



---

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

Work Package 08

## JRA2 - Co-Simulation based Assessment Methods

Deliverable D8.2

### D-JRA2.2: “Smart Grid ICT assessment method”

---

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

---

Contractual delivery date:	<b>31/10/2018</b>
Actual delivery date:	<b>23/12/2018</b>
Name of lead beneficiary for this deliverable:	<b>1 AIT Austrian Institute of Technology GmbH</b>
Deliverable Type:	<b>Report (R)</b>
Security Class:	<b>Public (PU)</b>
Revision / Status:	<b>released</b>

## Document Information

Document Version: 04  
Revision / Status: released

### All Authors/Partners

Edmund Widl, Michael Spiegel, Mislav Findrik, Agron Bajraktari / AIT  
Rishabh Bhandia / TUD  
Cornelius Steinbrink / OFF  
Kai Heussen, Tue Jensen / DTU  
Mantafounis Panagiotis-Timolewn, Aris Dimeas / ICCS-NTUA  
Matti Laukkanen, Poria Divshali / VTT  
Van Hoa Nguyen / GINP  
Nabil Akroud / OCT  
Przemyslaw Chodura / DNVGL

**Distribution List** ERIGrid consortium members

## Document History

Revision	Content / Changes	Resp. Partner	Date
1	Document structure generated	AIT	10.09.18
2	Draft inputs, first version for internal review	AIT, TUD, OFF, DTU, ICCS-NTUA, VTT, GINP, OCT, DNVGL	26.09.18
3	Included feedback from internal review	AIT	12.10.18
4	Included feedback from Security Advisory Group (SAG)	AIT	07.12.18

## Document Approval

Final Approval	Name	Resp. Partner	Date
Review Task Level	Edmund Widl	AIT	26.09.18
Review WP Level	Ron Brandl	Fraunhofer IEE	10.10.18
Review WP Level	Dimitrios Siagkas, Manikas Savvas	HEDNO	12.10.18
Review Security Adv. Group (SAG)	Kieran McLaughlin	QUB	28.11.18
Review Security Adv. Group (SAG)	Thomas Bleier	BSEC	06.12.18
Review Steering Com. Level	Thomas Strasser	AIT	23.12.18

## Disclaimer

This document contains material, which is copyrighted by certain ERIGrid consortium parties and may not be reproduced or copied without permission. The information contained in this document is the proprietary confidential information of certain ERIGrid consortium parties and may not be disclosed except in accordance with the consortium agreement.

The commercial use of any information in this document may require a licence from the proprietor of that information.

Neither the ERIGrid consortium as a whole, nor any single party within the ERIGrid consortium warrant that the information contained in this document is capable of use, nor that the use of such 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.

**Copyright Notice**

© The ERIGrid Consortium, 2015 – 2020

## Table of contents

Executive Summary .....	7
1 Introduction .....	8
1.1 Purpose of the Document .....	8
1.2 Scope of the Document .....	8
1.3 Structure of the Document .....	9
2 Motivation and Background.....	10
2.1 Smart Grid ICT Assessment .....	10
2.2 Drivers for ICT Assessments Based on Co-Simulation .....	11
3 Proposed Approach .....	13
3.1 General Requirements for Simulation-based Smart Grid ICT Assessment.....	13
3.2 The ERIGrid Holistic Test Description for Assessments Based on Co-Simulation.....	15
3.3 Smart Grid ICT Assessment Methodology .....	17
4 Smart Grid ICT Assessment Toolchain Implementation .....	20
4.1 Co-Simulation Environment .....	20
4.2 Power System Simulation .....	23
4.3 ICT Network Simulation .....	25
4.4 Coupling with Hardware Setups.....	28
5 Proof-of-Concept Validation .....	32
5.1 Power System Simulation with Cyclic Dependent Models (TC1).....	32
5.2 Combined Hardware and Software Simulation (TC2).....	39
5.3 Signal-based Synchronization between Simulators (TC3).....	48
6 Conclusions and Outlook .....	58
7 References .....	59
8 Annex .....	60
8.1 List of Figures .....	60
8.2 List of Tables .....	61
8.3 Updated Test Case Specification for TC1 .....	61
8.4 Updated Test Case Specification for TC2 .....	71
8.5 Updated Test Case Specification for TC3 .....	80

**Abbreviations**

<i>API</i>	Application Programming Interface
<i>ARP</i>	Address Resolution Protocol
<i>CC</i>	Creative Commons
<i>CS</i>	Co-Simulation (FMI specification)
<i>CVC</i>	Coordinated Voltage Control
<i>CMSA/CA</i>	Carrier-sense Multiple Access with Collision Avoidance
<i>DE</i>	Discrete Event (MoC)
<i>DER</i>	Distributed Energy Resources
<i>DES</i>	Discrete Event Simulation
<i>DoE</i>	Design of Experiments
<i>DSL</i>	DlgSILENT Simulation Language
<i>DT</i>	Discrete Time (MoC)
<i>Dul</i>	Domain under Investigation
<i>FMI</i>	Functional Mock-up Interface
<i>FMU</i>	Functional Mock-up Unit
<i>FRT</i>	Fault Ride Through
<i>Ful</i>	Function under Investigation
<i>FuT</i>	Function under Test
<i>HIL</i>	Hardware-in-the-Loop
<i>HLA</i>	High-Level Architecture
<i>HTD</i>	Holistic Test Description
<i>IO</i>	Input/output
<i>ICT</i>	Information and Communications Technology
<i>ISO</i>	International Standards Organization
<i>JRA</i>	Joint Research Activity
<i>LV</i>	Low Voltage
<i>ME</i>	Model Exchange (FMI specification)
<i>MoC</i>	Model of Computation
<i>MV</i>	Medium Voltage
<i>OLTC</i>	On-Load Tap Changer
<i>Oul</i>	Object under Investigation
<i>PCC</i>	Point of Common Coupling
<i>PLC</i>	Programmable Logic Controller
<i>POI</i>	Purpose of Investigation
<i>RMS</i>	Root Mean Square
<i>SA</i>	Sensitivity Analysis

<i>SCADA</i>	Supervisory Control and Data Acquisition
<i>SG</i>	Smart Grid
<i>SuT</i>	System under Test
<i>TC</i>	Test Case
<i>UDP</i>	User Datagram Protocol
<i>UML</i>	Unified Modelling Language
<i>WP</i>	Work Package
<i>WPP</i>	Wind Power Plant
<i>WTG</i>	Wind Turbine Generator

## Executive Summary

Work Package JRA2 has developed advanced simulation-based methods to check and validate Smart Grid scenarios, configurations and applications. The required models cover the various parts of Smart Grids, such as power system infrastructures, control algorithms, communication systems, market aspects or/and external factors like weather and people. The challenge is that the individual models of these parts are of very different nature (continuous, discrete, stochastic, etc.). From the technical perspective, this challenge has been overcome with the help of offline co-simulation, where the most suitable simulation tools for all considered domain can be coupled via the Functional Mock-up Interface specification. From the methodological perspective, this challenge has been addressed by basing the work in JRA2 on ERIGrid's holistic testing procedure.

From the modelling point of view, it is important to integrate commercial and open-source modelling and simulation frameworks, both specialized on particular system aspects (e.g., power system simulator, communication system simulator) and universal (e.g., general modelling environments like MATLAB/Simulink or Modelica-based tools). The specialized tools are usually equipped with validated component libraries, sophisticated import/export capabilities and well-designed user interfaces. The universal tools, on the other hand, are good for rapid prototyping of new and uncommon components, have extensive mathematical capabilities and are well-accepted in the scientific community. It is necessary to combine the best of both worlds.

The goal of work package JRA2 was to have easy-to-deploy software packages that can cover the required simulation needs, i.e., system scalability, model heterogeneity, and simulation performance. The focus was on integrating existing diverse solutions to achieve a common and flexible simulation solution to perform holistic Smart Grid scenarios, configurations and applications. Therefore, work in JRA2 has been based on the FMI specification, an emerging industry standard that allows the coupling of simulation packages and the encapsulation of models.

From the application point of view, one of the main challenges in designing, validating, and rolling out Smart Grid innovation is the size of the energy system, which puts the simulation packages under stress. Performance and accuracy are often traded against each other, which is not a desirable situation. Work package JRA2 has approached these challenges via:

- scenario handling and system modelling via specialized software;
- modular system of systems architecture with clean boundaries for separate optimization;
- cyber-security assessment.

This document shows how the approach outlined above enables Smart Grid information and communication technology assessments covering system stability, system scalability, component interoperability and cyber-security. The goal is to find operational limits and the sensitivity of these system properties towards specific system parameters. In combination with best practice approaches for modelling and simulation, such as design-of-experiments or Monte Carlo simulations, this approach allows to master the complexity of Smart Grid scenarios in offline simulations. In addition, first steps towards integrating these methods into online simulations (hardware-in-the-loop simulations) are shown.

## 1 Introduction

### 1.1 Purpose of the Document

Work Package (WP) Joint Research Activity (JRA) JRA2 has implemented 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 was technically feasible. Testing the applicability of the FMI specification and developing FMI-compliant models and tools for Smart Grid systems were thus a dominant aspect in JRA2.

Initial work conducted in JRA2 (see Deliverable D-JRA2.1) focused on the definition of concrete research challenges, based on which three distinct Test Cases (TC) have been devised. These test cases not only highlighted the relevance of the research challenges but were also used to derive technical requirements for JRA2's co-simulation approach.

This document gives an overview of the methods and tools developed in the context of work package JRA2 for the simulation-based assessment of ICT-enabled Smart Grid applications. This overview comprises a short review of the identified requirements, the proposed methodology and an outline of the implemented simulation toolchain. Furthermore, it is demonstrated how these tools and methods can be applied to assess Information and Communication Technology (ICT) enabled Smart Grid applications, by applying them to the above-mentioned test cases. These test cases are:

- *Power system simulation with cyclic dependent models (TC1)*: This test case verifies the low-voltage ride through capability of an onshore wind power plant that is interconnected to a small transmission system. The Fault Ride Through (FRT) curve is enforced at the plant's coupling point, whereas the grid interface of the converter, its controls, protection, and electromechanical conversion components ensure the compliance to this curve.
- *Combined hardware and software simulation (TC2)*: The purpose of this test case is to validate the operation of a controller for a transformer with On-Load Tap Changer (OLTC). Parts of this test case are implemented in real hardware, whereas the remainder of the system is implemented as a simulation model. TC2 will demonstrate the ability to interface the models and tools developed in JRA2 with online (hardware-in-the-loop) simulations.
- *Signal-based synchronization between simulators (TC3)*: This test case deals with the impact of communication delays and controller dead times 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 the OLTC transformer.

### 1.2 Scope of the Document

The goal of WP JRA2 was to have easy-to-deploy software packages that can cover the requirements for assessing ICT-enabled Smart Grid applications: system scalability, model heterogeneity and simulation performance. The focus was on integrating existing diverse solutions instead of developing a new monolithic special-purpose tool. This deliverable explains the methods that have been developed/adapted as well as the tools that have been implemented/extended for this purpose. Furthermore, the application of these methods and tools to simple yet instructive test cases is shown, demonstrating their basic functionality.

One of the main challenges in designing and engineering Smart Grid applications is the size of the energy system. This size and complexity put simulation tools under stress, such that performance and accuracy are often traded against each other, often leading to an undesirable situation. In order to address such issues, the methods and tools described in this deliverable have been developed with the goal to enable the assessment of large-scale systems. However, the description of how these methods and tools are applied to large-scale systems is not addressed in this document, but rather in deliverable D-JRA2.3 (Smart Grid simulation environment).



### 1.3 Structure of the Document

The document is structured as follows: Section 1 provides an overview of the subjects addressed in this document, including its scope and structure. Section 2 presents the motivation and background for the work presented in this document. Specifically, the definition of what a *Smart Grid ICT assessment* is in the context of work package JRA2 is given. Furthermore, the drivers for using co-simulation for such assessments are presented. Section 3 explains the approach for simulation-based Smart Grid ICT assessment devised in work package JRA2. First, the relation to ERIGrid's holistic testing procedure is explained. Then the technical requirements for simulation-based Smart Grid ICT assessments in the context of JRA2 are described. Based on this, the methodology adopted for JRA2 is presented. Section 4 gives an overview of the tools and methods that have been implemented/extended for the purpose of work package JRA2. This comprises a brief explanation of the co-simulation environment and the domain specific simulation tools for power system and ICT network simulation. Furthermore, the interface for integrating FMI-compliant simulation models as virtual components in hardware setups is presented. Section 5 provides a proof-of-concept validation of the methods and tools presented in Section 3 and Section 4, respectively. This is done by applying them for the assessment of ICT-enabled Smart Grid applications, namely, power system simulation with cyclic dependent models (TC1), combined hardware and software simulation (TC2) and signal-based synchronization between simulators (TC3). Section 6 presents the conclusion and outlook for the work presented in this document.

## 2 Motivation and Background

### 2.1 Smart Grid ICT Assessment

Electric power systems as Smart Grids are inherently multi-domain systems involving the power system dynamics, communication as well as control and supervision applications at various operational time scales. Assessment such as the validation of control solutions at higher levels of maturity (e.g., pre-deployment) and system integration requires that the solution components and functionality are accurately represented. The factors influencing this solution can be of physical, ICT or algorithmic and computational aspects, accordingly the assessment methodology and simulation has to account for the domain specific representations and procedures.

The development of a Smart Grid brings new requirements and new functionality to the power systems domain and associated ICT systems. More than just increasing in “scale”, this means functionality is to be provided beyond the boundaries of components and across different technology domains. Systems become multi-domain systems of increasing complexity. It can be expected that these developments require assessment approaches beyond those feasible in the state of the art. Explicit reflection of ICT and multi-domain interdependencies in simulation has to address complex interactions and behaviours internally and externally. As the system evolves in complexity and interdependency, it is conceivable to view the Smart Grid as a system-of-systems. In assessment of the Smart Grid as system-of-systems, the scope would shift from direct technical assessments to investigations on the modularity structure and resiliency of the overall energy infrastructure. Simulation-based assessment of complex and large-scale multi-domain systems does not aim at this level of abstraction, but instead focusses on explicit representation of technical elements and their assessment in a system context.

In the context of ERIGrid, Smart Grid ICT assessment describes the test-based qualification of Smart Grid ICT functions that entail continuous cyber-physical interactions (i.e., no grid planning or pure simulation but at least a closed-loop function). The assessment is aimed at anticipating and qualifying the behaviour of realistic systems where relevant phenomena arise from interactions of at least two domains. A key effort in multi-domain assessment is the separation of assessment procedures (and methodologies) from implementation, modelling and simulation techniques. Assessment procedures typically serve an assessment purpose and need and thus are problem-oriented. Relevant for Smart Grid ICT assessment are for instance:

- Control performance testing
- Interoperability testing
- Cyber-security testing

As the testing purposes used to be domain specific, the testing tools are domain specific as well, and thus the concrete realization of these procedures involved domain specific modelling and simulation tools. The domains of testing tools (both in hardware and in simulation) relevant for Smart Grid ICT assessments include:

- Physical simulations of power system and other energy forms (e.g., buildings)
- ICT/cyber systems and communication
- Control systems specification and verification tools

It is a premise in this work that the convergence of tooling for cross-domain simulation is more effective by means of tool coupling rather than convergence into monolithic simulation tools.

## 2.2 Drivers for ICT Assessments Based on Co-Simulation

### Overview

Simulation-based assessments of smart energy systems are typically employed to:

- *Complement analytical assessments* where scale, behaviour and nonlinearity of phenomena extends beyond those feasible to be addressed by analytical methods;
- *Assess the concept of a control system* for characterization or validation purposes, e.g., to check stability and robustness and quantify control performance with respect to alternative solutions or a baseline approach;
- *Prepare a control implementation*, as an intermediate step before transferring the control solution to an embedded system.

Co-simulation is practical when domain-specific models from several domains are required to reflect the relevant dynamics for a particular scenario (e.g., a power system model, controller implemented in a native language, and an emulation of communication): time is saved in the model development, and a possible model-translation error could be avoided. However, co-simulation also introduces a complication as compared to monolithic simulation approaches: the need for frequent data exchange (between simulators) and time-synchronization (across simulations) requires additional computational resources and implementation overheads, which typically slows simulations down. Obviously, there is a trade-off between co-simulation benefits and the expected slow-down and software complexity. In scaling up to larger simulation scenarios, this trade-off becomes even more prominent. In this context, the natural questions to ask are: In what cases should one resort to assessments based on co-simulation? Where are the trade-offs also acceptable when scaling up the simulation model to larger simulation scenarios? What is the alternative to co-simulation?

Avoiding co-simulation is generally achieved by the use of a native multi-domain simulation environment, such as *SystemModeler*, *MATLAB/Simulink*, and the like. Alternatively, a multi-domain simulation could be built from scratch. However, a simulator that supports multi-domain is not trivial to implement, due to the significant effort and expertise required. Also, models from different domains often target different time scales, Models of Computation (MoC) and specialized solvers. Building a simulator capable of providing appropriate environments, correct MoC, solvers and properly coordinating them internally is expensive and maybe not worth the effort.

### Linking Models of Computation

One of the most critical points of integration is the MoC behind a model and the simulator. It is the organizing principle of a model to represent the semantics of the interactions between modules, components or phenomena. A MoC is independent from the implementation technology (i.e., sequential or parallel) and language (i.e., Matlab, Python). MoCs can be classified as in [1]: Imperative (e.g., emulators), Finite-state Machines (e.g., a set of states, rule-based control), Dataflow (e.g., for ordinary differential equations of differential algebraic equations), Discrete Event (e.g., communication, zero-crossing) and others. Discrete Event simulation can be further broken down into event scheduling, process interaction and activity scanning [2].

Energy domain simulators often employ dataflow MoC, due to the fact that they derive mostly from sets of ordinary differential equations defining the peculiarity of the state variables and the environment factors of a system (e.g. steady-state simulations, electromagnetic transients and circuit simulations, or electromechanical phenomena). However, simulators for ICT, markets or controls often use Discrete Event or Finite-state Machine semantics. The discrete models react to events that occur at a given time instant and produce other events either at the same time instant or at some future time instant in a chronological execution order. More specifically, discrete event models are discrete, dynamic and stochastic in nature and are "run", whereas continuous model are "solved". Combining discrete event-based and continuous time-driven simulation requires mixing different MoC, such as

Discrete Events and Dataflow in a hierarchical way. This requires an interaction semantic that resolves the ambiguities caused by differences among MoCs. Events crossing the domains need to be totally ordered and associated with time stamps. Moreover, each domain (Simulator and MoC) must also support a rudimentary notion of time. The main difficulties of the integration of such different MoCs involves the handling of simultaneous events and zero-delay feedback loops [1]. Co-simulation can help to integrate and harmonize such models with different MoC.

### Domain Expert Support

A further argument against native multi-domain simulation is the need for models of professional quality in testing and operations assessments: domain experts are required to verify the correctness of models, which is easier to realize within domain-specific, accepted tools.

An assessment methodology based on co-simulation is necessary when system complexity requires a domain specific modelling approach. Often this complexity is viewed only as complexity of behaviour (nonlinearity, scale in number of nodes), but more important in a multi-domain setting is that each domain has a specific approach to system architecture, structure, and behaviour. For domain specific simulators, this means a domain specific approach to:

- *Architectures*: semantics of model lumping into sub-modules and coupling across modules
- *Structures*: coupling between modules basic simulation elements and data structures
- *Behaviours*: basic model of computation and mathematical representation of model elements

Professional domain expertise is tied to domain specific representations (e.g., networking and communications technology is modelled in network simulators; power system models including complex generator and load dynamics are modelled by power system engineers). Similarly, domain expertise is required in the semantics of a domain model in structure and architecture. For instance, cyber-security experts assessing vulnerability have to model an attack using a language that is descriptive of the cyber-domain (ICT network components, protocol layers, data streams, operating systems). For cyber-physical security assessments, the classical threat-model assessment approach has to be extended to a cyber-physical threat model and approach.

### Integration into Laboratory Environment

Finally, the development of control applications beyond the context of an embedded system is tied to more complex software, where transcoding a control description from a simulation model to implementation context can be similarly error prone, and thus at least a domain specific control description language is required (e.g., IEC61499). Testing of target control code in laboratory environment is limited due to the physical constraints of the lab environment. Here, a lab-integrated co-simulation can extend the physical environment with a virtual counterpart, enabling to explore scalability limits of the control code and to enhance the System under Test (SuT) with scale phenomena otherwise not feasible in a laboratory setup.

### Intellectual Property Issues

Another use case for upscaling co-simulations arises for grid integration of systems comprising (intellectual property-restricted) component models, such as for validation of third party systems, e.g. a “black box” control software.

### Extensibility

The above cases justify co-simulation in particular where domain-specific model complexity and expertise is the key driver. Another key driver is the simple notion that training and testing should be based on the most realistic and mature models, so that the model can be trusted. In that view, amending an existing and trusted simulation model with properties from a different domain (e.g. adding ICT communication delays) or an extended context (e.g. by extending a local electrical model with a more realistic demand behaviour) can only be realized with co-simulation.

### 3 Proposed Approach

#### 3.1 General Requirements for Simulation-based Smart Grid ICT Assessment

The following section provides an overview of the most important general requirements for Smart Grid assessments in the context of JRA2. These requirements are general in the sense that they are not domain-specific (i.e., not directly related to the assessment of power systems, automation and control, or communication), but rather comprise generic prerequisites for simulation-based assessments. As the focus of the work package lies on approaches based on co-simulation, these requirements reflect to some extent (but not exclusively) the challenges specific to assessment approaches based on co-simulation, also taking into account that the ultimate goal of JRA2 is the assessment of large-scale systems. In addition to listing these requirements, the following section also explains how the associated challenges are handled with mosaik, the co-simulation environment chosen for work package JRA2 (see Deliverable D-JRA2.1).

##### 3.1.1 System-of-Systems Architecture with Clean Boundaries for Separate Optimization

Smart Grid technologies are at their core the application of automation to the operation of electrical power systems. In the context of JRA2, the most relevant applications are those where (distributed) control systems are connected with smart sensors and actuators through ICT systems. Consequently, for the assessment of Smart Grid systems, it is of crucial importance to capture the interaction between the following three technical domains (see also Deliverable D-JRA2.1):

- *Power and energy domain:* This domain contains equipment for the generation, consumption and storage of electricity, as well as the technical infrastructure used to interconnect this equipment.
- *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.
- *Automation and control:* This domain contains control logic and algorithms.

From a conceptual point of view, this means that a Smart Grid system can be considered as a system-of-systems. Naturally, any assessment approach for Smart Grid applications should be able to capture and replicate system-of-systems properties. Within this context, an assessment approach based on co-simulation offers several advantages:

- *Modularity* is the most fundamental property of system-of-systems architectures. On the condition that appropriate interfaces are available that allow a proper linking of domain-specific tools, co-simulation allows the representation of (parts of) sub-systems with the most appropriate tool available. This encourages a modular representation of the SuT, with clean semantic and functional model boundaries along the real-world domain borders.
- *Hierarchical composition* is a prominent feature of many system-of-systems architectures, typically in the form of system layers with distinct functionality (e.g., an ICT layer “on top” of the physical power system). By following a modular approach, the combination of specialized tools in a co-simulation can also help to address typical issues of hierarchical modelling. For instance, co-simulation can help with mixing models of computation (e.g., continuous time-based and discrete event-driven simulations) or multi-rate simulations by encapsulating the individual models, each with its own dedicated solver.

The potential of assessment approaches based on co-simulation to capture systems-of-systems in terms of modularity and hierarchy, enables experts to focus on the assessment and optimization of their specific domain. At the same time, it provides them with the possibility to assess the effects of changes in the setup of their specific sub-system on the overall system, enabling a holistic assessment.

A substantial part of the work done in work package JRA2 focused on the subject of modularity and hierarchical modelling, specifically the development of appropriate simulation interfaces for the selected domain-specific tools (see Sections 4.2 and 4.3) as well as mosaik's capability of linking tools from different domains and with diverse models of computation (see Section 4.1). In addition, the domain-specific tools chosen for work package JRA2 (see deliverable D-JRA2.1) all use object-oriented component models, further reinforcing a modular and hierarchical modelling approach.

### 3.1.2 Scenario Handling and System Handling

The use of co-simulation for validation in holistic Smart Grid systems requires means of flexible scenario handling in complex setups. Such a feature allows for easy component replacement and system scaling, which make up the main advantages simulation-based validation has over lab-based validation.

In the co-simulation framework mosaik, flexible scenario handling is achieved via a dedicated scripting interface. This interface, called *Scenario Application Programming Interface* (Scenario-API), provides simple commands for the initialization of simulators, the instantiation and parameterization of model objects, and the connection of models. The fact that scripting is required leads to the need of users to possess some programming skills, but the slight loss in usability is easily outweighed by the gain in flexibility. If the employed simulators support instantiation, hundreds to thousands of model entities can be created with a single command allowing for fast upscaling of any co-simulation setup. Similarly, large sets of model entities can be easily connected to each other. Since the Scenario-API is based on the general-purpose programming language Python, structures for iterations and conditional control may be used for filtering of model entities and rule-based connections. Similarly, iterative structures may be used to initialize large sets of simulators.

Flexible scenario and system handling supports several of the system assessment efforts analysed in JRA2. On the one hand, higher numbers of modelled entities can easily be realized in a co-simulation setup, as mentioned above. On the other hand, integration of upscaled versions of single components is also possible. An example for this may be a power grid or communication system model with more nodes. Since the work in JRA2 is based on generally accepted and popular modelling tools as well as standardized interfaces (via FMI), it is relatively easy to replace, e.g. a 100-node model with a 1000-node model in a holistic co-simulation scenario. Finally, script-based scenario handling allows for easy creation of parameter studies with several parallel or sequential simulation processes exploring the parameter space of the scenario. This can either be used for gradual upscaling of a system simulation or for more general assessment approaches like sensitivity analysis or uncertainty quantification in form of a Monte Carlo study.

### 3.1.3 Distributed and Parallel Simulation

The mosaik co-simulation environment supports distributed simulation via the possibility to start simulation components in remote processes or connect to already running processes. Apart from enabling the distribution of a co-simulation across several hardware nodes, this approach allows to couple simulators with different run-time requirements executing on the same hardware node. For instance, the coupling of 32-bit and 64-bit applications on the same hardware node is possible, or Windows and Linux applications (using Cygwin, see for example Section 5.3). A distribution of the scheduling module itself, however, is not possible - unlike it is the case for co-simulation tools based on the High-Level Architecture (HLA).

In the context of parallel simulation, it is important to note that mosaik provides conservative scheduling of its simulator processes. This means that simulation components will stop their execution after a simulation step until new input data is provided. The alternative to this would be progressive scheduling that has components simulate in advance and force them to roll back in case of changing input data. Mosaik employs conservative over progressive scheduling since several simulators do

not support rollbacks and may thus only be employed in a conservative setting. Consequently, mosaic's potential to parallelize a single simulation scenario is limited. After all, conservatively scheduled simulators cannot be executed in parallel if one of them is expecting input data from the other. Instead, parallel simulation is only possible for those components that do not possess any direct or indirect data dependencies towards each other.

### 3.2 The ERIGrid Holistic Test Description for Assessments Based on Co-Simulation

The specification and execution of tests or experiments is central to any assessment approach. The holistic testing procedure proposed in ERIGrid aims to unify the approach to testing across different research infrastructures, different testbed types and to facilitate multi-domain testing. In practice, holistic testing here means to make the most of available infrastructure by providing a common frame for coupled simulations.

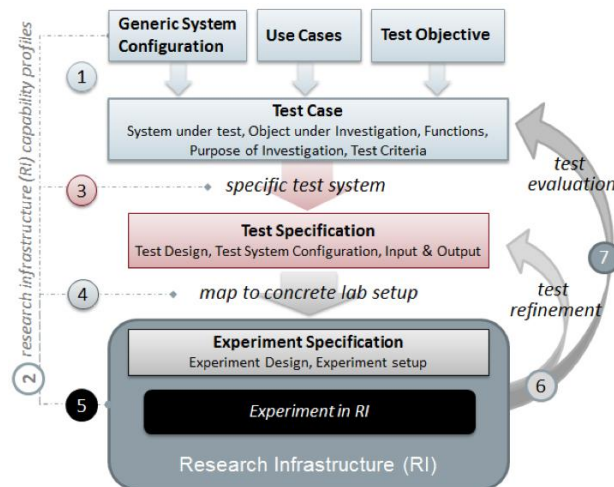


Figure 1: Overview of ERIGrid’s holistic testing procedure

A central element of the ERIGrid approach is the *Holistic Test Description* (HTD) method: it defines a number of specification elements for any test, to be identified independently of the particular assessment methodology. The specification elements comprise three main levels of specification:

- *Test Case*: a concise formulation of the overall test objectives and purpose of investigation, specifying the object under investigation, separating system structure from functions (system under test vs. functions) and identifying test criteria.
- *Test Specification*: a detailed description of the test system, control parameters and test signals to address a specific subset of the test criteria.
- *Experiment Specification*: how is the test specification realized in a given testbed? What compromise had to be made in the representation with respect to the test specification?

The method is supported by structured templates to facilitate the harmonized recording of test descriptions. Method detail is defined in detail in [3], and the description templates are provided under a Creative Commons (CC) license.

It is relevant here to emphasize how the HTD approach applies to co-simulation based Smart Grid ICT-assessment. First of all, the approach is *inherently multi-domain*, where the SuT outlines the generic multi-domain structure and interfaces relevant to a test case. Furthermore, the approach facilitates the separation of simulator embedding from specification of the test system structure (i.e., the model topology, also across simulators). To that end, the test specification is formulated independently of test beds but focusses on system parameters, including exchange variables. The description method is thus *independent of simulator embedding even in a co-simulation context* and

allows the detailing of test purposes before a simulation platform is selected. Figure 2 illustrates two such test system definitions. Third, the *framing of the assessment approach is generic* in a test case; it is not specific to any particular assessment method, so that it can be applied to any of the cyber-physical energy system and Smart Grid ICT-assessments investigated (e.g. controller validation, upscaling, cyber-security).

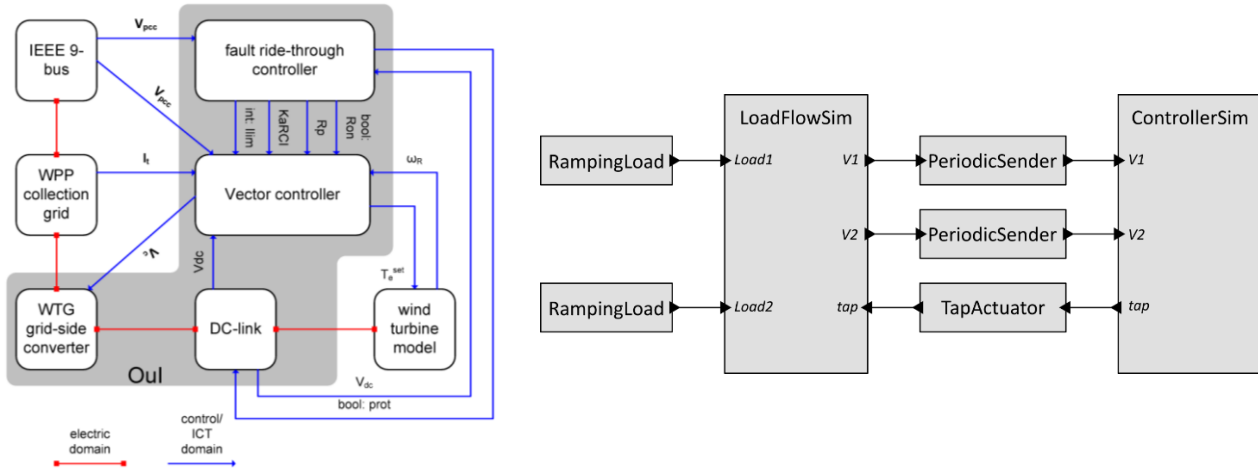


Figure 2: Example representations of test systems: components, terminals (with directionality) and domains are clearly distinguished

Two critical mapping steps are framed by the HTD: the step from generic test case to detailed test specification(s), and the step from test specification to realization in a specific set of simulation environments in an experiment setup. The former is addressed by a *Qualification Strategy*, which accounts for the division of tests, for example to first characterize the simulators before testing. In general, multiple individual test purposes are addressed by a single test case. Similarly, the implementation of test systems in a test-bed is well expressed by an illustration of simulator-embedded models, as illustrated for example in Figure 3.

In that, the HTD aids the specification of tests or experiments also for co-simulation testbeds:

- The formulation of test objectives in isolation from use case or simulation environments separates test purpose from test object and tooling; this aids technology-agnostic formulation of assessment methods.
- Co-simulation experiments, by definition, contain multiple simulation models; for the testing context, it remains essential to establish a shared topology of the experiment across domains; the notions of “test system” vs. “experiment setup” offer a natural framework for this distinction (compare Figure 2 to Figure 3).

