2 Results from Simulink/Arduino Experiment

After performing the reference simulation using the monolithic Simulink model, the controller was implemented on an open-source electronics platform (Arduino). The model has been adapted to the new platform, splitting it into two independent parts. The first part is the *area detection* and the second is the *stepper motor control*.

The results of this experiment are represented as digital outputs according to the voltage input, see Figure 4.17. To generate the output voltage for all possible inputs a test grid voltage has been simulated by the Simulink Signal Builder block. The first graph represents the input voltage (simulating over and under voltages), while the rest of the plots represent the digital outputs of Arduino required to drive the stepper motor control.

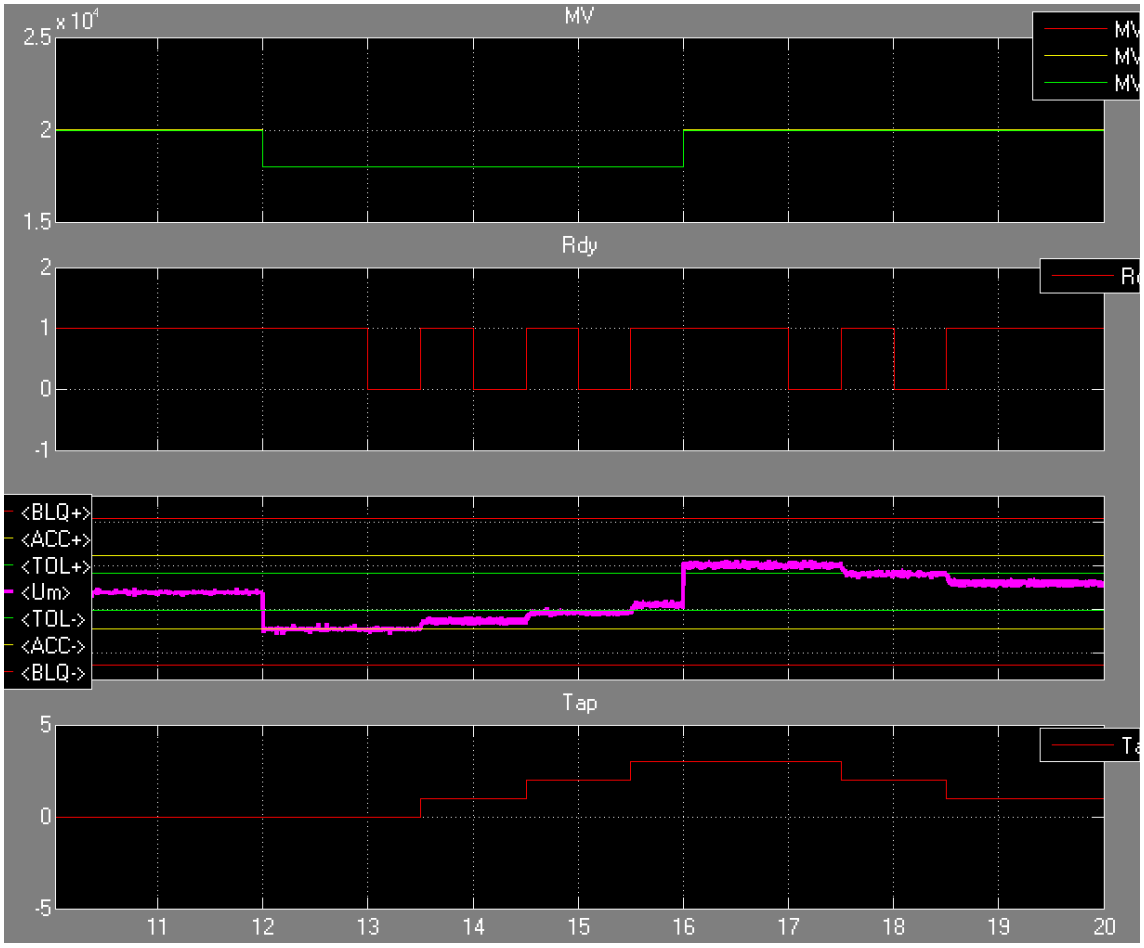


Figure 4.16: Results from the reference simulation (monolithic Simulink model)

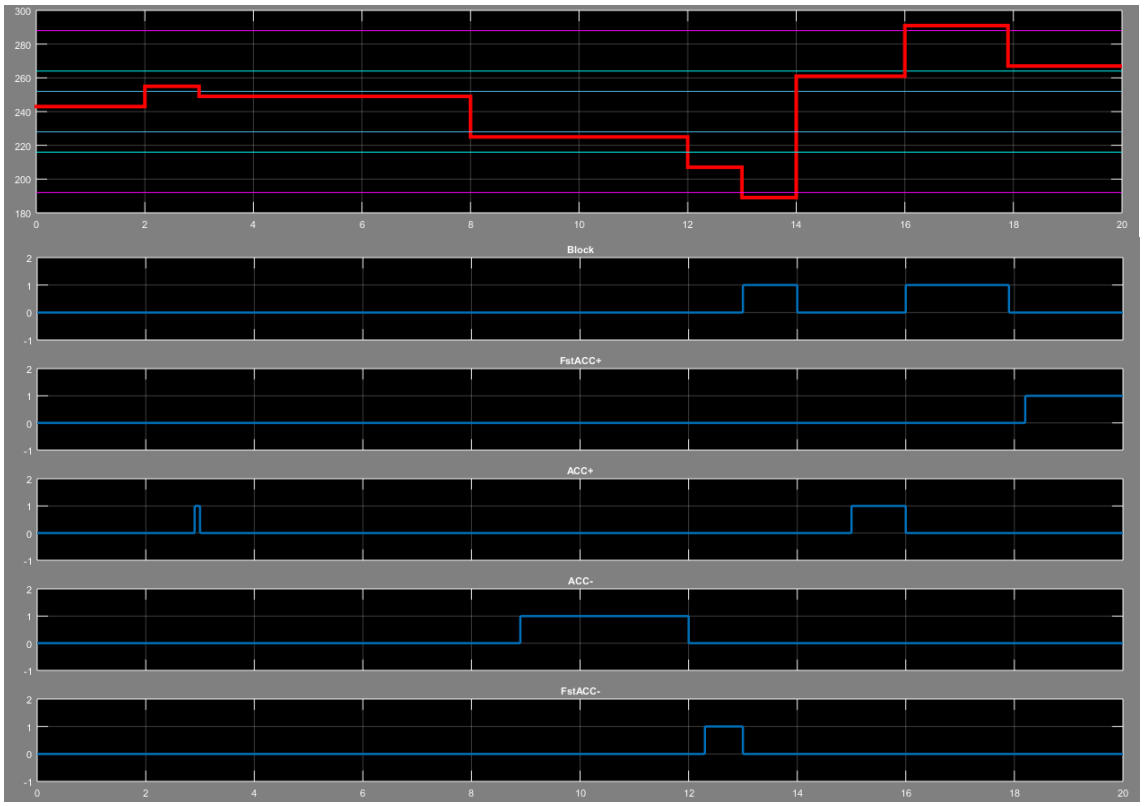


Figure 4.17: Results from the Simulink/Arduino experiment

4.3 Signal-based Synchronisation between Simulators (TC3)

4.3.1 Motivation

While the term smart grid covers a diverse range of technologies and application areas, the primary common factor between these is the increased application of (advanced) automation to the operation of electrical power systems. Some of these automation solutions are standalone systems such as e.g., an embedded controller in a PV inverter which injects reactive power into the grid depending on the voltage measured at the inverter's terminals. However, there is a growing trend towards network controlled solutions which enable the coordination and optimization of automation responses, be it between multiple units within close vicinity or across the whole power system.

Consequently, being able to accurately simulate the interaction between the electrical power system, embedded control hard- and software and the communication networks which link them is of great importance to smart grid research. Two aspects of communication are of particular interest in this context. The first aspect concerns the physical limitations of the communication channel, such as the communication latency (including any influence of channel congestion on latency) or jitter, i.e., the degree to which the latency is deterministic. These quantities can have a significant impact on the stability of a closed-loop controller, especially if their magnitude approaches the relevant time scale(s) of the controller in question.

The second aspect concerns the handling of communication errors, permanent as well as transient. This includes loss of information (e.g., dropped packets), changes to the information sequence or changes to the transmitted information itself (e.g., bit errors). Especially for unsupervised automation systems, it must be ensured that the system fails gracefully under all conditions.

An accurate representation of the interaction between the electrical power, control/automation and communication domains is of even higher importance if the interaction between entities is not limited to simple bilateral exchange of information, but involves complex dependency relations be-

tween multiple controllers which require many domain boundaries to be crossed and cause inaccuracies to accumulate. Examples of such complex relations include control hierarchies (e.g., layered control or multi-level aggregation), control loops which are closed across multiple devices and communication links or control systems based on communicating peers.

4.3.2 State-of-the-Art on Combined Power System and Communication Co-Simulation

4.3.2.1 Use Case and Test Case Relevance

For monitoring the state of the distribution network, in some EU countries DSOs could rely on the metering infrastructure. An Automated Metering Infrastructures (AMI, *smart metering*) comprises integrated systems of meters, communications networks, and data management systems; they enable a two-way communication between utilities and customers. Automated Meter Infrastructure is intended primarily for automatically collecting energy consumption data from premises and transmitting these data to the meter reading operator for billing and accounting.

The electricity meter can have either a Power Line Communication (PLC) or a Radio Frequency (RF) wireless interface with the concentrator, or a direct RF link with the central system. The choice of the technology depends on the *use cases* (i.e., the applications to be supported, in terms of availability, throughput, latency), on the land coverage, and on the rollout strategy. According to [73], smart metering infrastructure could promote a more efficient network operation, especially in presence of distributed generation, leading to a better quality of service and lower costs for customers; National Regulatory Authorities usually identify mechanisms to take this aspect into account.

Apart from billing purposes, data gathered from meters allow DSOs to have a more effective planning and operation of their networks. For example, reliable historical load profiles can support a better operational planning of the grid; and after a fault smart meters based on PLC can register the new electrical path up to the concentrator in the secondary substation. With respect to the test case considered here, meters can register the voltage value averaged on a fixed time horizon, voltage dips, and outages (cyclic collection of data started by the central system), and send a message to the central system when a given voltage threshold is exceeded.

Within this context, the selected test case is of particular relevance to control systems deployed in the distribution grid. Here, the large number of connections and the small control impact of individual units often dictate the choice of low-cost, shared-medium communication carriers such as PLC. Experience shows that the latency of these connections can be significant and is occasionally high enough to threaten the correct operation of controllers operating on a time scale of several minutes. Similarly, the aggregation of DER by commercial actors, e.g., demand response or the coordinated charging of electrical vehicles, often relies on residential broadband connections as the cheapest communication option. The comparatively low availability of these connections must be taken into account when designing a control strategy for such a system.

From the point of view of co-simulation, the selected test case represents a complex dependency relation as described in the previous section. The control loop is closed across three domains (power system, control/automation and communication) but crosses a total of eight domain boundaries because the automation system consists of four separate devices with communication links between them: Two smart meters, an application controller and the embedded controller of an OLTC. This is representative for many contemporary control and automation systems in which monolithic controllers with analogue sensor and actuator inputs and outputs have been superseded by distributed control systems where sensors (here: smart meter) and actuators (here: embedded OLTC controller) have embedded processing capabilities and are connected through a digital communication interface. Examples for existing systems which are, from a simulation perspective, similar in nature to the selected test case, are found in e.g., substation automation, wide-area protection, demand response or wind farm supervisory control.

4.3.2.2 Co-Simulation Examples from Literature

Coupling power grid and communication system simulators is a common application case of smart grid co-simulation. A comprehensive overview of relevant tools and platforms is given by [32] and [74]. Various suggested setups employ established tools for the simulation of the individual domains (power and communication), similar to the test case presented below, and aim to realise robust coupling between them. The Global Event-Driven Co-Simulation Framework for Interconnected Power System and Communication Network (GECO) [75] utilizes PSLF for power system simulation and ns-2 for communication simulation. The combined execution is based on discrete events in order to mitigate synchronization errors. A similar setup is given by the Electric Power and Communication Synchronizing Simulator (EPOCHS) [76] based on PSLF, PSCAD and ns-2. In contrast to GECO the synchronization is based on fixed time steps. A notable feature of the EPOCHS setup is the employment of HLA which allows for easy distribution of simulators on different computational nodes. Also based on HLA is the INSPIRE [77] approach, which used the commercial simulators DIgSILENT PowerFactory and OPNET Modeler. However, in contrast to GECO it relies on HLA's advanced time management services, enabling dynamic synchronization points assigned according to the chosen step size of the power system simulator.

4.3.2.3 Overview on FMI-based Interfaces for Similar Simulation Setups

The main focus for the development of the FMI standard has been the interfacing of time-continuous models and simulators. As a result, the interfaces specified by FMI for co-simulation and model exchange have reached a high level of maturity for this type of simulations.

However, for the coupled simulation of power systems and communication networks the adoption of discrete event-based modelling paradigms is necessary, in order to enable the construction of hybrid models (i.e., combinations of continuous time-driven and discrete event-based models). Interest in utilizing FMI-compliant interfaces in this context has been sparked only rather recently. First attempts to approach this topic have identified possible solutions for encoding alternative modelling paradigms (e.g., state machines, discrete-event systems, synchronous dataflow) with the help of the FMI specification [3]. For instance, it has been proposed to close the semantic gap between events and persistent signals (as defined by the FMI specification) by introducing a special value denoted as *absent*, indicating the absence of an event at a certain point in time. Output variables related to events would have this special value most of the time, except at time instances when an event occurs, in which case the value of an output variable is the value associated to the event. Other works have focused on the shortcomings of the FMI specification regarding hybrid co-simulation, pointing out the missing functionality and proposing new features to deal with these issues [78], [79].

Among others, these investigations lead to the formation of an official FMI development group on the subject, which is currently working on a proposal for future updates of the FMI specification. Within this context, the work done in ERIGrid for this test case can be seen as an additional input to this subject from an application-oriented point of view.

