POLITECNICO DI TORINO

Department of Electronics and Telecommunications

Master of Science

In

ICT for Smart Societies

Master Thesis

Design and Implementation of a Test Lab Setup to Perform Power Hardware-in-the-Loop Experiments

Relator

Prof. Ettore Francesco Bompard

Corelator

Dott. Abouzar Estebsari – Politecnico di Torino

Dott. Gianluca Fulli – European Commission

Student

Santi Insinga

March 2018

Acknowledgement

I want to thank my parents who gave me the possibility to study in Turin. I want to thank all my friends in Catania and in Turin for their day by day support during the Master period.

I want to thank my professor Bompard who gave me the chance to know the Joint Research Centre in Ispra. Thanks to all the Unit C3 of the JRC-Ispra that allowed me to work in a stimulating working environment. I want to thank also my friends Chiara, Francesco, Livia and Umberto with who I enjoyed all the best moments during the experience in Ispra.

I want to thank Alexandre, Mariano and Vangelis for their support in the Smart Grid Interoperability Laboratory; Abouzar, Giuseppe and Marco for having revised the thesis.

To my grandparents

Contents

Acronyms

SGAM	Smart Grid Architecture Model
HIL	Hardware-In-the-Loop
PHIL	Power Hardware-In-the-Loop
CHIL	Controller Hardware-In-the-Loop
HUT	Hardware Under Test
DUT	Device Under Test
DRTS	Digital Real-Time Simulation
RTDS	Real-Time Digital Simulator
PA	Power Amplifier
ITM	Ideal Transformer Model
DIM	Damping Impedance Model
VSS	Virtual Simulated System
ROS	Rest Of the System
GUI	Graphical User Interface
RMS	Root Mean Square
IP	Internet Protocol
NTP	Network Time Protocol
VPN	Virtual Private Network
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
OSI	Open Systems Interconnection model

LIST OF FIGURES

Chapter 1

1 Power Hardware-In-the-Loop: introduction and objectives

Which are the motivations that push us toward Power Hardware-In-the-Loop (PHIL) experiments? A Smart Grid promises to make the electrical power system more flexible and reliable. Electrical power and information seamlessly flow across an integrated infrastructure. The distribution system is changing and new actors are emerging within it. Renewable energy sources, electric vehicles and prosumers will penetrate the system and new control architectures are needed to govern the interactions and transactions. In order to assess their impact, real time simulation is a powerful mean for validating control algorithms and the new actors' behaviours in the grid. Real time simulation can be realized thanks to Real Time Digital Simulators, through which a real electrical network can be simulated at “wall clock time”. PHIL or Control Hardware-In-the-Loop (CHIL) is the means through which real hardware and the simulated network meet together. With these experiments, we can study in real time the behaviour of the prototyped hardware on a real distribution network at the development stage. PHIL is the means to validate the need for new technologies embedded into the traditional grid.

The chapter includes explanations on Real Time Simulations and the instruments used to get it, challenges and requirements to carry out PHIL experiments, examples of laboratory platforms, which are used for validation of models in Smart Grids in compliance with the Smart Grid Architecture Model.

1.1 Introduction

The traditional electrical grid is moving toward the new power system called Smart Grid. Inside a Smart Grid the power system and ICT domain work together to reach a joint objectives for the grid. The elements added with the new grid concept are control, automation, protection, sensing, monitoring, user interaction with the grid, etc. Despite the added complexity in control and protection the benefits are bidirectional flow of energy, distributed generation, loss reduction, reliability improvement and cost reduction. The Smart Grid has been modelled as a multilayer model called Smart Grid Architecture Model (SGAM). Whatever use case we want to realize, it is important to relate it with the SGAM in order to guarantee interoperability among all its layers.

The Smart Grid as such multi domains system that needs advanced design and testing methods. A faster way to validate the feasibility of new actors and scenarios at the different layers is needed. Real-time co-simulations can reproduce faithfully the behaviour of ICT and power system domains and testing them together directly in simulation. The integration and validation of renewable resource in the distribution system can be easily proved through this technique, which allows also to implement different control algorithms [1, 2].