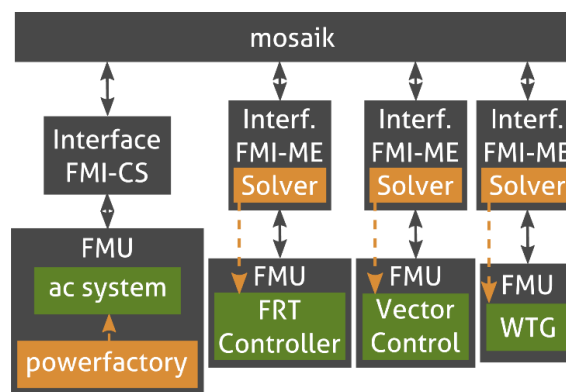


Figure 3: Representation of experiment setup (simulators representing testbed level)



### 3.3 Smart Grid ICT Assessment Methodology

The following section provides an overview of the methodical approaches that have been applied for the assessment of ICT-enabled Smart Grid applications using the toolchain developed in JRA2. The investigated methodical concepts represent general best practice approaches for simulation-based assessments (and beyond) and are themselves not limited to assessments of Smart Grid applications. However, by applying these methods to the JRA2 toolchain, a large spectrum of problems related to the assessment of ICT-enabled Smart Grid application can be addressed. The investigated methodical approaches are not exclusive, i.e., there are potentially others that can be successfully applied to this toolchain. However, during the course of the project, these approaches have been proven valuable for achieving the set goals. As such, they together form work package JRA2's methodology for Smart Grid ICT assessments.

#### 3.3.1 Monte Carlo Simulations

Monte Carlo simulations are a stochastic approach that is based on a (very) large number of similar random experiments. In the context of work package JRA2, randomness comes from processes whose properties can only be characterized statistically. For instance, ICT systems are typically modelled in discrete event simulations, with events corresponding to arrival of messages at various points in the system. As packages entering the system as well as processing delays at each node are typically modelled by random sampling, each model run will in general have different outcomes. This requires gathering statistics on outcomes from multiple model runs, as any single model run cannot represent the full range of possible outcomes. However, exploring all combinations in the case of multiple variable parameters may become computationally expensive. On the other hand, propagation of correlated variables may overestimate the output measure and statistical distributions.

JRA2 deals with controllers coupled through simulated ICT systems, making the use of Monte Carlo simulations necessary to quantify the impact of (randomized) ICT delays. This is demonstrated in test case TC3, where the effects of network delays and controller dead times on an OLTC transformer are analysed. In the example, two voltage meters periodically send their readings to a server, which calculates a new tap position for the OLTC transformer whenever a new measurement is available. The tap positions are then sent to the OLTC transformer where they are actuated. In this specific test case, it can be shown that the delays and dead times can lead to random, unexpected outcomes (see Section 5.3).

#### 3.3.2 Sensitivity-aware Parameter Variation and Design of Experiments

Smart Grid ICT systems are typically complex systems-of-systems with each sub-system possessing several parameters that determine its behaviour. The overall ICT system thus possesses a highly multi-dimensional *design space* which each parameter of each subsystem posing as one dimension. One point in this space stands for a particular system parameterization and a common task in ICT research is to identify the sets of parameters that lead to a system behaviour that is in some sense optimal. Obviously, in a high-dimensional and possibly continuous design space not all possible parameter value combinations can be tested. Instead, *Design of Experiments* (DoE) methods can be employed to obtain the maximal amount of information from a finite number of experiments or simulations.

While classical DoE is typically geared towards analysis of systems with intrinsic statistical fluctuations, the work in JRA2 is mostly focused on so-called “modern” DoE that is typically associated with software simulation experiments. Software models have the analytical benefit that they do not contain fluctuations (unless explicitly introduced into the model) so that one and the same set of parameters always leads to the same outcome. On the other hand, simulation models typically possess a larger number of tuneable parameters (factors) than hardware systems since they are used in earlier stages of the design process. Accordingly, modern DoE methods are often used for screening of the analysed system to gain information about its behaviour, which may be used to refine following experiments. With *space-filling* sampling strategies, modern DoE makes sure that large numbers of

simulation runs are efficiently used to account for nonlinear system behaviour. Similarly, high-dimensional design spaces can be efficiently filled. This way, a *Sensitivity Analysis* (SA) can be conducted that allows users to identify the most influential factors and value ranges for a given system. Furthermore, DoE techniques to restrain the parameter variation zone can be particularly helpful to reduce the number of Monte Carlo simulations to be performed and avoid unwanted design choice effects.

In JRA2, an exemplary screening experiment has been conducted in the context of test case TC1 (see Section 5.1). The analysed system contains a power grid, a wind turbine generator as well as its converter, and a control system influencing the converter behaviour. The factors analysed in the screening have been proportional control gain, the current limiting strategy followed by the control system, and the active power recovery rates. As the system response to value changes, its voltage and frequency behaviour have been analysed. As a result of the screening, the current limiting strategy has been identified as a negligible factor while control gain and recovery rates should be considered as important factors in following experiments. The importance of using DoE methods for such SA endeavours stems from the fact that factor interactions should be considered for SA instead of just single-factor effects. Furthermore, modern DoE also helps to consider nonlinear effects in a system, as mentioned before. This is typically associated with the term *global SA*.

### 3.3.3 Cyber-Security Assessment

Compared with legacy power systems, the ICT-enabled Smart Grid is envisioned to fully integrate high-speed, two-way communication technologies into a large number of devices. This will enable the creation of a dynamic and interactive infrastructure with new capabilities, such as energy management by means of active voltage control in medium and low voltage distribution grids. Therefore, the Smart Grid is a progressive evolution of existing power grids, whereby increased levels of automation and ICT technology will ultimately result in a large-scale system supporting complex interactions between various actors in the electricity sector (energy producers, consumers, distributors, etc.). One of the biggest challenges for such systems is to ensure that automation systems and supporting ICT networks are adequately designed and secure.

In 2009, the International Standards Organization (ISO) published the international standard for risk management ISO 31000:2009 [4] which is based on national approaches from Australia and New Zealand (AS/NZS 4360:2004, see [5]) as well as Austria (ONR 49000:2004, see [6]). This standard has evolved to a widely accepted standard, which – due to its generic approach – can be ubiquitously applied to basically every kind of organization, regardless of its type, perspective or size. The generic steps of every risk management process are shown in Figure 4.

This generic process has been adopted and tailored to Smart Grid applications within the research project SPARKS [7]. The project SPARKS suggested several good practices and tools for conducting the cyber-security risk assessment on Smart Grid systems. One of them is also consequence informed risk assessment, which can also be applied for cyber-security assessment in the context of work package JRA2. Specifically, the following steps of the risk management process along with the best practices for their application in the context of Smart Grids have been derived from the SPARKS project:

- *Establishing the context:* This step involves the definition of the scope of the risk assessment to be carried out and the overall goals of the stakeholders involved. To support the context establishment and provide guidance specific for Smart Grid assessments when necessary (including aspects such as the business goals and relevant actors) the underlying architecture is recommended to be modelled using the Smart Grid Architecture Model (SGAM) framework according to the SPARKS project.
- *Risk Identification:* The purpose of this step is to determine what could happen to cause a potential loss and further how, where and why the loss might happen. To answer these questions SPARKS recommends usage of threat libraries and structural graph-oriented techniques, such as attack graphs.

- **Risk Analysis:** This step addresses the assessment of impact and likelihood, and finally the determination of a risk level, which is based on the impact and likelihood values. On the one hand, for addressing likelihood, SPARKS recommends the usage of quantitative measures (low, medium, high). On the other hand, impact should be assessed from multiple perspectives (safety, financial, reputational, system impact, etc.). For deriving the system-level impact, it is recommended to use simulation-based approaches for the systematic identification of consequences in Smart Grid systems. These consequences also need to be placed in relation to threats identified in the previous step.
- **Risk Evaluation:** After completing a satisfactory risk assessment, risk treatment follows. Initially, security requirements and recommendations for the specific use case are derived. These can be viewed as high-level technical and organizational recommendations that can be realized using a number of different, more specific, controls. The SPARKS project recommends deriving specific security control based on the NISTIR 7628 [8] and ENISA guidelines.
- **Risk Treatment:** This step is concerned with the choice of security control to implement, such as firewalls and intrusion detection systems or authentication mechanisms, which will be based on an organization's risk tolerance and the budget allocated to cyber-security matters.

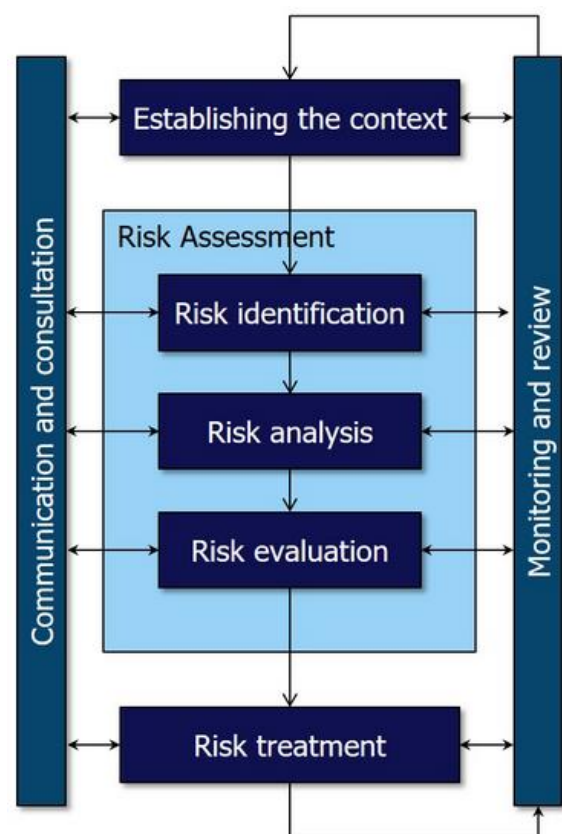


Figure 4: General Risk Management Process

In conclusion, designing security for (potentially large) Smart Grid systems is a complex process, which requires that cyber-security experts can assess an impact of ICT systems on physical infrastructure. One way to assist experts in performing such risk assessments are simulation-based holistic approaches, like the one developed in work package JRA2, as the results of the simulations enable better risk perception and decisions during the risk treatment phase.

## 4 Smart Grid ICT Assessment Toolchain Implementation

The following section gives an overview of the toolchain implemented in work package JRA2. The goal of work package JRA2 was to have easy-to-deploy software packages that are capable of covering the requirements for assessing ICT-enabled Smart Grid applications. At the same time, the goal was also *not* to implement yet another tool that just covers one view of the system, but to integrate existing solutions within a co-simulation approach. Therefore, work in JRA2 has been based on the FMI specification, an emerging industry standard that touches that allows for coupling simulation packages and encapsulating models. Therefore, the development work in JRA2 focused on two tasks:

- *FMI-compliant developments*: Several of the tools selected for the use in JRA2 do not provide or provide only limited support for FMI by themselves. Hence, FMI-compliant co-simulation interfaces have been developed, as well as tools and user-friendly scripts to generate FMI-compliant co-simulation components, so called *Functional Mock-up Units* (FMU).
- *mosaik developments*: The co-simulation framework mosaik also needed to be extended for the purpose of JRA2. This included a user-friendly way of handling FMUs (instantiation, data exchange, execution), the handling of cyclic dependencies between simulators and signal-based simulations.

At the time of writing, several of these developments are prototype implementations that will need to be improved and extended in the future. However, together they provide a proof-of-concept that FMI-based co-simulation is well suited for the assessment of ICT-enabled Smart Grid applications, able to address the research challenges defined for work package JRA2 (see Deliverable D-JRA2.1).

### 4.1 Co-Simulation Environment

The mosaik framework is an easy-to-deploy software package that facilitates the integration of new simulators as well as the creation of co-simulation experiments. This is achieved via a lightweight software core based purely on Python, a special *Component-API* for simulator integration, and a *Scenario-API* for flexible simulator coupling. Since the mosaik framework is still under active development, new features have been introduced based on requirements defined for work package JRA2 (see deliverable D-JRA2.1).

#### 4.1.1 FMI Support

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

To this end, the *FMI++ Python Interface* [9] has been developed, a Python package wrapping the *FMI++ Library* [10]. The FMI++ Library bridges the gap between the basic functionality provided by the FMI specification and the typical requirements of simulation tools. It provides high-level functionality that eases the handling and manipulation of FMUs, such as numerical integration, advanced event-handling or state predictions. This allows FMUs to be integrated more easily, e.g., into fixed time step or discrete event simulations.

For the purpose of work package JRA2, the FMI++ Python Interface and the mosaik framework have been successfully combined for the co-simulation of FMUs. Several examples of importing FMUs have been implemented using the FMI++ Python Interface to interact with the FMU and mosaik's high-level component API to integrate it into the co-simulation. For this, especially the functionality for conveniently handling FMUs was extensively used, such as extracting the FMU, parsing its model description or the ability to refer to input/output variables by name (rather than the numerical value reference associated to each variable).

Within JRA2, the main effort was to make the installation of the FMI++ Python Interface more user-friendly. Previously, users had to compile it from source code, which involved a rather complicated toolchain that only users with experience in software development were able to use. Therefore, easy-to-install binary packages (Python wheels) for Windows and source distribution packages for Linux have been made publicly available on the official Python Package Index [9].

#### 4.1.2 Handling of Cyclic Dependencies

##### Overview

The term “cyclic dependencies” here refers to a co-simulation setup in which two (or more) simulators require data from each other to advance their state in time. These data dependencies may lead to deadlocks with all simulators waiting for data from each other, halting the whole simulation process. Therefore, proper handling of cyclic dependencies is one of the most crucial tasks in co-simulation.

In the following, the proposed extension for enabling co-simulation of models with cyclic dependencies in the mosaik framework is presented. For concrete examples highlighting this extension, please refer to Sections 5.1 and 5.3.

##### Extension of mosaik

Most basic considerations about the topic stem from co-simulation research for continuous-time white box models. In this context, the simulation scheduling differentiates between micro steps for integration steps of the simulators and macro steps for the data exchange between simulators. Different procedures have been proposed for the scheduling of macro steps and iterative calculation of stable system states. Fundamentally, researchers differentiate between serial data exchange schemes (also called *Gauss-Seidel* type procedure) and parallel schemes (also called *Jacobi* type procedure). In Smart Grid co-simulation, simulators are typically treated as black box systems so that iterative refinement of the same time step is usually not considered. Nevertheless, the concepts of serial and parallel data exchange schemes also exist in black box Smart Grid co-simulation.

Additional aspects that are more pronounced in Smart Grid co-simulation than in classical white box systems are the ideas of *usability* and *flexibility*. Usability means that co-simulation solutions are expected to be applicable by different types of domain experts that are not necessarily simulation experts. Flexibility means that the solutions are expected to work in a variety of different application cases without requiring major changes to the employed tools.

The co-simulation framework mosaik has been developed with a strong focus on flexibility. Accordingly, the scheduling algorithm of mosaik is designed in a way to allow integration of any number of simulators. Furthermore, all integrated simulators may display different macro step sizes and even vary their step size over time. In order to guarantee the absence of deadlocks for any given setup, the handling of cyclic dependencies in mosaik has so far had some limiting characteristics. In particular, using mosaik’s intuitive connection capabilities to establish cyclic data exchange between two or more simulators has been prohibited. Instead, users had to extend the simulator interfaces to realize cyclic data exchange, which obviously decreases the usability of mosaik for researchers with little programming experience. Furthermore, the described solution in mosaik only supports serial data exchange schemes.

Within the scope of the ERIGrid project, the capabilities of mosaik have been extended to allow for higher usability in the handling of cyclic dependencies. The basic idea of the extension is the separation of data exchange into two stages: Simulators may receive data either *before* they are called to calculate a time step, or *after* they have calculated so that they store the data for the next time they are called. With this separation, priorities between simulators can be established so that deadlocks are avoided. Figure 5 illustrates different data exchange options between two simulators A and B. Connections for data exchange before calculations are called *standard connections* since they are part of the typical functionality of mosaik. The newly added connection type is called *time-shifted connection* since they provide data to simulators that already have been called for calculation.

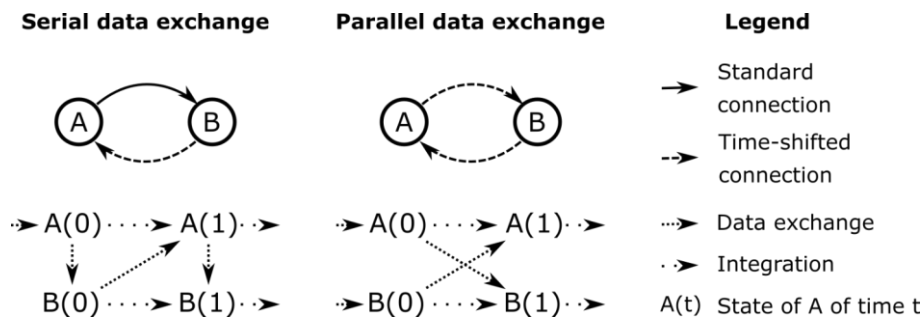


Figure 5: Data exchange schemes possible in mosaik

As Figure 5 shows, standard connections in mosaik provide data to a simulator for its calculation of the current time step while time-shifted connections provide data for the next time step to be calculated. Furthermore, mosaik provides the option to set default input data for the first calculation of a simulator that is addressed by time-shifted connections. This way, parallel data exchange schemes may also be realized if initial input data can be assigned to each simulator.

Overall, the extension of mosaik improves its usability and provides it with the most common options for handling cyclic dependencies in Smart Grid black box co-simulation.

#### 4.1.3 Signal-based Synchronization

##### Overview

One of the research challenges formulated for JRA2 was the detailed simulation of ICT-based communication in Smart Grid systems. This involved two challenges:

- **Discrete Event (DE) simulation:** The mosaik framework has been primarily designed for Discrete Time (DT) simulation, where the time step size between each call to a simulator is a fixed, constant value (even though the step size may be different for different simulators). Even though DT simulations are useful for the assessment of a large variety of Smart Grid application, they struggle to capture the details of discrete event-driven models (such as typical models of communication systems). Similarly, the API for FMUs for Co-Simulation is not primarily intended to be used in DE simulations (for instance, it lacks the functionality to predict internal simulation events).
- **Signals:** The co-simulation of “physical” systems mostly involves the exchange of information that corresponds directly to physical properties (voltages levels, temperatures, etc.). This is typically done – and the APIs of mosaik and the FMI specification are no exception here – by directly sending the values of the associated model variables from one simulator to another. However, communication systems typically do not just exchange values, instead information is transmitted with the help of protocols, which involve meta-data and data formats.

In the following, the proposed extensions for enabling signal-based simulations to the mosaik framework are presented. A discussion of the prototype for an FMU for Co-Simulation (CS) for signal-based simulations can be found in Section 4.3.1. For concrete examples showcasing these extensions please refer to Section 5.3.

##### Extension of the mosaik Framework

The intention for work package JRA2 was to extend mosaik in a way that it supports DE simulations. Unfortunately, a closer look showed that this would require more substantial modifications of mosaik’s core than previously assumed. Work on the subject started nevertheless, but it became clear that the final result would not be available in time for the purpose of JRA2. It was therefore decided to use a workaround that is functional, even though its drawbacks with respect to computational performance and time resolution compared to actual DE simulations are obvious.

In the following, the approach chosen for signal-based simulations in JRA2 is outlined. Its big advantage is that it does not require changes to the mosaik core, but only in the simulators APIs:

- *Logical time*: The internal representation of time in mosaik is an integer value. In one approach examined here, the internal mosaik time is dynamically scaled such that the necessary resolution is achieved (e.g., one tick of internal mosaik time may correspond to 5 milliseconds at one time, and 3 seconds at another time). The simulators have to be implemented such that they can translate mosaik's logical time to simulation time, which can be accomplished quite easily when calling an FMU's step function.
- *Absent values*: For signal-based simulations it is important to define the notion of a signal being absent. This means for instance that the output of a simulator only provides values whenever a signal is being sent. mosaik's implementation in Python allows to set outputs to the built-in constant *None*, which indicates the absence of a value. Simulators have to incorporate this notion accordingly, and only set input values for FMUs whenever a signal – and hence an associated value – is present or provide an output value whenever they send a signal.
- *Message IDs*: In contrast to mosaik, communication network simulators provide dedicated functionality to handle the details of data transmission protocols. Hence, it was decided to encapsulate these details in the simulators themselves, by not giving actual values as inputs to communication simulators. Rather, actual values being sent are associated with unique, integer-based message IDs, which are given to the communication simulator instead (with a special value representing an absent signal). The simulator then processes internally a dummy message associated to this message ID. Once the message is received, the corresponding message ID is given as output, which can then be retranslated to the initial value.

Taken together, these adaptations allow simulating discrete-event systems, albeit with a significant performance loss.

## 4.2 Power System Simulation

For the power system domain, two simulation tools – PowerFactory and PSCAD – have been identified that fulfil the general and domain-specific selection criteria (see deliverable D-JRA2.1 for details). Among other criteria, FMI compliance or at least the existence of a simulation API (or equivalent mechanism) was a determining factor. Since neither of the tools (nor any other comparable power system simulator) has an FMI-compliant interface, the work in JRA2 regarding to power system simulation focused on the development of such interfaces. The following section gives an overview of the accomplished work.

### 4.2.1 PowerFactory

#### Overview

DlgSILENT PowerFactory (DIgital SImuLation and Electrical NeTwork calculation program PowerFactory) [11] 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.

PowerFactory does not officially provide an FMI-compliant co-simulation interface. However, it provides an API that enables basic interactions with simulation models at run-time [12], e.g., setting/retrieving variable values and calculating power flows. A stand-alone tool for creating FMUs from PowerFactory models based on this API was already available before the project start. However, this implementation did only support consecutive power flow calculations. Therefore, to comply with the domain-specific criteria of JRA2, the interface has been extended to support time-domain simulations (referred to as “RMS simulations” in PowerFactory).

PowerFactory provides the possibility to issue so-called events during RMS simulations that can change the system state at a specified point in simulation time. This mechanism has been utilized to enable a dynamic interaction with simulation models at run-time. This mechanism is suited for co-simulation and has been integrated into the FMU exporter tool mentioned above.

### Creating Models for RMS Simulation

This approach relies on PowerFactory's so-called *DlgSILENT Simulation Language* (DSL), a domain-specific language used to program models for electrical controllers and other components used in electric power systems. PowerFactory provides the functionality to associate objects with user-defined DSL models via a composite model (type *ElmComp*). By sending a so-called *parameter event* (type *EvtParam*) to such a DSL model, the model's input parameters can be changed. This change of input parameters can be easily propagated to the parameters of any object by connecting the DSL model with the corresponding object in a *composite model*.

For sending events to a composite model, a dedicated compiled DSL block called *FMIAdapter* is provided. To use this DSL block it must be included into a composite model (see example in Figure 6). The block does not have to be connected directly to any other block, instead it sends events to other blocks (in the example to the block called *Controller*) to change their input parameters. To which blocks these events are sent is defined in the FMU export configuration.

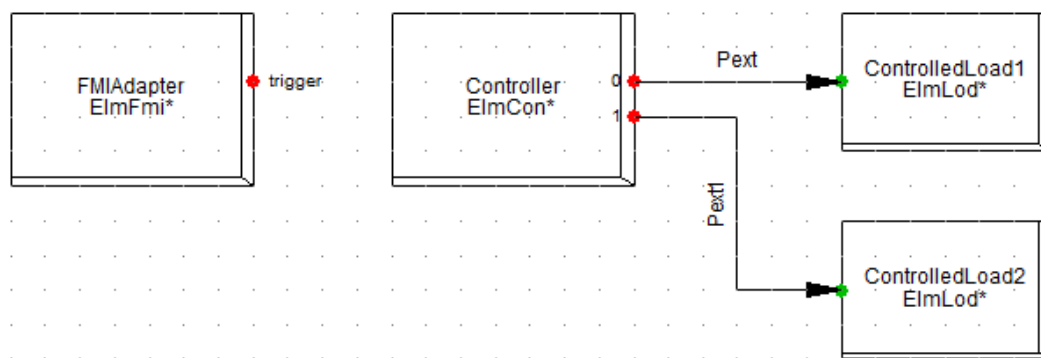


Figure 6: Example of a composite frame including DSL model *FMIAdapter*

### Exporting Models for RMS Simulation

To create an FMU from a PowerFactory model that is intended for RMS simulation, a dedicated Python script has to be executed. When calling this script, the user has to provide relevant information such as the FMU model identifier (i.e., the FMU's name), the PowerFactory model or lists of input and output variable names. For inputs intended to be sent as event via DSL block *FMIAdapter*, a dedicated naming convention has been devised that specifies the name of the receiving block and its input parameter name.

## 4.2.2 PSCAD

### Overview

PSCAD is a commercial general-purpose time domain simulation tool for studying transient behaviour of electrical networks. Its 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. As addressed in deliverable D-JRA2.1, the most recent versions of PSCAD provide a built-in interface for MATLAB/Simulink. However, this interface proved to perform poorly in TC1 scenarios, as data exchange was remarkably slow.



PSCAD provides a so-called *Automation API*, a Python interface that can be used to interact with the software, e.g., for opening and loading projects, setting model parameters and running simulations. Despite providing a wide range of functions, this API cannot access the simulation model's data signals that are needed for co-simulation. However, user components in PSCAD are programmable with Fortran, which in turn allows cross-compiling C/C++ code and thus enables the development of a co-simulation interface.

### Co-simulation Interface

A prototype of an FMI-compatible co-simulation interface for PSCAD has been developed in work package JRA2. It consists of two parts:

1. A back-end component used by PSCAD, and a front-end component used by the master algorithm. This part is implemented as user-defined components called *pscad\_send* and *pscad\_rcv* (see Figure 7 for an example). They take desired input/output signal names as arrays, and at configurable time intervals (co-simulation time-step) they send/receive data over a socket. Socket communication functions are written in C.
2. At the time of writing this report, the front-end component has been implemented as a “mock-up FMU”, i.e., a Python program that acts as a socket server, exchanging data with PSCAD and exposing it to the master algorithm. It also handles the time synchronization between the master and the slave. The developed co-simulation interface does not currently implement the FMI standard, but is compatible with it. After successfully demonstrating feasibility of the co-simulation with PSCAD, further work would include rewriting the front-end component in C++ with the help of the FMI++ library and encapsulating it into a real FMU.

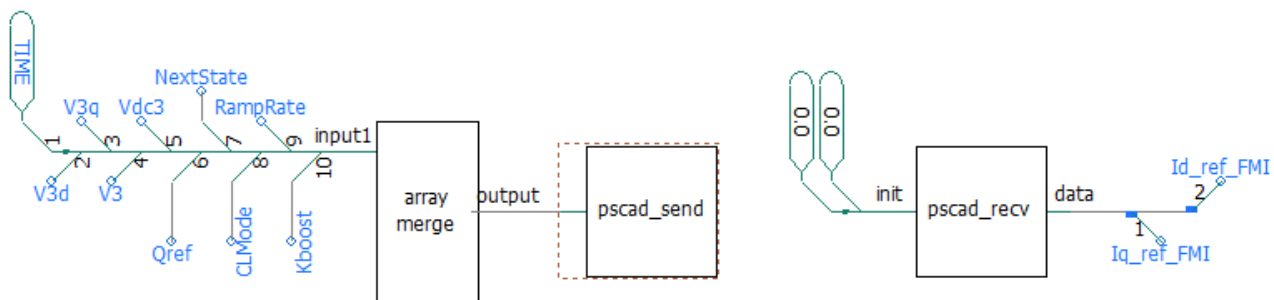


Figure 7: Example usage of PSCAD components for the FMI-compliant simulation back-end

## 4.3 ICT Network Simulation

### 4.3.1 ns-3

#### Overview

When implementing a co-simulation scenario where all the basic functionalities of a Smart Grid system are included, the telecommunication aspect of the system has to be addressed with caution. Today, many software tools are available for the assessment of communication networks, each one with its own benefits and drawbacks. In the scope of JRA2, the *ns-3* network simulator software was selected (see Deliverable D-JRA2.1). In recent years, *ns-3* has become very popular in the network simulation community, mainly due to its high flexibility, which allows programmers to add new attributes without modifying the “core” of the source code, or having to deal with a specific, restricted and complex API.

The development of *ns-3* is based on popular object-oriented programming languages (C++ and Python) and the design of *ns-3* also follows an object-oriented approach. The default version of *ns-3* comes with an extensive library of models, which can be used to describe the components and other

aspects of a communication network (e.g., devices, channels, interfaces, protocols). Additionally, ns-3 provides special objects that are associated with more general grid characteristics, such as the geographical grid extension, the (non-)presence of buildings and other obstacles or the mobility of a node.

### FMI Export Functionality

A dedicated ns-3 package called *fmippex* has been developed in work package JRA2, which provides all functionality needed for creating an FMU for CS from a user-defined ns-3 application (typically referred to as “script”). An FMU created with the help of this package implements a tool coupling mechanism that allows to control the execution of the ns-3 simulator and to establish a connection for data exchange during run-time.

This implementation represents a first prototype, which provides a proof-of-concept that an FMI-based co-simulation interface can indeed overcome the challenges associated with signal-based simulations (see Section 4.1.3). As such, the *fmippex* package can be successfully used to create FMUs from ns-3 models, even though there are some restrictions to its use:

- *Simple event queue*: The interaction with ns-3 is limited to the repeated execution of the same ns-3 script, i.e., whenever the FMU’s *doStep* method is called, ns-3 executes the same model (but different random seeds and hence different outputs). During one such simulation run, events and corresponding output values – such as receiving a message – can be added to an internal event queue. The entries of this event queue can then be accessed as the outputs of the corresponding FMU.
- *Internal FMU events*: Up to the most recent version of the FMI specification (FMI 2.0.1), there is no support for handling internal events in FMUs for CS.<sup>1</sup> Therefore, a “quick-and-dirty” solution has been implemented, which allowed to demonstrate the feasibility of the approach and that can be easily extended once FMI support is available. This solution uses so-called “iterations” (simulation steps with step size equal to zero) to trigger the FMU to process events (e.g., check for new inputs or re-run the ns-3 script). Furthermore, the FMUs define a dedicated output variable for event prediction, whose value always corresponds to the time of the next event in the queue.

The package is developed on top of the FMI++ library, which provides a generic approach for FMI-compliant tool coupling [13], see Figure 8. The approach relies on a *front-end component* to be used by the simulation master and a *back-end component* to be used by ns-3. Between these two components, a proper data management is established that is responsible for the communication and data exchange between both ends. To connect an ns-3 script with the back-end, the user has to implement class *SimpleEventQueueFMUBase*, which provides easy-to-use functions for declaring input and output variables or adding events to the event queue.

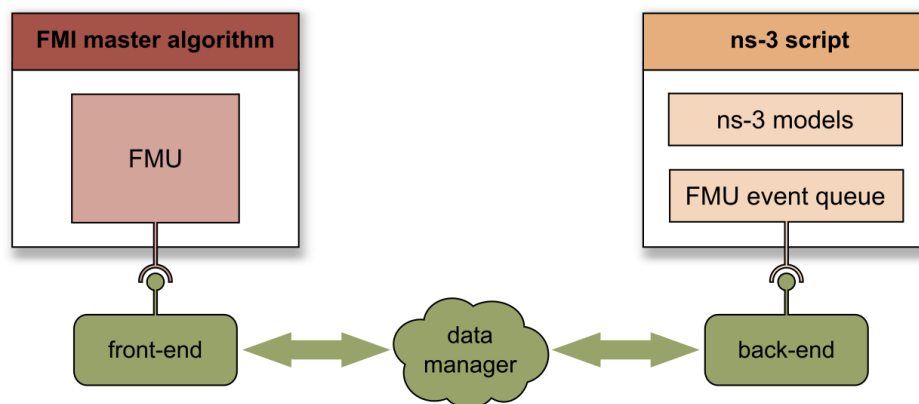


Figure 8: Schematic view of the FMI-compliant ns-3 interface

<sup>1</sup> However, there have been suggestions in literature how to overcome this issue (see deliverable D-JRA2.1 for an overview) and it is expected that FMI 3.0 will address it.

### Application Layer Models

For the purpose of work package JRA2, several application layer models were developed in order to meet the specific demands of the co-simulation scenarios. In the following, the application layer models used for test case TC3 are listed as an example:

- *Controller server*: This application layer model simulates the server functionality of a Coordinated Voltage Control (CVC) controller device. The role of this model is the same as the role of a server in every network, i.e., it creates a socket where the smart meters connect and sends data. In addition, after a packet is received, the headers are checked, and the end-to-end delay of the transmission is calculated and stored. Later in the simulation, this delay is used to calculate the corresponding timestamp and add an event to the event queue.
- *Smart meter custom client*: This application model simulates a smart meter device. The model's purpose is to establish connection to the CVC's server socket and send a packet of fixed size (100 bytes). For the calculation of the end-to-end delay, the time of the packet creation is added to it as part of a header.
- *Controller client*: This application model simulates the CVC controller's client aspect and helps to establish the connection between the controller and the OLTC transformer. More specifically, it implements a UDP client-like functionality, managing the creation and sending of packets (containing data associated with new setpoints) to the transformer. For the calculation of the end-to-end delay, the time of the packet creation is added to it as part of a header.
- *OLTC custom server*: This application model simulates the transformer server aspect and helps to establish the connection between the CVC and the OLTC controller. More specifically, it implements a UDP server-like functionality, receiving the packets (containing data associated with new setpoints) sent by the CVC. Here the header, containing the time the packet was sent by the CVC, is retrieved and the end-to-end delay between the CVC and the OLTC controller is calculated.

Please note that the choice of using UDP for the application models of the CVC controller client and the OLTC controller server is not meant as a recommendation of this technology for similar real-world systems. It was rather a convenient choice for the purpose of modelling an ICT network, resulting in simple yet realistic models for the proof-of-concept validation in Section 5.3.