4.3.3 Test Case Overview

The test deals with the impact of communication delays in a simple low voltage distribution grid, where two meters send information about local voltage levels via a communication network to a remote controller. Based on these meter readings, the controller actuates the tap position of an OLTC transformer. The aim of this test case is to demonstrate and assess the effect of long communication delays on the actuation pattern of the controller and the resulting effects in the low voltage distribution system. Since this test case aims at providing an illustrative example of what problems may arise from poor controller designs, a fundamentally flawed approach for handling long delays is implemented for the controller. Moreover it is demonstrated that a more realistic simulation is needed for distributed and centralized smart grid control algorithms, as well as a benchmark for evaluating communication network technologies and topologies for use in smart grid applications is demonstrated.

4.3.3.1 Experimental Setup

From the point of view of the co-simulation, TC3 is the most innovative and complex setup, due to FMI's current lack of support for discrete-event and signal-based simulations. Therefore, from the smart grid perspective, a very simple experimental setup has been chosen, in order to shift the focus to the aspects related to FMI-based co-simulation.

The simple low-voltage distribution network that will be used for simulating the power system domain is illustrated in Figure 4.18. Similarly, a simple layout for the communication network has been chosen, see Figure 4.19. The voltage controller logic is shown in Figure 4.20.

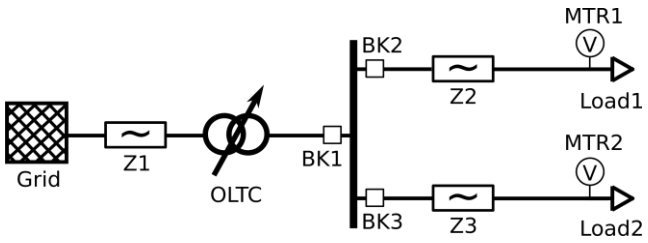


Figure 4.18: Power system layout for test case 3

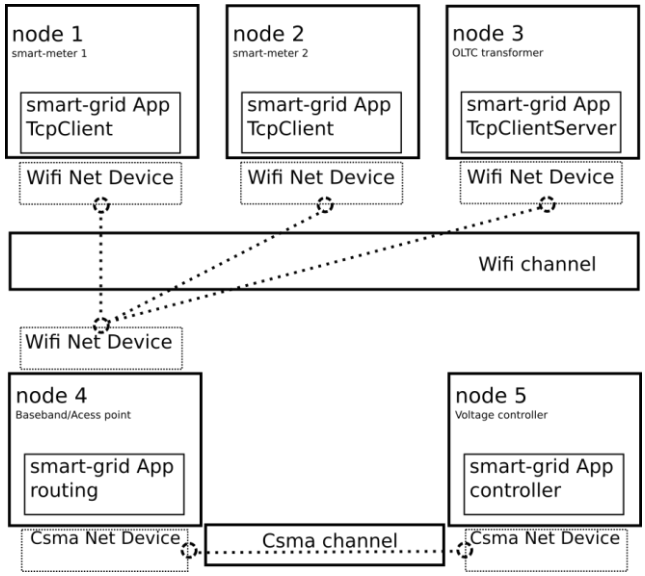


Figure 4.19: Communication network layout for test case 3

Figure 4.21 explains the signal-based interactions and timing synchronization between the actor nodes. The series of events that occur in TC3 can be described as follows. Two voltage metering nodes transmit a voltage measurement with a time period of TC3_period = 15 minutes, with a phase delay between the two measurements.

For voltage metering node A this translates to the following series of events:

1. *e1A:* smart meter transmits a new voltage measurement.
2. *e2A:* after transmission delay Td2A, the voltage measurement package arrives to the access point node, where after a processing delay Tp2A, routes the packet to the controller node.
3. *e3A:* after transmission delay Td3A, the voltage measurement package is routed to the controller node, where after a processing delay Tp3A, a decision is taken based on controller logic.
4. *e4A:* after transmission delay Td4A, the control command package is routed to the access point node, where after a processing delay Tp4A, the package is routed to the OLTC transformer node.

5. *e5A:* after transmission delay Td5A, the control command package is routed to the OLTC transformer node, where after a processing delay Tp5A, the transformer steps either up or down or takes no action depending on the control message.

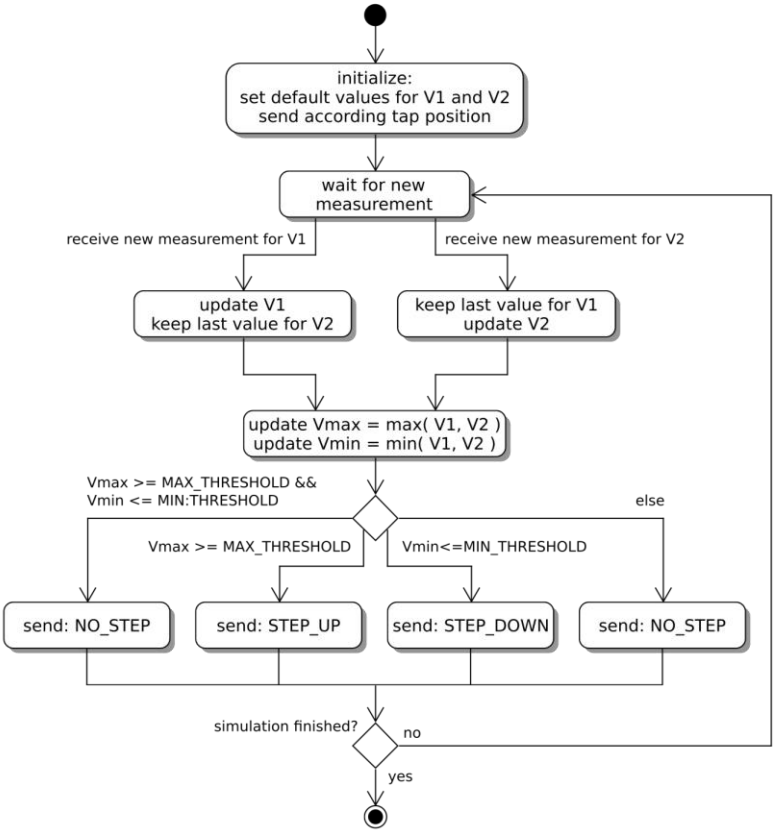


Figure 4.20: Schematic diagram of the voltage controller logic

The same concept applies for voltage metering node B.

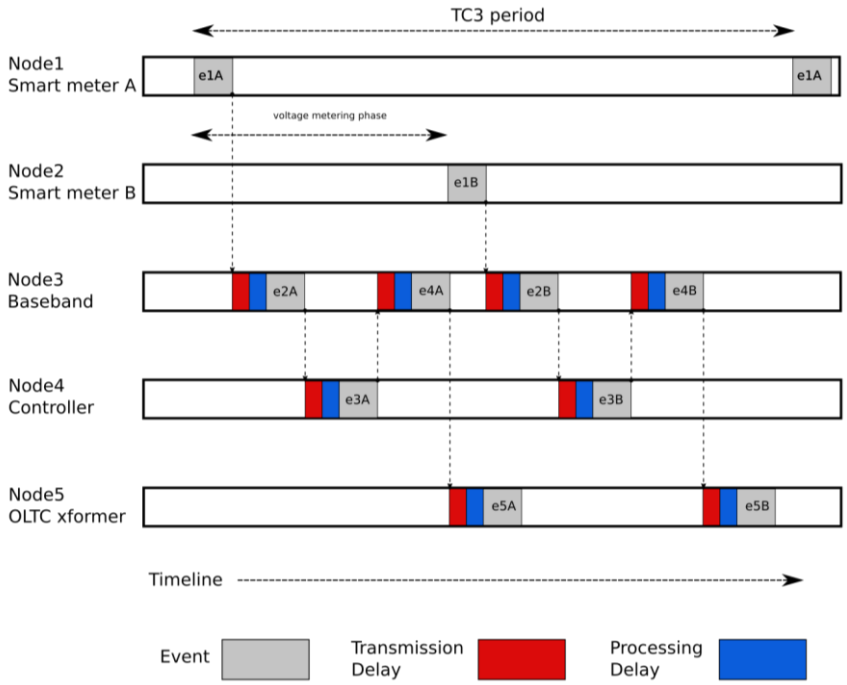


Figure 4.21: Signal-based interactions and timing synchronization between actor nodes

4.3.3.2 Reference Simulation Setup (ns-3 stand-alone)

In order to illustrate the challenges and needs for coupling network simulation with controller and/or power system simulators, a communication network-only simulation of TC 3 has been created. The topology illustrated in Figure 4.19 was created in ns-3. An overview of the created nodes to simulate TC3 with the assigned IP addresses is illustrated by the following diagram:

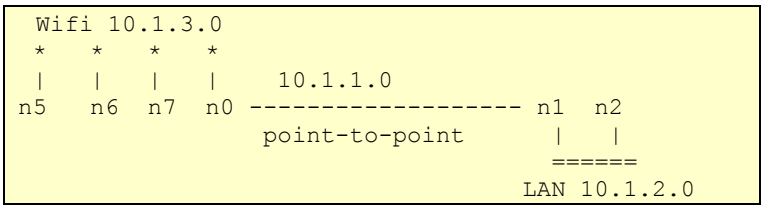


Figure 4.22: Created ns-3 nodes

Where

- n5: smart OLTC Transformer
- n6: smart meter A
- n7: smart meter B
- n0: Wifi access point
- n1: Ethernet router
- n2: smart grid controller
- point to point communication link representing the internet infrastructure

In order to properly simulate TC3, three new application layer models and their helper functions have been created:

- *smartGridSensor*
- *smartGridController*
- *smartGridActuator*

smartGridSensor is set to transmit in a periodic time interval (i.e., T_s = 15 min = 900 s) a specified packet length, representing in the two smart meters sending packets to the smart grid controller, which contains voltage measurements of the attached loads. The two meters are set to send data to the controller with a time difference T_p=1 s. A socket is created and bound on startup of the application. After each T_s a callback is invoked, which sends the voltage measurement packet.

smartGridController is the server application listening for packets from any *smartGridSensor* Node. For every packet that has been received, the *smartGridController* application node sends a control packet with predefined length to a specified *smartGridActuator* node, all using a specified time delay. Two sockets are created on startup, one for incoming connections with a registered callback to handle incoming packets and one for sending control packets to the *smartGridActuator*.

smartGridActuator represents the controlled OLTC transformer of TC3. On startup, the *smartGridActuator* application registers a callback in order to handle incoming control packets. Presently the *smartGridActuator* node notifies, using a log message, when the control packet has arrived. This is left as a placeholder for future handling for co-simulation purposes.

First the topology is created by creating a point to point communication link between an access point and an Ethernet switch, with a data rate and delay parameters representing the internet connection between the two local networks. Subequently, 3 nodes are added to the wifi network, and their respective network interface cards are created and “connected”. Then the controller node on the Ethernet is created as well and eventually connected with the Ethernet switch. After the topolo-

gy generation, the applications are installed at each node with the required set of parameters, and a simulation of 2 periods is run (so 2 times 900 s). A log file using the native ns-3 logging mechanism will show the timing of the communication data exchange.

4.3.3.3 Conceptual FMI-compliant Co-Simulation Setup

Figure 4.23 shows the conceptual co-simulation setup for TC3. The overall system configuration will be split in three parts: the power system (PowerFactory), the communication network (ns-3) and the voltage controller (MATLAB). Each of these parts will be represented by an individual FMU for CS and connected to mosaik. During the execution, mosaik uses the functionality provided by the FMUs and to interact with the simulators and orchestrate the interaction.

As explained above, from the point-of-view of FMI-based co-simulation this test case is the most innovative and challenging. On the one hand, the communication network uses message IDs as inputs and outputs, with a message ID equal to 0 indicating that no signal is present (compare with Section 2). Inside the ns-3 model, these message IDs are associated to dummy messages (of configurable size), which are used to simulate the processing of the message within the communication network. On the other hand, both the power system model and the voltage controller expect real-valued numbers as inputs and outputs and the corresponding tools (i.e., PowerFactory and MATLAB) are not designed to associate a signal with these inputs and outputs, in particular they lack the notion of a signal being absent. Hence, mosaik will implement a dedicated layer for mapping the signals and the corresponding message IDs to the inputs and outputs of the power system model and the voltage controller.

For instance, in the proposed setup there are two different reasons why mosaik would interact with the power system model:

1. Either one of the meters sends a measurement to the controller, or
2. The OLTC receives a new tap position from the controller.

In the first case, a mapping of the measurement to a signal (and a corresponding message ID) is needed, which is then provided as input to the communication network simulator. At the same time, the signal corresponding to the OLTC input would be absent (message ID equal 0), and no new value should be set. In the second case, a signal corresponding to a new OLTC input would be present (message ID not equal 0), which has to be translated and provided as input to the power system simulator. At the same time, the values corresponding to the current meter readings must not be translated to a new signal for the communication network simulator. In both cases, the additional layer on top of the FMI adapter (compare with Figure 4.23) has to decide based upon the current simulation time and the presence/absence of signals which action to perform.