In [3] they built a co-simulation platform through which they want to prove the power of such tool to analyse the performance of the multi domains smart grid in real-time. Co-simulation is the mean to meet all the requirements of the interoperability layers in the SGAM. An overview of different co-simulations platform is provided in [4] in order to show the effects of the communication layer with the physical layer.

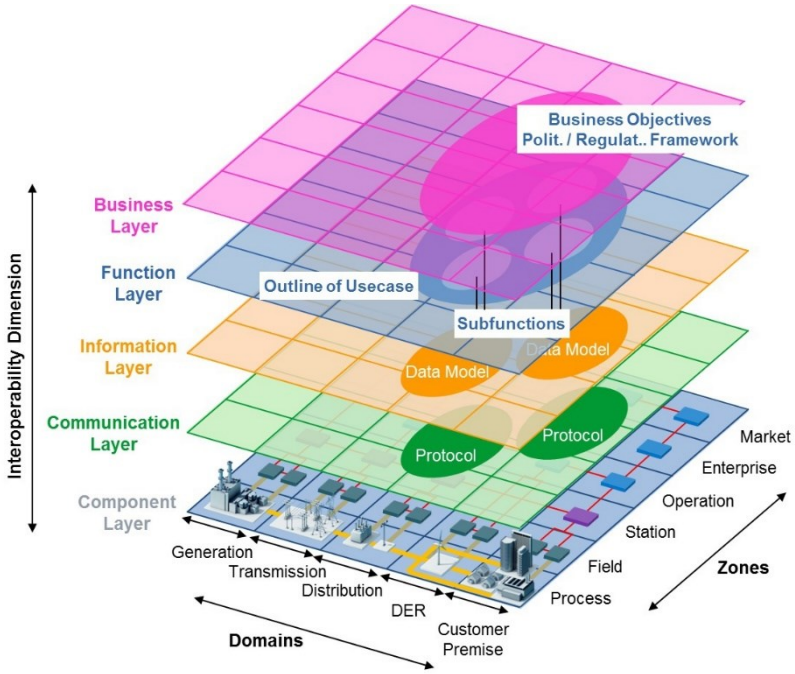

Figure 1-1: Smart Grid Architecture Model layers

1.2 What is Hardware-In-the-Loop Simulation?

Hardware-In-the-Loop (HIL) Simulation was born as a technique to develop and test embedded systems, and in particular for electrical devices with dynamic responses. The simulation is used to test the interactions of the devices within a virtual and controlled environment. Through this testing technique researchers want to analyse the output of the simulated dynamic response of the virtual system given the input of the prototype. The virtual system and the prototype are interfaced with an electronic controller allowing information exchange in real-time. The driving forces that pushed the usage of this testing technique is the

low cost of test and the flexibility on adapting to changes of the virtual or real system component [5].

1.3 Real-Time Simulation

"Digital Real-Time Simulation (DRTS) of the electric power system is the reproduction of output (voltage/currents) waveforms, with the desired accuracy that are representative of the behaviour of the real power system being modelled" [6]. This is for me the best explanatory definition of what it means Digital Real-Time Simulation. The simulation time step should be fast enough to be able to follow the dynamicity of the system we want to reproduce. This means that the time step, in the simulated system, should be close to the "real-world clock" of the real system. The quality and accurateness of the results is strictly related to the computation power performance. This technique is mostly used to study electromagnetic transient phenomenon. In DRTS the entire system is fully simulated in software. Hardware-In-the-Loop simulation belongs to the category of Real-Time Simulation. In this technique a component of the simulated system is replaced by a real hardware, such as for example an actuators, charging columns or loads. The virtual environment and the real hardware are connected through Inputs/Outputs interfaces. If the component under study is a controller, we are talking about Controller Hardware-In-the-Loop (CHIL). In CHIL control signals are exchanged between the virtual environment and the real world ($\pm 10 V_{PK}$, PK is the voltage peak). It is typically used to test control devices, such as relays. If the experiment requires a bidirectional power flow between the hardware and the simulated system, we need to use a power interface for generating the required real voltage or current. This is called Power Hardware-In-the-Loop (PHIL). It is useful to test equipment as if they were installed in the field with the real power grid. The power interface is made of a power amplifier and a set of sensors which monitor the hardware under test (HUT). The power interface is the key element in PHIL experiments. The power amplifier acts as a source or sink in order to respectively generate or absorb power. From the simulated grid it is possible to extract the values of voltage or current to control the output of the power amplifier in order to feed the HUT. If we equip the load with current or voltage sensors, we can extract the required load current or voltage quantity and they are sent as feedback to the software grid to close the simulation loop. Moreover, a high quality of the supplied voltage to the hardware under test can be obtained if higher is the frequency bandwidth of the power interface. Wider is the output bandwidth of the Power Amplifier and better the Power Amplifier can reproduce the fast voltage oscillations (at high frequency). A narrow bandwidth means that signals at higher frequency cannot be reproduced and so they are filtered out by the power interface [7, 8].