### **4.3.2 Handling of Cyber-Attacks**

#### Overview

Previous work on applying the risk management process explained in Section 3.3.3 on Smart Grid use cases has shown that ensuring cyber-security for (potentially large) cyber-physical systems is challenging. Cyber-systems (e.g. ICT and SCADA systems) are used to control physical systems and processes (tap changers, PV inverters, etc.). Since cyber-risks are addressed by cyber-security experts, a major challenge when conducting a risk assessment is the identification and quantifications of the physical consequences of cyber-attacks. This problem is even more challenging for large-scale cyber-physical systems, in which it is necessary to understand emerging behaviour and phenomena of complex systems as a whole and be able to judge whether a cyber-attack could cause unwanted phenomena or system-wide instabilities. However, an advanced co-simulation framework targeting a holistic assessment of Smart Grid system as developed within work package JRA2 can assist in understanding impacts and consequences of cyber-security attacks. Subsequently, this can be used within the risk management framework for large-scale Smart Grid systems explained in Section 3.3.3.

#### Consequence Identification Assessment

Designing new applications for Smart Grid systems is a complex process that requires careful testing and validation of functionality, in order to make sure it is correctly implemented, and design parameters are well tuned. As discussed above, such holistic technical assessments come with their own

set of requirements for the co-simulation environment and coupled tools, in order to be able to handle (potentially large) and complex Smart Grid configurations. Besides these challenges, it is also necessary to assess Smart Grid test cases with respect to potential cyber-security threats and identify potential consequences of cyber-attacks on the Smart Grid system. This consequence assessment is crucial to decide which Smart Grid components should be secured and how. But it can also be used to create evidence that certain functionality needs to be improved and made more resilient.

Within work package JRA2, a prototype for consequence identification has been implemented and integrated into the Smart Grid ICT assessment toolchain. In the following, the implemented features are illustrated, based on a dedicated version of test case TC3 focusing on cyber-security assessments. For details about the results of this test case refer to Section 5.3.

Figure 9 depicts a co-simulation setup implementing test case TC3. This voltage control test case is implemented on a simple electrical distribution grid topology, which comprises two loads and a transformer with an OLTC that regulates the voltage within specified operating conditions. In the figure, the voltage controller is labelled *ControllerSim*, the power grid is shown as *LoadFlowSim*, and the ICT network is shown as *CommSim*. The voltage controller periodically receives measurements from two voltage meters ( $U_3$ ,  $U_4$ ), based on whose values it periodically calculates the OLTC's tap position. Helper components called *Metaizer* are used to add meta-information to the simulated signals whenever needed. Similarly, this meta-information can be stripped from the signal using helper components called *Unmetaizer*.

In order to identify consequences of cyber-attack on the voltage control functionality, it is necessary to instantiate cyber-attack patterns on at least one of the control loop signals. Depending on the type of the control signal (continuous or discrete), either the *ContinuousSignalAttacker* or *DiscreteSignalAttacker* are introduced into the control loop. Once the cyber-attack has been instantiated on the control signal within the co-simulation setup, it is necessary to select an attack pattern that will fuzz the signal in order to "functionally mimic" a real cyber-attack.

The following attack patterns have been specified and implemented in the co-simulation framework:

- *Scaling attack*: The attacker modifies the true measurements to higher or lower values by multiplying it with a certain scaling factor.
- *Random attack*: The attacker adds random uniform values to the true measurements.
- *Ramp attack*: The attacker adds values to the true measurements gradually, based on a function that increases or decreases with time.

#### 4.4 Coupling with Hardware Setups

In order to leverage the flexibility of the FMI in reducing tool-specific implementation efforts for creating Hardware-in-the-Loop (HIL) setups, a generic interface application that couples simulation models and external hardware was designed and implemented [14]. The following section gives an overview of the accomplished work; results from several test cases are given in Section 5.2.

##### Virtual Component Interface Design

Although a large variety of numerical models can be represented as FMUs, hardware interfaces which border individual Hardware-under-Test strongly vary in different use cases and limit a general applicability. In addition, various functional and timing requirements lead to a vast amount of different solutions ranging from dedicated HIL simulator hardware which features time steps in the microsecond range up to tool-specific connections based on common ICT technologies. Many domains utilize standard automation infrastructure to implement controllers and to interface the controlled plants. Often standardized communication protocols such as Modbus and IEC 61850 are used to couple components from various vendors and access deployed automation equipment.

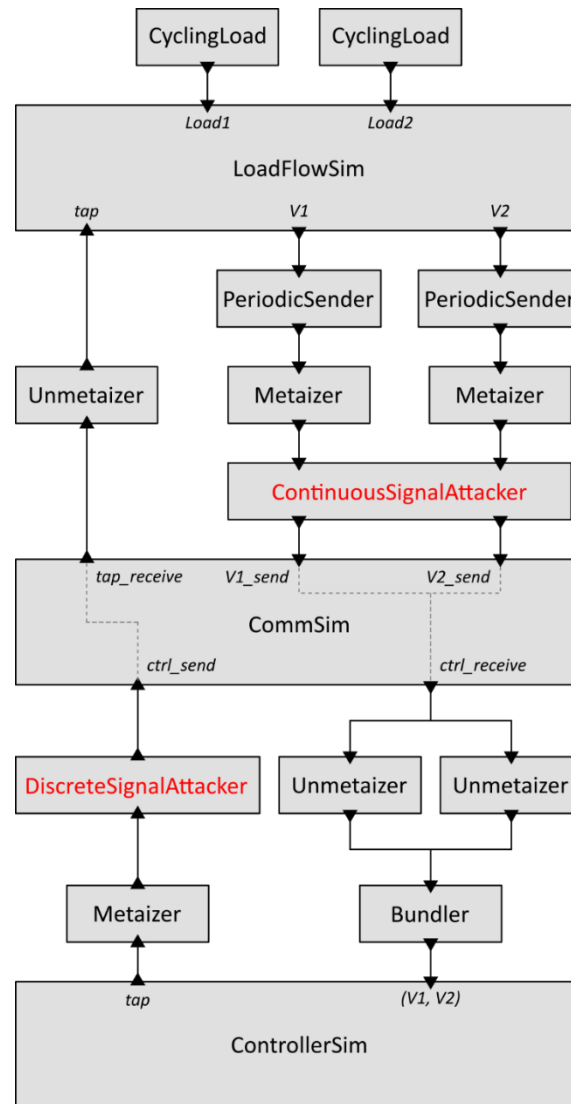


Figure 9: mosaik simulation setup including cyber-attack models (highlighted in red)

To enable a broad applicability of a generic hardware interface and to reduce case-specific implementation efforts, the concept of generic virtual components is developed. A generic virtual component is a simulated entity whose behaviour is defined by a numerical model and which can be accessed via standardized communication interfaces as if a real implementation of the simulated entity is present. Since both, the behavioural specification of the component in terms of a numerical model and the interface to existing hardware use established standards, a broad applicability beyond a specific use-case can be achieved.

The prototypical interface application which instantiates virtual components is designed as a stand-alone application without tight integration into a specific development environment. Such a tight integration may reduce the configuration effort of parametrizing a virtual component. It may utilize existing topological information from the automation system, but a tight integration reduces the vendor-neutral applicability. To configure a test setup, it is particularly necessary to specify the mapping of model variables and linked network variables. A scriptable command-line interface allows the configuration of every parameter, including the topological information, the coupling algorithm and the algorithms used to numerically solve a model.

Due to the vast amount of different communication protocols, it is unlikely that a single generic interface application implements all relevant standards. Hence, the interface application is designed in a

protocol agnostic way such that new protocols can be easily implemented. To prove the concept, a single ASN.1-based communication protocol, which is defined in the IEC 61499 standard, is implemented. Since the protocol is entirely event-based and does not require polling, communication schemes, which are not restricted to periodic communication, can be studied in detail.

### Coupling Algorithms

Since most automation systems implicitly operate in real time only, any communication between the virtual component and the automation system needs to be performed at the corresponding instant of real time. Hence, for each exchanged (synchronization) event, the notion of simulation time needs to be synchronized with the notion of real time. One common approach of coupling hybrid simulations with discrete event-based systems is periodic synchronization as illustrated in Figure 10. Between strictly periodic synchronization points, both, the model and the real components are executed independently. As soon as the predefined synchronization point is reached, results are exchanged, and one event is triggered in the discrete event-based system. Any change of a synchronized variable, either discrete or continuous in nature, must be delayed until the next synchronization point. Although the synchronization approach imposes few real-time constraints beyond solving each interval faster than real time, loss of information is introduced by artificially delaying results.

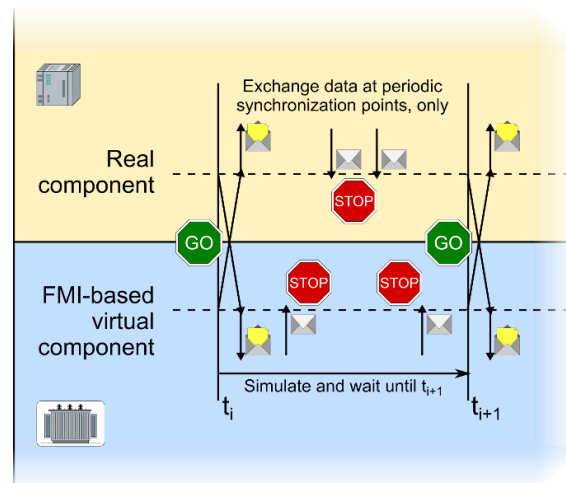


Figure 10: Periodic synchronization

To overcome the limitation, a predictive synchronization approach, which originates from offline coupling of FMI-based models and event-based simulations, is adapted for real-time operation. Figure 11 illustrates the predictive synchronization approach. To issue an event in time, the hybrid model is solved ahead under the assumption that no external event from the real component is triggered and that all operations finish in time. As soon as a discontinuity or a large deviation is encountered in the model, a discrete event is prepared which then can be triggered at the appropriate instant in real time. Still, the real component may trigger an external discrete event that invalidates any prediction beyond that time. Hence, future predicted events must be withdrawn, updated values must be applied and the prediction step starting with the external event will be repeated.

The repetition of some simulation steps imposes several subtle challenges regarding the reset operation. To limit the performance impact, a period should not be repeatedly solved unless the inputs change. Following the event-accurate synchronization paradigm, events are not restricted in time and consequently, intermediate model states are frequently recorded. By recalling previous records and interpolating values between state nodes, the state is reset to an arbitrary point in time. Since only some FMI-based models expose the discrete state, only the continuous state can be fully accessed and reset. Hence, reset operations must be limited to continuous intervals only and must not be issued beyond FMI events. If an external event is lately submitted after an FMI-event is executed, the model cannot be reset, and the late event cannot be applied in time.

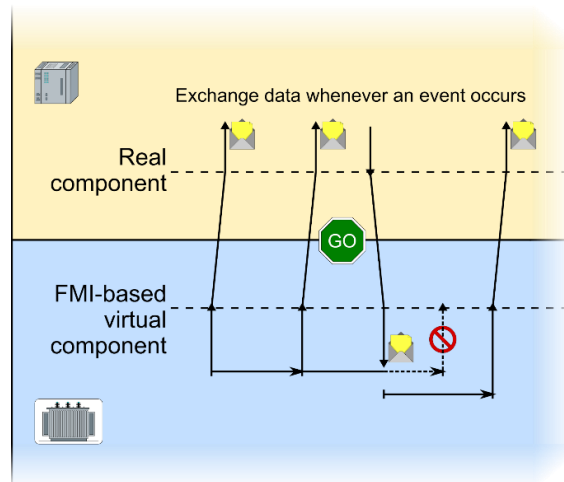


Figure 11: Predictive synchronization

Since communication is not restricted to predefined points in time only, the predictive approach imposes several strict real-time conditions that can hardly be guaranteed in practice. In particular, time between two consecutive events must be limited such that the next predicted event can be safely predicted in time. To allow a sound practical operation, the timing of each event in a simulation run is recorded and the real-time quality of a simulation run can be evaluated. By implementing the soft real-time approach, the need for detailed code and model analysis is eliminated but hard real-time systems such as some power hardware cannot be coupled without further safety measures.

#### Virtual Component Implementation

The prototypical interface application called *FMITerminalBlock* is implemented in C++ and released under an open-source license [15]. A unified program flow allows the support of both, periodic and predictive synchronization without changing the deployed event queue and real-time synchronization. Therefore, model events may get outdated, even if periodic synchronization is used. In case of periodic synchronization, no reset operation is performed, and the previously cached synchronization event is returned until it is actually scheduled.

To efficiently access and solve FMI-based models, the FMI++ library [16] has been used. Since FMI++ already requires several Boost libraries [17], it was decided to use Boost for platform independent network access, concurrent programming and automating test cases. A detailed description of the implementation, the program flow and the synchronization algorithms can be found in [14].

## 5 Proof-of-Concept Validation

### 5.1 Power System Simulation with Cyclic Dependent Models (TC1)

#### 5.1.1 Test Case Specification

The main test objective of this test case – referred to as test case TC1 – is to show the *Fault Ride Through* (FRT) capability of a *Wind Power Park* (WPP) that is connected to a small transmission system. The wind turbines inside the WPP are of type 4 and have a fully rated converter interface. The compliance to the grid codes of the WPP is tested. The entire WPP is modelled as one aggregate *Wind Turbine Generator* (WTG), with the collection system and the WTG treated as one single entity. The FRT curve is enforced at the coupling point.

Test case TC1 deals with cyclic dependencies between the electrical domain and the control domain. The definition of TC1 comprises a power system that is simulated in a domain-specific power system simulator, while its control mechanisms are simulated in another simulator. During the simulation, a short-circuit event occurs, which triggers the control mechanisms to interact with the power system. From the co-simulation point of view, the main challenge here is the proper data exchange during runtime, which has to consider the cyclic dependencies between the co-simulation models.

The goal of TC1 is to demonstrate the feasibility of the FMI-based co-simulation toolchain developed in work package JRA2 for the assessment of models with cyclic dependencies. Therefore, the simulations have been conducted in two stages:

1. A simulation is conducted entirely with the power system simulator, using a monolithic model of both the power system and the controllers.
2. A co-simulation using the JRA2 toolchain is used, where the controls and the power system are simulated in different simulators.

The results of the monolithic simulation and the co-simulation are then evaluated to measure the performance of the co-simulation toolchain.

#### 5.1.2 System under Test

The SuT defines the system configuration of a specific test case, comprising the list of systems, subsystems, components, interactions and behaviours in a specific test case. The SuT definition also includes the *Object under Investigation* (Oul) and the *Domain under Investigation* (Dul). As the name suggests, the Oul comprises the systems and/or components that are to be characterized or validated. The Dul identifies the relevant domains and their connectivity in the test case.

The SuT of the TC1 monolithic simulation is shown in Figure 13. It consists an IEEE 9-bus transmission grid (see Figure 12) with one of its synchronous generators (G3) replaced by a WTG. The WTG must comply with the grid code specification of a low-voltage ride through time against the voltage profile. Hence, a FRT curve is enforced at the PCC and the electro-mechanical conversion components.

Similarly, Figure 14 shows the SuT for the co-simulation experiments. The main changes from the monolithic simulations are the wind turbine aerodynamic model as well as the DC-link model, for which the wind speed and pitch angle are assumed to be constant during the assessment.



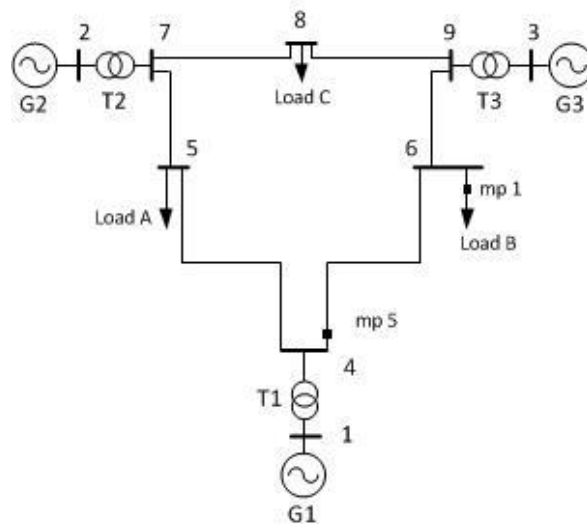


Figure 12: IEEE 9-bus transmission system

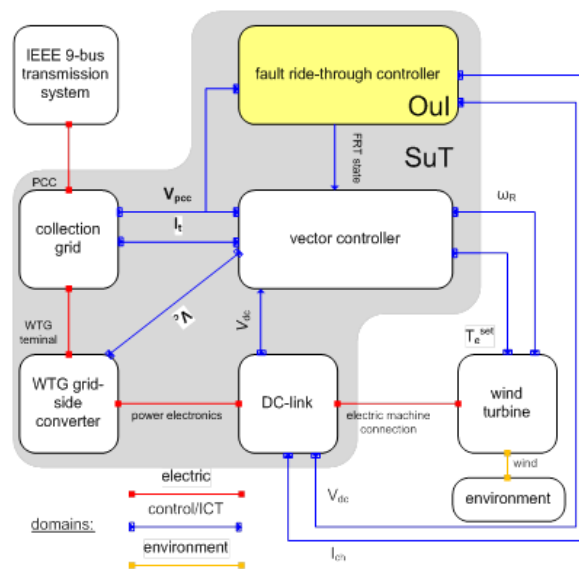


Figure 13: SuT for the TC1 monolithic simulations

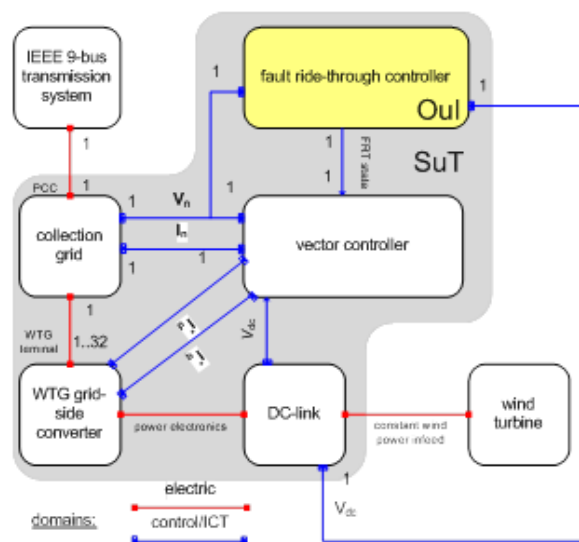


Figure 14: SuT for the TC1 co-simulations

### 5.1.3 Monolithic Reference Simulation Implementation

The monolithic reference simulation is performed as an RMS dynamic simulation in DigSILENT PowerFactory. For this purpose, the static generator was equipped with a dynamic controller built in the proprietary DSL environment. The controller does not implement all functions to a degree common in commercial wind turbines. However, it is sufficient for the purpose of demonstration of dynamic behaviour, yet complex enough for a reasonable validation of the co-simulation.

The dynamic model includes continuous measurement of active power, reactive power and voltage at the terminals of the wind turbines. The measurements are passed on to the converter controller model, which calculates the current orders for the static generator. A PLL measurement device connected to the terminals of the static generator provides the voltage reference for the static generator. The overview of this logic is depicted in Figure 15. The figure also shows that current output values are fed from the static generator into the controller, which is done only for the purpose of initialization of the controller.

The controller model is shown in Figure 16. The measured active power is used together with an active power setpoint (parameter) in a PI controller to control the current in the d-axis. The current in the q-axis is also controlled by a PI controller, which uses as input either the measured reactive power and the reactive power setpoint (in case of operation in reactive power control mode), or the measured voltage and the voltage set point (in case of operation in voltage control mode). Furthermore, the q-current can be boosted in case the FRT flag is triggered. The FRT flag is triggered by a corresponding block in case voltage drops below 90% of nominal and the q-axis current boost is proportional to voltage drop. To prevent excessive recovery time, both PI controllers have wind-up limiters of the integrators set to 110% of nominal current. The generated current orders are passed through current limiter, which ensures that resulting current does not exceed 110% of nominal current. The current limiter operates with q-axis preference. Lastly, the d-axis current has a ramp-rate limiter.

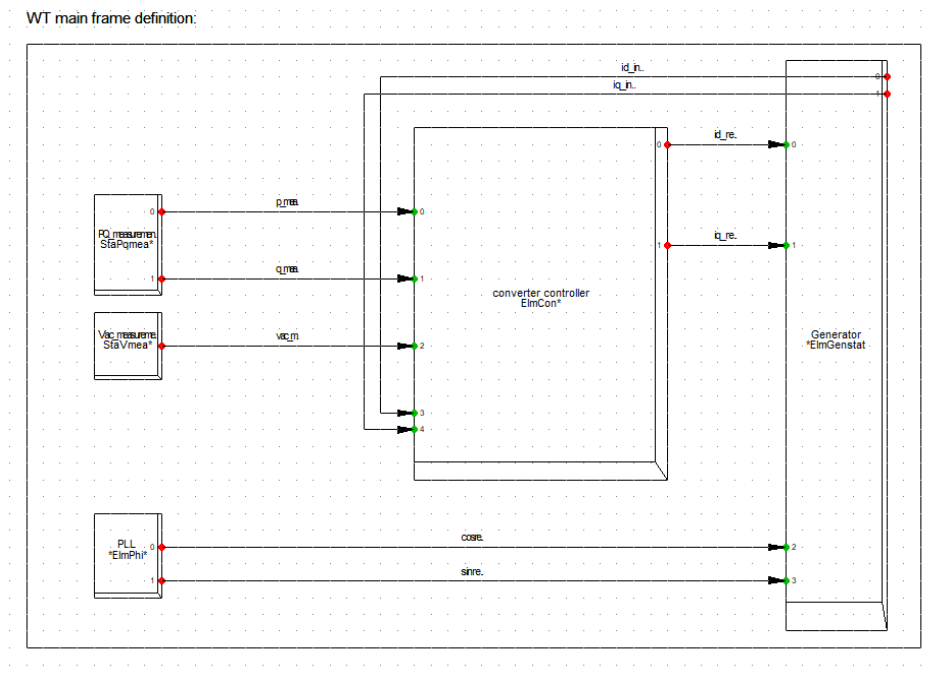


Figure 15: Dynamic model overview for TC1

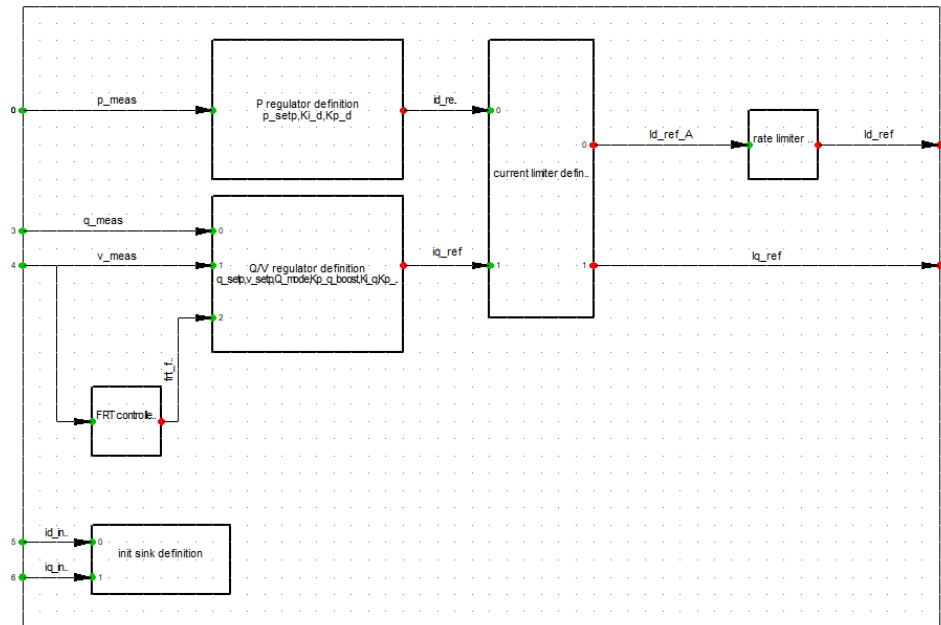


Figure 16: Controller model overview for TC1

#### 5.1.4 Co-simulation Implementation Using PowerFactory and Simulink

The co-simulation is implemented between PowerFactory and Simulink, using the co-simulation tool-chain developed for work package JRA2. The FRT controller model and converter controller model for the WPP are exported from Simulink as FMUs for Model Exchange. The transmission system along with the WPP are modelled in PowerFactory and exported as FMU for CS. Figure 17 shows an overview of the co-simulation setup.

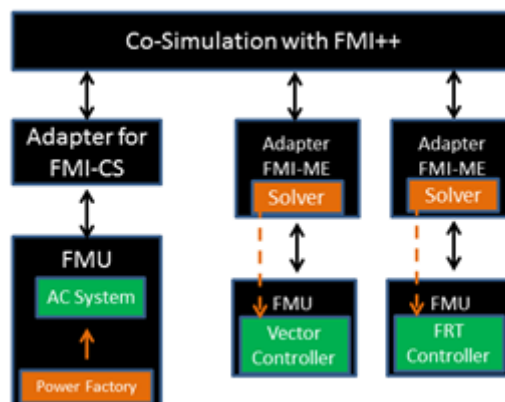


Figure 17: Implementation of the TC1 co-simulation using FMUs from Simulink (ME) and PowerFactory (CS)

In PowerFactory, the IEEE 9-bus system is again used as the transmission system. The synchronous generator G3 is replaced by the WPP. The WPP is connected to the rest of the transmission system via an equivalent impedance, as depicted in Figure 18.

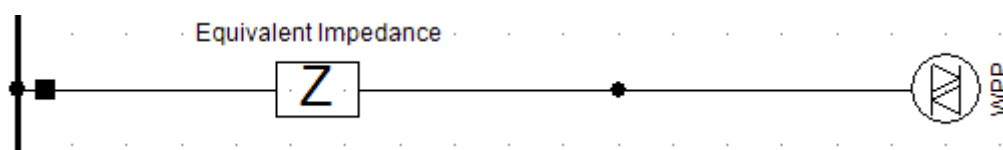


Figure 18: WPP connected to IEEE 9 bus via Equivalent Impedance

The WPP itself is represented as a static generator and equivalent impedance. The static generator represents lumped converters of wind turbines with applicable controls. In load flow calculations, the static generator is a PQ node, whereas in dynamic RMS simulations it behaves as a current source. The static generator works in the d-q reference frame and requires as input a voltage reference (from PLL measurement device) and current orders in d- and q-axis respectively (from the controller). The equivalent impedance represents the impedance of the collection grid, step-up transformer for connection as well as internal impedance of wind turbines, as seen from the grid.

Figure 19 shows the FMI adapter slot logic inside PowerFactory, which facilitates the data exchange of the  $I_d$  and  $I_q$  reference values with the converter controller FMU.

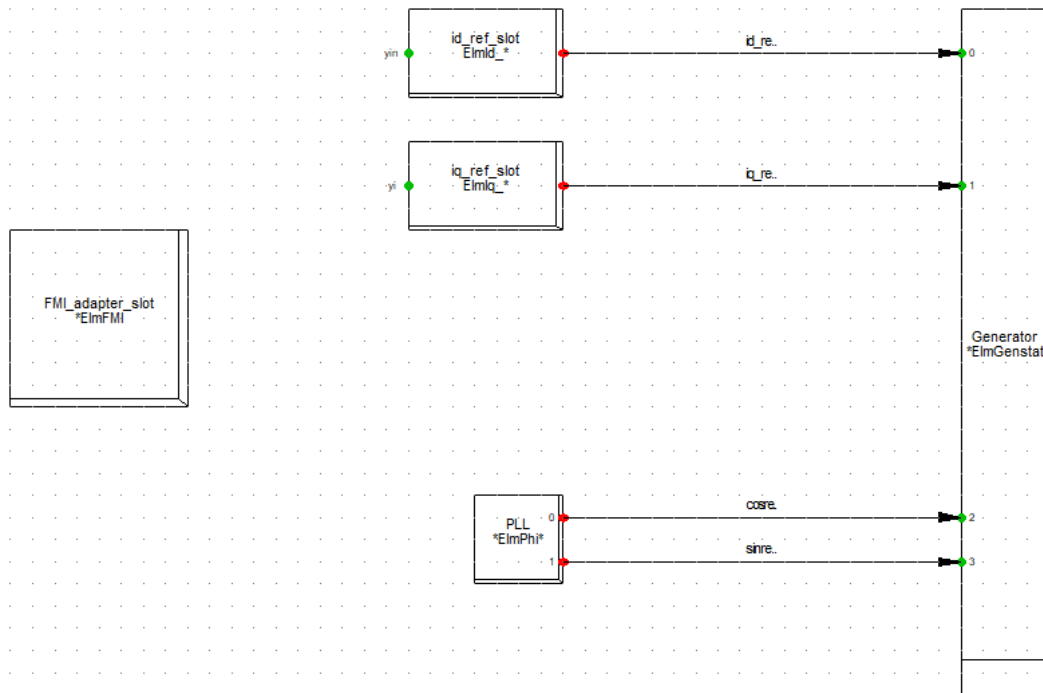


Figure 19: FMI adapter slot logic (PowerFactory co-simulation)

### 5.1.5 Co-simulation Implementation Using PSCAD and Simulink

Using an analogous approach as in Section 5.1.4, the WTG controllers, including the FRT and converter controller, are modelled in Simulink and are exported as FMU. The transmission system, i.e., the dynamic IEEE 9-bus system, along with the WPP are modelled in PSCAD, as shown in Figure 20. Analogous to Section 5.1.4, the synchronous generator G3 is replaced by a WTG, which is modelled in PSCAD by a converter (*Turbine* block in Figure 20).

The entire PSCAD simulation is exported as an FMU, using the *pscad\_send* and *pscad\_recv* blocks, developed for the co-simulation interface of PSCAD (see Section 4.2.2). By using this interface, the co-simulation can be conducted with the help of the FMI++ Python Interface (see Section 4.1.1). Via the master program, these blocks send measurements to the controller implemented in Simulink, e.g. time,  $V_d$ , and  $V_q$  and in turn receive the controller's outputs, i.e., the reference values for the WPP converter.

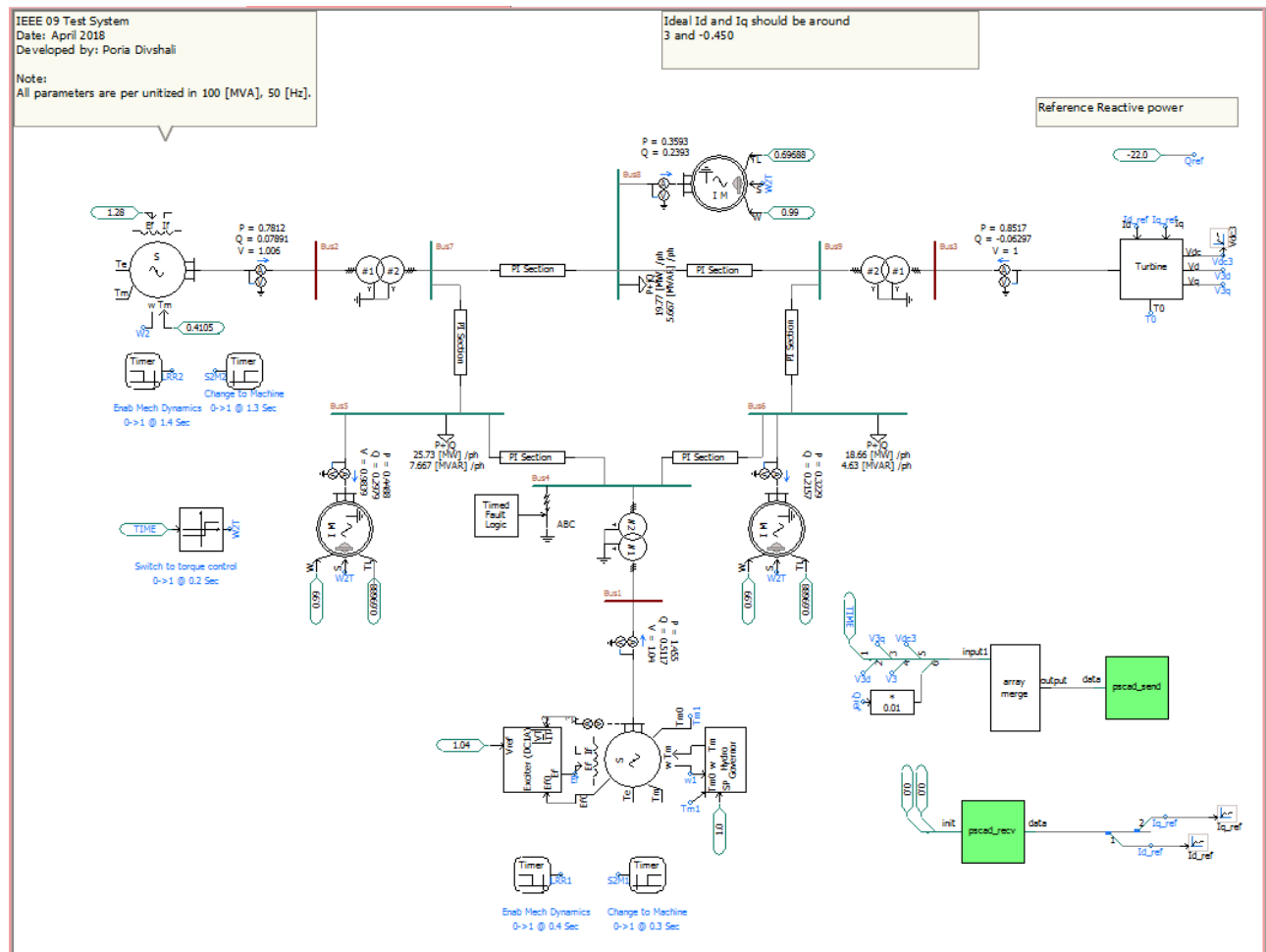


Figure 20: The dynamic IEEE 9-bus system model, implemented in PSCAD

### 5.1.6 Results

Both the monolithic reference simulation and the co-simulation implement the same test case and the results are then compared. A fault event is triggered at simulation time 1 s with a fault duration of 200 ms.