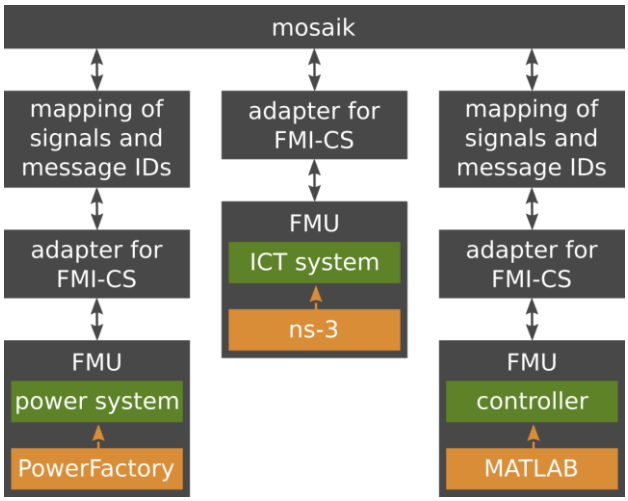


Figure 4.23: Conceptual view of the co-simulation setup for test case 3

4.3.4 Experimental Results

4.3.4.1 Results from Reference Experiment (ns-3 stand-alone)

The standalone simulation in ns-3 is the list of events shown in the log-file output below:

```
At time 1s (smartMeterA) sent 1024 bytes to (Controller) 10.1.2.2 port 9
At time 1.00902s (Controller) received 1024 bytes from  (smartMeterA) 10.1.3.3 port 49153
At time 1.00902s (Controller) sent 1024 bytes to (smartTransformer) 10.1.3.1 port 49153
At time 1.02105s (Transformer) received 1024 bytes from  (Controller) 10.1.2.2 port 49153
At time 2s (smartMeterB) sent 1024 bytes to 10.1.2.2 port 9
At time 2.00525s (Controller) received 1024 bytes from 10.1.3.2 port 49153
At time 2.00525s (Controller) sent 1024 bytes to (smartTransformer) 10.1.3.1 port 49153
At time 2.01714s (Transformer) received 1024 bytes from  (Controller) 10.1.2.2 port 49153
At time 901s (smartMeterA) sent 1024 bytes to 10.1.2.2 port 9
At time 901.014s (Controller) received 1024 bytes from 10.1.3.3 port 49153
At time 901.014s (Controller) sent 1024 bytes to (smartTransformer) 10.1.3.1 port 49153
At time 901.023s (Transformer) received 1024 bytes from  (Controller) 10.1.2.2 port 49153
At time 902s (smartMeterB) sent 1024 bytes to 10.1.2.2 port 9
At time 902.005s (Controller) received 1024 bytes from 10.1.3.2 port 49153
At time 902.005s (Controller) sent 1024 bytes to (smartTransformer) 10.1.3.1 port 49153
At time 902.017s (Transformer) received 1024 bytes from  (Controller) 10.1.2.2 port 49153
```

Figure 4.24: Standalone ns-3 simulation results

It can be seen that latencies are correctly represented in the simulation setup: each sample time (900 s) the smart meters send their measured values according to the previously described communication framework. The presence of delays clearly showcases the efficacy of modelling the communication layer on top of the OLTC controls, and stresses the need to include this behaviour when holistically assessing the system under test.

5 Conclusions and Outlook

Cyber-physical energy systems constitute a significant challenge for system testing and validation. JRA2 takes a co-simulation approach to tackle the associated modelling and simulation challenges. JRA2 currently has 2 ongoing tasks:

- Simulator couplings and interfaces
- Extended model-libraries covering power system and ICT components

The first one is centred on FMI for co-simulation based interfacing approaches, and aims on selecting focal simulation tools, developing FMI-CS interfaces for them, and testing their capabilities using the mosaik co-simulation platform. The second one focuses on FMI for model-exchange, and aims to develop a model library based on FMI-ME covering a diverse set of smart-grid components for various domains. Testing and validation of the smart grid model library is conducted using mosaik. This report described the progress that has been made in both tasks during the first year of research.

5.1 Approach

The general approach in JRA2 was as follows. First the boundary conditions were outlined:

- The implementation of the FMI specification for co-simulation and model exchange
- The application of the corresponding FMUs in the mosaik co-simulation platform

Then the related co-simulation interfacing and modelling challenges were defined:

1. The treatment of cyclic-dependencies between simulators in mosaik
2. Achieve software to hardware coupling using the FMI specification
3. The application of the FMI specification for signal-based synchronisation and modelling

For each objective focal domains were specified. This led to a selection of tools for which the FMI-based interfaces were to be developed: the power system domain (PowerFactory), the ICT domain (ns-3), and the control and automation domain (MATLAB/Simulink and OpenModelica). The above objectives have subsequently been encapsulated into working groups, which aimed to specify elementary test cases according to the holistic test case description method of Network Activity NA5, implement FMI-based interfaces for the focal tools, and develop a smart grid model library.

5.2 Results and Next Steps

5.2.1 Test Case 1

In mosaik, cyclic-dependencies must be tackled at a scenario specification level but also at the synchronisation layer of the framework. Mastering the latter has been the focus of this deliverable by adopting the Gauss-Seidel synchronisation method. The test case itself comprised a wind power plant connected to an external system. During faults, both have significant functional interactions being present in both monolithic and heterogeneous simulation implementations. The monolithic simulation was used as a base experiment because it offers a convenient platform to develop FMI-ME based component models, and it brings about a model that can be used as a reference to compare the co-simulation experiments against.

Next steps include:

- Coupling of PowerFactory to mosaik through FMI-CS using the FMUs of the wind power plant controls developed in the base experiment
- Testing the numerical aspects of scaling the system size (i.e., number of simulators)
- Resolve arising scalability issues.

5.2.2 Test Case 2

With hardware-software coupling it is aimed to utilise the best of two worlds: The flexibility of scrutinising the behaviour of large-scale multi-domain systems by simulation and the accuracy of validated hardware components. As sophisticated real-time HIL aspects are being covered in JRA3, the application of the FMI specification for such applications at an elementary level are covered. This needs an industry grade communication protocol (like Modbus) to couple the hardware components accordingly. Like for TC1, a base experiment has been built in Simulink serving as a reference the other experiments and a platform to contribute to the smart grid component library. An additional step has been made by emulating a controller of an OLTC on an Arduino board while keeping the rest of the system under test on Simulink.

The primary focus of the remainder of this task is to

- Develop an FMI-Modbus interface
- Test the co-simulation performance of this combined virtual/non-virtual experiment.

5.2.3 Test Case 3

As their particulars differ significantly, coupling event-based and continuous time based simulators is an important challenge that needs to be mastered in smart grid simulation. Test case 3 revolves around FMI-based coupling of PowerFactory, ns-3, and MATLAB, and implementing adapters of these simulators in mosaik. The first steps for this TC have been taken by defining the system under test and composing and testing a distributed voltage controller inside PowerFactory and ns-3.

Next steps include:

- Develop & implement coupling of event-based models and simulations using the FMI specification
- Tackle synchronisation issues that arising in the mosaik scheduler

5.3 Outlook

As for JRA2 in general, the dedicated FMI adapters built for mosaik and the focal tools need to be further developed and tested. These are then used as an implement to thoroughly study the behaviour of co-simulation of large-scale scenarios, which need a large number of simulators, and eventually improve their numerical performance. Furthermore, the smart grid library based on the FMI-ME specification is aimed to sustain not just the upcoming research and testing activities in ERIGrid, but also serve the smart grid community in general.

6 References

[1] „The Functional Mock-up Interface," [Online]. Available: https://fmi-standard.org/.

[2] V. Galtier, S. Vialle und et al., „FMI-Based Distributed Multi-Simulation with DACCOSIM," in *Symposium on Theory of Modeling and Simulation (TMS'15)*, Alexandria, VA, USA, 2015.

[3] S. Tripakis und D. Broman, „Bridging the Semantic Gap Between Heterogeneous Modeling Formalisms and FMI," University of California, Berkeley, CA, USA, 2014.

[4] „DIgSILENT PowerFactory," [Online]. Available: http://www.digsilent.com/.

[5] M. Stifter, R. Schwalbe, F. Andren und T. Strasser, „Steady-state co-simulation with Power-Factory," in *2013 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, Berkeley, CA, USA, 2013.

[6] „The FMI++ PowerFactory FMU Export Utility," [Online]. Available: http://powerfactory-fmu.sourceforge.net.

[7] „The FMI++ Library: A High-level Utility Package for FMI-based Software Development," [Online]. Available: http://fmipp.sourceforge.net.

[8] „The Boost C++ Libraries," [Online]. Available: http://www.boost.org/.

[9] „PSCAD/EMTDC power system simulation," [Online]. Available: https://hvdc.ca/pscad/.

[10] „PSS®E: Power Transmission System Planning Software," [Online]. Available: http://w3.siemens.com/smartgrid/global/en/products-systems-solutions/software-solutions/planning-data-management-software/planning-simulation/pages/pss-e.aspx.

[11] S. Uski, B. Lemström, J. Kiviluoma, S. Rissanen und P. Antikainen, „Adjoint wind turbine modeling with ADAMS, Simulink and PSCAD/EMTDC," in *Nordic wind power conference (NWPC'04)*, Gothenburg, Sweden, 2014.

[12] „The FMI++ Python Interface for Windows," [Online]. Available: https://pypi.python.org/pypi/fmipp.

[13] „The ns-3 discrete-event network simulator for Internet systems," [Online]. Available: https://www.nsnam.org/.

[14] S. Ciraci, J. Daily, J. Fuller, A. Fisher, L. Marinovici und K. Agarwal, „FNCS: a framework for power system and communication networks co-simulation," in *Proceedings of the Symposium on Theory of Modeling & Simulation*, Tampa, FL, USA, 2014.

[15] „MATLAB," [Online]. Available: http://www.mathworks.com.

[16] „The Modelon FMI Toolbox," [Online]. Available: http://www.modelon.com/products/fmi-tools/fmi-toolbox-for-matlabsimulink/.

[17] „FMI Kit for Simulink," [Online]. Available: https://www.3ds.com/products-services/catia/products/dymola/fmi/.

[18] „The mosaik Smart Grid co-simulation framework," [Online]. Available: http://mosaik.offis.de/.

[19] S. Schütte, *Simulation Model Composition for the Large-Scale Analysis of Smart Grid Control Mechanisms,* University of Oldenburg, 2013.

[20] I. Hafner, M. Rössler, B. Heinzl, A. Körner, F. Breitenecker, M. Landsiedl und W. Kastner, „Using BCVTB for co-simulation between dymola and matlab for multi-domain investigations of product plants," in *Proceedings of the 9th International MODELICA Conference*, Munich, Germany, 2012.

[21] H. Neema, J. Sztipanovits und M. Burns, „C2WT-TE: A model-based open platform for integrated simulations of transactive smart grids," in *Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, Vienna, Asutria, 2016.

[22] S. Rohjans, E. Widl, W. Müller, S. Schütte und S. Lehnhoff, „Gekoppelte Simulation komplexer Energiesysteme mittels mosaik und FMI," *At-Automatisierungstechnik,* Bd. 62, Nr. 5, pp. 325-336, 2014.

[23] F. Andrén, G. Lauss, R. Bründlinger, P. Svec, C. Seitl und T. Strasser, „Smart Grid Laboratory

Automation Approach Using IEC 61499," in *Distributed Control Applications: Guidelines, Design Patterns, and Application Examples with the IEC 61499*, A. Zoitl und T. Strasser, Hrsg., CRC Press, 2015, pp. 463-482.

[24] A. Eisentraut und A. Brown, „HEATING WITHOUT GLOBAL WARMING: Market Developments and Policy Considerations for Renewable Heat," International Energy Agency, 2014.

[25] A. Thavlov und H. Madsen, „A non-linear stochastic model for an office building with air infiltration," *Int. J. Sustain. Energy Plan. Manag.,* Bd. 7, Nr. 0, pp. 55-66, 2015.

[26] A. Roscoe, I. Abdulhadi und G. Burt, „P and M Class Phasor Measurement Unit Algorithms using Adaptive Cascaded Filters," *IEEE Transactions on Power Delivery,* Bd. 28, Nr. 3, pp. 1447-1459, 2013.

[27] M. Kennedy und A. O'Hagan, „Bayesian calibration of computer models," *Journal of the Royal Statistical Society, Series B,* Bd. 63, Nr. 3, p. 425–464, 2001.

[28] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana und S. Tarantola, Global Sensitivity Analysis: The Primer, Wiley, 2008.

[29] C. Steinbrink und S. Lehnhoff, „Challenges and Necessity of Systematic Uncertainty Quantification in Smart Grid Co-Simulation," in *16th International Conference on Computer as a Tool*, Linz, Austria, 2015.

[30] E. Keogh und A. Mueen, „Curse of dimensionality," in *Encyclopedia of Machine Learning*, Springer, 2011, pp. 257-258.

[31] A. Giunta, S. Wojtkiewicz und M. Eldred, „Overview of Modern Design of Experiments Methods for Computational Simulations," in *41st Aerospace Sciences Meeting and Exhibit*, Reno, NV, USA, 2003.

[32] K. Mets, J. Ojea und C. Develder, „Combining power and communication network simulation for cost-effective smart grid analysis," *IEEE Communications Surveys & Tutorials,* Bd. 16, Nr. 3, pp. 1771-1796, 2015.

[33] W. Li und X. Zhang, „Simulation of the smart grid communications: Challenges, techniques, and future trends," *Computers & Electrical Engineering,* Bd. 40, Nr. 1, pp. 270-288, 2014.

[34] A. Razaq, B. Pranggono, H. Tianfield und H. Yue, „Simulating smart grid: Co-simulation of power and communication network," in *50th International Universities Power Engineering Conference (UPEC)*, Stoke-on-Trent, UK, 2015.

[35] P. Palensky, A. van der Meer, C. Lopez, A. Jozeph und K. Pan, „Co-Simulation of Intelligent Power Systems - Fundamentals, software architecture, numerics, and coupling," *IEEE Industrial Electronics Magazine,* Bd. 11, Nr. 1, pp. 34-50, 2017.

[36] S. Lehnhoff, O. Nannen, S. Rohjans, F. Schlögl, S. Dalhues, L. Robitzky, U. Häger und C. Rehtanz, „Exchangeability of power flow simulators in smart grid co-simulations with mosaik," in *Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, Seattle, WA, USA, 2015.

[37] „IEEE SA - 1516-2010 - IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)-- Framework and Rules," IEEE ComputerSociety, 2010.

[38] „Simantics - An open operating system for modeling and simulation," [Online]. Available: https://www.simantics.org/.