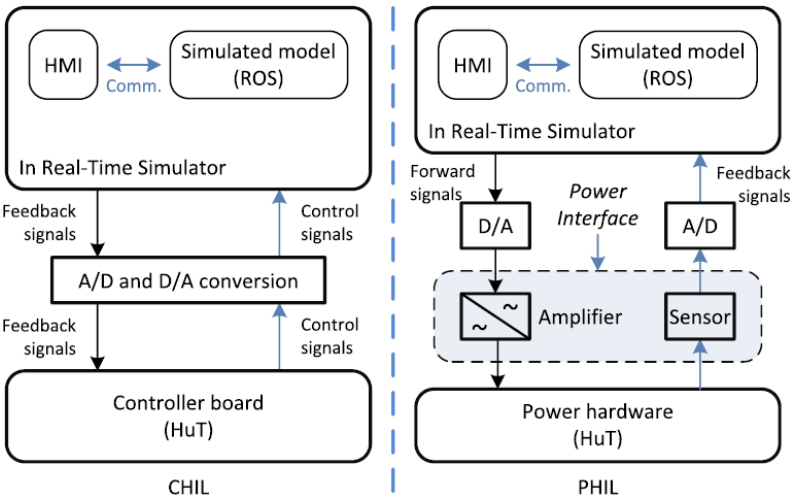

Figure 1-2: Different kinds of Hardware-In-the-Loop experiments

In the paper [8] there is a general survey on the all applications of DRTS, it can be useful as a reference to understand the requirements for each application. Usually we do not know prior the behaviour or the mathematical model of our HUT. DRTS allows us to study the prototype as a black-box and instead of modelling it, we use directly the real hardware. This is a way to characterize the behaviour of an unknown hardware. Using the real hardware is like to have a more accurate model of the system with respect to the case in which we model the device in a software, because we do not have approximation errors introduced by the model. From the simulated system we get information at fixed sampling time. The sampling time is set based on the kind of phenomenon we want to study. Within the defined time step is important that the simulator completes three tasks: "*data acquisition, computations* and *data restitution*". If these three tasks are not completed, the application is not in real-time, and it is called offline. The power interface needs to guarantee specific performances that will be better explained in the dedicated chapter of the power interface [6].

1.4 What is a RTDS Simulator?

In Hardware-In-the-Loop experiment the hardware is interfaced with a Real-Time Simulator (RTS). The RTDS Simulator is a real time power system simulator. It is used to solve the continuous real time electromagnetic transient algorithms. This simulator is widely used in the power system field. The design of the network is done in a block programming way. Each portion of the network is assigned to a rack that is made of communication and processor cards linked to each other. Each rack has a Workstation InterFace (WIF) card which synchronizes that rack and it also manages the communications with the rest of the

simulation. Nowadays the RTDS Simulator has on board two types of processor cards: Triple Processor Card (3PC) and the Giga Processor Card (GPC). The RTDS Simulator objective is to favour the exchange of a huge amount of signals from the simulator to external equipment, without affecting the time step. Usually the simulation time step for real time experiences is 50 μs, but for real time power electronics test the time step is 3 μs – 4 μs to get a certain accuracy results. RTDS can be also useful for "Large Scale Real Time Simulation", where there is the focus more on a detailed representation of the network behaviour rather than the electromagnetic transient. RSCAD is the development environment for RTDS. In RSCAD Draft we define the grid components and their interconnection; while in RSCAD RunTime we build the Graphical User Interface for the analysis of the grid status [9].