The results of the co-simulation using PowerFactory and Simulink have been compared with the results from the PowerFactory monolithic simulation and are presented below, see Figure 21 to Figure 24. As can be seen, the simulation traces of the monolithic reference simulation and the co-simulation are in very good agreement, indicating that the co-simulation basically produces the same results as monolithic simulation.

To validate the co-simulation using PSCAD and Simulink, the results of the co-simulation have been compared with the results from the PSCAD monolithic simulation and are presented below. The simulation is performed for 20 seconds, and the fault occurs at simulation time 0.4 s (link in the case of PowerFactory). Since PSCAD – unlike Simulink or PowerFactory – starts from zero initial condition, the first 10 seconds (before simulation time 0 s) are used for initialising the model. As shown in Figure 25 to Figure 28, the co-simulation and monolithic simulations trace the same and overlap each other indicating that the co-simulation produces the same results as the monolithic simulation.

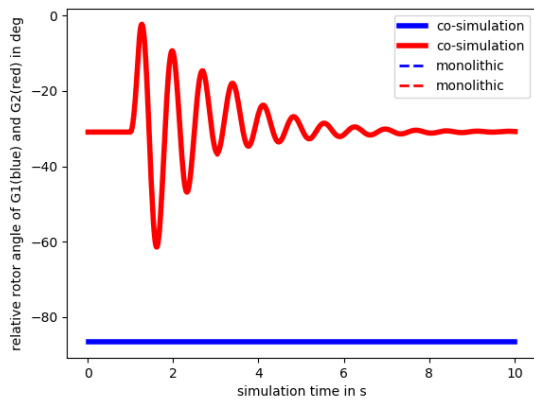


Figure 21: Relative rotor angles of G1 and G2

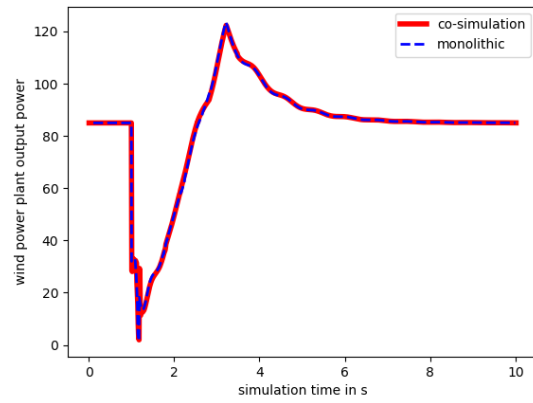


Figure 22: WPP output power

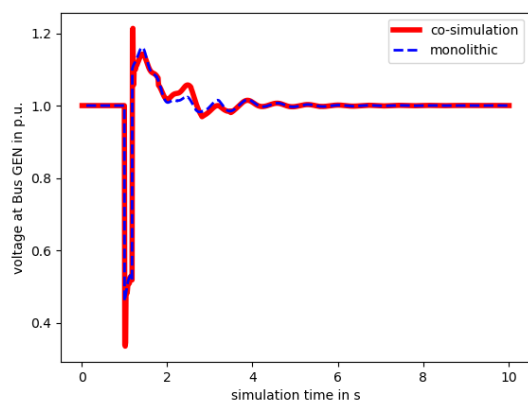


Figure 23: Voltage at bus 9 (where the WPP replaces generator G3)

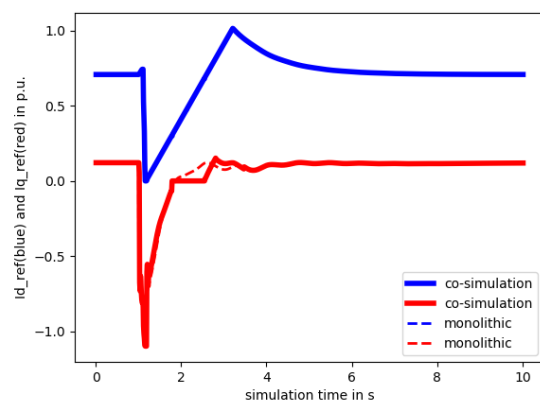


Figure 24: Id and Iq reference currents

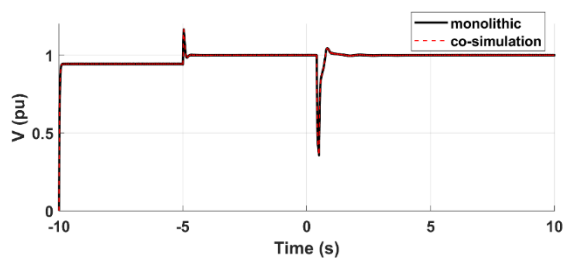


Figure 25: The output voltage of the WPP

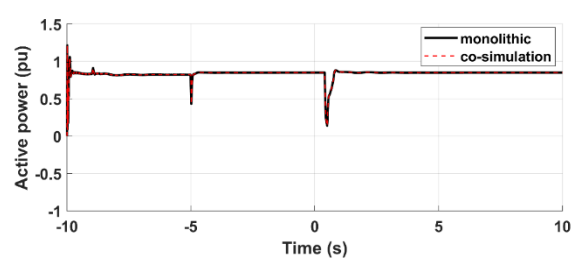


Figure 26: The active power output of the WPP

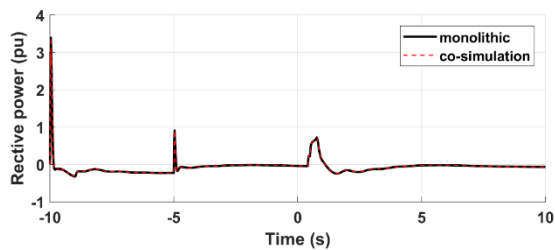
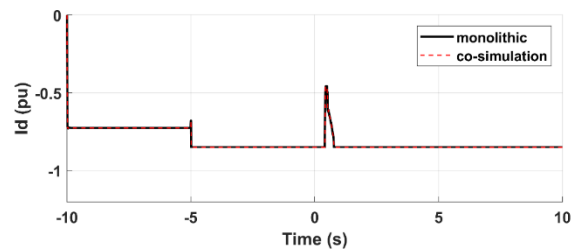


Figure 27: The reactive power output of the WPP

Figure 28: The reference current ( $I_d$ ) of the WPP

## 5.2 Combined Hardware and Software Simulation (TC2)

### 5.2.1 Overview

Deliverable D-JRA 2.1 introduces the rationale of a combined hardware and software simulation (TC2) in more detail. The herein presented work extends the findings of D-JRA 2.1 and documents improvements on the original experiment design.

As described in deliverable D-JRA 2.1, the test case aims at demonstrating and refining model-based hardware and control development methods. All experiments are based on a single SuT in the context of a smart electricity distribution system. First, relevant models are created and then used to guide further hardware development. Therefore, the model is used to automatically derive controller implementations and to assess an embedded controller hardware. A novel FMI-based method that links hardware via generic interfaces is applied and evaluated.

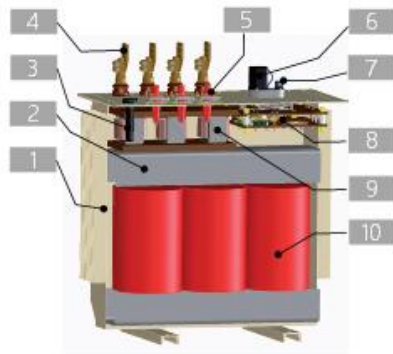
Test case TC2 presents two main contributions. First, a method for coupling event-based simulations via the FMI standard for model exchange was extended to event-based automation systems [14]. In particular, a roll-back mechanism is used to prevent the emission of outdated events. In [14], relevant real-time properties and constraints are analysed in detail. The method is compared to a conventional periodic synchronization approach. Two practical experiments demonstrate both coupling approaches and link the results to a reference simulation. It is shown that the event-based approach can couple models, which cannot be accurately linked with periodic synchronization.

Second, a method for creating HIL setups based on the concept of generic virtual components is presented and implemented into a prototypical generic interface application. In contrast to existing solutions, virtual components use standardized communication interfaces and the FMI to interface components under test. Hence, they provide cost-effective development options without the need of cost-intensive HIL equipment. Special attention was put in the flexibility of the interface program, which allows to adapt various parameters such as the synchronization approach and numerical integration properties. The application of generic virtual components and the prototypical interface application is demonstrated in several experiments, which cover various model-driven development stages. Additionally, proprietary modelling and hardware coupling techniques are included to assess existing techniques and compare them to the generic virtual component concept.

### 5.2.2 System under Test

In the ongoing evolution of existing power grids towards Smart Grids, the intermittent behaviour of Distributed Energy Resources (DER), integrated into the distribution system are changing existing power systems. Maintaining the voltage within the tolerance limits is currently a challenge in the power grid and refers to the ability of a power system to maintain steady voltages at all buses in the system, in particular after it has been subjected to a disturbance. To deal with this phenomenon, many transformers are equipped with an OLTC changing the transformer ratio whilst energized. Therefore, OLTCs and their control infrastructure collect voltage signals from the controlled busses and change the tap positions of the transformers according to the received signals.

One challenge to use this technology in MV networks is how to avoid internal arcs inside transformer dielectric liquid tank. At *Ormazabal Corporate Technology* (an Ormazabal subsidiary), engineers are developing this on-load tap change technology using vacuum interrupters to deal with this challenge. Figure 29 illustrates the main components of a transformer, which is equipped with an OLTC.



1. Tank and dielectric liquid.
2. Ferromagnetic core.
3. Temperature sensor.
4. MV plug-in bushings.
5. Low voltage (LV) terminals.
6. OLTC motor.
7. OLTC position indicator.
8. OLTC gearing system.
9. OLTC reactors.
10. MV and LV windings.

Figure 29: Sketch of a transformer with integrated OLTC

Another challenge is to develop OLTC controllers that need to effectively and safely drive the OLTC such that stable voltages are maintained. To guarantee a stable and efficient system, both the OLTC and transformer hardware and the control algorithm need to be jointly considered. It is not sufficient to develop and test the OLTC controller without detailed knowledge of the controlled system. Hence, advanced development and engineering methodologies are necessary, which allow a rapid development of controller implementations while considering specifics of the controlled equipment. The SuT as studied in the outlined test case is comprised of an OLTC, its controller, and its associated test environment such as power sources and loads. A focus is put on various OLTC controller implementations, which were selected as Oul.

### 5.2.3 Experiment 1: Monolithic Reference Simulation

In the first experiment, a monolithic reference simulation of the SuT is created via state-of-the-art modelling and simulation tools. The model is then used in the experiment to assess the intended control functionality. Various over and under voltage conditions are applied to test the control algorithm and its ability to maintain stable voltage levels.

#### Monolithic Reference Model Implementation

This section describes a model of an OLTC controller, which simulates the behaviour of a real hardware device. This OLTC controller model has been designed to interface with the existing OLTC transformer model available within MATLAB/Simulink, which itself has been adapted. The model is developed further as an FMU for wider use with other simulation tools.

The purpose of this specific model is to emulate the real hardware device (OLTC) controller connected to the transformer. The model was set up in the form of a masked subsystem in order to be more user-friendly. Thus, the user can specify the basic parameters of it through a simple interface. The IO ports of the subsystem are denoted with descriptive names of each input-output signal as well as the units of each quantity.

The most important feature of the model is the fact that the parameters are directly those given by the user and there is no need for adaptation or other calculations prior to using them. The selected model is described by the logical steps shown in Figure 30 and the parameters given in Table 1. The general layout of the model is given in Figure 31, showing the model inputs ( $U_m$ ,  $Pulses$ ) and outputs ( $Transformer Up$ ,  $Transformer Down$ ). The model consists of one function block, which represents the logical steps previously described.



```

IF input  $\notin$  [BLQ-BLQ+] THEN turn off OLTC control
IF ELSE
    IF Input  $\in$  [ACC-BLQ] During [Delay2] THEN move Motor Down fast
    IF Input  $\in$  [TOL-ACC] During [Delay1] THEN move Motor Down
    IF Input  $\in$  [TOL-TOL] THEN do not move
    IF Input  $\in$  [ACC-TOL] During [Delay1] THEN move Motor Up
    IF Input  $\in$  [BLQ-ACC] During [Delay2] THEN move Motor Up fast

```

Figure 30: Pseudo code for OLTC controller algorithm in TC2

Table 1: Parameters of the OLTC controller algorithm in TC2

Concept	Typical Value <sup>2</sup>	Unit	Function
SetP	420	V	Set point
Tap Step	2,5%	V	Tap Step
BLQ+	20%	V	Block Control
ACC+	10%	V	Step down (Delayed)
TOL+	+5%	V	Step Down
TOL-	- 5%	V	Step Up
ACC-	-10%	V	Step Up (Delayed)
BLQ-	-20%	V	Block Control
Delay	5s	s	Delay Variable
Tap Number	[-5, 5]	-	
Ts	0.0001	s	Sampling Time

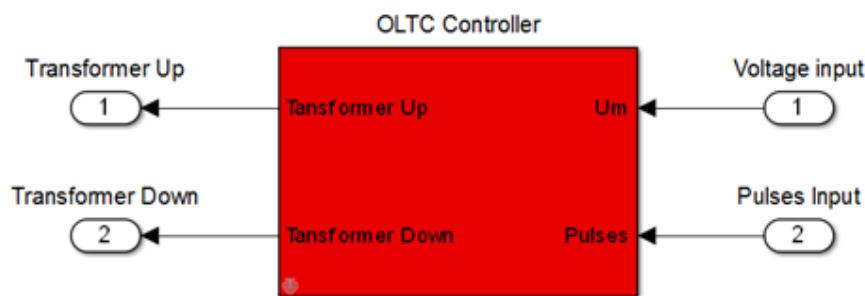


Figure 31: General layout of the implemented OLTC controller model

This model can be controlled either by changing one of the fixed parameters via the model's mask before each test, or by modifying the two input variables. Specifically, all constant parameters coming from the transformer end-user's requirements have to be defined and/or modified before the test. The model initialization functions are predefined inside the model initial mask.

The interconnection of the model to the outside world is done by the basic inputs – *Voltage* and *Pulses* – and the outputs – *Transformer Up* and *Transformer Down* commands.

<sup>2</sup> These values are not standardized, and are used only for this practical use case, they can be defined according to the end-user requirements.

The voltage input corresponds to the measurement of the transformer secondary side. While the Pulses input is a ready-state input coming from the driving mechanism of the OLTC in order to enable or disable the controller during the operation time of the hardware. The purpose of this whole simulation setup is the interconnection of two model outputs to the transformer block, in order to trigger the operation of the OLTC driving mechanism, see Figure 32.

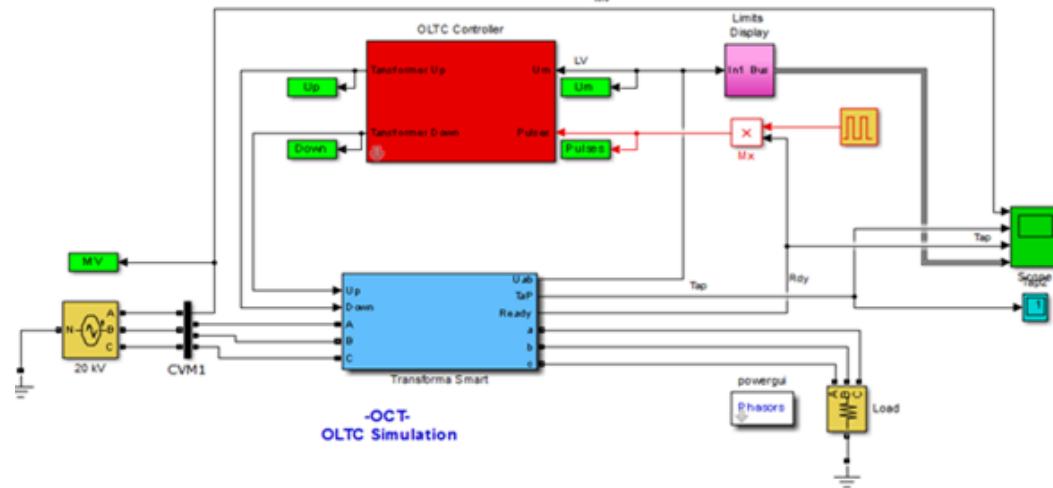


Figure 32: Interconnections between the model and the transformer

The set of the input parameters of the model in normal operation mode is shown in the menu diagram in Figure 33. The data used for the input parameters correspond to the typical value detailed in Table 1.

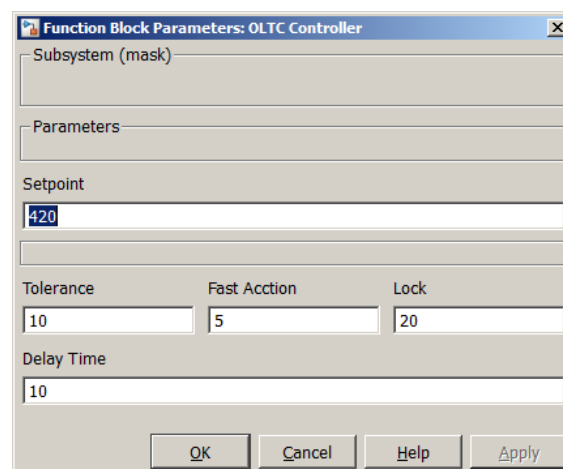


Figure 33: Snapshot of the parameters selection for normal operation.

### Model Validation and Simulation Results

For the simulation and validation of the model a number of tests were executed. These involved the variation of the voltage input during a certain amount of time to check the performance of the whole system according to the specification. Output voltage and transformer tap position were measured under the various conditions and compared to the expected results.

Figure 34 presents the results of a simulation of the model. It shows graphically the performance of the OLTC parameters during the experiment. The top row displays the medium voltage input and an induced voltage drop. As a consequence of the disturbance, the tap position (shown in the second row) is increased until the simulated low voltage reaches the destined voltage band again. Addition-

ally, one can see that the ready signal (shown in the third row) is de-asserted as soon as the transformer performs a switching operation. From these results, it is evident that the model follows the behaviour predicted by theory, that is, the voltage control performed by the OLTC when the network voltage drops. The correctness of the model is also validated by comparing the results of the simulation with the grid code requirements.

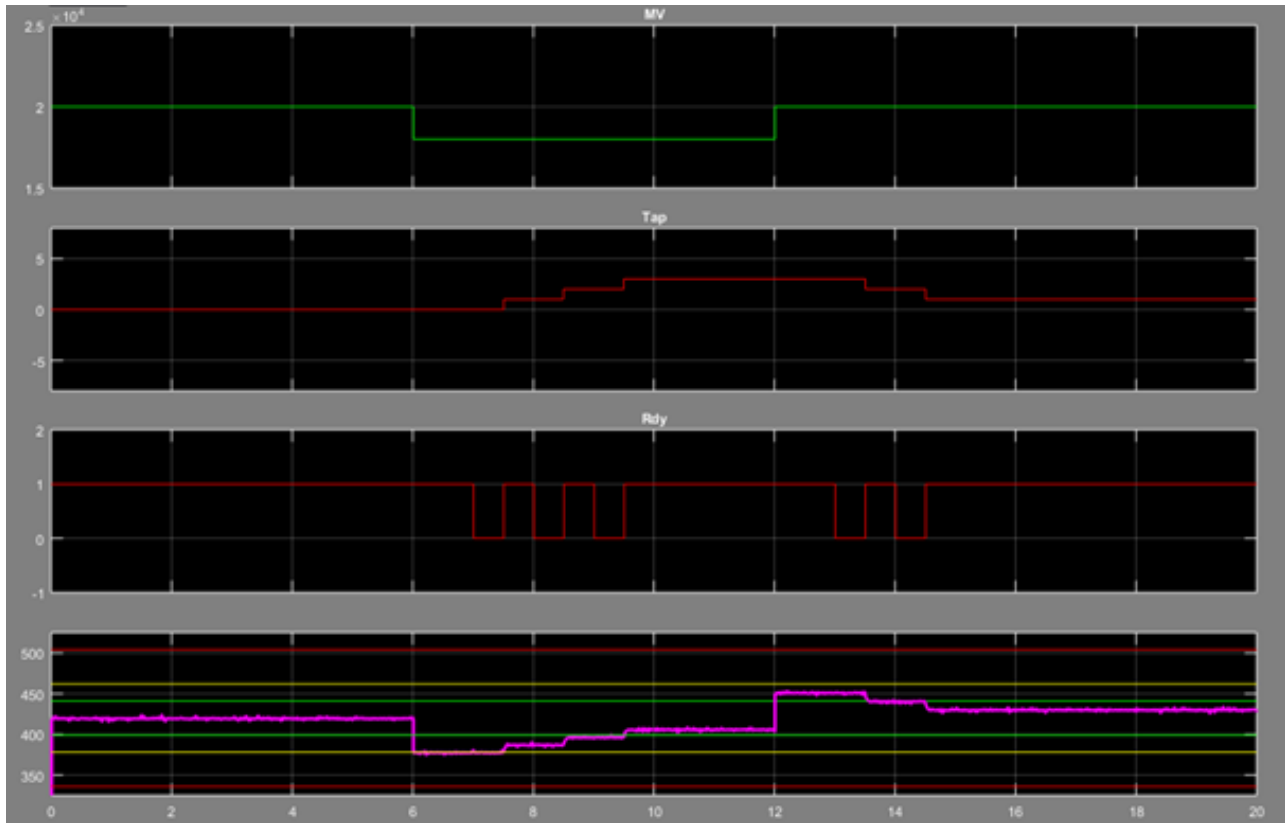


Figure 34: OLTC parameters during execution of experiment  
(first row: MV input, second row: tap position, third row: ready signal, fourth row: LV output)

#### 5.2.4 Experiment 2: Hardware OLTC Controller

In the second experiment, the controller model developed in the first experiment is used to automatically derive the embedded hardware implementation. The hardware controller is coupled via proprietary communication protocols with the plant model. In a consequent step, the whole setup is again tested by applying various voltage conditions. Results of this experiment have already been documented in Deliverable D-JRA 2.1.

#### 5.2.5 Experiment 3: OLTC Controller as FMU for Model Exchange

##### Test Case Specification and Implementation

Another rapid prototyping and controller validation method is presented in the third experiment, where the control logic is exported as an FMI-compatible model. The model is then instantiated as a virtual component, which can control a physical plant without the need of manually deriving a controller prototype. To thoughtfully validate the FMI-based virtual controller implementation without restricting tests to safe conditions only, it is decided to use a mock-up interface instead of a real transformer implementation. The mock-up interface allows to precisely emulate voltage readings in various over- and under-voltage conditions and outputs an analogue signal which is then sensed by the IO terminals of the controller. Figure 35 illustrates the experimental setup.

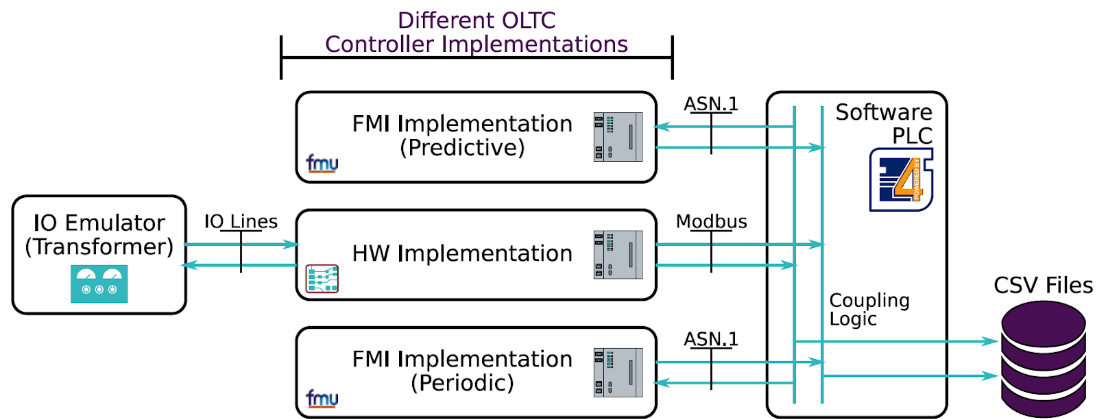


Figure 35: Experiment 3 setup [14]

The hardware controller implementation is used as Modbus IO terminal from which the voltage readings are collected and to provide another control output reference at the same time. To further study two synchronization approaches coupling virtual components, the experiment is extended and two virtual component configurations (predictive and periodic) are instantiated simultaneously. In an off-line simulation, the input values are applied to the references simulation and the outputs of all controller implementations are contrasted.

### Results and Discussion

The open loop experiment covers various voltage transitions, which result in 76 switching commands issued by each controller implementation [14]. In total, a time span of more than 450 s is covered. For each switching command, every controller implementation issues the same control action, but the timing of each action slightly varies. Figure 36 shows one command and the corresponding timing variations in detail. The first row plots the voltage readings obtained from the controller hardware implementation and the connected IO emulator. One can note that two values are given. An instant value reflects the voltage reading at the time the Modbus slave is polled and the average value corresponds to the filtered value, which is taken to calculate the control outputs. The second and third row illustrate the actual control outputs. Each rising edge initiates a single control action.

The timing of each control action throughout the entire experiment is compared to the timing of the reference controller. Table 2 lists the delay statistics of every configuration. One may note that the hardware controller issues its control outputs up to 516 ms before the reference simulation because the controller does not rely on a network connection to gather the input readings. On the one hand, due to the limited resources of the controller, the nominal polling cycle of 200 ms is delayed and a corresponding action delay of -516 ms is observed. Positive extreme hardware delays of 169 ms, on the other hand, can be tracked down to scheduling latencies within the software programmable logical controller (PLC) instance which distributes and records all signals.

Since all voltage readings are distributed to the virtual controller instances after they are recorded by the software PLC, no negative delay value of a virtual component is observed. The maximum delay of nearly 500 ms can again be tracked down to scheduling latencies of the software PLC. On removing the single outlier, a maximum delay of 169 ms for both virtual component configurations is observed. These delays directly correspond to the expectations from the internal real-time tracking of the interface application.

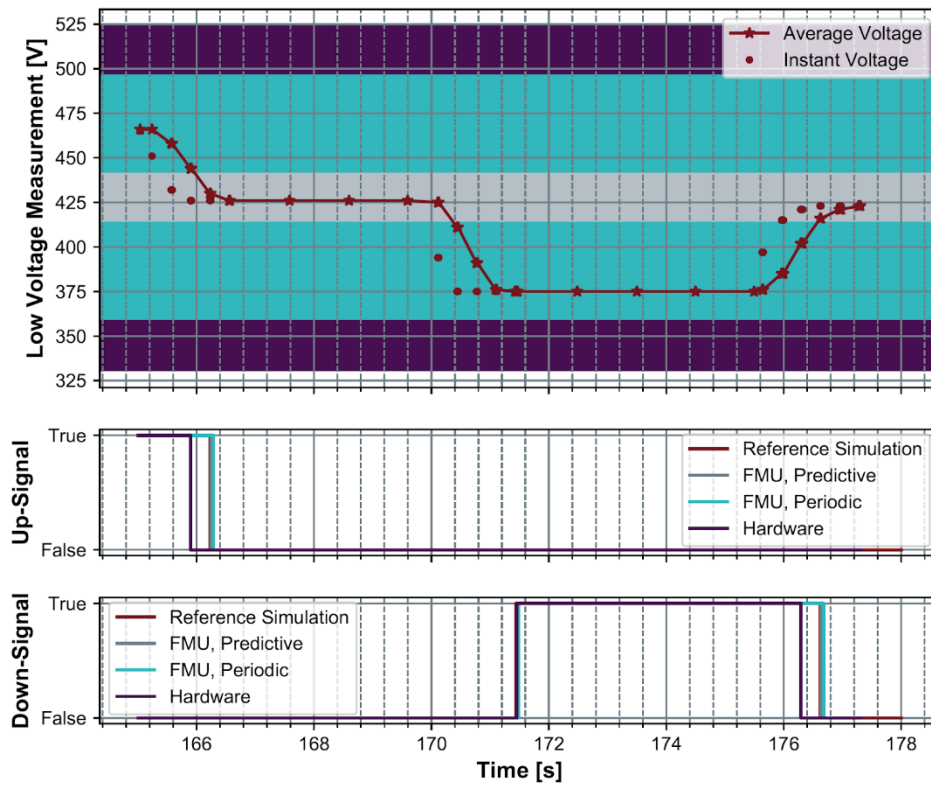


Figure 36: Switching operation in experiment 3 [14]

Table 2: Control action delay statistics of experiment 3 [14]

Configuration	Signal	Samples	Delay Statistics				
			Median [s]	Mean [s]	Variance [s <sup>2</sup> ]	Min. [s]	Max. [s]
Predictive	Down	46	0.03	0.0246	0.000162	0	0.084
	Up	30	0.016	0.0428	0.007932	0	0.498
Periodic	Down	46	0.062	0.0567	0.0002295	0.03	0.094
	Up	30	0.062	0.0781	0.006728	0.031	0.498
Hardware	Down	46	-0.327	-0.264	0.02085	-0.516	0.015
	Up	30	-0.327	-0.236	0.02868	-0.515	0.169
Predictive	Both	76	0.023	0.0318	0.003307	0	0.498
Periodic	Both	76	0.062	0.0651	0.002904	0.03	0.498
Hardware	Both	76	-0.327	-0.253	0.02413	-0.516	0.169

## 5.2.6 Experiment 4: FMI-based OLTC Controller Hardware-in-the-Loop

### Test Case Specification and Implementation

The fourth experiment as illustrated in Figure 37 explores the capabilities of using FMI-based virtual components to test a controller implementation in a controller-hardware-in-the-loop setup. Therefore, the transformer model is exported as FMU and coupled with an embedded hardware controller implementation. To set the voltage readings from the virtual transformer and to query the control outputs, the controller is directly accessed via a Modbus TCP/IP connection.

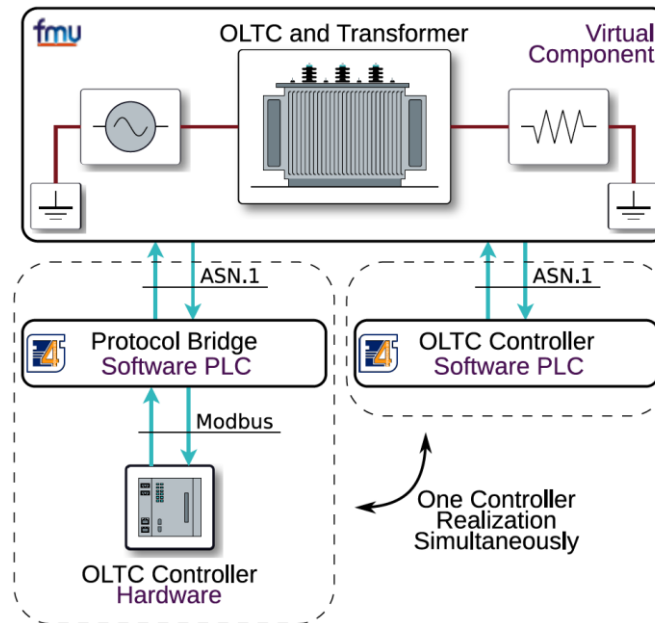


Figure 37: Experiment 4 setup [14]

In contrast to the first version of the experiment design, inaccuracies are avoided by using the Modbus interface of the controller instead of its analogue IO lines. Hence, no additional digital-to-analogue and analogue-to-digital conversion is needed between the controller and the virtual components. To further study the timing properties of the virtual components, the experiment was extended by implementing a functionally equivalent, event-based controller based on a software PLC. Since the PLC-based controller can directly act on changed conditions without awaiting the next polling cycle, the effects of a virtual component implementation can be observed. Again, multiple virtual component configurations are tested and compared to the outcome of the monolithic reference simulation conducted in the first experiment.

### Results and Discussion

Three experiment runs in different controller configurations and one reference simulation is performed in the fourth experiment. Each experiment run covers multiple induced medium voltage transitions to assess the capabilities of the connected controllers. Figure 38 shows one medium voltage transition and the corresponding control actions of all closed-loop experiments in detail. The first row indicates the position of the induced medium voltage deviation, which is created within the model. The second row plots the resulting low voltage reading from the transformer, which needs to be brought back to the normal inner voltage level by the connected controller. The low voltage plot is followed by two lines, which show the control signals of every experiment run. In the last row, the resulting tap position as given by the transformer model is visualized.

One may note the different pulse widths of the control action signals, which directly originate from processing, synchronization and communication delays. A status signal of the transformer is fed back to the controller, which clears the control signal as soon as the status changes. In the reference simulation, the resulting pulse of exactly one integrator step is formed by a delay element, which breaks an algebraic loop. The first rising control signal edge of the predictive PLC configuration is delayed by 21 ms. The transformer model changes the status signal as soon as it receives the control signal, but since the previous event is still processed, the status signal is distributed late. Consequently, delays accumulate to 193 ms until the control signal is cleared. Since subsequent transformer status changes are independent of the falling control edges, only communication delays of the rising control edges accumulate in subsequent tap switching operations. In the hardware controller configuration, Modbus polling cycles of 200 ms additionally contribute to the observed delay.