[39] K. M. Chandy und J. Misra, „Distributed Simulation: A Case Study in Design and Verification of Distributed Programs," *IEEE Trans. Softw. Eng.,* Bde. %1 von %2SE-5, Nr. 5, p. 440–452, 1979.

[40] K. M. Chandy und J. Misra, „Asynchronous Distributed Simulation via a Sequence of Parallel Computations," *Commun ACM,* Bd. 24, Nr. 4, p. 198–206, 1981.

[41] D. R. Jefferson, „Virtual Time," *ACM Trans Program Lang Syst,* Bd. 7, Nr. 3, p. 404–425, 1985.

[42] O. Berry und G. Lomow, „Speeding Up Distributed Simulation Using the Time Warp

Mechanism," in *Proceedings of the 2Nd Workshop on Making Distributed Systems Work*, New York, NY, USA, 1986.

[43] S. Olcoz, L. Entrena und L. Berrojo, „A VHDL virtual prototyping technique for mechatronic systems design," in *Int. Conference on Recent Advances in Mechatronics*, Istanbul, Turkey, 1995.

[44] P. L. Marrec, C. A. Valderrama, F. Hessel, A. A. Jerraya, M. Attia und O. Cayrol, „Hardware, soft-ware and mechanical cosimulation for automotive applications," in *Proceedings. Ninth International Workshop on Rapid System Prototyping*, Leuven, Belgium, 1998.

[45] M. Zwolinski, C. Garagate, Z. Mrcarica, T. J. Kazmierski und A. D. Brown, „Anatomy of a simulation backplane," *IEE Proc. - Comput. Digit. Tech.,* Bd. 142, Nr. 6, p. 377–385, 1995.

[46] J. Buck, S. Ha, E. A. Lee und D. G. Messerschmitt, „Ptolemy: A Framework for Simulating and Prototyp-ing Heterogeneous Systems," *Int J. Comput. Simul.,* Bd. 4, p. 155–182, 1994.

[47] A. R. W. Todesco und T. H. Y. Meng, „Symphony: a simulation backplane for parallel mixed-mode co-simulation of VLSI systems," in *33rd Design Automation Conference Proceedings*, Las Vegas, NV, USA, 1996.

[48] W. Sung und S. Ha, Efficient and Flexible Cosimulation Environment for DSP Applications, 1998.

[49] S. Xu und L. F. McGinnis, „Optimistic-Conservative Synchronization in Distributed Factory Simulation," in *Proceedings of the 2006 Winter Simulation Conference*, Monterey, CA, USA, 2006.

[50] X. Wang, S. J. Turner, M. Y. H. Low und B. P. Gan, „Optimistic synchronization in HLA based distributed simulation," in *18th Workshop on Parallel and Distributed Simulation*, Kufstein, Austria, 2004.

[51] X. Qin und J. L. Baer, „A comparative study of conservative and optimistic trace-driven simulations," in *Proceedings of Simulation Symposium*, Phoenix, AZ, USA, 1995.

[52] C. D. Carothers und K. S. Perumalla, „On deciding between conservative and optimistic approaches on massively parallel platforms," in *Proceedings of the 2010 Winter Simulation Conference*, Baltimore, MD, USA, 2010.

[53] M. U. Awais, P. Palensky, W. Müller, E. Widl und A. Elsheikh, „Distributed Hybrid Simulation Using the HLA and the Functional Mock-up Interface," in *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*, Vienna, Austria, 2013.

[54] P. Palensky, E. Widl, M. Stifter und A. Elsheikh, „Modeling Intelligent Energy Systems: Co-Simulation Platform for Validating Flexible-Demand EV Charging Management," *IEEE Transactions on Smart Grid,* Bd. 4, Nr. 4, pp. 1939 - 1947, 2013.

[55] E. Widl, W. Müller, D. Basciotti, S. Henein, S. Hauer und K. Eder, „Simulation of Multi-domain Energy Systems Based on the Functional Mock-up Interface Specification," in *International Symposium on Smart Electric Distribution Systems and Technologies (EDST)*, Vienna, Asutria, 2015.

[56] M. Pazold, S. Burhenne, J. Radon, Her, S. Herkel und F. Antretter, „Integration of Modelica models into an existing simulation software using FMI for Co-Simulation," in *9th International Modelica Conference*, Munich, Germany, 2012.

[57] T. Nouidui, M. Wetter und W. Zuo, „Functional mock-up unit for co-simulation import in EnergyPlus," *Journal of Building Performance Simulation,* Bd. 7, Nr. 3, pp. 192-202, 2014.

[58] V. Jalili-Marandi, V. Dinavahi, K. Strunz, J. A. Martinez und A. Ramirez, „Interfacing Techniques for Transient Stability and Electromagnetic Transient Programs IEEE Task Force on Interfacing Techniques for Simulation Tools," *IEEE Transactions on Power Delivery,* Bd. 24, Nr. 4, pp. 2385-2395, 2009.

[59] A. A. van der Meer, M. Gibescu, M. A. M. M. van der Meijden, W. L. Kling und J. A. Ferreira, „Advanced Hybrid Transient Stability and EMT Simulation for VSC-HVDC Systems," *IEEE Transactions on Power Delivery,* Bd. 30, Nr. 3, pp. 1057-1066, 2015.

[60] F. Plumier, P. Aristidou, C. Geuzaine und T. V. Cutsem, „Co-simulation of Electromagnetic Transients and Phasor Models: a Relaxation Approach," *IEEE Transactions on Power Delivery,* Bd. PP, 2016.

[61] Q. Huang und V. Vittal, „Application of Electromagnetic Transient - Transient Stability Hybrid Simulation to FIDVR Study," *IEEE Transactions on Power Systems,* Bd. 31, Nr. 4, pp. 2634-2646, 2016.

[62] P. Top, Y. Qin und L. Min, „Integration of functional mock-up units into a dynamic power systems simulation tool," in *IEEE Power and Energy Society General Meeting*, Boston, MA, USA, 2016.

[63] S. Uski, „Aggregate wind power plant collection network modeling — Error sources and magnitudes," in *IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)*, Bangkok, Thailand, 2015.

[64] E. Muljadi, C. P. Butterfield, A. Ellis, J. Mechenbier, J. Hochheimer, R. Young, N. Miller, R. Delmerico, R. Zavadil und J. C. Smith, „Equivalencing the collector system of a large wind power plant," in *IEEE Power Engineering Society General Meeting*, Montreal, Canada, 2006.

[65] Y. Yang und X. Zha, „Aggregating wind farm with DFIG in power system online analysis," in *IEEE 6th International Power Electronics and Motion Control Conference*, Wuhan, China, 2009.

[66] „Simscape Power Systems," [Online]. Available: https://www.mathworks.com/products/simpower.html.

[67] P. M. Anderson und A. A. Fouad, Power System Control and Stability, The Iowa State University Press, 1977.

[68] X. Pang, T. S. Nouidui, M. Wetter, D. Fuller, A. Liao und P. Haves, „Building energy simulation in real time through an open standard interface," *Energy & Buildings,* Bd. 117, p. 282–289, 2016.

[69] M. H. Spiegel, F. Leimgruber, E. Wild und G. Gridling, „On using FMI-based models in IEC 61499 control applications," in *Workshop on*, Seattle, WA, USA, 2015.

[70] J. Zehetner, G. Stettinger, H. Kokal und B. Toye, „Echtzeit-Co-Simulation für die Regelung eines Motorprüfstands," *ATZ - Automobiltechnische Zeitschrift,* Bd. 116, Nr. 2, pp. 40--45, 2014.

[71] I. Orue, I. Gilbert, J. Larrieta und J. A. Sanchez, „Making Faults to Protect Power Networks," in *Proceedings CIRED Workshop*, Helsinki, Finland, 2016.

[72] I. Gilbert, P. Mulroy, A. Hurtado, N. Akroud und I. Orue, „Practical Experience of a Partial Discharge Monitoring Application on an Experimentation MV Distribution Network," in *Proceedings CIRED 23rd International Conference on Electricity Distribution*, Lyon, France, 2015.

[73] „Cost-benefit analyses & state of play of smart metering deployment in the EU-27: Accompanying the document Report from the Commission Benchmarking smart metering deployment in the EU-27 with a focus on electricity," European Commission.

[74] S. C. Müller, H. Georg, J. Nutaro und et al., „Interfacing Power System and ICT Simulators: Challenges, State-of-the-Art, and Case Studies," *IEEE Transactions on Smart Grid,* Bd. PP, 2016.

[75] H. Lin, S. S. Veda, S. S. Shukla, L. Mili und J. Thorp, „GECO: Global Event-Driven Co-Simulation Framework for Interconnected Power System and Communication Network," *IEEE Transactions on Smart Grids,* Bd. 3, Nr. 3, pp. 1444-1456, 2012.

[76] K. Hopkinson, X. Wang, R. Giovanini, J. Thorp, K. Birman und D. Coury, „EPOCHS: A platform for agent-based electric power and communication simulation built from commercial off-the-shelf components," *IEEE Transactions on Power Systems,* Bd. 21, Nr. 2, pp. 548-558, 2006.

[77] H. Georg, S. C. Müller, C. Rehtanz und C. Wietfeld, „Analyzing Cyber-Physical Energy Systems: the INSPIRE Co-Simulation of Power and ICT Systems Using HLA," *IEEE Trans.*

Ind. Informat., Bd. 10, Nr. 4, pp. 2364-2373, 2014.

[78] D. Broman, C. Brooks, L. Greenberg, E. A. Lee, M. Masin, S. Tripakis und M. Wetter, „Determinate composition of FMUs for co-simulation," in *Proceedings of the 11th ACM International Conference on Embedded Software (EMSOFT)*, Montreal, Canada, 2013.

[79] D. Broman, L. Greenberg, E. A. Lee, M. Masin, S. Tripakis und M. Wetter, „Requirements for hybrid cosimulation standards," in *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control (HSCC)*, Seattle, WA, USA, 2015.

[80] S. Tripakis und D. Broman, „Bridging the Semantic Gap Between Heterogeneous Modeling Formalisms and FMI," University of California, Berkeley, CA, USA, 2014.

7 Annex

7.1 List of Figures

7.2 List of Tables

7.3 Annex A: Formal Test Case Specifications

7.3.1 Definitions

Holistic testing is the process and methodology for the evaluation of a concrete function, system or component (object under investigation) within its relevant operational context (system under test), corresponding to a purpose of investigation.

A *test case* provides a set of conditions under which a test can determine whether or how well a system, component or one of its aspects is working given its expected function.

A *test specification* aims to clarify the object under investigation, test objective, and by what means a test is to be carried out (i.e., test setup and test design): what is to be tested, why, and how.

An *experiment specification* builds on a given test specification and the specifics of a given lab infrastructure and provides the additional information required to carry out a concrete test or experiment in the lab.

A *component* is a constituent part of a system which cannot be divided into smaller parts without losing its particular function for the purpose of investigation. In a system configuration, components cannot further be divided; connections are established between components.

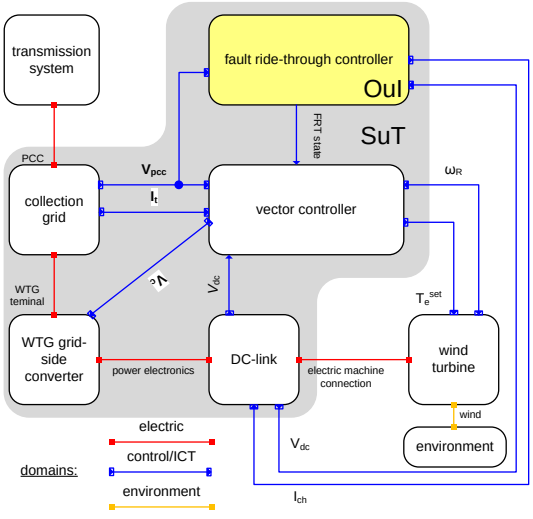
A system is a set of interrelated elements considered in a defined context as a whole and separated from their environment. In a system configuration, a system represents a grouping of components, which may be divided into sub-systems; interfaces between systems a system.

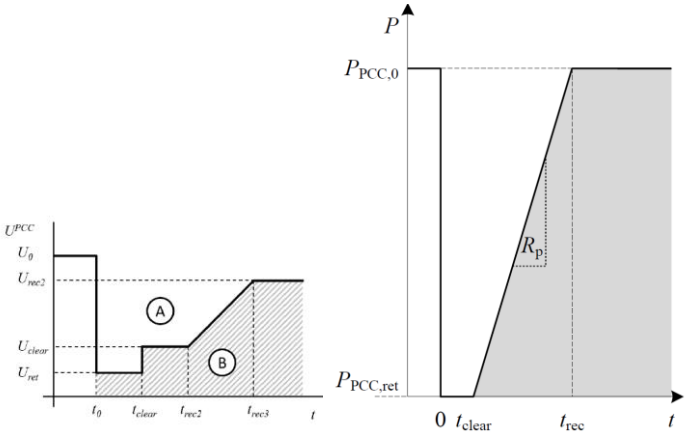
A *domain* is an area of knowledge or activity in the context of smart grids characterized by a set of concepts and terminology understood by practitioners in that area. In a system configuration, domains represent a categorization of the connections between systems; a domain can be divided into sub-domains; domains interface with other domains via components.

A *system(s) configuration* is an assembly of (sub-)systems, components, connections, domains, and attributes relevant to a particular test case.