In the Smart Grid landscape HIL is useful to validate the introduction of controller, such as switchers or inverters, and their control algorithms inside the grid. It is then possible, based on the output of the device under test, to understand the overall impact and response on the simulated grid built on the RTDS software. The RTDS Technologies Company provides a useful guide for users who approach for the first time this kind of experiments [10]. The guide explains the elements on which we have to focus in order to do PHIL experiments:

- Simulation Time Step: It is the main parameter on the overall delay calculation in the PHIL experiment. It is crucial to set an accurate value in order for the solver to compute all the calculation in time, and at the same time it should be close as possible to the time variation of the signal under study. This is necessary to reduce the delay between the time step of the signal and the time step of the solver.
- Power Amplifier: it is necessary to have a four quadrants amplifier in order to generate and absorb power. It is mostly used as a voltage or current source, to be provided to the HUT. It is important to have the knowledge of the amplifier we are using. For this reason it is suggested to characterize the interface in order to obtain the required gains, amplifier behaviour, noise errors and time delay.
- Noise reduction: Because many hardware are required, and each one is a source of noise, it is also important to find a way to reduce the impact of the noise on the measurements. Filtering is the common technique, but it introduces magnitude attenuation and additional delays on the PHIL experiment.
- Interface algorithms (IA): It determines the way to realize the interconnection among the virtual environment and the HUT and also how signals are exchanged between them.

Chapter 2

2 Facing problems in Power Hardware-In-the-Loop experiments

A power interface between the hardware and the virtual system is needed in order to supply power to the hardware under test in PHIL experiments. While offering a great testing flexibility, PHIL requires careful consideration of the system stability and the results accuracy. The introduction of the power interface in the test setup creates an additional close loop, which possibly injects errors, time delay, and distortion that may cause severe instability issues or lead to inaccurate results. To counteract these problems Interface Algorithms (IAs) are used. The choice of the right IA can guarantee increased stability and more accurate results. Moreover, due to the bidirectional power exchange between the virtual system and the hardware, it becomes important to monitor the safe running of the experiment. Safety controls for the Hardware Under Test and the involved personnel are needed.

The chapter includes an explanation on the power amplification interface, the closed loop system stability and accuracy analysis, an introduction to the most popular interface algorithms, an example of an instability case.

2.1 Power amplification interface

The quality of the PHIL experiment is affected by the power interface. In the real world, the HUT and the Rest of the System (ROS) are naturally coupled; it means that they are directly connected to each other. In the PHIL experiment ROS and HUT are separated by a power interface which provides power to the HUT. The quality of the power interface depends on its transparency between the ROS and the HUT, it introduces less errors if the total system does not perceive the disjunction. For the power interface transparency it is necessary to have “unity gain with infinite frequency bandwidth and zero time delay” [11]. This is only an ideal condition and it never happens that one out of the three parameters (gain, bandwidth or delay) respects the perfect value. The real power interface will introduce errors due to its not ideal behaviour.

The power amplification interface is not only composed by a power amplifier, but it has also a set of sensors used to acquire measurements from the HUT. For this reason we say that HIL simulations are closed loop, because the response of the hardware is fed back to the simulator. The measurements may bring with themselves stability problems. The way in

which we connect the ROS with the HUT is through Interface Algorithm (IA). We use IA in order to counteract the stability problems.

The dynamicity of the power amplification interface is what introduces the instability issues into PHIL simulation. The choice of doing a PHIL experiment depends a lot from the availability of the equipment and the IA used.

There are three different types of power amplifiers: Switched Mode Amplifier, Generator Type Amplification and Linear Power Amplification Unit. The focus goes on the Switched Mode one because it is what we have in our Smart Grid Interoperability Laboratory (SGILAB).

The Switched Mode Power Amplification is a four quadrant amplifier, it is able to produce power to the HUT from the utility grid and to sink power from the HUT to the grid. The power amplifier receives a low level voltage signal from the real-time simulator, which is rationally scaled up and injected to the HUT. This kind of amplifier can control either current or voltage injected to the HUT by applying specific control algorithm. This gives higher flexibility, but on the other hand this kind of amplifier introduces higher level of time delay and lower accuracy.