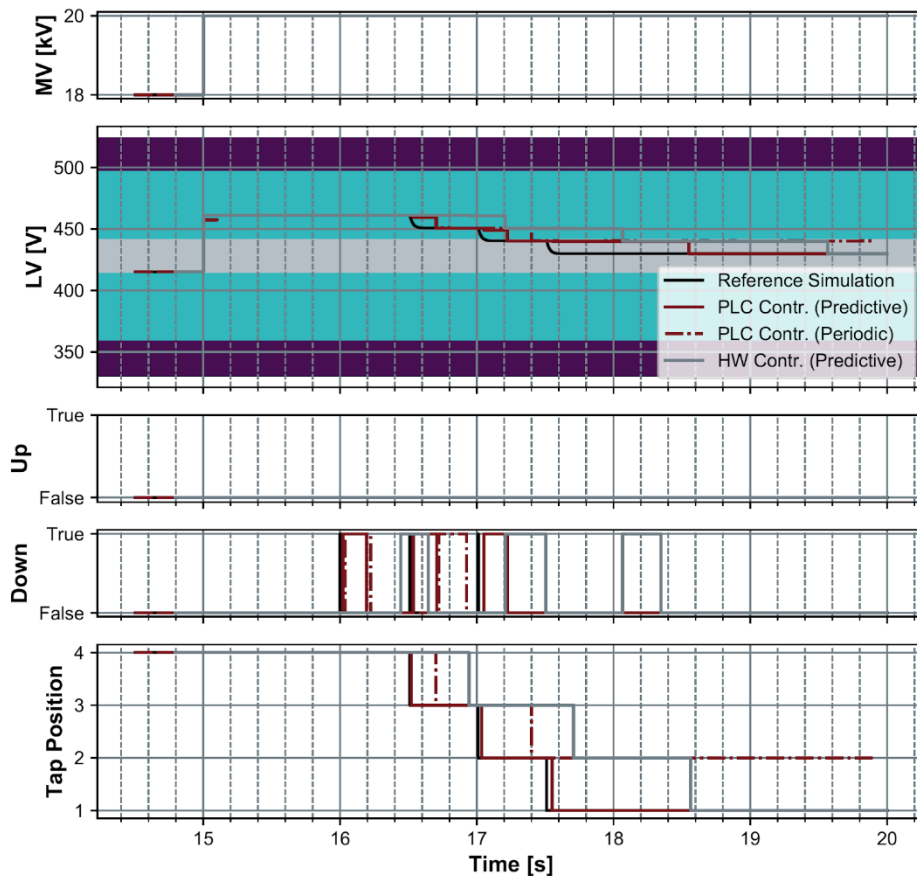


Figure 38: Control operation in experiment 4 [14]

One may note that for the periodic configuration, similar pulse durations to the predictive configuration can be observed, but one switching action is omitted. Since the transformer model also covers various dynamic effects, a non-instantaneous low voltage transition at each switching operation is observed. In case of the predictive configurations, the low voltage reading can be sampled at the control event just before the transition. Hence, the same switching operations as in the reference simulation are observed. For periodic synchronization, the voltage reading at the switching event is delayed until the next synchronization point. At that instant in simulation time, the transient reaches its steady state and the last switching action is omitted because the low voltage reading is already in the destined voltage band.

### 5.2.7 Combined Results of Test Case 2

In experiment 1 to 4, the use of co-simulation and HIL setups at various processing stages was demonstrated. In experiment 1, a common reference of the OLTC transformer setup and the controller was established. The second experiment successfully demonstrated state-of-the-art co-simulation, which includes an embedded controller and the plant model (see deliverable D-JRA2.1). Nevertheless, several difficulties in extending the automatically generated controller implementation for the subsequent experiments were found.

To enable a communication via established protocols for automation systems, the controller is rewritten and used in experiment 3 and 4. In both experiments, which deploy industrial communication protocols, the effects of increased delay and the need of polling variables is observed. It is demonstrated that such effects influence the timing of the open- and closed-loop control and consequently the practical implementation. Additionally, it is shown that limited communication bandwidth, which



prevents massive oversampling, requires special attention in coupling physical and virtual components. Particularly, work conducted in experiment 4 demonstrates that the coupling approach significantly influences achieved results.

Furthermore, the feasibility of FMI-based co-simulation that also includes coupled hardware is demonstrated. Although efficient proprietary solutions exist, they showed its limits in including industrial communication protocols. The FMI-based virtual component allows to quickly interface with automation systems at the cost of maintaining a separate interface program which is not tightly integrated into the modelling environment.

### 5.3 Signal-based Synchronization between Simulators (TC3)

#### 5.3.1 Overview

This test case deals with the impact of ICT-related aspects 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, see Figure 39 for an overview. ICT-related aspects of interest are technical features (communication delays, controller dead times) and consequences of cyber-security attacks (scaling attacks, ramping attacks, random scaling factor attacks, digital signal tap attacks).

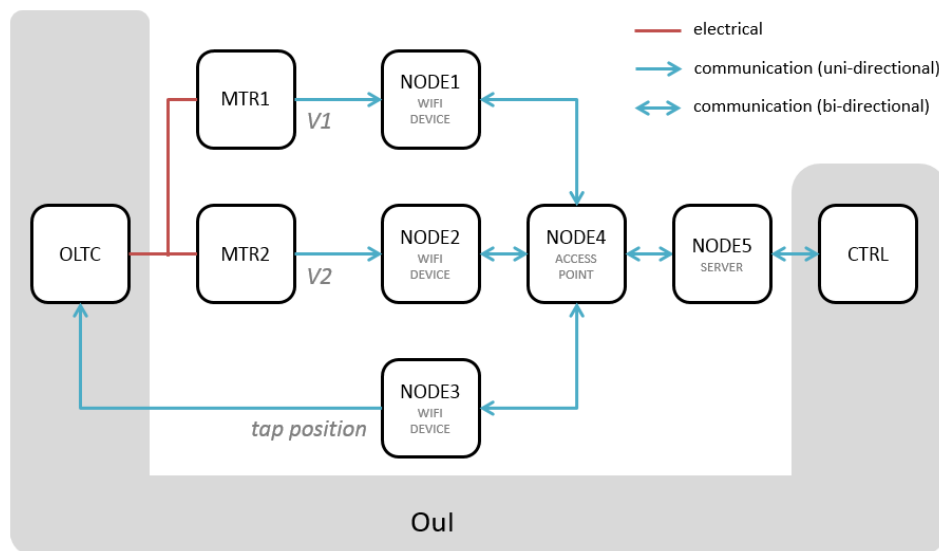


Figure 39: Overview of the Oul of test case TC3

The aim of this test case is to demonstrate and assess the effect of communication networks on the actuation pattern of the CVC and the resulting physical effects in the low voltage distribution system. Figure 40 shows the details of the simple algorithm the controller implements. 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 delays is implemented for the controller. Similarly, it is assumed that cyber-security measures are insufficient (e.g., due to the usage of UDP or ARP), opening the possibility to implement cyber-attacks.

Moreover, this test case aims to demonstrate the importance of realistic simulation approaches for assessing distributed and centralized Smart Grid control algorithms as well as benchmarking communication network technologies and topologies for use in Smart Grid applications.



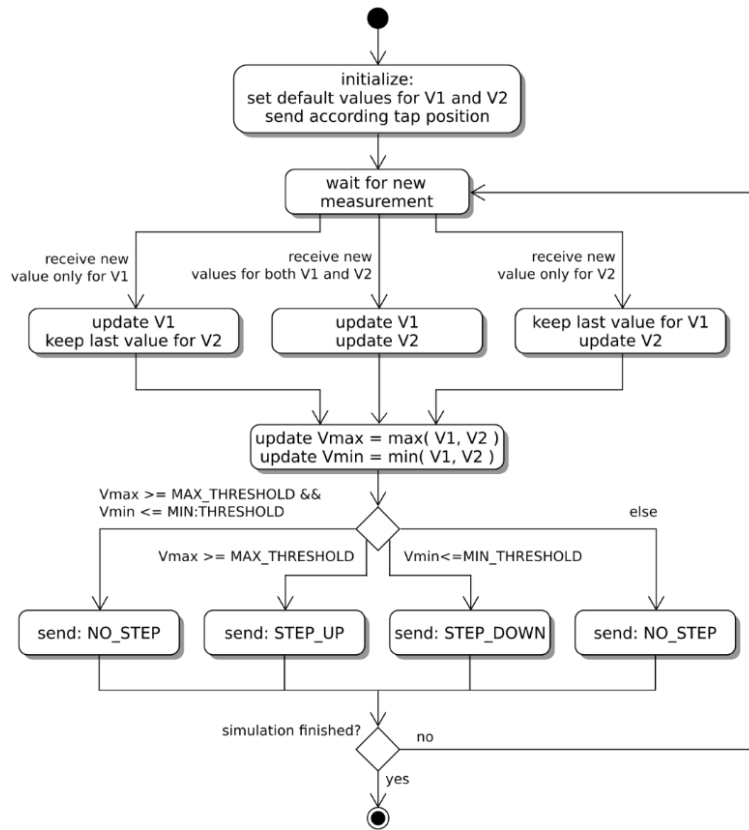


Figure 40: UML state chart of the CVC algorithm for test case TC3

### 5.3.2 System under Test

The SuT was chosen to be as simple as possible, in order to focus on the ICT-related effects rather than aspects related to classical power system engineering or control engineering:

- **CVC controller:** A simple rule-based control algorithm has been chosen for the CVC, see Figure 40. It calculates tap position setpoints for the OLTC transformer depending on the meter readings and runs on a dedicated server, separate from transformer and meters. When receiving a new measurement and computing a new tap position setpoint, the controller becomes unresponsive for a short time period (dead-time), i.e., further measurements arriving during this time are not processed.
- **Low voltage distribution system:** The electrical system comprises an OLTC MV/LV transformer connected to the external grid and two loads with voltage meters, which send their measurements at regular intervals to the controller. Changing the OLTC's tap position is assumed to take 3 s in total, during which the OLTC is not responsive to new setpoints. See Figure 41 for a schematic overview.
- **Communication network:** A simple topology has been chosen for the communication network. It comprises, two smart meters and one transformer connected via Wi-Fi to an access point, which is connected via Ethernet to the server, see Figure 42. The IEEE 802.11ac (Wi-Fi, 5 GHz) protocol along with the Ethernet protocol (100 mbps) were used.

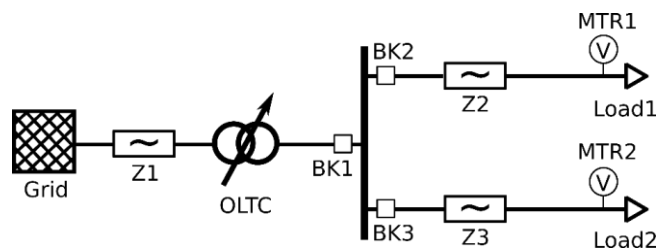


Figure 41: Schematic of the low voltage distribution system used in test case TC3

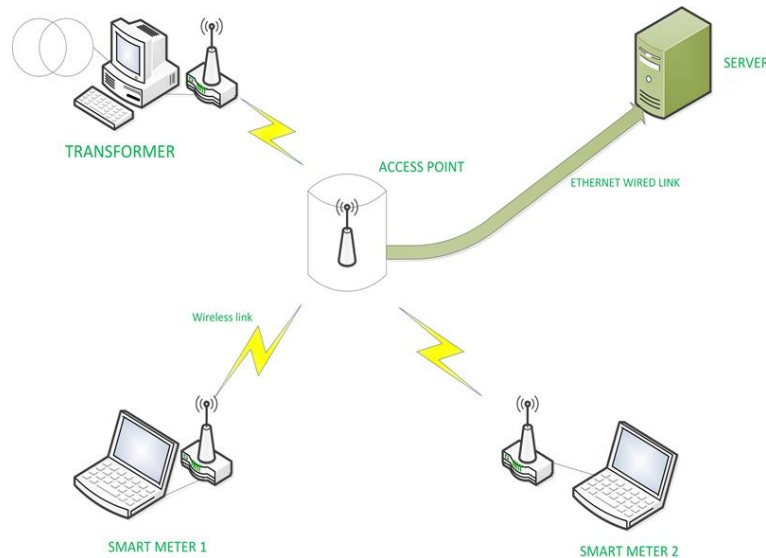


Figure 42: Communication network topology for test case TC3

### 5.3.3 Validation of Network Communication Models

#### Test Case Specification and Implementation

A standalone simulation experiment including only the communication network has been carried out in order to test and validate the ns-3 models developed for test case TC3 (see Section 4.3.1). The system was simulated for a full day, with the meters sending measurements to the controller and the controller sending tap position setpoints to the OLTC transformer:

- The meters send every 15 minutes a packet of fixed size (100 bytes) containing a voltage measurement to the access point (Wi-Fi protocol). Subsequently, the data is sent to the server (Ethernet protocol), where the CVC algorithm is implemented.
- In the next minute after the packets are received, the controller sends a tap position setpoint to the OLTC transformer, which is also connected to the access point (Wi-Fi).

Please note that the choice in this test case of using Wi-Fi networks (particularly the use of ARP for resolving IP-MAC addresses) to connect to the meters is not meant as a recommendation of this technology for similar real-world systems. It was rather a convenient choice for the purpose of modelling an ICT network, resulting in simple yet realistic models for the proof-of-concept validation in test case TC3.

#### Results

The traffic of the topology and the transmission stages during a sending event from the meters are represented using the *pyViz* tool [18] in Figure 43.

When data is transmitted on a Wi-Fi network, it is transmitted to every station within the transmission range of the broadcasting station. Also, because there is no way to detect packet collisions in the IEEE 802.11 protocol, the CSMA/CA protocol (carrier-sense multiple access with collision avoidance) is used. In this protocol, after the receiver receives the corresponding packet, it transmits an acknowledgement frame to notify the sender that the packet was successfully transmitted. Therefore, when the meters send a packet, the signal is transmitted to every station within the Wi-Fi network, as can be seen in Figure 43 (first and second stage). However, it is captured only by the target receiver (access point), which then sends the acknowledgement frame back to the sending station.

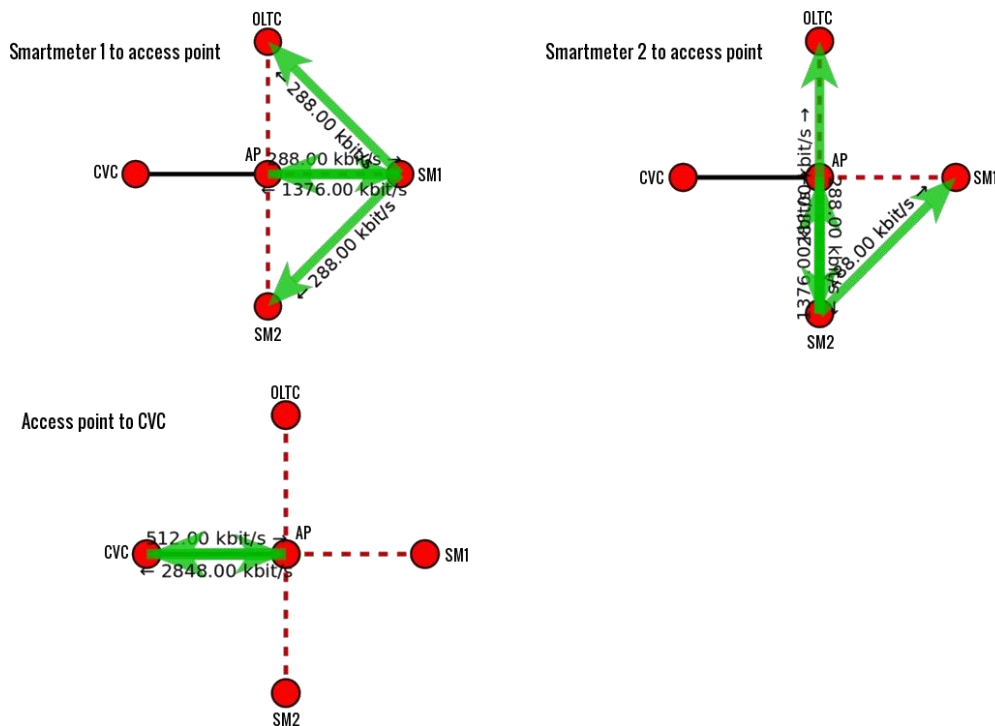


Figure 43: Transmission stages for the meter data, showing the first stage (upper left), the second stage (upper right) and the third stage (lower left)

On the other hand, in the CSMA network (Ethernet) the stations are able to detect collisions directly through the signal power (CSMA/CD protocol). Because of this, no acknowledgement is required to secure data integrity. When a station notices that a collision has occurred, it will send a pattern to inform all the network stations about the event, halting all transmissions. Because the access point tries to send two packets at the same time, the CVC controller detects a collision in the network and will therefore send a collision detection pattern to the access point, as can be seen in Figure 43 (third stage).

Similarly, without the packet collision in the CSMA network, the traffic of the topology and the transmission stages for the controller sending event are represented using the pyViz tool in Figure 44. The simulated end-to-end delays are shown in Figure 45.

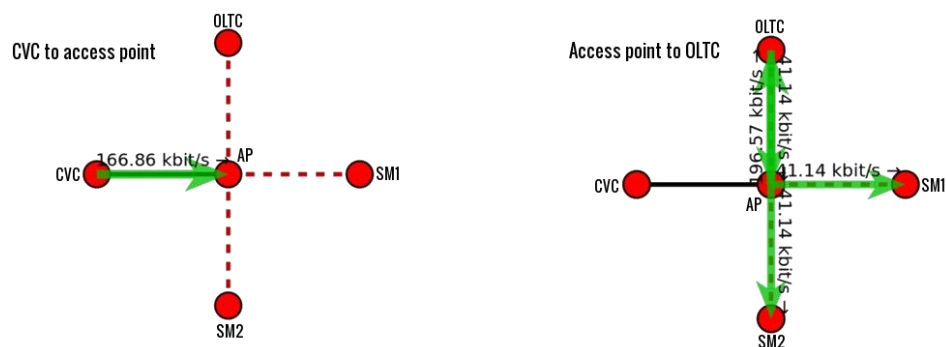


Figure 44: Transmission stages for the controller

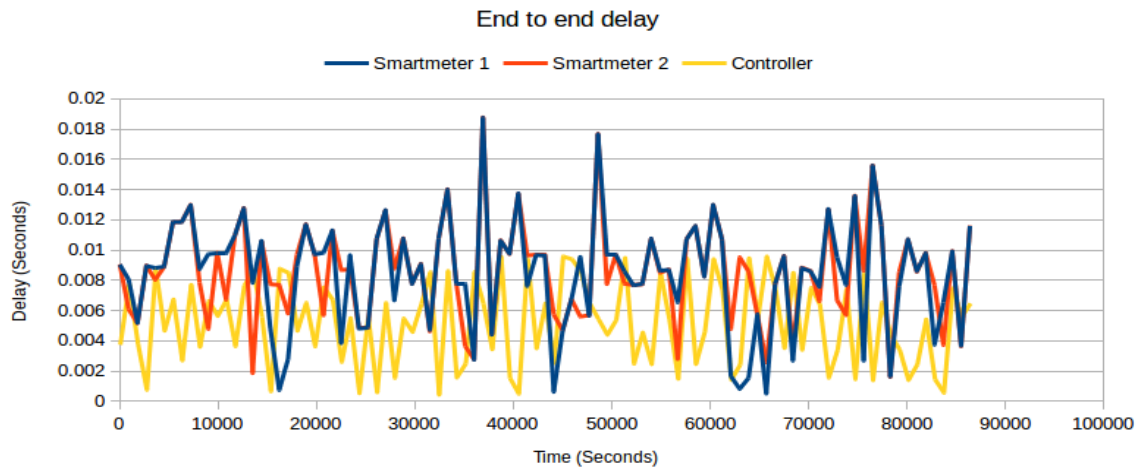


Figure 45: Simulated end-to-end delays over the course of one full day

### 5.3.4 Technical Assessment

#### Test Case Specification

This test assesses the impact of communication delays and controller dead times on the actuation pattern of the OLTC. The impact is evaluated by comparing the final realized tap position with the expected tap position (from a baseline simulation).

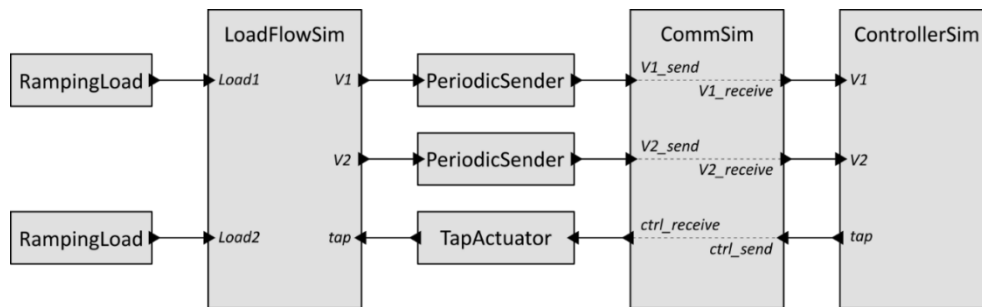


Figure 46: Overview of the simulation components for the technical assessment in test case TC3

The test takes 2 minutes, during which both loads are linearly ramped up. The meters send their measurements to the controller in regular intervals ( $T_{\text{sender}} = 1 \text{ min}$ ) with a small delay ( $\Delta t_{\text{sender}}$ ), beginning at the start of the simulation ( $t = 0 \text{ min}$ ). Whenever the controller receives new measurements, a new value for the tap position is calculated and sent to the OLTC transformer, see Figure 46.

The transmission of data from the meters to the controller (voltage measurements) and from the controller to the OLTC's tap actuator (tap position setpoint) happens with a delay, calculated by the communication network simulator. After receiving a new voltage measurement, the controller enters its dead time ( $\delta t_{\text{ctrl}}$ ) and becomes unresponsive. Similarly, after receiving a new tap position setpoint, the tap actuator enters its dead time and becomes unresponsive.

Voltage V1 (voltage measured at Load1) is expected to stay within the operational limits throughout the test. Voltage V2 (voltage measured at Load2) falls beyond the lower threshold within 1 minute, and a change in the tap position (from 0 to -1) is expected to happen the next time the meters transmit their measurements to the controller ( $t = 1 \text{ min}$ ).

### Experiment Specification

The experiment is implemented as co-simulation using mosaik with a constant simulation step size of 2 ms (compare with Section 4.1.3). Dedicated simulation components are implemented as FMUs for CS:

- *Power system simulation*: The power system is implemented as PowerFactory model, using consecutive power flow calculations to simulate the power system.
- *Controller*: The algorithm for calculating the tap position setpoint is implemented as a (simple) MATLAB script.
- *Communication network simulation*: The communication network delays are simulated with the help of ns-3.

The other simulation components and the FMI-compliant adapters are implemented in Python on top of mosaik's high level API. Apart from the FMI-compliant wrappers, this includes the following components:

- *Ramping load*: Ramps the loads linearly up or down.
- *Periodic sender*: Periodically sends the up-to-date voltage measurement. The individual senders can be configured to send with (small) delays relative to each other.
- *Tap actuator*: Actuates a new setpoint from the controller at the OLTC. When actuating a new tap position, it is not responsive for the next 3 seconds (dead time).

The use of PowerFactory requires this simulation to use Windows as operating system. However, since ns-3 is developed for Linux operating systems, it is run in a Cygwin environment (compare with Section 3.1.3). Figure 47 gives an overview of the implemented setup.

The communication network model uses message IDs as inputs and outputs, with a message ID equal to 0 indicating that no signal is present (compare with Section 4.1.3 and Section 4.3.1). 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. However, 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). Therefore, the mosaik wrapper for the ns-3 FMU implements a mapping between message IDs and signal values.

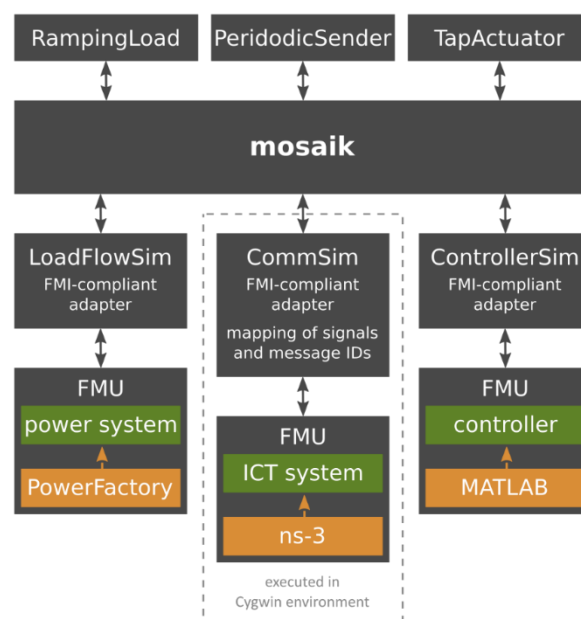


Figure 47: Implementation of the technical assessment of test case TC3 in the mosaik environment

## Results

This is an advanced co-simulation setup using FMUs and communication delays. In order to incorporate the “randomness” of the communication delays, an ensemble of 100 simulation runs with different random number generator seeds has been evaluated for every considered combination of  $\Delta t_{\text{sender}}$  and  $\delta t_{\text{ctrl}}$  (Monte Carlo approach). All simulations are compared to an “ideal” reference case, where no relative delay between the periodic senders, no communication delays and no controller dead time are present. The test specification is such that voltage V2 falls below the lower voltage threshold within the first minute. This should trigger a response from voltage controller, changing the transformer’s tap position from 0 to -1, see Figure 48b.

However, in a more realistic simulation setup that takes into account the relative delay between the periodic senders, the communication delays and the controller dead time, the final tap position can be different from -1. In fact, there are two other possible outcomes:

- *Final tap position 0:* Voltage measurement V3 may arrive at the controller before measurement V4, either because of the delay between the senders and/or the communication delay. This may cause measurement V4 to arrive at the controller when it is unresponsive (dead time) and/or it arrives too late at tap actuator (3 s dead time), see Figure 48a.
- *Final tap position -2:* In case measurement V4 arrives at the controller before V3, the controller calculates a new tap position setpoint as -1. Furthermore, if measurement V4 arrives after the controller’s dead time, then the undervoltage is still not resolved, such that the controller is again triggered and calculates yet another tap position setpoint as -2. Subsequently, if the second tap position setpoint arrives at the tap actuator first (due to the random communication delay), it will get actuated (whereas the first tap position setpoint will arrive during the tap actuator’s dead time), see Figure 48c.

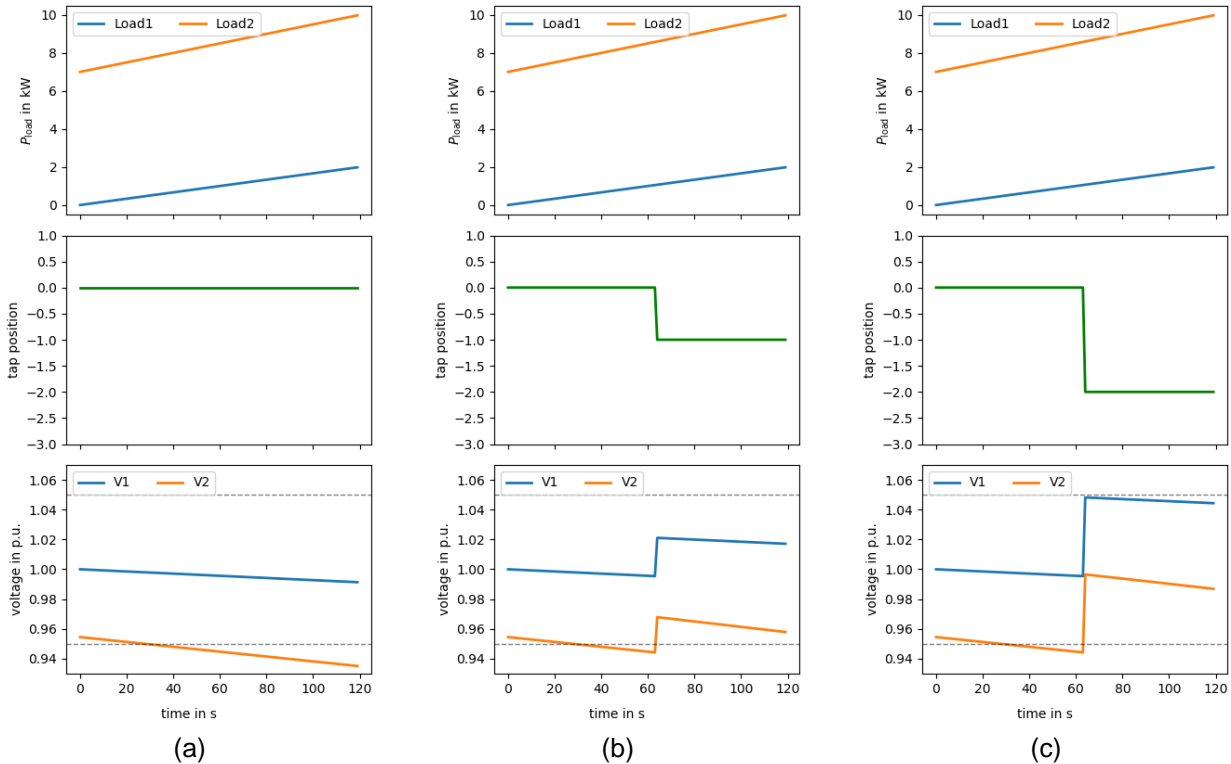


Figure 48: Possible outcomes of the simulation experiments for test case TC3 with  
(a) final tap position 0, (b) final tap position -1 and (c) final tap position -2

Figure 49 gives an overview of the results from the Monte Carlo simulations, showing the observed relative frequency of final tap positions depending on the relative delay between the senders  $\Delta t_{\text{sender}}$  and the controller dead time  $\delta t_{\text{ctrl}}$ . This shows that for the chosen values of  $\Delta t_{\text{sender}} \in \{-14 \text{ ms}, -12 \text{ ms}, \dots, 12 \text{ ms}, 14 \text{ ms}\}$  and  $\delta t_{\text{ctrl}} \in \{2 \text{ ms}, 10 \text{ ms}, 20 \text{ ms}\}$ , the final tap position is basically non-deterministic and can with a certain probability lead to unexpected results.

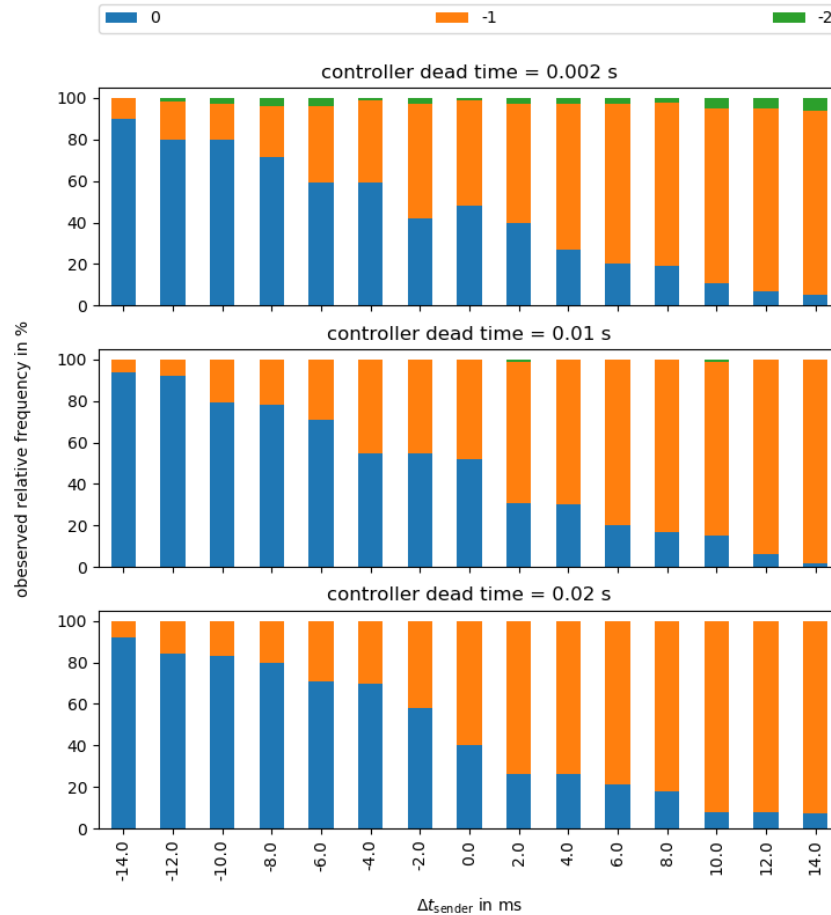


Figure 49: Relative observed frequency of final tap positions (blue: 0, orange: -1, green: -2) for different values of  $\Delta t_{\text{sender}}$  and controller dead times

### 5.3.5 Cyber-Security Assessment

#### Test Case Specification and Implementation

The cyber-security consequence assessment has been carried out using the setup described in Section 4.3.2 and shown in Figure 9. The assessments of all types of cyber-attack consequences are run for several hours (simulation time), with the meters sending their measurements to the controller in regular intervals ( $T_{\text{sender}} = 1 \text{ min}$ ) in perfect synchronization ( $\Delta t_{\text{sender}} = 0 \text{ s}$ ), beginning at the start of the simulation ( $t = 0 \text{ min}$ ). In contrast to the technical assessment above, the voltage controller only sends new tap position setpoints every 15 minutes.