7.3.2 Formal Specification of Test Case 1

7.3.2.1 Test Case

Narrative “a storyline summarizing motivation, scope and purpose of the test case.”	The test verifies the low-voltage ride through capability of an onshore WPP that is interconnected to a small transmission system. The WPP comprises type 4 wind turbines, which have a fully rated converter interface. The wind power plant must comply to the grid code specification of a low-voltage ride through time against voltage profile. This profile stipulates at the coupling location the minimum profile at which the WPP must stay connected.
System under Test (*SuT*): “a (specific) system configuration that includes all relevant properties, interactions and behaviours (closed loop I/O and electrical coupling), that are required for evaluating an OuI as specified by the test criteria.“ A list of systems, subsystems, components included in the test case or test setup.	The wind park (collection system+wind turbine generators (WTG)) is treated by the system operator as one single entity, the wind power plant. The fault ride-through (FRT) curve is enforced at the coupling point, whereas the grid interface of the converter, its controls, protection, and electromechanical conversion components ensure the compliance to this curve. Hence the SuT comprises: 

	• Coupling point • Collection grid, • Step up transformers, • Converters, • WTG converters • WTG FRT controller • WTG protection schemes • WTG DC links • WTG electrical machines
Object under Investigation (*OuI*) "the component(s) (1..n) that are to be characterized or validated"	The fault ride-through controller
Domain under Investigation (*DuI*) "Identifies the relevant domains or sub-domains of test parameters and connectivity."	• Electrical • control/ICT • primary source (environment)
Functions under Test (*FuT*) "the functions relevant to the operation of the system under test, as referenced by use cases"	• The fault ride-through functionality of the converter, • Fast reactive power support, • The physical response of the external system interacting with the OuI, • Post fault active power recovery functionality, • Normal operating controls of the WTGs, • Current limit of the converters, • Direct voltage control of the DC link of the wind turbine
Function(s) under Investigation (*FuI*) "the referenced specification of a function realized (operationalized) by the object under investigation"	The fault ride-through capability of the converter, fast reactive power support, active power recovery by the WTGs
Purpose of Investigation *(PoI)* "a formulation of the relevant interpretations of the test purpose (e.g., in terms of Characterization, Verification, or Validation)"	Verification of the converter dynamics and the converters' capability to comply to the FRT curve after a voltage dip at the coupling point of the WPP, caused by a 3-phase short circuit upstream in the (sub-)transmission system.
Test criteria "the measures of satisfaction that a need to be evaluated for a given test to be considered successful." A formalization of the purpose of investigation wrt. SuT and FuT attributes.	• Converter must stay connected during and after the fault (see FRT curve for in test metrics section) • Direct voltage operating region is not violated • WTGs remain synchronised to the grid • Transient and frequency stability must be maintained
target metrics (criteria) A numbered list of measures to qualify (quantify) each identified Purpose of Investigation	FRT curve: A: WPP must stay connected, B: WPP may (temporarily) disconnect from transmission system, active power recovery curve: minimum ramping active power rate the WTG has to comply to. Test criterion is violated in case a slower recovery rate is required to maintain synchronism with the external grid. 

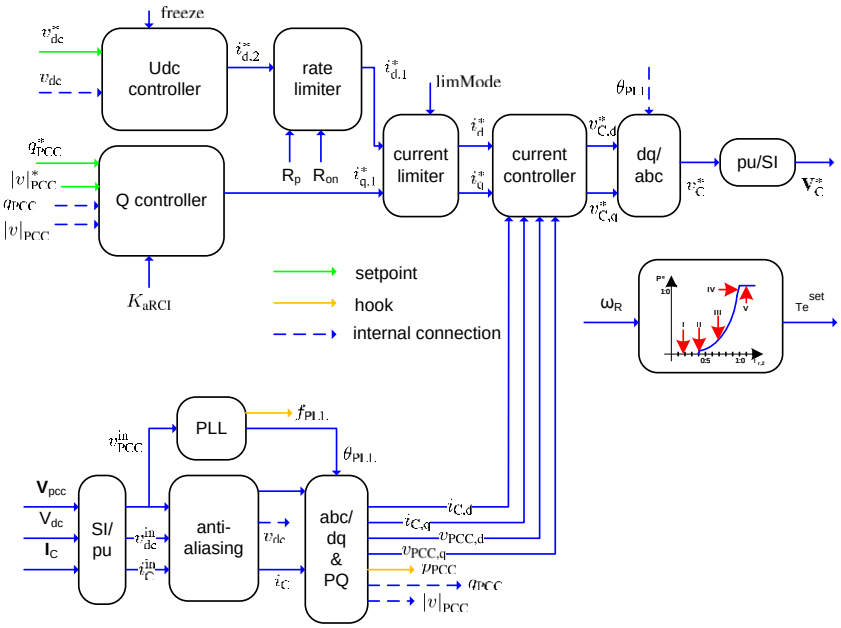
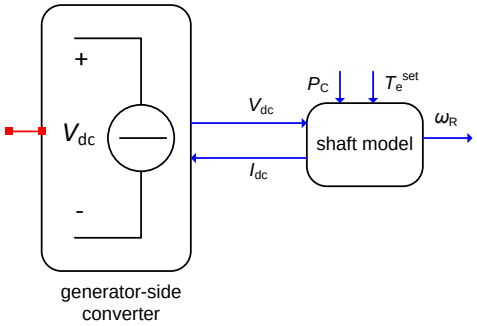
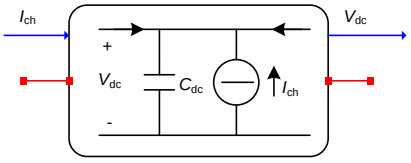
	variability attributes (test factors): identification of the sets of attributes (controllable or uncontrollable parameters) and qualification of the required variability; includes reference to purpose of investigation.	• Short circuit duration (primary versus backup protection) • Short circuit location (causing different dip depths)
	quality attributes (thresholds): with reference to purpose of investigation and/or target metrics, the threshold level required to pass a test or the certainty/precision level (e.g., probabilistic measure) required for the quality of a characterization	FRT curve tests: • short duration (100 ms), FRT control on • short duration (100 ms), FRT control disabled • long duration (200 ms), FRT control engaged • long duration (200 ms), FRT control disabled Test is successful in case the direct voltage remains within +/- 15% of the nominal value, the PLL angle remains within +/- 180 deg., and active and reactive power are properly restored after fault clearance.

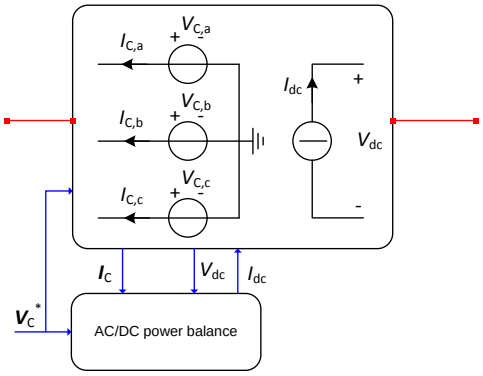
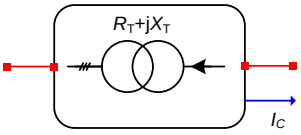
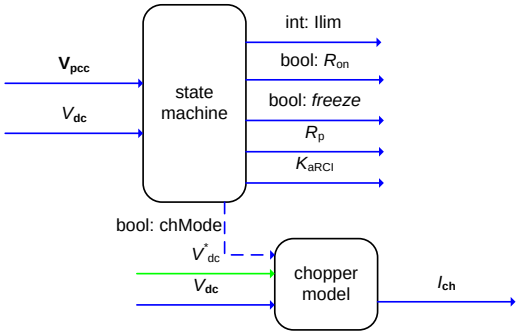
7.3.2.2 Test Specification

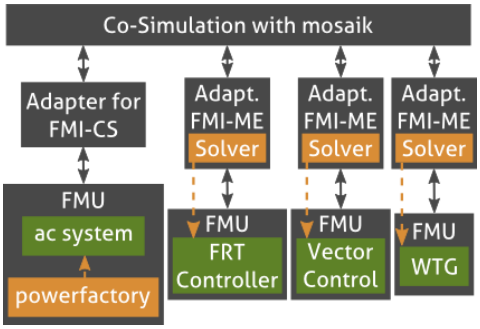
Test Setup (also graphical)	Please consider the grid configuration in the test case descriptions as a reference. The variables between the components are the domain specific interface variables. The connections in the control domain have a directional component. The type, descriptions, and units of the interfacing variables inside the test system are described below. • V_{pcc}: 3x1 array with nodal voltages [V] • I_t: 3x1 array with equivalent branch currents [A] • V_c: 3x1 array with converter voltages [V] • V_{dc}: voltage between + and – pole [V] • T_e^{set}: electric torque setpoint [pu] • ω_R: rotor (or shaft) speed [pu] • I_{lim}: limiting scheme (0=no limiting, 1=d-axis priority, 2=q-axis priority, 3=proportional limiting) [-] • K_{aRCI}: additional reactive current injection gain [pu] • R_p: active power recovery ramp rate [pu/s] • R_{on}: ramp rate on/off [-] • prot: chopper on/off [-]
Input and output parameter	Controllable input parameters: fault duration, fault location, FRT control mode (on/off) Uncontrollable parameters: voltage at coupling point implicitly set by the fault characteristics, wind turbine rotor speed Measured parameters: DC voltage, phase angle of PLL
Test Design	• Determine operating point • Set short circuit location to x • Initiate short circuit at t=0.1 • Clear fault at t=y • Assess test criteria • Vary x and y and repeat (2-5)
Initial system state	The WPP replaces G2 from the IEEE 9-bus system, inheriting its operating point (P,Q at coupling point) from the original system. The initial values of the voltages, active, and reactive powers are given in Table 4.2 in Section 4.1
Evolution of system state and test signals	Test events: See test design. Target metrics: All deterministic cases (so all test criteria for all parameter variations) must be successful. Internal boundary conditions: The IGBT current limit of the converter is 110% of the rated current, the minimum active power recovery rate is 5pu/s, i.e., in 200 ms the wind turbines must be able to recover to the prefault power output. The wind turbine speed and the corresponding pitch controller are not modelled. Their boundaries and time constants are hence not taken into consideration for FRT operation.

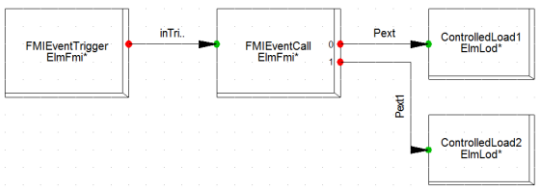
Temporal resolution	• Time constants inside SuT in between 50 µs and 5 sec • Continuous simulation, time step size depends on software experiment • Components exhibit physical behaviour, the FRT controller is a discrete controller (state machine)

7.3.2.3 Experiment Specification

Title	**Experiment 1: Reference simulation**
Experiment realisation	A monolithic approach is used by simulating the system inside Simulink. The models are developed in Simulink and exported as FMUs. These FMUs can be used for other experiments not involving Simulink. The connections between the subsystems and components resemble the interactions shown in the SuT diagram. Inside the subsystems these interactions are represented in a similar fashion.
Experiment Setup (concrete lab equipment)	• All control connections bear directionality, internal and external signals alike • The units of the signals are disregarded • All electrical connections (red) are bi-directional. Vector controller:  Wind turbine model:  DC-link model: 

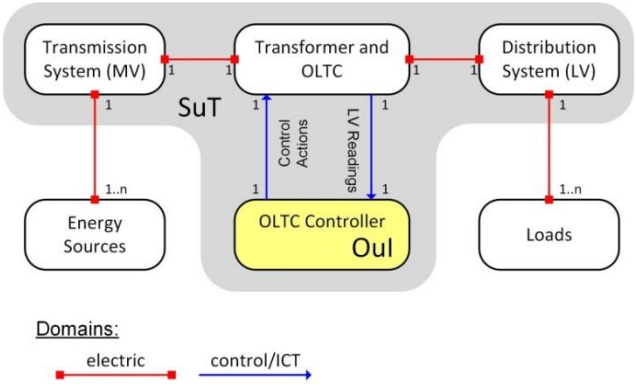
	Grid interface of the WTG:  Collection grid equivalent:  FRT controller: 
Experimental Design	• Three-phase short-circuit location: bus 4 • Variation of fault duration: 100 ms and 200 ms • Variation of control mode: FRT enabled versus disabled

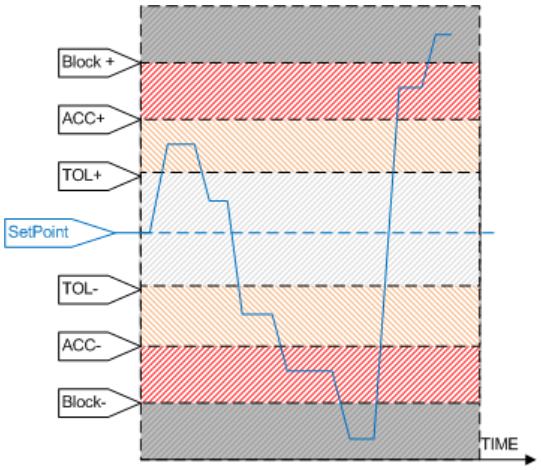
Title	**Experiment 2: FMI-based co-simulation using mosaik**
Experiment realisation	Using the same models and control modes as compared to experiment 1, a co-simulation approach simulation is applied here. The IEEE 9-bus system, the collection grid, and the grid interface of the aggregated wind turbine are modelled in the RMS partition of PowerFactory, while underlying controls are FMUs. The controllers are developed in Simulink and exported as FMUs. These FMUs are coupled in a co-simulation with the help of mosaik.
Experiment Setup (concrete lab equipment)	 The interface between PowerFactory and mosaik is based on FMI for CS, whereas the controllers are being interfaced through FMI for MA. The I/O in PowerFactory has been discussed in Section 2, and is depicted below for the sake of completeness:

	
Experimental Design	• Three-phase short-circuit location: bus 4 • Variation of fault duration: 100 ms and 200 ms • Variation of control mode: FRT enabled versus disabled

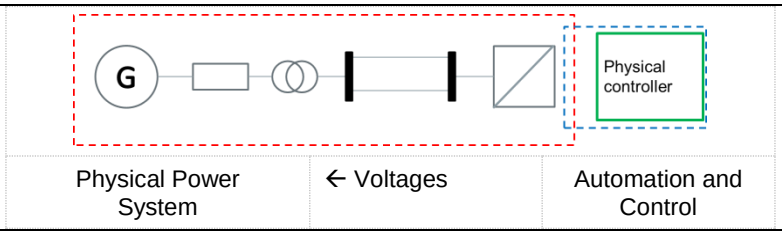
7.3.3 Formal Specification of Test Case 2

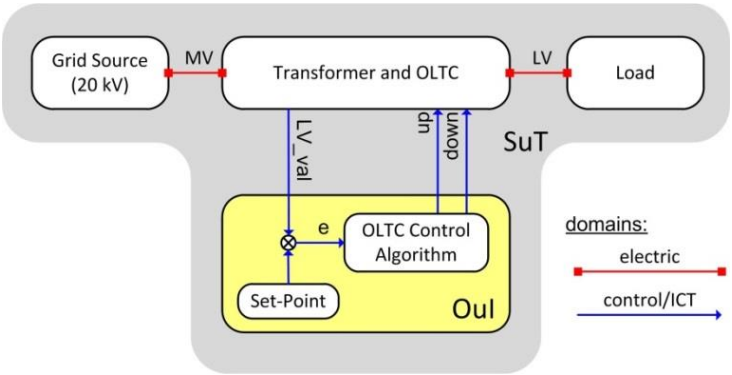
7.3.3.1 Test Case

Narrative "a storyline summarizing motivation, scope and purpose of the test case."	The purpose of this Test Case is to have the first step towards the implementation of a control block using FMU for model exchange functionality. In addition, an interface between a simulated part of the power system and a physical device can be carried out using an embedded controller board such as an Arduino, in which data exchange is performed in both directions. Voltage regulation closed loop can be implemented using not only a simulated controller but also using a physical controller embedded in Arduino card.
System under Test (*SuT*): "a (specific) system configuration that includes all relevant properties, interactions and behaviours (closed loop I/O and electrical coupling), that are required for evaluating an OuI as specified by the test criteria." A list of systems, subsystems, components included in the test case or test setup.	The On Load Tap Changer (OLTC), its electrical vicinity, and its corresponding control algorithm are the concerned system under test. The concerned sub-systems are: • Grid supply • Voltage sensor • Transformer • OLTC's control block • Actuators which operate the OLTC • Load • Communication links between the controls and the transformer The following system diagram illustrates the interaction of the major system components. 
Object under Investigation (*OuI*) "the component(s) (1..n) that are to be characterized or validated"	The OLTC's controller.
Domain under Investigation (*DuI*) "Identifies the relevant domains or sub-domains of test parameters and connectivity."	• Electrical & electronic domains • Control / ICT domain
Functions under Test (*FuT*) "the functions relevant to the operation of the system under test, as referenced by use cases"	• The OLTC control functionality, • Voltage regulation within the admissible limits, • Control functionality implemented in an embedded hardware device (Arduino card). • Control functionality exported into an FMU for model exchange

Function(s) under Investigation (*FuI*) "the referenced specification of a function realized (operationalized) by the object under investigation"	The voltage control function.
Purpose of Investigation *(PoI)* "a formulation of the relevant interpretations of the test purpose (e.g., in terms of Characterization, Verification, or Validation)"	The described tests aim at verifying the functionality of various OLTC controller realizations. Each realization covers a different design and development stage of the controller under test. Especially, the capability of the OLTC controller realizations to successfully maintain the terminal voltage at the high voltage side of the transformer within the specified boundaries is evaluated. Certain controller realizations such as the encapsulation into an FMU or a realization by an embedded device may impose some major challenges and may influence the behaviour of the controller. In order to characterize each realization and to verify whether the control logic is still functional, the described tests will be performed.
Test criteria "the measures of satisfaction that a need to be evaluated for a given test to be considered successful." A formalization of the purpose of investigation wrt. SuT and FuT attributes.	OLTC control block must be able to exchange data with other blocks within the specified time-step using its software/physical versions of implementation. The voltage operation region must not be violated
target metrics (criteria) A numbered list of measures to qualify (quantify) each identified Purpose of Investigation	Voltage regulation regions: 
variability attributes (test factors): identification of the sets of attributes (controllable or uncontrollable parameters) and qualification of the required variability; includes reference to purpose of investigation.	• Voltage set-point • Load variations
quality attributes (thresholds): with reference to purpose of investigation and/or target metrics, the threshold level required to pass a test or the certainty/precision level (e.g., probabilistic measure) required for the quality of a characterization	A voltage variation test outcome is considered to be positive if and only if • The admissible voltage range of [TOL-,TOL+] is re-established and maintained within the first 10 s after the induced disturbance, and • The system operates without system disconnections (shutdowns).

7.3.3.2 Test Specification

Test Setup (also graphical)	



The types, descriptions, and units of the interface variables are as follows:

Name	Unit	Description
MV	[V]	Nodal voltages
LV	[V]	Nodal voltages
LV_val	[V]	Voltage measurement
set-point	[V]	Voltage set-point
e	[V]	Voltage error
up	[-]	Next transformer level (coil)
down	[-]	Previous transformer level (coil)

Input and output parameter

Controllable input parameters:

- Voltage set-point
- Load value, delays

Uncontrollable input parameters:

- Transformer taps
- Main source voltage value and frequency

Measured parameters:

- Load voltage value
- Controller states with respect to “target metrics” (criteria) as follows

Concept	Value	unit	Notes
SetP	420	V	Set point
Tap Step	2.5%*	V	Tap Step
BLQ+	20%*	V	Block Control
ACC+	10%*	V	Step down (Delay)
TOL+	+5%*	V	Step Down
TOL-	- 5%*	V	Step Up
ACC+	-10%*	V	Step Up (Delay)
BLQ-	-20%*	V	Block Control

* depends on the rates of the deployed transformer.

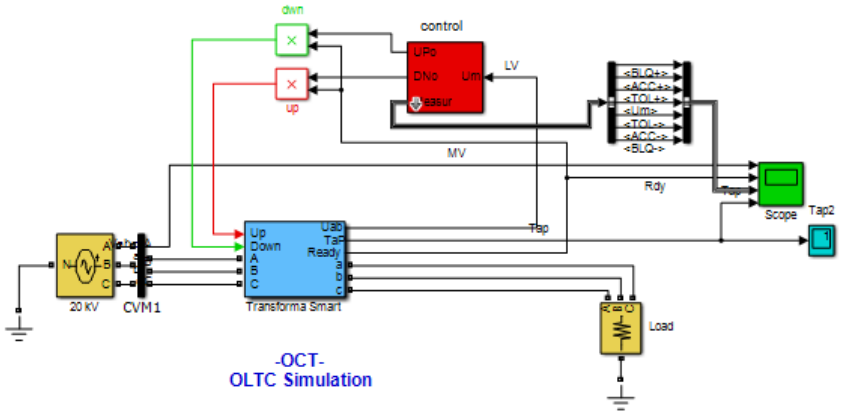
Test Design

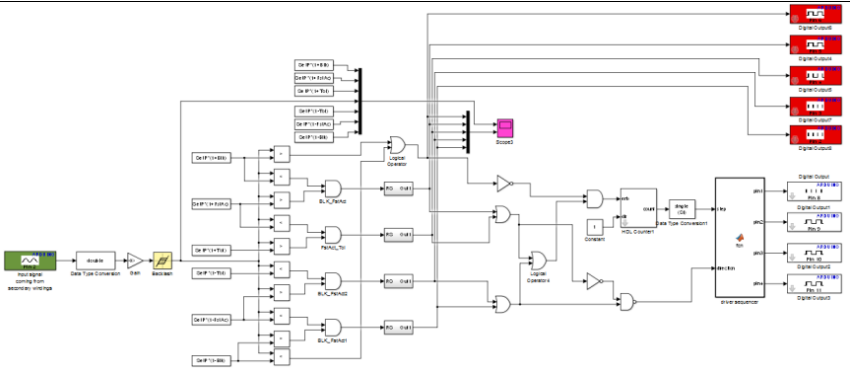
The test aims at validating and verifying the voltage control functionality of the controller in various experiments. The controller must be able to maintain a low voltage value within a certain voltage region. In order to assess the functionality, first, a predefined set of border cases will be applied. For each test iteration, one input variation from the predefined set is chosen. The test criteria which are defined above will then be applied to evaluate each test run. The predefined set of inputs features an improved comparability of the experiments.

If some test iteration show difficulties such as failed test criteria, additional test iterations with manually adjusted inputs will be scheduled. Such additional test iterations

	are used to gain further insights. Since the controllable inputs of each test run are mostly defined beforehand, a systematic/factorial test is performed. In particular, the following steps will be executed in each experiment: • Determine the operating set-point • Wait until the output is stabilized • Vary the MV input voltage and/or the set-point according to the current border case • Assess test criteria • Repeat 2-4 until all predefined border cases are tested The following border cases are defined: • Maximally decrease MV input voltage, use constant voltage set-point • Maximally increase MV input voltage, use constant voltage set-point • Decrease voltage set-point from maximal to minimal value, do not artificially vary MV input voltage • Increase voltage set-point from minimal to maximal value, do not artificially vary MV input voltage
Initial system state	Initial power flow conditions: Load voltage value (output) = voltage set-point
Evolution of system state and test signals	Test events: See test design. Target metrics: The test is successful in the case of the load voltage is always regulated within the interval [TOL-,TOL+] in both cases soft/hard OLTC control regardless of load voltage variation.
Temporal resolution	The simulation of virtual components uses fixed time steps of 0.1 ms. Any communication with external equipment such as the embedded controller may be done with a lower time resolution of up to 200 ms.

7.3.3.3 Experiment Specification

Title	**Experiment 1: Software OLTC controller**
Experiment realisation	In the first initial experiment, the SuT is simulated by a monolithic model which covers all components of the test setup. The monolithic model and the simulation results are used to test the functionality of the controller and to create a reference for further experiments. The initial model has been built in Simulink using blocks contained in the Physical Systems Simulation toolbox (Simscape). The tools were chosen in order to simulate the whole system and to be able to optimize the simulation via a phasor solver to be as accurate as possible.
Experiment Setup (concrete lab equipment)	The Simulink block diagram of the SuT is shown below.  The OLTC controller (red block) is tested in a monolithic simulation and is implemented as follows:

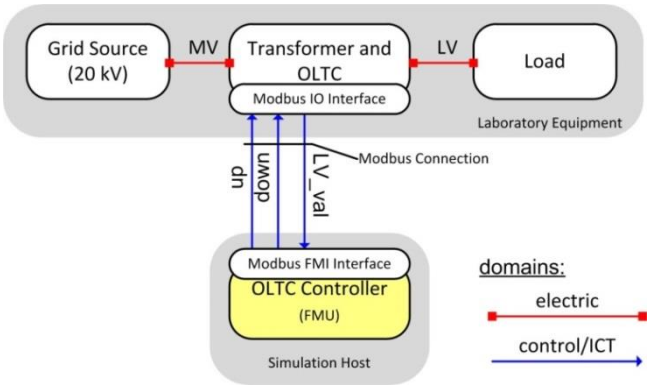


The following types are used to implement the interface variables which are described in the test setup section:

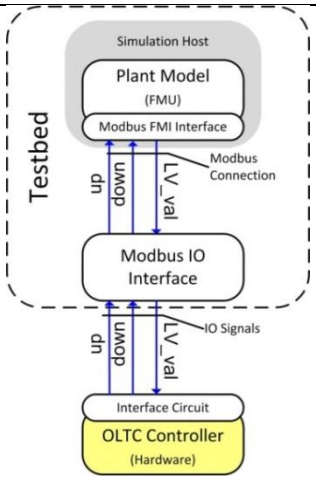
Name	Type	Unit	Description
MV	double, 3x1 array	[V]	Nodal voltages
LV	double, 3x1 array	[V]	Nodal voltages
LV_val	double	[V]	Voltage measurement
set-point	Uint	[V]	Voltage set-point
e	double	[V]	Voltage error
up	bool	[-]	Next transformer level (coil)
down	bool	[-]	Previous transformer level (coil)

Title	**Experiment 2: Hardware OLTC controller**
Experiment realisation	In the second experiment, the SuT is separated in two parts. The hardware part is composed by the controller of the medium voltage transformer built in an open-source electronics platform along with the transformer itself and the software part includes the simulation model built in Simulink. Once the first experiment is done correctly, we choose a new model based in a discrete solver (no continuous states) with fixed-step size=0.0001 simulation to facilitate its integration into the open-source electronics platform featuring an AT-Mega2560 microcontroller. Hence, the SuT is split into a control part being emulated on an Arduino, and an electrical part simulated inside Simulink. The interface variables are listed in the test specification. The controller hardware and the simulated components will be interfaced by means of Simulink and Matlab. The control program may be extended such that data exchange between the controller and the Simulink instance can be achieved.
Experiment Setup (concrete lab equipment)	The Simulink block diagram of the SuT is identical to the block diagram of the first experiment. The OLTC controller is implemented in an Arduino card in order to test the communication between soft/hard enviroments. The implementation of the OLTC controller is also identical to the controller which is shown in experiment 1.

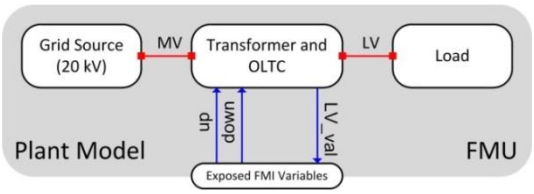
Title	**Experiment 3: OLTC controller as FMU-ME**
Experiment realisation	This test concerns transferring the previously tested controller into an FMU model exchange block. Then this block can be simulated with other system blocks or components in order to investigate and compare its behaviour to the case in Experiment 1. The experiment covers a prototyping stage which integrates the physical plant (i.e., the OLTC transformer and the measurement equipment) with a simulated controller. Since the embedded controller on the Arduino platform is not capable of directly executing an FMU, an industrial communication protocol will be used to connect the simulation host which executes the controller to the IO interfaces driving the plant.
Experiment Setup (concrete lab equipment)	The grid supply, the OLTC including its electircal actuators, the transformer itself, the voltage sensors, and the electircal load are physically present within the laboratory setup. The OLTC and the voltage readings can be accessed via industrial IO devices. As a communication protocol, Modbus RTU over EIT/TIA 485 or Modbus TCP/IP is used. In

	any case, the simulation host (x86-based Workstation) which is connected to the IO devices may actively poll all data points. Hence, the IO devices act as Modbus slaves. On the simulation host, the exported FMU and all interface components are executed such that a closed-loop operation between the simulated controller and the plant is achieved. The interface components periodically poll the voltage readings via Modbus and send the outputs of the controller to the actuators. The actual communication period depends on the capabilities of the equipment. A period of 100 ms to 200 ms is targeted. The following graphic illustrates the laboratory setup. The realization of the controller FMI is specified by the Simulink diagram in Experiment 1. The plant model in the previous experiment (the gird source, smart transformer, and load) resembles the laboratory setup in the current experiment.  In order to evaluate the experiment, the parameters listed in the test specification will be recorded by the laboratory equipment.