The following attack patterns are applied:

- Scaling attack:** Voltage measurements V1 and V2 sent to the controller are modified by the ContinuousSignalAttacker component, representing an influence of a cyber-attacker in the system. Both voltage measurements are scaled negatively with a scaling factor of  $\lambda_s = 0.01$ , i.e., the scaling is  $y(t) * (1 + \lambda_s)$ .

- b) *Ramping attack*: Voltage measurements V1 and V2 sent to the controller are modified by the ContinuousSignalAttacker component, representing an influence of a cyber-attacker in the system. Both voltage measurements are modified according to the ramped signal attack pattern, i.e.,  $y(t) + \lambda_r * t$ , where  $t$  is the simulation time and  $\lambda_r$  is the ramping factor.
- c) *Random attack*: Voltage measurements V1 and V2 sent to the controller are modified by the ContinuousSignalAttacker component, representing an influence of a cyber-attacker in the system. Both voltage measurements are modified according to the random signal attack pattern, i.e.,  $y(t) + \text{rand}(a,b)$ , where  $t$  is simulation time and  $a$  and  $b$  are lower and upper bounds of random numbers.
- d) *Digital signal tap attack*: Voltage measurements V1 and V2 sent to the controller remain unmodified. Rather the tap position setpoints sent from the controller are modified by the DiscreteSignalAttacker component, representing an influence of a cyber-attacker in the system.

Figure 50 shows the implementation of this setup in the mosaik environment. In contrast to the technical assessment, only the communication network simulator uses an FMU for CS.

### Results

Figure 51 shows results from applying the attack patterns described above, depicting both the normal system operation (solid lines) and system operation under attack (dashed lines). In all cases, the impact of the attacks is clearly discernible, with the control actions causing significant violations of the operating conditions. The assessment of the consequences could further be used as input to risk assessment process (see Section 3.3.3), where the potential security threats causing such consequences could be determined and appropriate countermeasures could be developed. Identifying consequences via simulation is particularly beneficial for large scale Smart Grid use-cases, in which it might not be obvious how a system is affected, due to its complexity in terms of number of components and interactions among them.

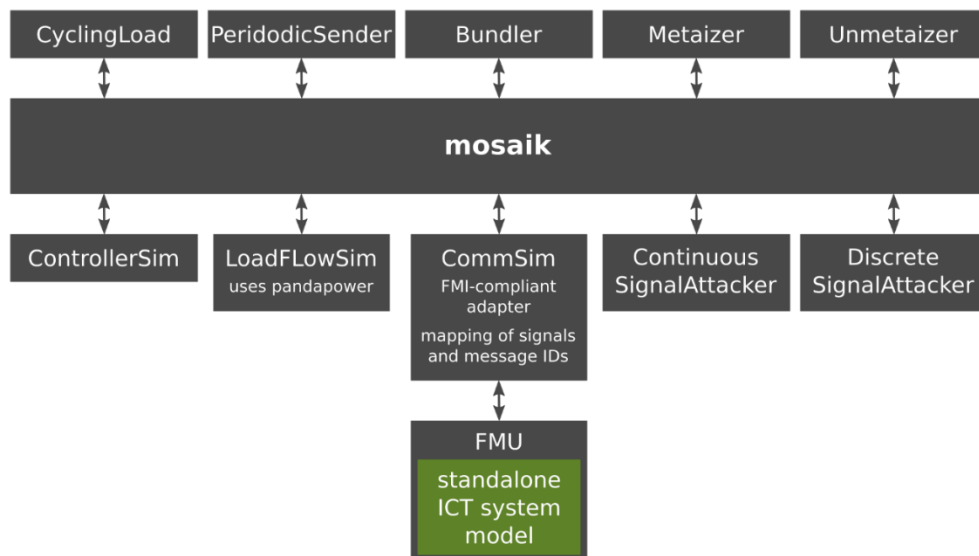


Figure 50: Implementation of the cyber-security assessment of test case TC3 in the mosaik environment



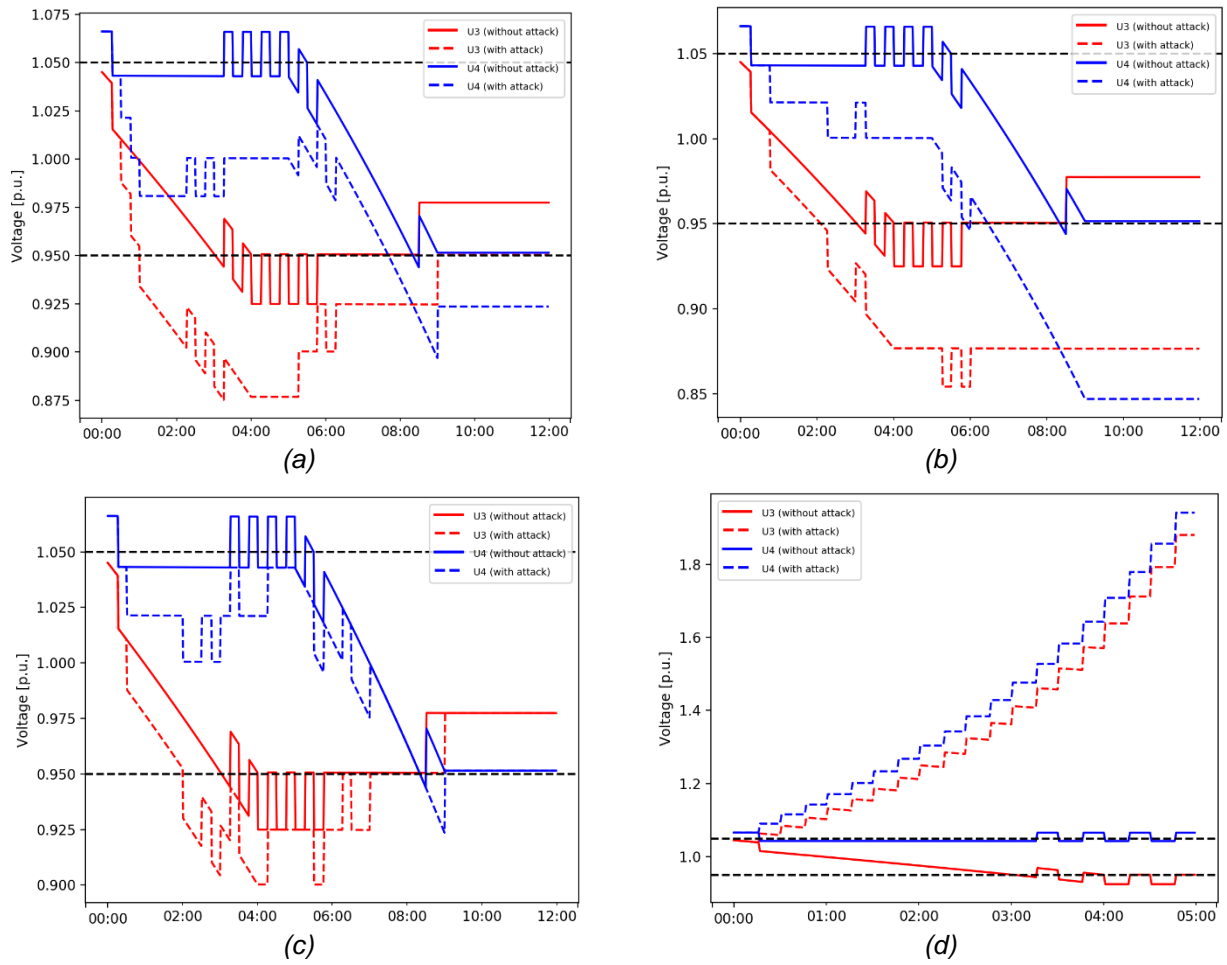


Figure 51: Results from the cyber-security assessments for  
 (a) scaling attacks, (b) ramping attacks, (c) random attacks and (d) digital signal tap attacks

## 6 Conclusions and Outlook

Work package JRA2 has been successful in demonstrating the feasibility of an FMI-based co-simulation approach for the Smart Grid ICT assessments. To this end, easy-to-deploy software packages have been extended and/or developed, which can cover the required simulation needs, i.e., system scalability, model heterogeneity, and simulation performance. The focus was on integrating existing diverse solutions instead of developing a new monolithic special-purpose tool. Furthermore, it is demonstrated how these tools and methods can be applied to assess ICT-enabled Smart Grid applications, by applying them to three relevant test cases.

In future work, the mosaik co-simulation framework is to be extended by a new scheduling module that is fully capable of supporting discrete-event simulation. The Discrete Event Simulation (DES) paradigm is the basis for most ICT models, especially communication networks. While it is possible to integrate such models into a co-simulation that is scheduled on a discrete-time basis, this will typically require very small-time steps (i.e. a high sampling frequency) which leads to a loss in performance. Therefore, providing DES scheduling capabilities will likely increase the performance and scalability of application cases that involve communication simulation, like TC3. The same is true for the simulation of some socially determined dynamics like market-based trading or energy consumer behaviour. DES can be integrated into co-simulation by establishing a central event queue that is managed by the co-simulation master. Every data exchange between simulators is then registered as an event with a given timestamp and inserted in the queue. The co-simulation process is organized by resolving the events in the queue, starting with the lowest timestamp. Each execution of a simulator may lead to the creation of more events that have to be sorted into the queue based on their timestamps. In the case of equal time stamps the co-simulation master needs a way to prioritize events. This can be realized via topological sorting based on the data dependencies between simulators. Furthermore, DES co-simulation needs a way of replacing obsolete events with newer ones. Finally, proper support of communication simulation requires the possibility of simulators to resend data (events) in the case of packet loss. Once these features are implemented into mosaik, the framework will enable users to do much broader research in the context of ICT and socio-technical application cases.

Another important topic for future work is the concept of *simulation as a service* that has gained increasing attention in the recent years, especially in the application of (co-)simulation in complex research fields like cyber-physical energy systems. The basic idea is to provide users with (typically web-based) access to simulation processes instead of having them install the full simulation software. This concept has several advantages. On the one hand, the CPU time for running the simulation is also part of the services, which makes high-performance simulation more accessible to institutions that do not possess the necessary hardware. On the other hand, vendors of specialized simulation software gain an additional stream of revenue. Researchers may not have enough application cases to purchase the full software, but still may be willing to pay to use it a limited number of times. At the same time, company secrets are protected since users will only have access to an interface and not to the code itself. Co-simulation provides an especially good fit with simulation as a service due to its modular layout. Web-based co-simulation environments could provide the framework were several simulation users and model providers come together. This would enable researchers to study complex scenarios with detailed models on powerful hardware, without the requirement to pay for the long-term use of all of these components. They may even be able to integrate and test their own simulation software in these setups or extend them by hardware laboratories for hardware-software co-simulation.

## 7 References

- [1] W.-T. Chang, S. Ha and E. A. Lee, "Heterogeneous Simulation - Mixing Discrete-Event Models with Dataflow," *J. VLSI Signal Process. Syst.*, vol. 15, no. 1-2, pp. 127-144, 1997.
- [2] V. Kesaraju and F. Ciarallo, "Integrated simulation combining process-driven and event-driven models," *J Simulation*, 2012.
- [3] M. Blank, S. Lehnhoff, K. Heussen, D. E. Morales Nondy, C. Moyo and T. Strasser, "Towards a foundation for holistic power system validation and testing," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2016.
- [4] International Organization for Standardization, "ISO 31000 - Risk management," [Online]. Available: <https://www.iso.org/iso-31000-risk-management.html>. [Accessed 12. 10. 2018].
- [5] Standard New Zealand, "Risk management," [Online]. Available: <https://www.standards.govt.nz/search-and-buy-standards/standards-information/risk-managment/>. [Accessed 12. 10. 2018].
- [6] Austrian Standards, "Serie ONR 49000," [Online]. Available: <https://www.austrian-standards.at>. [Accessed 12. 10. 2018].
- [7] SPARKS - Smart Grid Protection Against Cyber-Attacks, [Online]. Available: <https://project-sparks.eu/>. [Accessed 26. 09. 2018].
- [8] National Institute of Standards and Technology, "Introduction to NISTIR 7628," [Online]. Available: <https://www.nist.gov/document-13017>. [Accessed 12. 10. 2018].
- [9] The FMI++ Python Interface, [Online]. Available: <https://pypi.org/project/fmipp/>. [Accessed 26. 09. 2018].
- [10] FMI++ Library, [Online]. Available: <http://fmipp.sourceforge.net/>. [Accessed 31. 10. 2017].
- [11] DlgSILENT PowerFactory, [Online]. Available: <http://www.digsilent.de/>. [Accessed 12. 10. 2018].
- [12] "The FMI++ PowerFactory FMU Export Utility," [Online]. Available: <http://powerfactory-fmu.sourceforge.net>.
- [13] E. Widl and W. Müller, "Generic FMI-compliant Simulation Tool Coupling," in *Proceedings of the 12th International Modelica Conference*, 2017.
- [14] M. H. Spiegel, Linking Simulation and Automation Infrastructure - A Study Based on the FMI and IEC 61499 (Diplomarbeit), TU Wien, 2018.
- [15] FMITerminalBlock, [Online]. Available: <https://github.com/AIT-IES/FMITerminalBlock>. [Accessed 26. 09. 2018].
- [16] E. Widl, W. Müller, A. Elsheikh, M. Hörtenhuber and P. Palensky, "The FMI++ library: A high-level utility package for FMI for model exchange," in *2013 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, 2013.
- [17] "The Boost C++ Libraries," [Online]. Available: <http://www.boost.org/>.
- [18] ns-3 Documentation, "PyViz," [Online]. Available: <https://www.nsnam.org/wiki/PyViz>. [Accessed 26. 09. 2018].

## 8 Annex

### 8.1 List of Figures

Figure 1: Overview of ERIGrid's holistic testing procedure .....	15
Figure 2: Example representations of test systems: components, terminals (with directionality) and domains are clearly distinguished .....	16
Figure 3: Representation of experiment setup (simulators representing testbed level) .....	16
Figure 4: General Risk Management Process .....	19
Figure 5: Data exchange schemes possible in mosaik .....	22
Figure 6: Example of a composite frame including DSL model FMIAdapter .....	24
Figure 7: Example usage of PSCAD components for the FMI-compliant simulation back-end .....	25
Figure 8: Schematic view of the FMI-compliant ns-3 interface .....	26
Figure 9: mosaik simulation setup including cyber-attack models (highlighted in red) .....	29
Figure 10: Periodic synchronization .....	30
Figure 11: Predictive synchronization .....	31
Figure 12: IEEE 9-bus transmission system .....	33
Figure 13: SuT for the TC1 monolithic simulations .....	33
Figure 14: SuT for the TC1 co-simulations .....	33
Figure 15: Dynamic model overview for TC1 .....	34
Figure 16: Controller model overview for TC1 .....	35
Figure 17: Implementation of the TC1 co-simulation using FMUs from Simulink (ME) and PowerFactory (CS) .....	35
Figure 18: WPP connected to IEEE 9 bus via Equivalent Impedance .....	35
Figure 19: FMI adapter slot logic (PowerFactory co-simulation) .....	36
Figure 20: The dynamic IEEE 9-bus system model, implemented in PSCAD .....	37
Figure 21: Relative rotor angles of G1 and G2 .....	38
Figure 22: WPP output power .....	38
Figure 23: Voltage at bus 9 (where the WPP replaces generator G3) .....	38
Figure 24: Id and Iq reference currents .....	38
Figure 25: The output voltage of the WPP .....	38
Figure 26: The active power output of the WPP .....	38
Figure 27: The reactive power output of the WPP .....	38
Figure 28: The reference current (Id) of the WPP .....	38
Figure 29: Sketch of a transformer with integrated OLTC .....	40
Figure 30: Pseudo code for OLTC controller algorithm in TC2 .....	41
Figure 31: General layout of the implemented OLTC controller model .....	41
Figure 32: Interconnections between the model and the transformer .....	42
Figure 33: Snapshot of the parameters selection for normal operation .....	42
Figure 34: OLTC parameters during execution of experiment (first row: MV input, second row: tap position, third row: ready signal, fourth row: LV output) .....	43
Figure 35: Experiment 3 setup [14] .....	44
Figure 36: Switching operation in experiment 3 [14] .....	45
Figure 37: Experiment 4 setup [14] .....	46
Figure 38: Control operation in experiment 4 [14] .....	47
Figure 39: Overview of the Oul of test case TC3 .....	48
Figure 40: UML state chart of the CVC algorithm for test case TC3 .....	49
Figure 41: Schematic of the low voltage distribution system used in test case TC3 .....	49
Figure 42: Communication network topology for test case TC3 .....	50
Figure 43: Transmission stages for the meter data, showing the first stage (upper left), the second stage (upper right) and the third stage (lower left) .....	51
Figure 44: Transmission stages for the controller .....	51
Figure 45: Simulated end-to-end delays over the course of one full day .....	52
Figure 46: Overview of the simulation components for the technical assessment in test case TC352	

Figure 47: Implementation of the technical assessment of test case TC3 in the mosaik environment .....	53
Figure 48: Possible outcomes of the simulation experiments for test case TC3 with (a) final tap position 0, (b) final tap position -1 and (c) final tap position -2.....	54
Figure 49: Relative observed frequency of final tap positions (blue: 0, orange: -1, green: -2) for different values of $\Delta t_{\text{sender}}$ and controller dead times.....	55
Figure 50: Implementation of the cyber-security assessment of test case TC3 in the mosaik environment.....	56
Figure 51: Results from the cyber-security assessments for (a) scaling attacks, (b) ramping attacks, (c) random attacks and (d) digital signal tap attacks.....	57

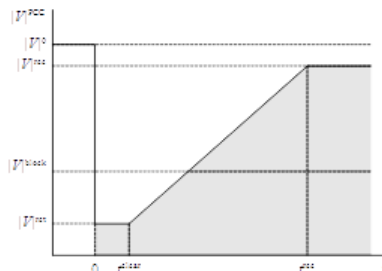
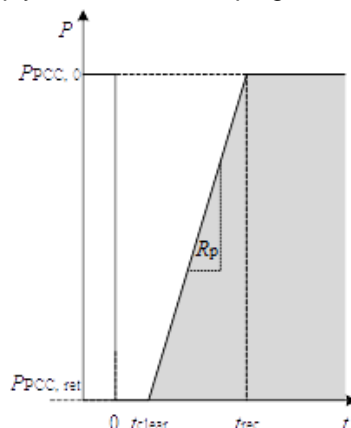
## 8.2 List of Tables

Table 1: Parameters of the OLTC controller algorithm in TC2 .....	41
Table 2: Control action delay statistics of experiment 3 [14].....	45

## 8.3 Updated Test Case Specification for TC1

### 8.3.1 Test Case

<b>Name of the Test Case</b>	JRA2-TC1
<b>Narrative</b>	<p>This test case aims to study and evaluate cyclic dependencies between continuous simulators. The interaction between electricity network and converter interfaced devices is of main interest in this test case as converters generally exhibit non-linear behaviour during faults.</p> <p>The experiments in the test case verify the low-voltage ride through capability of an onshore wind power plant (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.</p>
<b>Function(s) under Investigation</b>	The fault ride through capability of the converter, fast reactive power support, active power recovery by the WTGs.
<b>Object under Investigation</b>	The fault ride through capability of the converter, fast reactive power support, active power recovery by the WTGs
<b>Domain under Investigation</b>	<ul style="list-style-type: none"> <li>• Electrical</li> <li>• Control/ICT</li> <li>• Environment</li> </ul>
<b>Purpose of Investigation</b>	Verification of the converter dynamics and the converters' capability to comply to the FRT curves after a 3-phase short circuit upstream in the (sub-)transmission system, causing a voltage dip at the coupling point of the WPP.
<b>System under Test</b>	<p>The wind park (collection system plus WTG) is treated by the system operator as one single entity, the wind power plant. The 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:</p> <ul style="list-style-type: none"> <li>• the coupling point</li> <li>• the collection grid</li> <li>• the step-up transformers</li> <li>• the converters</li> </ul>

	<ul style="list-style-type: none"> <li>the WTG converters</li> <li>the WTG FRT controller</li> <li>the WTG protection schemes</li> <li>the WTG DC links</li> <li>the WTG electrical machine</li> </ul>
<b>Functions under Test</b>	<ul style="list-style-type: none"> <li>FRT functionality of the converter</li> <li>fast reactive power support</li> <li>physical response of external system interacting with Oul</li> <li>post fault active power recovery functionality</li> <li>normal operating controls of the WTGs</li> <li>current limit of the converters</li> <li>direct voltage control of the DC link of the wind turbine</li> </ul>
<b>Test criteria</b>	<ul style="list-style-type: none"> <li>converter must stay connected during and after the fault</li> <li>direct voltage operating region is not violated</li> <li>WTGs remain synchronised to the grid</li> <li>Transient and frequency stability must be maintained</li> </ul>
<b>target metrics</b> (test factors)	<p><u>FRT curve:</u></p> <ul style="list-style-type: none"> <li>WPP must stay connected</li> <li>WPP may temporarily disconnect from transmission system</li> </ul>  <p>Active power recovery curve:</p> <ul style="list-style-type: none"> <li>WTG has to comply to minimum ramping active power rate</li> </ul> 
<b>variability attributes</b>	<ul style="list-style-type: none"> <li>short circuit duration (primary versus backup protection)</li> </ul>
<b>quality attributes</b> (thresholds)	<p>FRT curve tests:</p> <ul style="list-style-type: none"> <li>short duration (200ms), deep dip</li> <li>criteria are fulfilled in case FRT controls keep direct voltage and WTG speed within design boundaries, and phase locked loop (PLL) maintains synchronisation.</li> </ul>



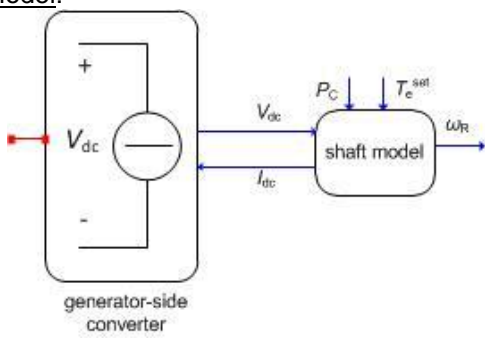
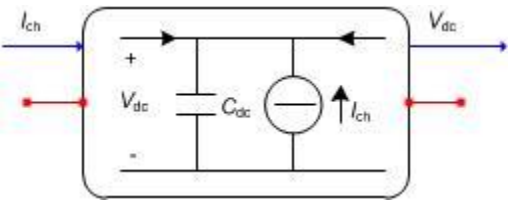
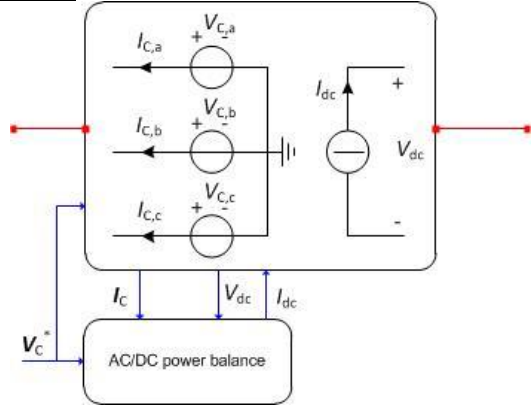
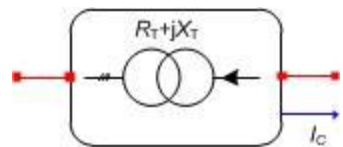
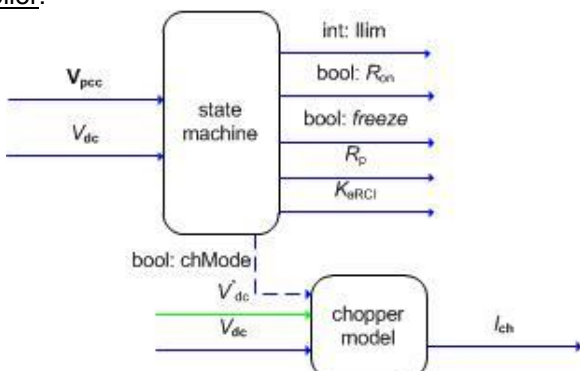
	<ul style="list-style-type: none"> <li>• <math>I_{lim}</math>: limiting scheme (0=no limiting, 1=d-axis priority, 2=q-axis priority, 3=proportional limiting) [-]</li> <li>• <math>K_{aRCI}</math>: additional reactive current injection gain [p.u.]</li> <li>• <math>R_p</math>: active power recovery ramp rate [p.u./s]</li> <li>• <math>R_{on}</math>: ramp rate on/off [-]</li> <li>• <math>p_{rot}</math>: chopper on/off [-]</li> </ul>
<b>Target measures</b>	<ul style="list-style-type: none"> <li>• FRT curve compliance</li> <li>• WPP remain synchronized to transmission grid</li> <li>• Correct initialization of converter controller and FRT controller</li> <li>• Direct Voltage operating region is not violated</li> </ul>
<b>Input and output parameters</b>	<p><u>Controllable input parameters:</u></p> <ul style="list-style-type: none"> <li>• fault duration</li> <li>• fault location</li> <li>• FRT control mode (on/off)</li> </ul> <p><u>Uncontrollable parameters:</u></p> <ul style="list-style-type: none"> <li>• voltage at coupling point implicitly set by the fault characteristics</li> <li>• wind turbine rotor speed</li> </ul> <p><u>Measured parameters:</u></p> <ul style="list-style-type: none"> <li>• DC voltage,</li> <li>• phase angle of PLL</li> </ul>
<b>Test Design</b>	<ul style="list-style-type: none"> <li>• determine operating point</li> <li>• set short circuit location to x</li> <li>• initiate short circuit at <math>t=1</math></li> <li>• clear fault at <math>t=y</math></li> <li>• assess test criteria</li> </ul>
<b>Initial system state</b>	WPP replaces G3 from IEEE 9 bus system inheriting it's operating point (P,Q at coupling point)
<b>Evolution of system state and test signals</b>	<p><u>Test events:</u> See test design.</p> <p><u>Target metrics:</u> All deterministic cases (so all test criteria for all parameter variations) must be successful.</p> <p><u>Internal boundary conditions:</u> The IGBT current limit of the converter is 110% of the rated current, the minimum active power recovery rate is 5 p.u./s, i.e., in 200 ms the wind turbines must be able to recover to the pre-fault 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.</p>
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	<ul style="list-style-type: none"> <li>• time constants inside SuT in between 50 <math>\mu</math>s and 5 s</li> <li>• continuous simulation, time step size depends on software experiment</li> <li>• components exhibit physical behaviour, the FRT controller is a discrete controller (state machine)</li> </ul>
<b>Source of uncertainty</b>	N/A



<b>Suspension criteria / Stopping criteria</b>	<ul style="list-style-type: none"> <li>violation of WTG synchronism with grid</li> <li>transient and frequency stability is violated</li> </ul>
--	---

### 8.3.2.2 Test Specification JRA2-TC1.TS2

<b>Reference to Test Case</b>	JRA2-TC1
<b>Title of Test</b>	Co-simulation of JRA2-TC1
<b>Test Rationale</b>	Perform and evaluate a co-simulation with the power system, the FRT controller and the converter controller implemented in separate models.
<b>Specific Test System (graphical)</b>	<p><u>Test specific SuT:</u></p> <p>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:</p> <ul style="list-style-type: none"> <li><math>V_{pcc}</math>: 3x1 array with nodal voltages [V]</li> <li><math>I_t</math>: 3x1 array with equivalent branch currents [A]</li> <li><math>I_d</math>: 3x1 array with converter currents [A]</li> <li><math>I_q</math>: 3x1 array with converter currents [A]</li> <li><math>V_{dc}</math>: voltage between + and – pole [V]</li> <li><math>I_{lim}</math>: limiting scheme (0=no limiting, 1=d-axis priority, 2=q-axis priority, 3=proportional limiting) [-]</li> <li><math>K_{aRCI}</math>: additional reactive current injection gain [p.u.]</li> <li><math>R_p</math>: active power recovery ramp rate [p.u./s]</li> <li><math>R_{on}</math>: ramp rate on/off [-]</li> <li><math>p_{rot}</math>: chopper on/off [-]</li> </ul> <p><u>Vector Controller:</u></p>

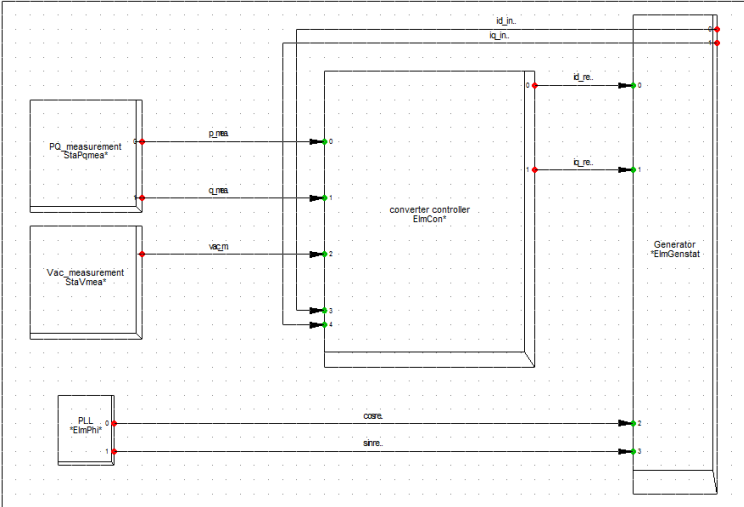
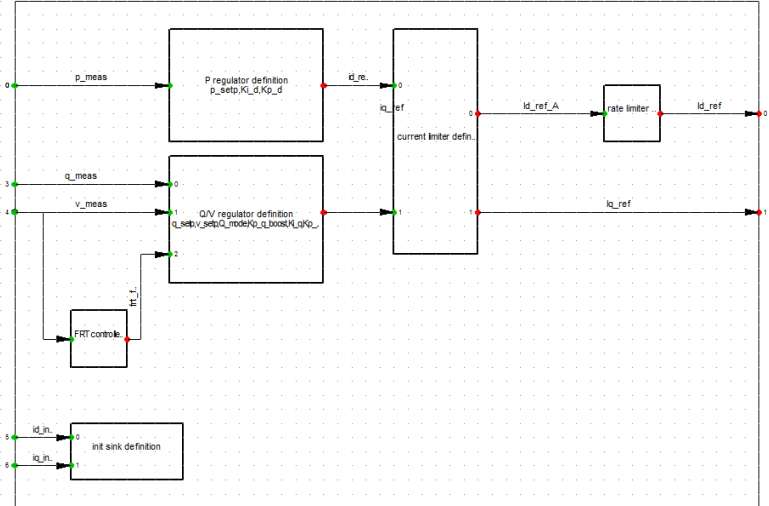
	<p><u>Wind Turbine Model:</u></p>  <p><u>DC Link Model:</u></p>  <p><u>Grid Side Converter:</u></p>  <p><u>Collection Grid:</u></p>  <p><u>FRT Controller:</u></p> 
<b>Target measures</b>	<ul style="list-style-type: none"> <li>• FRT curve compliance</li> <li>• WPP remain synchronized to transmission grid</li> <li>• correct initialization of converter controller and FRT controller</li> <li>• direct voltage operating region is not violated</li> </ul>

<b>Input and output parameters</b>	<u>Controllable input parameters:</u> <ul style="list-style-type: none"> <li>• fault duration</li> <li>• fault location</li> <li>• FRT control mode (on/off)</li> </ul> <u>Uncontrollable parameters:</u> <ul style="list-style-type: none"> <li>• voltage at coupling point implicitly set by the fault characteristics</li> <li>• wind turbine rotor speed</li> </ul> <u>Measured parameters:</u> <ul style="list-style-type: none"> <li>• DC voltage</li> <li>• Id and Iq currents</li> </ul>
<b>Test Design</b>	<ul style="list-style-type: none"> <li>• determine operating point</li> <li>• set short circuit location to x</li> <li>• initiate short circuit at t=1</li> <li>• clear fault at t=y</li> <li>• assess test criteria</li> </ul>
<b>Initial system state</b>	WPP replaces G3 from IEEE 9-bus system inheriting it's operating point (P,Q at coupling point)
<b>Evolution of system state and test signals</b>	<u>Test events:</u> See test design. <u>Target metrics:</u> All deterministic cases (so all test criteria for all parameter variations) must be successful. <u>Internal boundary conditions:</u> <ul style="list-style-type: none"> <li>• IGBT current limit of the converter is 110% of the rated current</li> <li>• minimum active power recovery rate is 5 p.u./s, i.e., in 200 ms the wind turbines must be able to recover to the pre-fault power output.</li> </ul> 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.
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	<ul style="list-style-type: none"> <li>• time constants inside SuT in between 50 <math>\mu</math>s and 5 s</li> <li>• continuous simulation, time step size depends on software experiment</li> <li>• components exhibit physical behaviour, the FRT controller is a discrete controller (state machine)</li> </ul>
<b>Source of uncertainty</b>	N/A
<b>Suspension criteria / Stopping criteria</b>	<ul style="list-style-type: none"> <li>• Violation of WTG synchronism with grid</li> <li>• If transient and frequency stability is violated</li> </ul>

### 8.3.3 Mapping Strategy

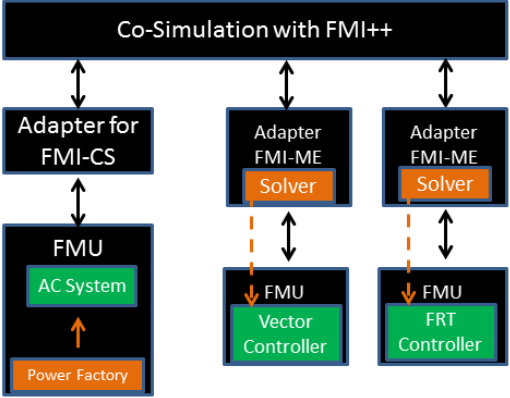
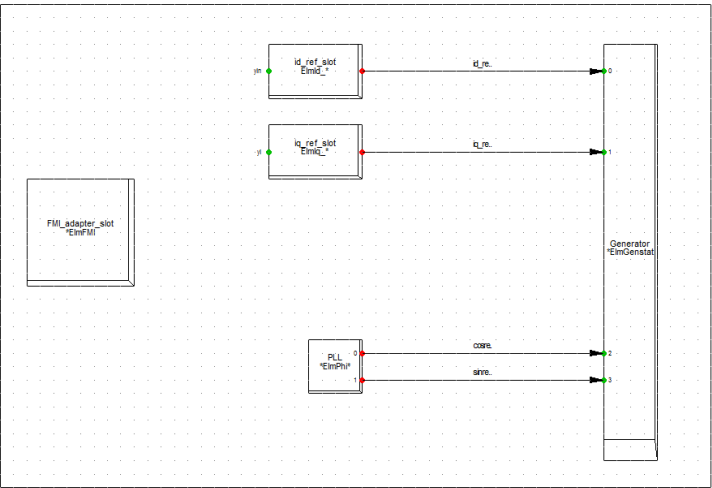
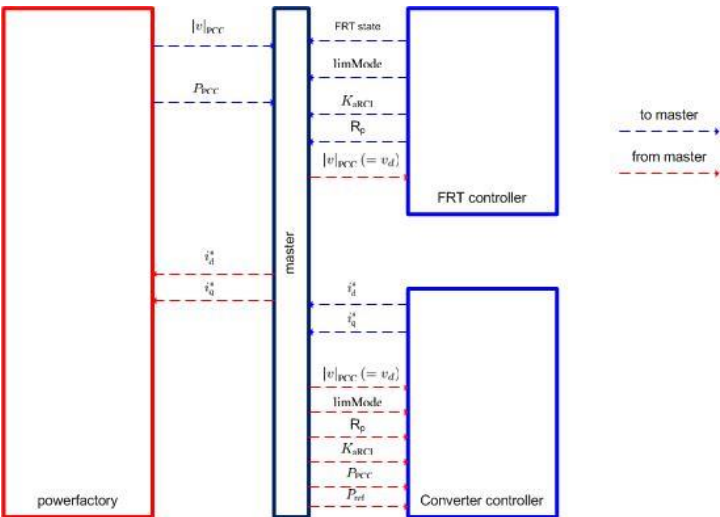
#### 8.3.3.1 Experiment Specification JRA1-TC1.TS1.monolithic

<b>Reference to Test Specification</b>	JRA2-TC1.TS1
<b>Title of Experiment</b>	Monolithic Simulation Using PowerFactory
<b>Experiment Realisation</b>	A monolithic simulation approach is applied for simulating the entire test case in DIgSILENT Power Factory. The models are developed and simulated in PowerFactory.

<p><b>Experiment Setup</b> (concrete lab equipment)</p>	<p><b>Vector Controller:</b></p> <p>WT main frame definition:</p>  <p><b>Wind Turbine Frame:</b></p> <p>controller definition:</p> 
<p><b>Experimental Design and Justification</b></p>	<ul style="list-style-type: none"><li>• 3-phase short circuit location: bus 4</li><li>• variation of fault duration: 200ms</li></ul>
<p><b>Precision of equipment</b></p>	<p>N/A</p>
<p><b>Uncertainty measurement</b></p>	<p>N/A</p>
<p><b>Storage of data</b></p>	<p>Simulations results are stored in PowerFactory for further analysis.</p>

8.3.3.2 Experiment Specification JRA1-TC1.TS2.powerfactory-simulink

<p><b>Reference to Test Specification</b></p>	<p>JRA2-TC1.TS2</p>
<p><b>Title of Experiment</b></p>	<p>Co-Simulation of PowerFactory and Simulink using the FMI++ Python Interface</p>

<b>Experiment Realisation</b>	<p>The FMI++ Python Interface is used to perform a co-simulation between Simulink and PowerFactory.</p> <p>The controller converter and FRT controller are modelled in Simulink. 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, with control signals coming from the other FMUs</p>
<b>Experiment Setup</b> (concrete lab equipment)	<p><u>Co-simulation setup:</u></p>  <p><u>PowerFactory wind turbine frame (with <math>I_d</math> and <math>I_q</math> control):</u></p> <p>Main FMI frame:</p>  <p><u>PowerFactory co-simulation IO interface:</u></p> 

<b>Experimental Design and Justification</b>	<ul style="list-style-type: none"> <li>3-phase short circuit location: bus 4</li> <li>variation of fault duration: 200ms</li> </ul>
<b>Precision of equipment</b>	N/A
<b>Uncertainty measurement</b>	N/A
<b>Storage of data</b>	For further evaluation, recorded data points are stored in a CSV file on the simulation host.

### 8.3.3.3 Experiment Specification JRA1-TC1.TS2.pscad-simulink

<b>Reference to Test Specification</b>	JRA2-TC1.TS2
<b>Title of Experiment</b>	Co-simulation of PSCAD and Simulink using the FMI++ Python Interface
<b>Experiment Realisation</b>	<p>The WTG controllers, including the FRT and converter controller, are modelled in Simulink and are exported as FMUs for ME. The transmission system, i.e., the dynamic IEEE 9-bus system, along with the WPP are modelled in PSCAD. The synchronous generator G3 is replaced by a WTG, which is modelled in PSCAD by a converter. The entire PSCAD simulation is exported as an FMU for CS, using the pscad_send and pscad_rcv blocks, developed for the co-simulation interface of PSCAD. All FMUs are connected via the FMI++ Python Interface.</p>
<b>Experiment Setup</b> (concrete lab equipment)	<p><u>Co-simulation setup:</u></p> <ul style="list-style-type: none"> <li>analogous to JRA2-TC1.TS2.powerfactory-simulink</li> </ul> <p><u>PSCAD system model:</u></p> <p><u>PSCAD co-simulation IO interface:</u></p>
<b>Experimental Design and Justification</b>	<ul style="list-style-type: none"> <li>3-phase short circuit location: bus 4</li> <li>variation of fault duration: 200ms</li> </ul>

<b>Precision of equipment</b>	N/A
<b>Uncertainty measurement</b>	N/A
<b>Storage of data</b>	For further evaluation, recorded data points are stored in a CSV file on the simulation host.

#### 8.3.3.4 Experiment Specification JRA1-TC1.TS2.mosaik

<b>Reference to Test Specification</b>	JRA2-TC1.TS2
<b>Title of Experiment</b>	Co-Simulation of PowerFactory and Simulink using mosaik
<b>Experiment Realisation</b>	Analogous to JRA1-TC1.TS2.powerfactory-simulink, only that mosaik is used to couple the FMUs.
<b>Experiment Setup</b> (concrete lab equipment)	<p><u>Co-simulation setup:</u></p> <p>Otherwise analogous to JRA1-TC1.TS2.powerfactory-simulink.</p>
<b>Experimental Design and Justification</b>	Analogous to JRA1-TC1.TS2.powerfactory-simulink.
<b>Precision of equipment</b>	N/A
<b>Uncertainty measurement</b>	N/A
<b>Storage of data</b>	The outputs from the individual simulation components are stored as time series data (HDF5 data format).

### 8.4 Updated Test Case Specification for TC2

#### 8.4.1 Test Case

<b>Name of the Test Case</b>	JRA2-TC2
<b>Narrative</b>	OLTC transformer setups including the required control logic need thoughtful validation and testing beyond monolithic simulations. The purpose of this test case is to have the first step towards the implementation of a control block using FMU for ME 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. Closed-loop voltage regulation can be implemented using not only a simulated controller but also using a physical controller embedded in an Arduino card.
<b>Function(s) under Investigation</b>	The focus of this investigation is the voltage control function for the OLTC transformer.

<b>Object under Investigation</b>	<ul style="list-style-type: none"> <li>OLTC controller</li> </ul>
<b>Domain under Investigation</b>	<ul style="list-style-type: none"> <li>Electrical &amp; electronic domains</li> <li>Control / ICT domain</li> </ul>
<b>Purpose of Investigation</b>	<p>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 low voltage side of the transformer within the specified boundaries is evaluated.</p> <p>Certain controller realizations, such as the encapsulation into an FMU or a realization by an embedded device, may impose 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 experiments described below will be performed.</p>
<b>System under Test</b>	<p>The OLTC, its electrical vicinity, and its corresponding control algorithm comprise the SuT. The relevant sub-systems are:</p> <ul style="list-style-type: none"> <li>grid supply</li> <li>voltage sensor</li> <li>transformer</li> <li>OLTC control block</li> <li>actuators that operate the OLTC</li> <li>load</li> <li>communication links between the controls and the transformer</li> </ul> <p>The following system diagram illustrates the interaction of the major system components.</p> <p><b>Domains:</b>  <span style="color: red;">—</span> electric      <span style="color: blue;">—</span> control/ICT</p>
<b>Functions under Test</b>	<ul style="list-style-type: none"> <li>OLTC control functionality</li> <li>voltage regulation within the admissible limits</li> <li>control functionality implemented in an embedded hardware device (Arduino card)</li> <li>control functionality exported into an FMU for ME</li> </ul>
<b>Test criteria</b>	<ul style="list-style-type: none"> <li>OLTC control block must be able to exchange data with other blocks within the specified time-step using its software/physical versions of implementation</li> <li>voltage operation region not violated</li> </ul>
<b>target metrics</b> (test factors)	Voltage regulation regions (LV):



<b>variability attributes</b>	<ul style="list-style-type: none"> <li>• low-voltage set-point</li> <li>• load variations</li> <li>• medium-voltage variations</li> </ul>
<b>quality attributes</b> (thresholds)	<p>A voltage variation test outcome is considered to be positive if and only if:</p> <ul style="list-style-type: none"> <li>• the admissible voltage range of [TOL-, TOL+] is re-established and maintained within the first 10 s after the induced disturbance, and</li> <li>• the system operates without system disconnections (shutdowns).</li> </ul>

## 8.4.2 Qualification Strategy

### 8.4.2.1 Test Specification JRA2-TC2.TS1

<b>Reference to Test Case</b>	JRA2-TC2
<b>Title of Test</b>	Voltage Control Function Assessment
<b>Test Rationale</b>	This test aims at verifying the behavior of various OLTC controller implementations in the context of the SuT. It specifically targets the interaction of the Oul with surrounding system components and the resulting control actions.
<b>Specific Test System</b> (graphical)	

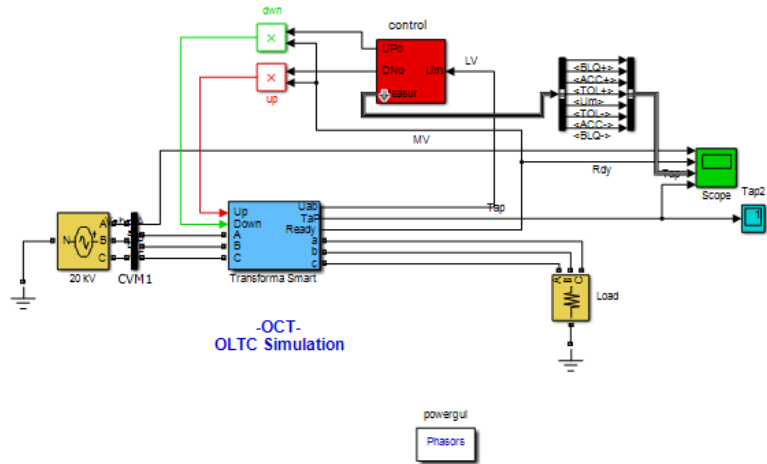
	<div>The types, descriptions, and units of the interface variables are as follows:</div> <table><tr><th>Name</th><th>Unit</th><th>Description</th></tr><tr><td>MV</td><td>[V]</td><td>Nodal voltages</td></tr><tr><td>LV</td><td>[V]</td><td>Nodal voltages</td></tr><tr><td>LV_val</td><td>[V]</td><td>Voltage measurement</td></tr><tr><td>set-point</td><td>[V]</td><td>Voltage set-point</td></tr><tr><td>e</td><td>[V]</td><td>Voltage error</td></tr><tr><td>up</td><td>[-]</td><td>Next transformer level (coil)</td></tr><tr><td>down</td><td>[-]</td><td>Previous transformer level (coil)</td></tr></table>	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)												
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)																																			
Target measures	<ul style="list-style-type: none"><li>controller low voltage</li></ul>																																				
Input and output parameters	<div><div>Controllable input parameters:</div><ul style="list-style-type: none"><li>voltage set-point</li><li>load value</li><li>delays</li></ul><div>Uncontrollable input parameters:</div><ul style="list-style-type: none"><li>transformer taps</li><li>main source voltage value</li><li>frequency.</li></ul><div>Measured parameters:</div><ul style="list-style-type: none"><li>load voltage value</li><li>controller states with respect to “target metrics” (see table below)</li></ul><table><tr><th>Concept</th><th>Value</th><th>unit</th><th>Notes</th></tr><tr><td>SetP</td><td>420</td><td>V</td><td>Set point</td></tr><tr><td>Tap Step</td><td>2.5% *</td><td>V</td><td>Tap Step</td></tr><tr><td>BLQ+</td><td>50% *</td><td>V</td><td>Block Control</td></tr><tr><td>ACC+</td><td>5% *</td><td>V</td><td>Step down (Delay)</td></tr><tr><td>TOL+</td><td>+1% *</td><td>V</td><td>Step Down</td></tr><tr><td>TOL-</td><td>- 1% *</td><td>V</td><td>Step Up</td></tr><tr><td>ACC+</td><td>-5% *</td><td>V</td><td>Step Up (Delay)</td></tr><tr><td>BLQ-</td><td>-50% *</td><td>V</td><td>Block Control</td></tr></table><div>* depends on the rates of the deployed transformer</div></div>	Concept	Value	unit	Notes	SetP	420	V	Set point	Tap Step	2.5% *	V	Tap Step	BLQ+	50% *	V	Block Control	ACC+	5% *	V	Step down (Delay)	TOL+	+1% *	V	Step Down	TOL-	- 1% *	V	Step Up	ACC+	-5% *	V	Step Up (Delay)	BLQ-	-50% *	V	Block Control
Concept	Value	unit	Notes																																		
SetP	420	V	Set point																																		
Tap Step	2.5% *	V	Tap Step																																		
BLQ+	50% *	V	Block Control																																		
ACC+	5% *	V	Step down (Delay)																																		
TOL+	+1% *	V	Step Down																																		
TOL-	- 1% *	V	Step Up																																		
ACC+	-5% *	V	Step Up (Delay)																																		
BLQ-	-50% *	V	Block Control																																		
Test Design	<div>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. In each test iteration one input variation from the predefined set is chosen. The test criteria defined above will then be applied to evaluate each test run. The predefined set of inputs features an improved comparability of the experiments. If a test iteration shows 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:</div>																																				

	<ol style="list-style-type: none"> <li>1. determine the operating set-point</li> <li>2. wait until the output is stabilized</li> <li>3. vary the MV input voltage and/or the set-point according to the current border case</li> <li>4. assess test criteria</li> <li>5. repeat 2-4 until all predefined border cases are tested</li> </ol> <p>The following border cases are defined:</p> <ul style="list-style-type: none"> <li>• maximally decrease MV input voltage, use constant voltage set-point</li> <li>• maximally increase MV input voltage, use constant voltage set-point</li> <li>• decrease voltage set-point from maximal to minimal value, do not artificially vary MV input voltage</li> <li>• increase voltage set-point from minimal to maximal value, do not artificially vary MV input voltage</li> </ul>
<b>Initial system state</b>	<p>Initial power flow conditions:</p> <ul style="list-style-type: none"> <li>• Load voltage value (output) = voltage set-point</li> </ul>
<b>Evolution of system state and test signals</b>	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.
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	The simulation of virtual components uses fixed time steps of 0.1 ms. Any periodic communication with external equipment such as the embedded controller may be done with a lower time resolution of up to 200 ms.
<b>Source of uncertainty</b>	LV and MV measurement error, parametric deviations (manly within the transformer), timing deviations
<b>Suspension criteria / Stopping criteria</b>	All test patterns are successfully applied.

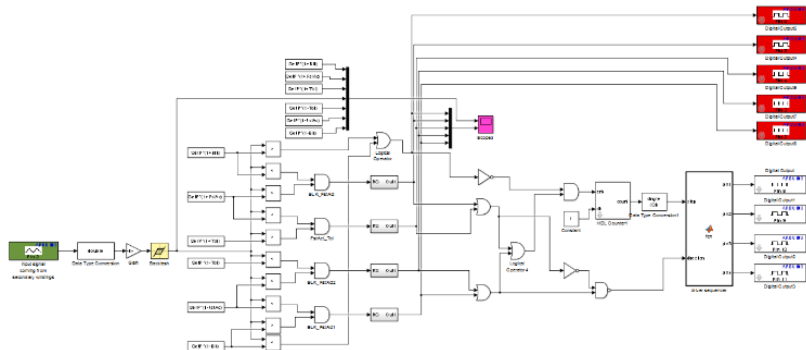
### 8.4.3 Mapping Strategy

#### 8.4.3.1 Experiment Specification JRA2-TC2.TS1.1

<b>Reference to Test Specification</b>	JRA2-TC2.TS1
<b>Title of Experiment</b>	Software OLTC Controller
<b>Experiment Realisation</b>	In the 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 (branded 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.
<b>Experiment Setup</b> (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)

Experimental Design and Justification

Induced MV-Variations: (at 20kV nominal voltage)

Start Time [s]	MV Set-Point [p.u.]
0	1
12	0.9
16	1

LV 3-Phase parallel RLC load:

Parameter	Value	Unit
Configuration	Y (Neural)	

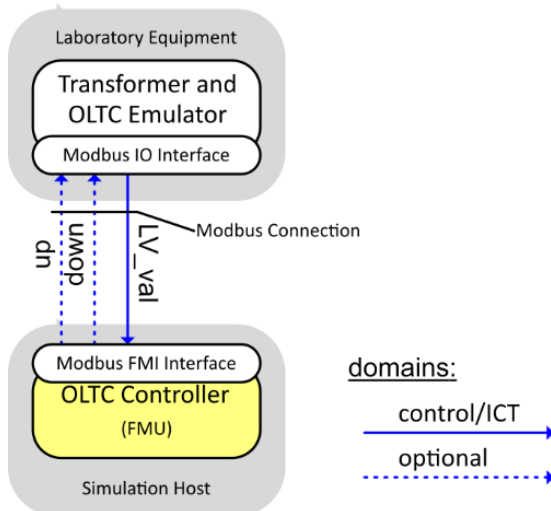
		Nominal Phase-to-phase voltage	420	[V]	
		Unbalanced-Power	Off		
		Active Power	36	kW	
		Inductive Reactive Power	0	[var]	
		Capacitive Reactive Power	0	[var]	
Precision of equipment	• relative solver tolerance = 1e-4.				
Uncertainty measurement	N/A				
Storage of data	For further evaluation, the simulation outcome will be stored in a CSV file on the simulation host.				

#### 8.4.3.2 Experiment Specification JRA2-TC2.TS1.2

<b>Reference to Test Specification</b>	JRA2-TC2.TS1
<b>Title of Experiment</b>	Hardware OLTC Controller
<b>Experiment Realisation</b>	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. The software part includes the simulation model built in Simulink Matlab. Once the first experiment is done correctly, we choose a new model based on a discrete solver (no continuous states) with fixed-step size=0.1 ms to facilitate its integration into the open-source electronics platform featuring an AT Mega 2560 microcontroller.
<b>Experiment Setup</b> (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 environments. The implementation of the OLTC controller is also identical to the controller which is shown in experiment JRA2-TC2.TS1.1.
<b>Experimental Design and Justification</b>	The simulated MV value is adjusted in order to induce a voltage drop on the LV side. The applied MV pattern is determined during the experiment runs such that the LV values reside within all previously defined LV bands and such that all control actions of the Oul can be fully observed.
<b>Precision of equipment</b>	<ul style="list-style-type: none"> <li>fixed simulation and communication steps: 0.1 ms</li> </ul>
<b>Uncertainty measurement</b>	Involved uncertainties are quantified through a direct comparison to the results off the reference simulation.
<b>Storage of data</b>	For further evaluation, recorded data points will be stored in a CSV file on the simulation host.

#### 8.4.3.3 Experiment Specification JRA2-TC2.TS1.3

<b>Reference to Test Specification</b>	JRA2-TC2.TS1
<b>Title of Experiment</b>	OLTC Controller as FMU-ME
<b>Experiment Realisation</b>	This test concerns transferring the previously tested controller into an FMU for ME block. This block can be simulated with other system blocks or components in order to investigate and compare its behaviour to the case in experiment JRA2-TC2.TS1.1. The experiment covers a prototyping stage that integrates an emulated physical plant (i.e., the OLTC transformer and the measurement equipment represented by the analogue

	<p>voltage measurement signals) 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 emulation.</p>
<p><b>Experiment Setup</b> (concrete lab equipment)</p>	<p>The grid supply, the OLTC including its electrical actuators, the transformer itself, the voltage sensors, and the electrical load will be emulated by the physical 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) connected to the IO devices may actively poll all data points. Hence, the IO devices act as Modbus slaves.</p> <p>On the simulation host, the exported FMU and all interface components are executed such that end-to-end communication between the plant emulation and the simulated controller is achieved. The interface components periodically poll the voltage readings via Modbus and send the outputs of the controller to a logging facility. Logged data includes the control outputs and the inputs read from the plant emulator. Stored recordings allow to evaluate the controller FMU in detail and must also contain timing information up to the limits imposed by the communication infrastructure. The actual communication period depends on the capabilities of the equipment. A period of 100 ms to 200 ms is targeted.</p> <p>The following graphic illustrates the laboratory setup. The realization of the controller FMI is specified by the Simulink diagram in Experiment JRA2-TC2.TS1.1. The plant model in the previous experiment (the grid source, smart transformer, and load) is represented by the plant emulator in the current experiment.</p>  <p>The diagram illustrates the laboratory setup. It shows two main components: 'Laboratory Equipment' and 'Simulation Host'. The 'Laboratory Equipment' contains a 'Transformer and OLTC Emulator' and a 'Modbus IO Interface'. The 'Simulation Host' contains a 'Modbus FMI Interface' and an 'OLTC Controller (FMU)'. A 'Modbus Connection' links the two. Data flows are indicated: 'up' (dashed blue arrow), 'down' (dashed blue arrow), and 'LV_val' (solid blue arrow). A legend on the right defines the arrow types: solid blue for 'control/ICT' and dashed blue for 'optional'.</p>
<p><b>Experimental Design and Justification</b></p>	<p>For each experiment run, LV_val has to be dynamically adjusted within the permissible input range such that at least 50 control output transitions are triggered. Inputs may be manually adjusted and hence, the precise input voltage sequence may only be available after a single experiment run.</p>
<p><b>Precision of equipment</b></p>	<ul style="list-style-type: none"> <li>relative solver tolerance: 1e-4</li> <li>event search precision: 1 ms</li> <li>max. 200 ms Modbus polling cycle</li> </ul>
<p><b>Uncertainty measurement</b></p>	<ul style="list-style-type: none"> <li>recording and evaluation of timing variations</li> </ul>
<p><b>Storage of data</b></p>	<p>For further evaluation, recorded data points will be stored in a CSV file on the simulation host.</p>

#### 8.4.3.4 Experiment Specification JRA2-TC2.TS1.4

<b>Reference to Test Specification</b>	JRA2-TC2.TS1						
<b>Title of Experiment</b>	FMI-based OLTC Controller Hardware in the Loop						
<b>Experiment Setup</b> (concrete lab equipment)	<p>The standalone control software is derived from the Simulink model and uploaded into the embedded controller board, which is also used in experiment JRA2-TC2.TS1.1. The LV_val IO signal of the controller is connected to the IO interface that drives the signal. Similarly, the digital output signals of the embedded controller are connected to the IO interface. 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 executing 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.</p> <p>Interface components on the simulation host run the plant model encapsulated into an FMU and manage the communication between the IO interface and the FMU. The FMU itself contains an exported version of the Simulink plant model of experiment JRA2-TC2.TS1.1. The following figure illustrates the experimental setup.</p> <pre> graph TD     subgraph Simulation_Host [Simulation Host]         PM[Plant Model FMU]         MFI[Modbus FMI Interface]     end     subgraph Testbed [Testbed]         MII[Modbus IO Interface]     end     IC[Interface Circuit]     OC[OLTC Controller Hardware]      PM &lt;--&gt; Modbus Connection  MFI     MFI &lt;--&gt; up LV_val, down LV_val  MII     MII &lt;--&gt; up LV_val, down IO Signals  IC     IC &lt;--&gt; OC   </pre>						
<b>Experiment Realisation</b>	<p>In order to test the embedded OLTC controller including its IO interfaces, it will be coupled to a testbed mimicking the behaviour of the plant. The testbed may consist of industrial IO interfaces that drive the IO lines of the controller and a plant model that is interfaced via FMI for ME. Alternatively, the testbed may be directly added to the control software such that a Modbus testing interface is provided. The simulated low voltage value (LV_val) is transferred to the testbed that applies it to the controller. The signal is then processed by the physically available OLTC controller. Likewise, the control commands of the OLTC controller are sensed and transferred to the plant model. The software of the controller is still derived from the Simulink model, but in contrast to experiment JRA2-TC2.TS1.2, the controller is interfaced by its IO facilities.</p>						
<b>Experimental Design and Justification</b>	<p>Induced MV variations: (at 20kV nominal voltage):</p> <table border="1"> <thead> <tr> <th>Start Time [s]</th><th>MV Set-Point [p.u.]</th></tr> </thead> <tbody> <tr> <td>0</td><td>1</td></tr> <tr> <td>5</td><td>0.9</td></tr> </tbody> </table>	Start Time [s]	MV Set-Point [p.u.]	0	1	5	0.9
Start Time [s]	MV Set-Point [p.u.]						
0	1						
5	0.9						

		15	1	
		25	0.8	
		35	1	
		45	0.5	
		50	0.8	
		60	1	
	LV 3-phase parallel RLC load:			

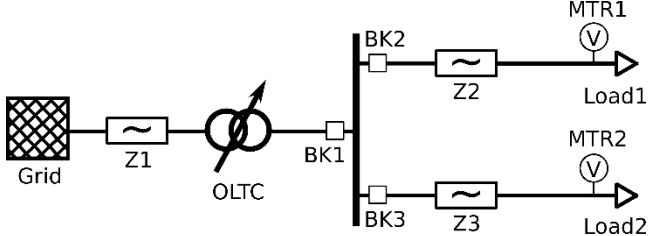
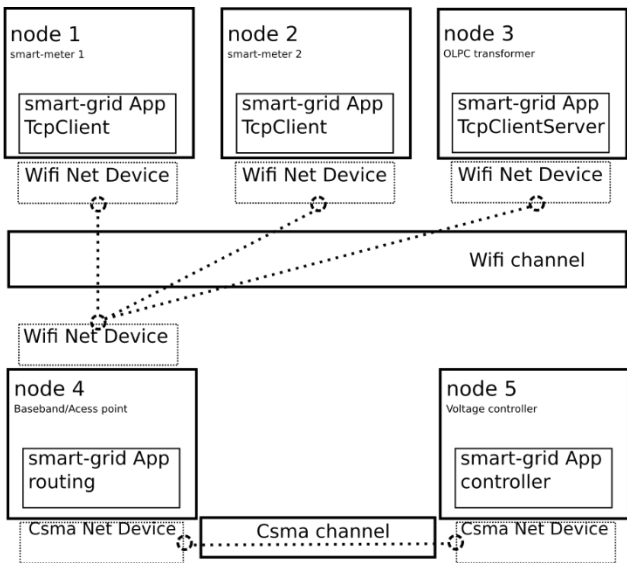
## 8.5 Updated Test Case Specification for TC3

### 8.5.1 Test Case

<b>Name of the Test Case</b>	JRA2-TC3
<b>Narrative</b>	<p>This test case deals with the impact of ICT-related aspects 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. ICT-related aspects of interest are technical features (communication delays, controller dead times) and consequences of cyber-security attacks (scaling attacks, ramping at-tacks, random scaling factor attacks, digital signal tap attacks).</p> <p>The aim of this test case is to demonstrate and assess the effect of communication networks on the actuation pattern of the controller and the resulting physical 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 delays is implemented for the controller. Similarly, it is assumed that cyber-security measures are insufficient, opening the possibility to implement cyber-attacks.</p> <p>Moreover, this test case aims to demonstrate the importance of realistic simulation approaches for assessing distributed and centralized Smart Grid control algorithms as well as benchmarking communication network technologies and topologies for use in Smart Grid applications.</p>



<b>Function(s) under Investigation</b>	<p>The focus of this investigation is the actuation pattern of a voltage controller in the presence of communication delays, controller dead times and cyber-attacks. This controller calculates setpoints for the tap positions of an OLTC transformer based on the readings of two voltage meters.</p>
<b>Object under Investigation</b>	<ul style="list-style-type: none"> <li>voltage controller</li> <li>OLTC transformer</li> </ul>
<b>Domain under Investigation</b>	<ul style="list-style-type: none"> <li>electrical (voltage levels)</li> <li>ICT (data transmission, cyber-attacks)</li> <li>control (calculation of setpoints, OLTC actuation)</li> </ul>
<b>Purpose of Investigation</b>	<p>Characterize the response of the voltage controller in the presence of communication delays, controller dead times and cyber-attacks.</p>
<b>System under Test</b>	<p><u>Controller:</u></p> <ul style="list-style-type: none"> <li>simple rule-based control algorithm, calculating tap position setpoints for the OLTC transformer depending on the meter readings</li> <li>runs on a dedicated server, separate from transformer and meters</li> <li>when receiving a new measurement and computing a new tap position setpoint, the controller becomes unresponsive for a short time period (dead time), i.e., further measurements arriving during this time</li> </ul>

	<p>are not processed</p> <p><u>Low voltage distribution system:</u></p> <ul style="list-style-type: none"> <li>• OLTC MV/LV transformer</li> <li>• changing the OLTC's tap position is assumed to take 3s in total, during which the OLTC is not responsive to new setpoints</li> <li>• 2 loads with voltage meters (measurement devices and/or smart meters) that send their measurements at regular intervals to the controller</li> </ul>  <p><u>Communication network:</u></p> <ul style="list-style-type: none"> <li>• base station/access point for wireless communication</li> <li>• 1 node (sender) for each voltage meter, wireless connection to the base station</li> <li>• 1 node (receiver) for the transformer, wireless connection to the base station</li> <li>• 1 node (sender/receiver) for the server (controller), wired internet connection to the base station</li> </ul> 				
<b>Functions under Test</b>	<ul style="list-style-type: none"> <li>• setpoint calculation from voltage controller</li> <li>• setpoint actuation at OLTC</li> <li>• data transmission of meter readings and controller setpoints</li> </ul>				
<b>Test criteria</b>	<p>Even in the presence of communication delays, controller dead times and cyber-attacks, the actuation of the OLTC transformer should result in acceptable operational conditions (voltage levels according to grid codes). More specifically, the tap position should coincide with the “idealized” base case, where no communication delays, controller dead times or cyber-attacks are present.</p>				
<table border="1"> <tr> <td data-bbox="199 1877 549 1966"><b>target metrics</b> (test factors)</td><td data-bbox="549 1877 1442 1966"> <ul style="list-style-type: none"> <li>• tap position</li> <li>• voltage levels</li> </ul> </td></tr> <tr> <td data-bbox="199 1966 549 2007"><b>variability attributes</b></td><td data-bbox="549 1966 1442 2007"> <ul style="list-style-type: none"> <li>• <math>T_{\text{sender}}</math>: time between sending two measurements</li> </ul> </td></tr> </table>	<b>target metrics</b> (test factors)	<ul style="list-style-type: none"> <li>• tap position</li> <li>• voltage levels</li> </ul>	<b>variability attributes</b>	<ul style="list-style-type: none"> <li>• <math>T_{\text{sender}}</math>: time between sending two measurements</li> </ul>	
<b>target metrics</b> (test factors)	<ul style="list-style-type: none"> <li>• tap position</li> <li>• voltage levels</li> </ul>				
<b>variability attributes</b>	<ul style="list-style-type: none"> <li>• <math>T_{\text{sender}}</math>: time between sending two measurements</li> </ul>				

	<ul style="list-style-type: none"> <li>• <math>t_{\text{sender}}</math>: time offset between voltage measurements</li> <li>• <math>t_{\text{ctrl}}</math>: controller dead time</li> <li>• random number generator seed for ICT network simulator</li> </ul>
<b>quality attributes</b> (thresholds)	Acceptable operational conditions for the voltage levels are between 0,95 p.u. and 1,05 p.u.