Title	**Experiment 4: FMI-based OLTC controller hardware in the loop**
Experiment realisation	In order to test the embedded OLTC controller including its IO interfaces, it will be coupled to a testbed which mimics the behaviour of the plant. The testbed mainly consists of industrial IO interface which drive the IO lines of the controller and a plant model which is interfaced via the FMI for model exchange. The simulated low voltage value (LV_val) is transferred to the industrial IO interface which converts them to an analogous signal. The signal is then processed by the physically available OLTC controller. Likewise, the control commands of the OLTC controller are sensed by the IO interface of the testbed and transferred to the plant model. The software of the controller is still exported into the embedded development board via Simulink, but in contrast to Experiment 1, the controller is interfaced by its dedicated analogous and digital IO lines. Hence, the embedded controller can be tested comprehensively without the need of adding a debugging link to the control software.
Experiment Setup (concrete lab equipment)	The standalone control software is uploaded from the Simulink Model into the embedded controller board which is also used in Experiment 1. The analogous LV_val IO line of the controller is connected to an industrial IO interface which drives the signal. Similarly, the digital output lines of the embedded controller are connected to the industrial IO interface. If the voltage or current ratings of the interfaces do not match, an appropriate interface circuit will be deployed. The IO interface of the testbed is accessed via an industrial communication protocol. Modbus RTU over EIT/TIA 485 or Modbus TCP/IP is used between the IO device and the simulation host which executes the plant model. The simulation host periodically polls the digital inputs and sets the analogue output via Modbus. The polling period depends on the capabilities of involved devices but a period of 100 ms to 200 ms is targeted. Interface components on the simulation host run the plant model which is encapsulated into an FMU and manage the communication between the IO devices and the FMU. The FMU itself contains an exported version of the Simulink plant model of Experiment 1. The following graphic illustrates the experimental setup.



The plant model consists of the following components which are exported from the initial Simulink model of Experiment 1.

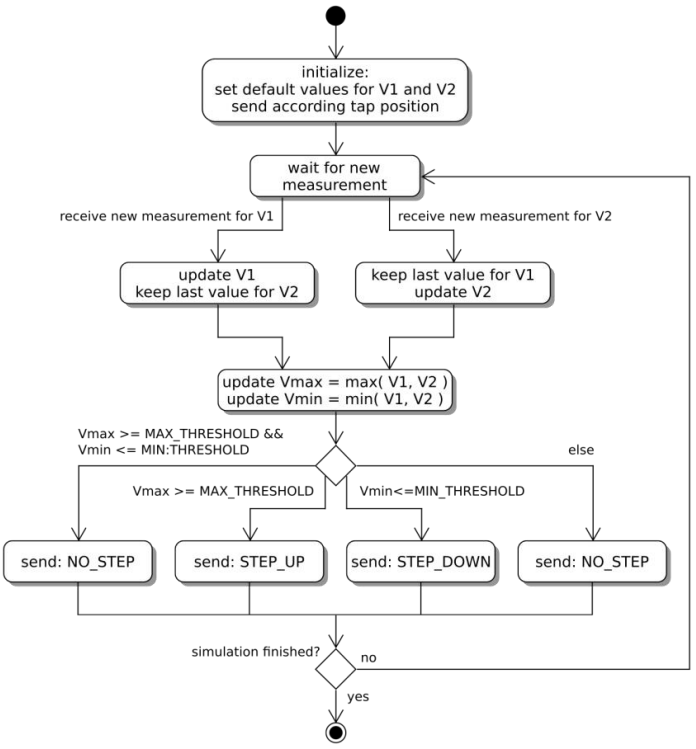


In order to evaluate the experiment, the parameters listed in the test specification will be recorded by the simulation host.

7.3.4 Formal Specification of Test Case 3

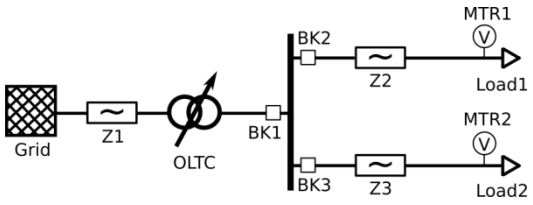
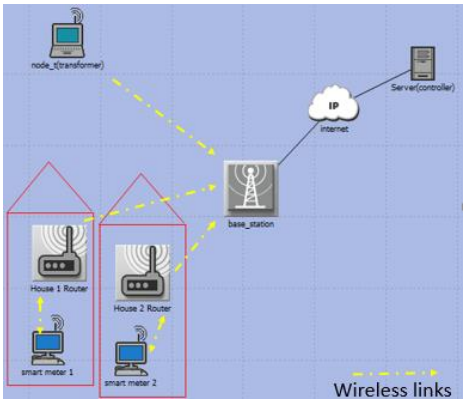
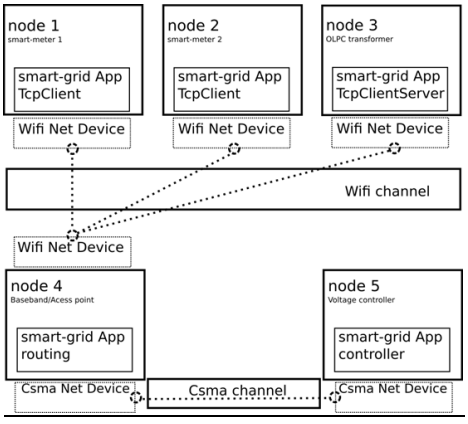
7.3.4.1 Test Case

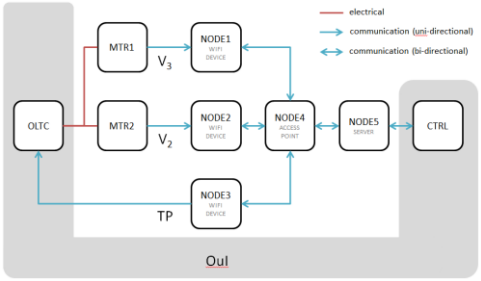
Narrative "a storyline summarizing motivation, scope and purpose of the test case."	The test deals with the impact of communication delays in a simple low voltage distribution grid, where two meters send information about local voltage levels via a communication network to a remote controller. Based on these meter readings, the controller actuates the tap position of an OLTC transformer. The aim of this test case is to demonstrate and assess the effect of long communication delays on the actuation pattern of the controller and the resulting effects in the low voltage distribution system. Since this test case aims at providing an illustrative example of what problems may arise from poor controller designs, a fundamentally flawed approach for handling long delays is implemented for the controller. Moreover it is demonstrated that a more realistic simulation is needed for distributed and centralized smart grid control algorithms, as well as a benchmark for evaluating communication network technologies and topologies for use in smart grid applications is demonstrated.
System under Test (*SuT*): "a (specific) system configuration that includes all relevant properties, interactions and behaviours (closed loop I/O and electrical coupling), that are required for evaluating an OuI as specified by the test criteria." A list of systems, subsystems, components included in the test case or test setup.	<u>Low voltage distribution system:</u> • OLTC MV/LV transformer • 2 loads with voltage meters (measurement devices and/or smart meters) <u>Controller:</u> • Simple rule-based controller, changing the tap of the transformer depending on the meter readings • Dedicated server, separate from transformer and meters <u>Communication network:</u> • Base station/access point for wireless communication • 1 node (sender) for each voltage meter, wireless connection to the base station • 1 node (receiver) for the transformer, wireless connection to the base station

	• 1 node (sender/receiver) for the server (controller), wired internet connection to the base station
Object under Investigation (*OuI*) "the component(s) (1..n) that are to be characterized or validated"	• Boltage meters • OLTC transformer • Communication channel (protocol, delay, throughput, utilization)
Domain under Investigation (*DuI*) "Identifies the relevant domains or sub-domains of test parameters and connectivity."	• ICT • Control
Functions under Test (*FuT*) "the functions relevant to the operation of the system under test, as referenced by use cases"	• The functionality of the voltage controller • Voltage regulation within the admissible limits
Function(s) under Investigation (*FuI*) "the referenced specification of a function realized (operationalized) by the object under investigation"	The voltage controller logic can be described by the following algorithm (see below). Attention: The controlles calculates and transmits tap position every time a new measurement arrives (using the latest values available). In this system configuration, more complex rules usually adopted for OLTC operation (i.e., voltage dead band, maximum allowed steps, etc.) are neglected. 
Purpose of Investigation *(PoI)* "a formulation of the relevant interpretations of the test purpose (e.g., in terms of Characterization, Verification, or Validation)"	• The test will characterize the robustness of the controller against (long) communication delays. • Identify system performance, for different communication network performance scenarios, such as performance over a saturated communication channel. • Investigate performance over different protocols and communication network utilization conditions. • Validate the scaling of the control algorithm given certain topology and infrastructure limits.
Test criteria "the measures of satisfaction that a need to be evaluated for a given test to be considered successful." A formalization of the purpose of investigation wrt. SuT and FuT attributes.	• The resulting actuation of the OLTC transformer has to result in acceptable operational conditions (voltage levels according to grid codes) • Tap changes should not occur more frequently than once every 15 minutes

	target metrics (criteria) A numbered list of measures to qualify (quantify) each identified Purpose of Investigation	• Communication network average delay, QoS • Communication channel utilization
	variability attributes (test factors): identification of the sets of attributes (controllable or uncontrollable parameters) and qualification of the required variability; includes reference to purpose of investigation.	• Mean and standard deviation of delays (order of minutes) • Frequency of voltage measurements (every 5-15 minutes) • Time offset (□t) between voltage measurements • Controller dead time (order of seconds to minutes) • Voltage meter transmitted packet size • Network protocol stack • Access point to controller line throughput and utilization (existing network traffic).
	quality attributes (thresholds): with reference to purpose of investigation and/or target metrics, the threshold level required to pass a test or the certainty/precision level (e.g., probabilistic measure) required for the quality of a characterization	• Voltage levels at loads and transformer have to stay within predefined levels (stricter than the range for slow voltage variations in EN 50160, for example ± 1.08 p.u.) • Tap changes should not occur more frequently than once every 15 minutes

7.3.4.2 Test Specification

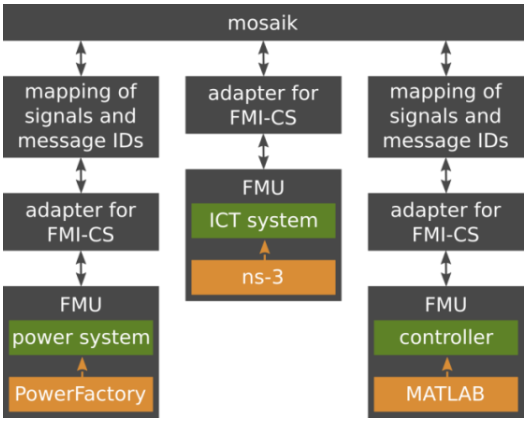
Test Setup (also graphical)	Low voltage distribution system:  Communication network:  Communication network in ns-3 terms topology: 

	Interfaces:  • V_1, V_2 … voltage meter readings • TP … tap position
Input and output parameter	controllable input parameters: • Tap position uncontrollable parameters: • Power consumption of loads measured parameters: • Voltages at loads
Test Design	• The meters send their measurements in regular intervals to the controller. • Whenever the controller receives a new individual measurement, new values for the tap position will be calculated with the newest values available. • Whenever the controller calculates a new value for the tap positions, it is sent to the OLTC transformer. • Communication channel throughput and utilization between the access point and the controller is varied in order to affect communication delay and overall performance.
Initial system state	• Start simulation in uncritical state (no over- or undervoltages, medium tap position) • Prepare load profiles such that under- or overvoltages occur during the simulation
Evolution of system state and test signals	See test design.
Temporal resolution	Low voltage distribution system: • Continuous RMS simulation (*check*: slow voltage variations > Load flow may-be it is sufficient) • Exact time step size depends on software experiment Communication network: • Event based simulation • Events happen at random times • Events are triggered by voltage measurement and controller nodes smart grid applications. Controller: • Discrete controller (state machine) • Returns results immediately without causing a delay in simulation time • Recalculates and transmits tap positions every time a new measurement signal is received (using latest available values)

7.3.4.3 Experiment Specification

Title	**Reference simulation (ns-3 stand-alone)**
Experiment realisation	This reference simulation is a stand-alone simulation of the communication system (implemented solely in ns-3) without considering the immediate effects on the power system. It serves as a baseline for the communication network performance
Experiment Setup (concrete lab equipment)	3 new application layer models and their helper functions have been created: • smartGridSensor • smartGridController • smartGridActuator

	smartGridSensor is set to transmit in a periodic time interval a specified packet length, representing in TC3 the two smart meters sending packets to the smart grid controller containing voltage measurements of the attached loads. A socket is created and bound on startup of the application. And every Ts amount of time a callback that sends the voltage measurement packet is called. smartGridController is the server application listening for packets from any smartGridSensor Node. For every packet received, after a specified time delay, smartGridController application node sends a control packet of a defined packet length to a specified smartGridActuator node. Two sockets are created on startup, one for incoming connections with a registered callback to handle incoming packets and one for sending control packets to the smartGridActuator. smartGridActuator represents the controlled OLTC transformer. The smartGridActuator application registers on startup a callback in order to handle incoming control packets. Currently the smartGridActuator node notifies with a log message when the control packet has arrived, and this is left as a placeholder for future handling for co-simulation purposes.
Experimental Design	First the topology is created, by creating a point to point link between an access point and an Ethernet switch, with a Data rate and a delay parameters representing internet connection between the two local networks. Then 3 nodes are added to the WiFi network, and their respective network interface cards are created and "connected". Then the controller node on the Ethernet is also created and connected with the Ethernet switch. After the topology generation, the applications are installed at each node with the required set of parameters, and a simulation of 2 periods is run. A log file using the native ns3 logging mechanism will show the timing of the communication data exchange. smartGridSensor is set to transmit in a periodic time interval Ts=15min(900sec). The two meters are set to send data to the controller with a time difference Tp=1sec.

Title	**Co-simulation experiment**
Experiment realisation	An FMI-based co-simulation approach is applied here with the help of mosaik. The models of all sub-system, i.e., the power system, the communication network and the voltage controller, are coupled via FMUs for CS. Mosaic takes care of bridging the semantic gaps between the various models (signal-based vs. continuous-time).
Experiment Setup (concrete lab equipment)	 The co-simulation interfaces for PowerFactory, ns-3 and MATLAB have been discussed in Section 2
Experimental Design	Specified one level up

7.4 Annex B: FMI-ME Appraisal

In the initial literature review, it was identified that MATLAB/Simulink and OpenModelica (OM) offer the capability to export developed models as FMUs for ME. OM inherently offers the capability to export FMUs, however an additional toolbox is required for MATLAB/Simulink. It was further identified that there are two toolboxes available for MATLAB/Simulink to export models as FMUs for ME: *a)* FMI Toolbox from Modelon and *b)* FMI ToolKit from Dassault Systems. Three short studies were undertaken and a summary of them has been presented below.

7.4.1 Case Study 1: FMI Toolbox/ToolKit Performance Evaluation

As part of the study, a simple feedback controller model was developed in MATLAB/Simulink (as shown in Figure 7.1a. Three different configurations were studied: *(i)* feedback controller with all elements from MATLAB/Simulink, *(ii)* the PID controller (kp=1, ki=0, kd=0) was exported as FMU from MATLAB/Simulink using the FMI toolbox from Modelon and imported within MATLAB/Simulink using the FMI toolbox from Modelon and *(iii)* the PID controller was exported as FMU from MATLAB/Simulink using the FMI Toolkit from Dassault Systems and imported within MATLAB/Simulink using the FMI Toolkit from Dassault Systems.

A 30 s simulation with time step as 500 µs and 0.5 s was conducted on the setup as shown in Figure 7.1a and the results have been presented in Figure 7.1b (for 500 µs) and Figure 7.1c (for 0.5 s). From the results presented, it can be observed that for both the time steps, the outputs of the PID and the TF for the three configurations are identical.

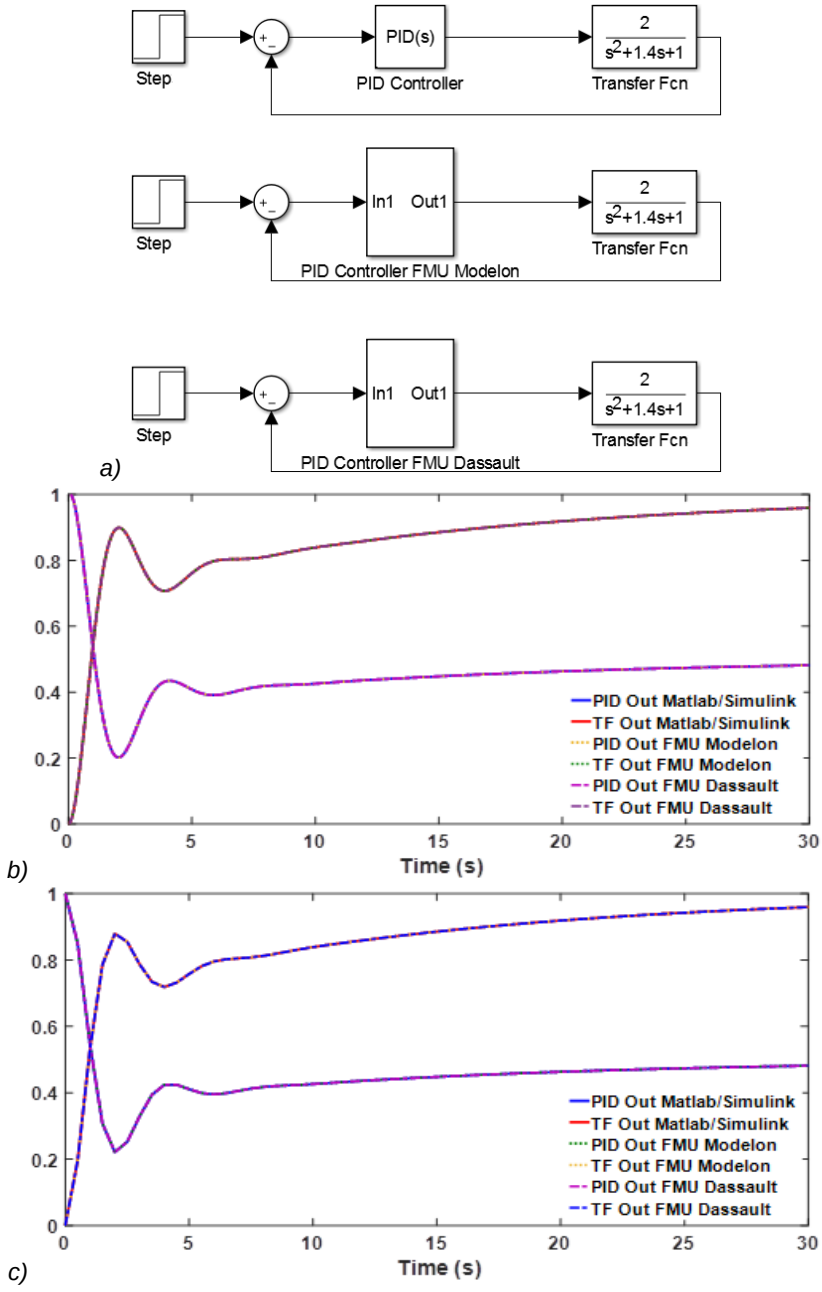


Figure 7.1: FMI Toolbox/Toolkit performance evaluation: a) feedback controller configurations, b) simulation results for time step 500 µs, c) simulation results for time step 0.5 s

7.4.2 Case Study 2: SimPowerSystems Model Export

From Case Study 1, it was evident that the two tools for FMU export from MATLAB/Simulink allow for control systems to be exported without any issue. As the interest of the consortium was to develop a model library for smart grid applications, it was important to assess the ability to export physical components as FMUs. The system under consideration for this study is shown in Figure 7.2. The goal was to export the Synchronous Machine (SM) as an FMU. It was identified that an FMU cannot have electrical connections with the outside world. However, if only control signals are to be exchanged with the outside world, the FMU can comprise of electrical components.

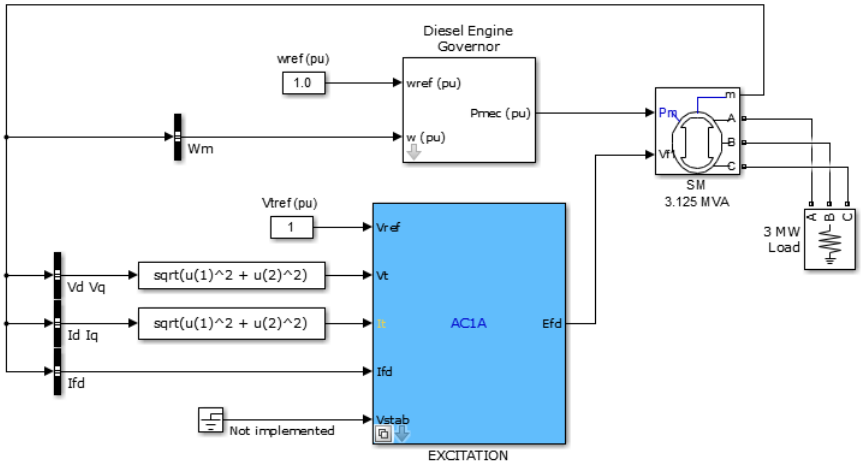


Figure 7.2: SimPowerSystems Model for case study 2

This issue is not confined to electrical connections only but to all physical components that use acausal terminals, as FMU export is limited to models with causal terminals. This does present limitations to what can be simulated using FMUs, however, workarounds to incorporate physical models can be investigated. This makes the simulation setup quite complex and defeats the purpose of easy reusability of models. Within the ERIGrid project, this limitation should be explored further and the capabilities of FMI-ME to support electrical components expanded. One of the possible ways is to develop “adapters” that allow physical signals to be converted to control signals and vice-versa. The capability of OM for this purpose was not investigated.

7.4.3 Case Study 3: Testing Tool Independent Modelling

The aim of the short study undertaken was to ensure the interoperability between MATLAB/Simulink and OM. As part of the study, the simple feedback controller model (as in Figure 7.1a) was chosen. The model was built in both OM and MATLAB/Simulink. As can be seen from Figure 7.1, the feedback controller comprises of a Step function, a PID controller and a Transfer Function (TF). First, each of the elements were exported individually from OM and imported within MATLAB/Simulink. For this purpose, the FMU toolbox from Modelon was utilized. The time step for the simulation was chosen as 500 µs. The results from the comparative study were in close proximity. A similar exercise was undertaken, where individual elements were exported from MATLAB/Simulink and imported within OM. An example representation of the comparative study, where the PID controller exported from MATLAB/Simulink is imported within OM, is shown in Figure 7.3a and the results presented in Figure 7.3b. As can be seen from the results, the performance of the two systems, one built entirely of OM elements and the other with PID controller as FMU from MATLAB/Simulink, are identical.

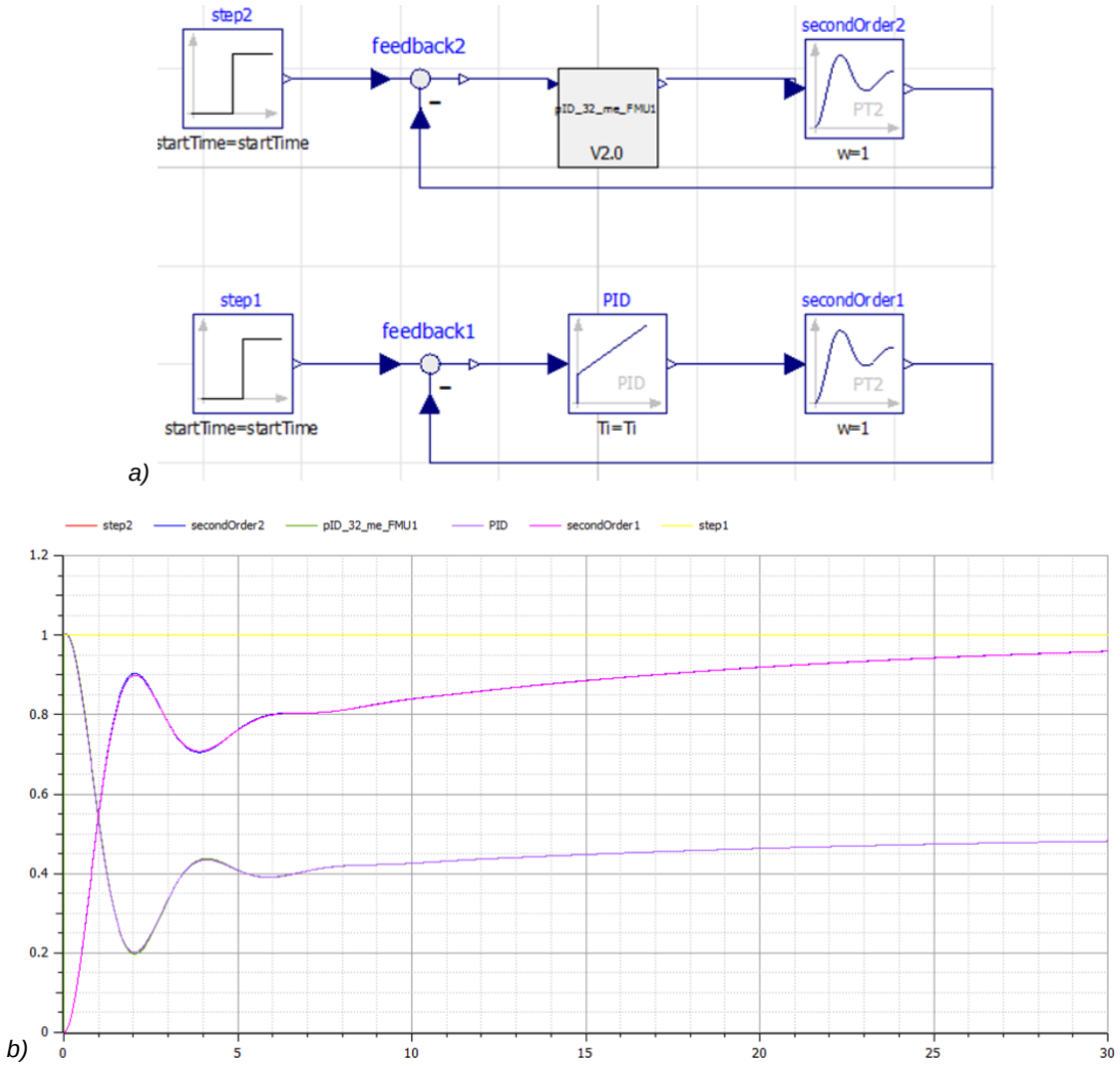


Figure 7.3: Testing of tool independent modelling: a) feedback controller in OM, (b) simulation results