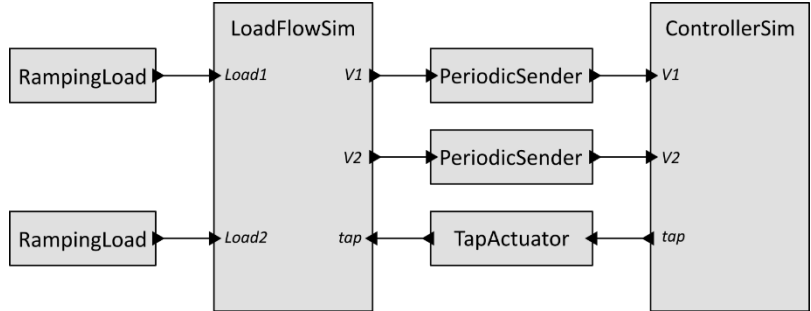
## 8.5.2 Qualification Strategy

### 8.5.2.1 Test Specification JRA2-TC3.TS1

<b>Reference to Test Case</b>	JRA2-TC3
<b>Title of Test</b>	validation of network communication models
<b>Test Rationale</b>	This is a standalone experiment including only the communication network simulator, in order to test and validate the ns-3 models developed for the test case.
<b>Specific Test System</b> (graphical)	The test system comprises the communication network as described above.
<b>Target measures</b>	The simulated transmission of data between the nodes has to happen according to the defined protocols (Wi-Fi, Ethernet)
<b>Input and output parameters</b>	<u>Measured parameters:</u> <ul style="list-style-type: none"> <li>• timing information about data packet transmission</li> <li>• end-to-end delays of data transmission between nodes</li> </ul>
<b>Test Design</b>	<ul style="list-style-type: none"> <li>• The meters send every 15 minutes a packet of fixed size (100 bytes) containing a voltage measurement to the access point (Wi-Fi protocol). Subsequently, the data is sent to the server (Ethernet protocol), where the CVC algorithm is implemented.</li> <li>• In the next minute after the packets are received, the controller sends a tap position setpoint to the OLTC transformer, which is also connected to the access point (Wi-Fi).</li> </ul>
<b>Initial system state</b>	The meters start sending at $t = 0$ min, no other data traffic otherwise.
<b>Evolution of system state and test signals</b>	N/A
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	internal time resolution of communication network simulator
<b>Source of uncertainty</b>	communication delays
<b>Suspension criteria / Stopping criteria</b>	N/A

### 8.5.2.2 Test Specification JRA2-TC3.TS2\_baseline

<b>Reference to Test Case</b>	JRA2-TC3
<b>Title of Test</b>	Baseline simulation without communication delays, controller dead times or cyber-attacks
<b>Test Rationale</b>	The outcome of this test is the reference for test specification JRA2-TC3.TS2. By neglecting ICT-related effects, this represents the "idealized" case.

<b>Specific Test System</b> (graphical)	<p>Overview of individual simulation components:</p>  <pre> graph LR     RL1[RampingLoad] --&gt; L1[Load1]     RL2[RampingLoad] --&gt; L2[Load2]     L1 -- V1 --&gt; PS1[PeriodicSender]     L2 -- V2 --&gt; PS2[PeriodicSender]     PS1 -- V1 --&gt; CS[ControllerSim]     PS2 -- V2 --&gt; CS     CS -- tap --&gt; TA[TapActuator]     TA -- tap --&gt; L1     TA -- tap --&gt; L2 </pre>
<b>Target measures</b>	<p>This test serves as baseline for reference for test specification JRA2-TC3.TS2. As such, there is no qualification strategy for the output of this test itself.</p>
<b>Input and output parameters</b>	<p><u>Controllable input parameters:</u></p> <ul style="list-style-type: none"> <li>tap position</li> </ul> <p><u>Measured parameters:</u></p> <ul style="list-style-type: none"> <li>voltages at loads</li> </ul> <p><u>Uncontrollable parameters:</u></p> <ul style="list-style-type: none"> <li>power consumption of loads</li> <li><math>T_{\text{sender}} = 1 \text{ min}</math></li> <li><math>t_{\text{sender}} = 0 \text{ s}</math></li> <li><math>t_{\text{ctrl}} = 0 \text{ s}</math></li> </ul>
<b>Test Design</b>	<ul style="list-style-type: none"> <li>The test takes 2 minutes of simulation time, during which both loads are linearly ramped up.</li> <li>The meters send their measurements to the controller in regular intervals (<math>T_{\text{sender}}=1 \text{ min}</math>) in perfect synchronization (<math>t_{\text{sender}}=0 \text{ s}</math>), beginning at the start of the simulation (<math>t=0 \text{ min}</math>).</li> <li>Whenever the controller receives new measurements, a new value for the tap position is calculated and sent to the OLTC transformer.</li> <li>The transmission of data from the meters to the controller (voltage measurements) and from the controller to the OLTC's tap actuator (tap position setpoint) happens without delays.</li> <li>Voltage <math>V1</math> (voltage measured at <i>Load1</i>) is expected to stay within the operational limits throughout the test. Voltage <math>V2</math> (voltage measured at <i>Load2</i>) falls beyond the lower threshold within 1 minute, and a change in the tap position (from 0 to -1) is expected to happen the next time the meters transmit their measurements to the controller (<math>t=1 \text{ min}</math>).</li> </ul>
<b>Initial system state</b>	<ul style="list-style-type: none"> <li>initial tap position (at <math>t=0 \text{ min}</math>): 0</li> </ul>
<b>Evolution of system state and test signals</b>	<ul style="list-style-type: none"> <li>Load1: active power consumptions ramps linearly from 0 to 2 kW within 2 minutes</li> <li>Load2: active power consumptions ramps linearly from 7 to 10 kW within 2 minutes</li> <li>For the sake of simplicity, the effect of changing a tap should become effective at the end of the OLTC's actuation deadtime (instantaneous event, no continuous process).</li> </ul>
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	1 second
<b>Source of uncertainty</b>	N/A
<b>Suspension criteria / Stopping criteria</b>	N/A

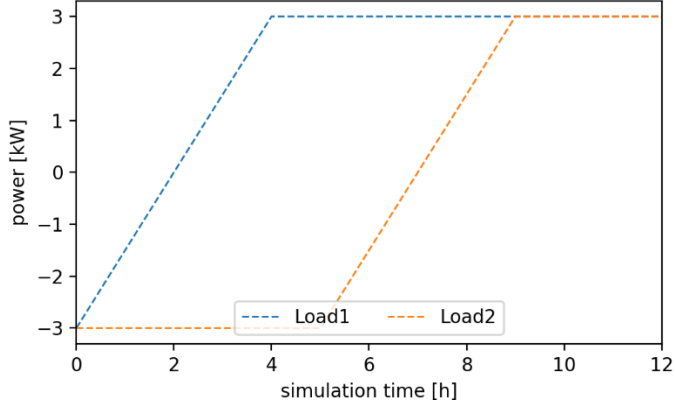
### 8.5.2.3 Test Specification JRA2-TC3.TS2

<b>Reference to Test Case</b>	JRA2-TC3
<b>Title of Test</b>	Assessment of impact of communication delays and controller dead times
<b>Test Rationale</b>	This test assesses the impact of communication delays and controller dead times on the actuation pattern of the OLTC and the voltage levels. The impact is evaluated by comparing the final realized tap position with the expected tap position (from the base line simulation JRA2-TC3.TS2_baseline). Furthermore, the voltage levels at the end of the simulation are expected to be within the operational limits.
<b>Specific Test System (graphical)</b>	<p><u>Overview of individual simulation components:</u></p> <p><b>Attention:</b> The controller calculates and transmits tap position every time a new measurement arrives (using the latest values available).</p>
<b>Target measures</b>	<ul style="list-style-type: none"> <li>expected final tap position (at <math>t=2</math> min): -1</li> <li>voltage levels are expected to be within operational limits</li> </ul>
<b>Input and output parameters</b>	<p><u>Controllable input parameters:</u></p> <ul style="list-style-type: none"> <li>tap position</li> </ul> <p><u>Measured parameters:</u></p> <ul style="list-style-type: none"> <li>voltages at loads</li> </ul> <p><u>Uncontrollable parameters:</u></p> <ul style="list-style-type: none"> <li>power consumption of loads</li> <li><math>T_{\text{sender}} = 1</math> min</li> <li><math>\Delta t_{\text{sender}} \in \{-14 \text{ ms}, -12 \text{ ms}, \dots, 12 \text{ ms}, 14 \text{ ms}\}</math></li> <li><math>\Delta t_{\text{ctrl}} \in \{2 \text{ ms}, 10 \text{ ms}, 20 \text{ ms}\}</math></li> </ul>
<b>Test Design</b>	<ul style="list-style-type: none"> <li>The test takes 2 minutes of simulation time, during which both loads are linearly ramped up.</li> <li>The meters send their measurements to the controller in regular intervals (<math>T_{\text{sender}} = 1</math> min) with a small relative delay to each other (<math>\Delta t_{\text{sender}}</math>), beginning at the start of the simulation (<math>t = 0</math> min).</li> <li>Whenever the controller receives new measurements, a new value for the tap position is calculated and sent to the OLTC transformer.</li> <li>The transmission of data from the meters to the controller (voltage measurements) and from the controller to the OLTC's tap actuator (tap position setpoint) happens with a delay (calculated by the communication network simulator).</li> <li>After receiving a new voltage measurement, the controller enters its dead time (<math>\Delta t_{\text{ctrl}}</math>) and becomes unresponsive.</li> <li>After receiving a new tap position setpoint, the tap actuator enters its dead time and becomes unresponsive.</li> <li>Voltage <math>V1</math> (voltage measured at <i>Load1</i>) is expected to stay within the operational limits throughout the test. Voltage <math>V2</math> (voltage measured at <i>Load2</i>) falls beyond the lower threshold within 1 minute, and a change in the tap position (from 0 to -1) is expected to happen the</li> </ul>

	next time the meters transmit their measurements to the controller (t = 1 min).
<b>Initial system state</b>	<ul style="list-style-type: none"> <li>initial tap position (at t = 0 min): 0</li> </ul>
<b>Evolution of system state and test signals</b>	<ul style="list-style-type: none"> <li>Load1: active power consumptions ramps linearly from 0 to 2 kW within 2 minutes</li> <li>Load2: active power consumptions ramps linearly from 7 to 10 kW within 2 minutes</li> <li>For the sake of simplicity, the effect of changing a tap should become effective at the end of the OLTC's actuation deadtime (instantaneous event, no continuous process).</li> </ul>
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	2 milliseconds
<b>Source of uncertainty</b>	communication delays
<b>Suspension criteria / Stop-ping criteria</b>	N/A

#### 8.5.2.4 Test Specification JRA2-TC3.TS3\_baseline

<b>Reference to Test Case</b>	JRA2-TC3
<b>Title of Test</b>	Baseline simulation without cyber-attacks
<b>Test Rationale</b>	The outcome of this test is the reference for test specification JRA2-TC3.TS3. By neglecting ICT-related effects this represents the "idealized" case.
<b>Specific Test System (graphical)</b>	<p><u>Overview of individual simulation components:</u></p> <p><i>Attention:</i> In this test, the controller periodically calculates and transmits tap position setpoints, using that latest values available. This means, that it does not transmit a new tap position setpoints every time a new measurement arrives (as is happening in JRA2-TC3.TS2).</p> <pre> graph TD     CL1[CyclingLoad] --&gt; L1[Load1]     CL2[CyclingLoad] --&gt; L2[Load2]     L1 --&gt; LFS[LoadFlowSim]     L2 --&gt; LFS     LFS -- tap --&gt; U1[Unmetaizer]     U1 --&gt; LFS     LFS -- V1 --&gt; P1[PeriodicSender]     P1 --&gt; M1[Metaizer]     M1 --&gt; CSA[ContinuousSignalAttacker]     CSA -- V1_send --&gt; CS[CommSim]     LFS -- V2 --&gt; P2[PeriodicSender]     P2 --&gt; M2[Metaizer]     M2 --&gt; CSA     CSA -- V2_send --&gt; CS     CS -- ctrl_send --&gt; U2[Unmetaizer]     U2 --&gt; CS     CS -- ctrl_receive --&gt; DSA[DiscreteSignalAttacker]     DSA --&gt; M3[Metaizer]     M3 -- tap --&gt; CSim[ControllerSim]     CSim -- (V1, V2) --&gt; B[Bundler]     B --&gt; U3[Unmetaizer]     U3 --&gt; B     </pre>

<b>Target measures</b>	This test serves as baseline for reference for test specifications JRA2-TC3.TS3_scaling, JRA2-TC3.TS3_ramping, JRA2-TC3.TS3_random and JRA2-TC3.TS3_tap_position. As such, there is no qualification strategy for the output of this test itself.
<b>Input and output parameters</b>	<p><u>Controllable input parameters:</u></p> <ul style="list-style-type: none"> <li>tap position</li> </ul> <p><u>Measured parameters:</u></p> <ul style="list-style-type: none"> <li>voltages at loads</li> </ul> <p><u>Uncontrollable parameters:</u></p> <ul style="list-style-type: none"> <li>power consumption of loads</li> <li><math>T_{\text{sender}} = 1 \text{ min}</math></li> <li><math>T_{\text{ctrl}} = 15 \text{ min}</math></li> <li><math>\square t_{\text{sender}} = 3 \text{ s}</math></li> <li><math>\square t_{\text{ctrl}} = 0 \text{ s}</math></li> </ul> <p><u>Attacker module parameters:</u></p> <ul style="list-style-type: none"> <li>DisreteSignalAttacker=OFF</li> <li>ContinuousSignalAttacker=OFF</li> </ul>
<b>Test Design</b>	<ul style="list-style-type: none"> <li>The test takes 12 hours of simulation time, during which both loads are ramped up (see below).</li> <li>The meters send their measurements to the controller in regular intervals (<math>T_{\text{sender}} = 1 \text{ min}</math>) with a small relative delay to each other (<math>\square t_{\text{sender}} = 3 \text{ s}</math>), beginning at the start of the simulation (<math>t = 0 \text{ min}</math>).</li> <li>The controller periodically calculates and transmits tap position set-points (<math>T_{\text{ctrl}} = 15 \text{ min}</math>), using that latest values available, beginning at the start of the simulation (<math>t = 0 \text{ min}</math>).</li> <li>The transmission of data from the meters to the controller (voltage measurements) and from the controller to the OLTC's tap actuator (tap position setpoint) happens over the communication network simulator ("CommSim" component).</li> </ul>
<b>Initial system state</b>	<ul style="list-style-type: none"> <li>initial tap position (at <math>t = 0 \text{ min}</math>): 0</li> </ul>
<b>Evolution of system state and test signals</b>	<p>The active power consumptions of Load1 and Load2 ramp according to the following chart:</p> 
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	1 second
<b>Source of uncertainty</b>	communication delays
<b>Suspension criteria / Stopping criteria</b>	N/A

### 8.5.2.5 Test Specification JRA2-TC3.TS3\_scaling

<b>Reference to Test Case</b>	JRA2-TC3
<b>Title of Test</b>	Assessment of scaling attacks
<b>Test Rationale</b>	This test characterizes the impact of a scaling attack (scaling the voltage measurements) on the OLTC's actuation pattern and the voltage levels.
<b>Specific Test System</b>	See test specification JRA2-TC3.TS3_baseline
<b>Target measures</b>	<ul style="list-style-type: none"> <li>tap actuation according to baseline (JRA2-TC3.TS3_baseline)</li> <li>voltage levels are expected to be within operational limits</li> </ul>
<b>Input and output parameters</b>	<u>Attacker module parameters:</u> <ul style="list-style-type: none"> <li>DisreteSignalAttacker=OFF</li> <li>ContinuousSignalAttacker=ON</li> <li><math>\square_s = 0.01</math></li> </ul> Remaining parameters according to test specification JRA2-TC3.TS3_baseline.
<b>Test Design</b>	Basic design like in test specification JRA2-TC3.TS3_baseline. In addition, the scaling attack pattern is introduced. This means, both voltage measurements ( $V1$ and $V2$ ) are modified by the ContinuousSignalAttacker component, representing an influence of a cyber-attacker in the system and scaling them by the factor $\square_s$ . Scaling rule for the voltage measurements is $y(t) \square (1 + \square_s)$ .
<b>Initial system state</b>	<ul style="list-style-type: none"> <li>initial tap position (at <math>t=0</math> min): 0</li> <li>attacker activation time at <math>t=0</math> min</li> </ul>
<b>Evolution of system state and test signals</b>	See test specification JRA2-TC3.TS3_baseline
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	See test specification JRA2-TC3.TS3_baseline
<b>Source of uncertainty</b>	See test specification JRA2-TC3.TS3_baseline
<b>Suspension criteria / Stopping criteria</b>	N/A

### 8.5.2.6 Test Specification JRA2-TC3.TS3\_ramping

<b>Reference to Test Case</b>	JRA2-TC3
<b>Title of Test</b>	Assessment of ramping attacks
<b>Test Rationale</b>	This test characterizes the impact of a ramping attack (ramping the voltage measurements) on the OLTC's actuation pattern and the voltage levels.
<b>Specific Test System</b>	See test specification JRA2-TC3.TS3_baseline
<b>Target measures</b>	<ul style="list-style-type: none"> <li>tap actuation according to baseline (JRA2-TC3.TS3_baseline)</li> <li>voltage levels are expected to be within operational limits</li> </ul>
<b>Input and output parameters</b>	<u>Attacker module parameters:</u> <ul style="list-style-type: none"> <li>DisreteSignalAttacker=OFF</li> <li>ContinuousSignalAttacker=ON</li> <li><math>\square_r = 2.4e-4</math></li> </ul>



	Remaining parameters according to test specification JRA2-TC3.TS3_baseline.
<b>Test Design</b>	Basic design like in test specification JRA2-TC3.TS3_baseline. In addition, the ramping attack pattern is introduced. This means, both voltage measurements (V1 and V2) are modified by the ContinuousSignalAttacker component, representing an influence of a cyber-attacker in the system and ramping them by the factor $\square_r \square t$ , where t is the simulation time. The ramped signal attack pattern is $y(t) + \square_r \square t$ .
<b>Initial system state</b>	<ul style="list-style-type: none"> <li>initial tap position (at t=0 min): 0</li> <li>attacker activation time at t=0 min</li> </ul>
<b>Evolution of system state and test signals</b>	See test specification JRA2-TC3.TS3_baseline
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	See test specification JRA2-TC3.TS3_baseline
<b>Source of uncertainty</b>	See test specification JRA2-TC3.TS3_baseline
<b>Suspension criteria / Stopping criteria</b>	N/A

#### 8.5.2.7 Test Specification JRA2-TC3.TS3\_random

<b>Reference to Test Case</b>	JRA2-TC3
<b>Title of Test</b>	Assessment of the random attack pattern
<b>Test Rationale</b>	This test characterizes the impact of a random attack (adding random noise to the voltage measurements) on the OLTC's actuation pattern and the voltage levels.
<b>Specific Test System</b>	See test specification JRA2-TC3.TS3_baseline
<b>Target measures</b>	<ul style="list-style-type: none"> <li>tap actuation according to baseline (JRA2-TC3.TS3_baseline)</li> <li>voltage levels are expected to be within operational limits</li> </ul>
<b>Input and output parameters</b>	<u>Attacker module parameters:</u> <ul style="list-style-type: none"> <li>DisreteSignalAttacker=OFF</li> <li>ContinuousSignalAttacker=ON</li> </ul> Remaining parameters according to test specification JRA2-TC3.TS3_baseline.
<b>Test Design</b>	Basic design like in test specification JRA2-TC3.TS3_baseline. In addition, the random attack pattern is introduced. This means, both voltage measurements (V1 and V2) are modified by the ContinuousSignalAttacker component, representing an influence of a cyber-attacker in the system with the modification pattern $y(t) + \text{rand}(a, b)$ , where t is the simulation time and a and b are constants.
<b>Initial system state</b>	<ul style="list-style-type: none"> <li>initial tap position (at t=0 min): 0</li> <li>attacker activation time at t=0 min</li> </ul>
<b>Evolution of system state and test signals</b>	See test specification JRA2-TC3.TS3_baseline
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	See test specification JRA2-TC3.TS3_baseline
<b>Source of uncertainty</b>	<ul style="list-style-type: none"> <li>random attack pattern</li> <li>communication delays</li> </ul>

<b>Suspension criteria / Stopping criteria</b>	N/A
--	-----

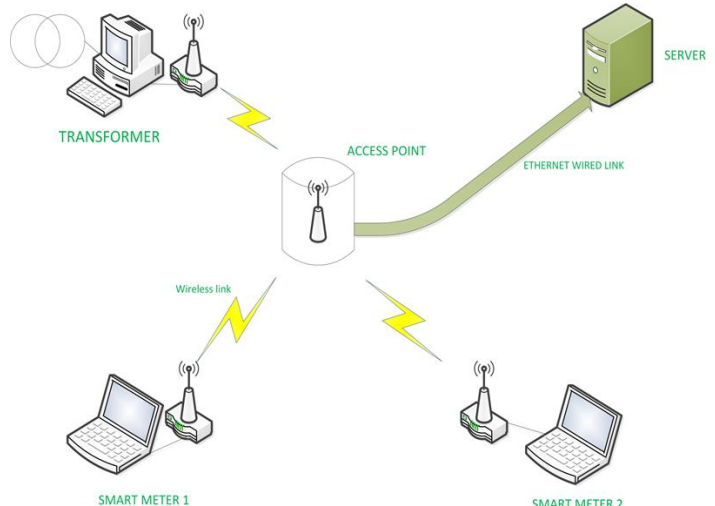
### 8.5.2.8 Test Specification JRA2-TC3.TS3\_tap\_position

<b>Reference to Test Case</b>	JRA2-TC3
<b>Title of Test</b>	Assessment of an attack on the tap setpoint commands
<b>Test Rationale</b>	This test characterizes the impact of a direct attack on the OLTC's actuation pattern, assessing the impact on the voltage levels.
<b>Specific Test System</b>	See test specification JRA2-TC3.TS3_baseline
<b>Target measures</b>	<ul style="list-style-type: none"> <li>tap actuation according to baseline (JRA2-TC3.TS3_baseline)</li> <li>voltage levels are expected to be within operational limits</li> </ul>
<b>Input and output parameters</b>	<u>Attacker module parameters:</u> <ul style="list-style-type: none"> <li>DisreteSignalAttacker=ON</li> <li>ContinuousSignalAttacker=OFF</li> </ul> Remaining parameters according to test specification JRA2-TC3.TS3_baseline.
<b>Test Design</b>	Basic design like in test specification JRA2-TC3.TS3_baseline, but with a total simulation time of only 5 hours. In addition, an attack pattern is introduced where the voltage measurements remain unmodified, but rather the OLTC tap position setpoints sent from the controller are intercepted and modified directly. This means that the voltage measurements sent to the voltage controller have the correct values and the controller will always calculate the correct setpoints based on the correct input. However, the attacker is assumed to be able to intercept the OLTC setpoint commands and modify them by decreasing their current value by 1. As a result, the voltages rise until the systems becomes unstable.
<b>Initial system state</b>	See test specification JRA2-TC3.TS3_baseline
<b>Evolution of system state and test signals</b>	See test specification JRA2-TC3.TS3_baseline
<b>Other parameters</b>	N/A
<b>Temporal resolution</b>	See test specification JRA2-TC3.TS3_baseline
<b>Source of uncertainty</b>	See test specification JRA2-TC3.TS3_baseline
<b>Suspension criteria / Stopping criteria</b>	N/A

### 8.5.3 Mapping Strategy

#### 8.5.3.1 Experiment Specification JRA2-TC3.TS1.ns-3

<b>Reference to Test Specification</b>	JRA2-TC3.TS1
<b>Title of Experiment</b>	Validation of network communication models in ns-3
<b>Experiment Realisation</b>	The experiment is implemented in the ns-3 communication network simulator.

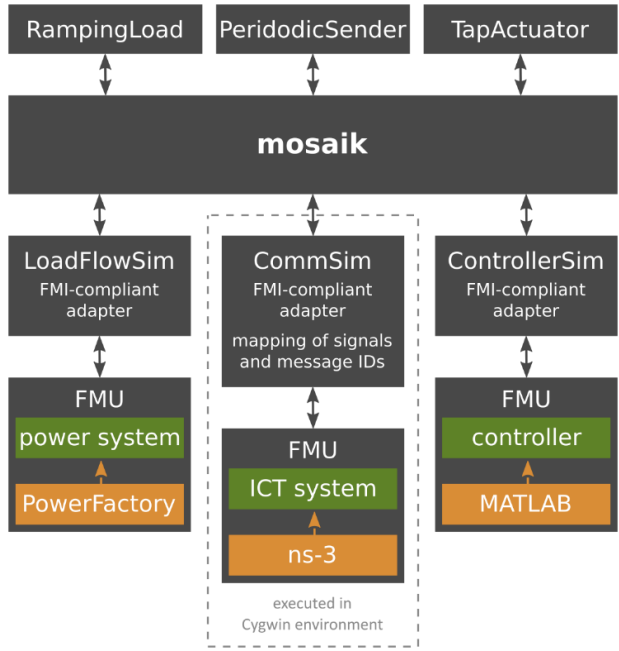
	 <p>The diagram illustrates a network topology. A central 'ACCESS POINT' is connected to a 'SERVER' via an 'ETHERNET WIRED LINK'. The 'ACCESS POINT' also communicates wirelessly with a 'TRANSFORMER' (represented by a computer monitor and tower), 'SMART METER 1' (represented by a laptop), and 'SMART METER 2' (represented by a laptop). The wireless links are indicated by yellow lightning bolts.</p>
<b>Experiment Setup</b>	<p>The ns-3 communication network simulator provides all models for simulating the data transfer via Wi-Fi and Ethernet. In addition, the following dedicated application layer models are used for simulating the nodes:</p> <ul style="list-style-type: none"> <li>• <i>Model Tc3ControllerServer</i>: This application layer model simulates the server functionality of a CVC controller device. The role of this model is the same as the role of a server in every network, i.e., it creates a socket where the smart meters connect and sends data. In addition, after a packet is received, the headers are checked, and the end-to-end delay of the transmission is calculated and stored. Later in the simulation, this delay is used to calculate the corresponding timestamp and add an event to the event queue.</li> <li>• <i>Model SmartmeterCustomClient</i>: This application model simulates a smart meter device. The model's purpose is to establish connection to the CVC's server socket and send a packet of fixed size (100 bytes). For the calculation of the end-to-end delay, the time of the packet creation is added to it as part of a header.</li> <li>• <i>Model Tc3ControllerClient</i>: This application model simulates the CVC controller's client aspect and helps to establish the connection between the controller and the OLTC transformer. More specifically, it implements a UDP client-like functionality, managing the creation and sending of packets (containing data associated with new setpoints) to the transformer. For the calculation of the end-to-end delay, the time of the packet creation is added to it as part of a header.</li> <li>• <i>Model OltcCustomServer</i>: This application model simulates the transformer server aspect and helps to establish the connection between the controller and OLTC controller. More specifically, it implements a UDP server-like functionality, receiving the packets (containing data associated with new setpoints) sent by the controller. Here the header, containing the time the packet was sent, by the controller is retrieved and the end-to-end delay is calculated.</li> </ul>
<b>Experimental Design and Justification</b>	<p>This is a basic ns-3 simulation using the dedicated application layer models implemented for this test case.</p>
<b>Precision of equipment</b>	<p>N/A</p>
<b>Uncertainty measurement</b>	<p>N/A</p>
<b>Storage of data</b>	<p>Information regarding data packet transfer is visualized and stored (graphically) via the pyViz tool. End-to-end delays are stored in CSV files for further evaluation.</p>

### 8.5.3.2 Experiment Specification JRA2-TC3.TS2\_baseline.mosaik

<b>Reference to Test Specification</b>	JRA2-TC3.TS2_baseline
<b>Title of Experiment</b>	Implementation of base line simulation in mosaik
<b>Experiment Setup</b> (concrete lab equipment)	<p>Dedicated simulation components are implemented as FMUs for Co-Simulation:</p> <ul style="list-style-type: none"> <li>Power system simulation: the power system is implemented as PowerFactory model, using consecutive power flow calculations to simulate the power system</li> <li>Controller: the algorithm for calculating the tap position setpoint is implemented as a (simple) MATLAB script</li> </ul> <p>The other simulation components and the FMI-compliant adapters are implemented in Python on top of mosaik's high level API.</p> <p>The use of PowerFactory requires this simulation to use Windows as operating system.</p>
<b>Experiment Realisation</b>	<p>The experiment is implemented as co-simulation using mosaik with a constant simulation step size of 1 second.</p>
<b>Experimental Design and Justification</b>	This is a basic co-simulation setup using the mosaik environment with FMUs.
<b>Precision of equipment</b>	N/A
<b>Uncertainty measurement</b>	N/A
<b>Storage of data</b>	The output from the individual simulation components is stored as time series data (HDF5 data format).

### 8.5.3.3 Experiment Specification JRA2-TC3.TS2.mosaik

<b>Reference to Test Specification</b>	JRA2-TC3.TS2
<b>Title of Experiment</b>	Characterization of effect of communication delays and controller dead times using mosaik

<b>Experiment Setup</b> (concrete lab equipment)	<p>Dedicated simulation components are implemented as FMUs for Co-Simulation:</p> <ul style="list-style-type: none"> <li>Power system simulation: the power system is implemented as PowerFactory model, using consecutive power flow calculations to simulate the power system</li> <li>Controller: the algorithm for calculating the tap position setpoint is implemented as a (simple) MATLAB script</li> <li>Communication network simulation: the communication network delays are simulated with the help of ns-3</li> </ul> <p>The other simulation components and the FMI-compliant adapters are implemented in Python on top of mosaik's high level API.</p> <p>The use of PowerFactory requires this simulation to use Windows as operating system. However, since ns-3 is developed for Linux operating systems, it is run in a Cygwin environment.</p>
<b>Experiment Realisation</b>	<p>The experiment is implemented as co-simulation using mosaik with a constant simulation step size of 2 milliseconds.</p> 
<b>Experimental Design and Justification</b>	<p>This is an advanced co-simulation setup using FMUs and communication delays. In order to incorporate the “randomness” of the communication delays, an ensemble of simulation runs with different random number generator seeds has to be evaluated for every considered combination of <math>\square_{t_{\text{sender}}}</math> and <math>\square_{t_{\text{ctrl}}}</math> (Monte Carlo approach).</p> <p>The communication network model uses message IDs as inputs and outputs, with a message ID equal to 0 indicating that no signal is present. 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. However, 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). Therefore, the mosaik wrapper for the ns-3 FMU implements a mapping between message IDs and signal values.</p> <p>Furthermore, all FMU wrappers and mosaik simulators are implemented such that they understand an input of type <i>None</i> as absent signal and react accordingly (e.g., remain idle in case the signal is absent).</p>
<b>Precision of equipment</b>	N/A

<b>Uncertainty measurement</b>	Since this is a computer simulation, there is no real source of uncertainty or randomness. The communication network simulator uses an integer-valued seed for its pseudo random number generator.
<b>Storage of data</b>	The output from the individual simulation components is stored as time series data (HDF5 data format).

#### 8.5.3.4 Experiment Specification JRA2-TC3.TS3.mosaik

<b>Reference to Test Specification</b>	JRA2-TC3.TS3_baseline, JRA2-TC3.TS3_scaling, JRA2-TC3.TS3_ramping, JRA2-TC3.TS3_random, JRA2-TC3.TS3_tap_position
<b>Title of Experiment</b>	Cyber-attack assessment
<b>Experiment Realisation</b>	<p>The experiment is implemented as co-simulation using mosaik with a constant simulation step size of 1 second.</p> <pre> graph TD     subgraph TopRow         direction LR         CL[CyclingLoad]         PS[PeriodicSender]         B[Bundler]         M[Metaizer]         U[Unmetaizer]     end     subgraph BottomRow         direction LR         CS[ControllerSim]         LFS[LoadFlowSim uses pandapower]         CommSim[CommSim FMI-compliant adapter mapping of signals and message IDs]         CSA[Continuous SignalAttacker]         DSA[Discrete SignalAttacker]     end     FMU[FMU standalone ICT system model]      CL &lt;--&gt; Mosaik[mosaik]     PS &lt;--&gt; Mosaik     B &lt;--&gt; Mosaik     M &lt;--&gt; Mosaik     U &lt;--&gt; Mosaik     Mosaik &lt;--&gt; CS     Mosaik &lt;--&gt; LFS     Mosaik &lt;--&gt; CommSim     Mosaik &lt;--&gt; CSA     Mosaik &lt;--&gt; DSA     CommSim &lt;--&gt; FMU   </pre>
<b>Experiment Setup</b> (concrete lab equipment)	<p>The communication network delays are simulated with the help of a simple standalone ICT system model, which calculates delays by drawing from a random distribution. This standalone model is provided as an FMU for CS.</p> <p>Cyber-attacks are simulated via the ContinuousSignalAttacker and DiscreteSignalAttacker components, which can be turned on/off and parameterized to yield the desired attack pattern (scaling, ramping, etc.).</p> <p>All simulation components, including the FMI-compliant adapter, are implemented in Python on top of mosaik's high level API.</p>
<b>Experimental Design and Justification</b>	<p>This is a co-simulation setup that can be used for part of the risk assessment of cyber-attacks, as suggested by the SPARKS methodology. It assists experts in performing such risk assessments, as the results of the simulations enable better risk perception and decisions during the risk treatment phase.</p> <p>The communication network model uses message IDs as inputs and outputs, see experiment specification JRA2-TC3.TS2.mosaik.</p>
<b>Precision of equipment</b>	N/A
<b>Uncertainty measurement</b>	Since this is a computer simulation, there is no real source of uncertainty or randomness. The communication network simulator and the random attack pattern use an integer-valued seed for their pseudo random number generator.
<b>Storage of data</b>	The output from the individual simulation components is stored as time series data (HDF5 data format).