For the stability and accuracy analysis of the entire system is important to characterize the whole power amplification system. It means to study the introduced delay, the bandwidth, the noise and the gain. In literature the Transfer Function for this kind of amplifier is derived from the introduced time delay and the output filter of the converter (DC/AC).

Its Laplace representation is:

$$T_{\mathrm{AMP}} = e^{-sT_d} \cdot T_f$$

Where T_{AMP} is the transfer function of the Power Amplifier, T_d is the time delay introduced by the power amplifier and T_f is the filter transfer function and unity gain [11].

2.2 Stability and accuracy problems in PHIL, usage of Interface Algorithm

In order to connect the ROS with HUT we need to use an IA. This IA introduces stability problems and the non-ideality of the entire loop introduces problems related to the accuracy of the results. Based on the topology chosen (voltage or current control) and on its parameters (value of impedances) we are able to reach stability region. The interface algorithm describes basically the kind of signal being transmitted and the way in which the signal are processed (e.g. low pass filter, etc.). In power system applications the typical signal transmitted are voltage or current.

The figures below contain the most utilized IAs used in literature [10] for the following two kinds of working operations: voltage type in Figure 2-1, where we control the output voltage of the power amplifier, and current type in Figure 2-2, where we control the output current of the power amplifier.

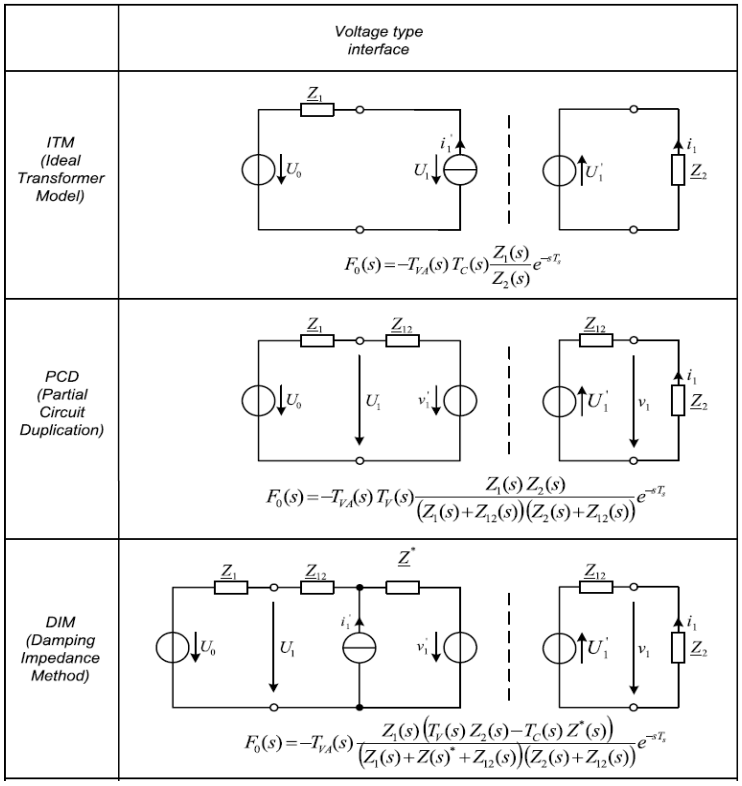

Figure 2-1: Voltage type Interface Algorithm **[10]**

For its easiest implementation, the Ideal Transformer Model (ITM) is the most used in literature. Depending on the exchanging signal, we distinguish voltage-type ITM and current-type ITM. In the Figures above are represented the transfer function for both types of interfaces. Usually the transfer function of the power amplifier is considered with unity gain inside the working bandwidth range; the impedance Z_1 and Z_2 are the equivalent impedance of the ROS and of the HUT. Depending on the type of interface, the stability depends on the ratio of the two impedances. With this interface we get a good accuracy, but we may be in trouble with the stability. To face the stability issue, we use to filter out the current/voltage measurements of the HUT.

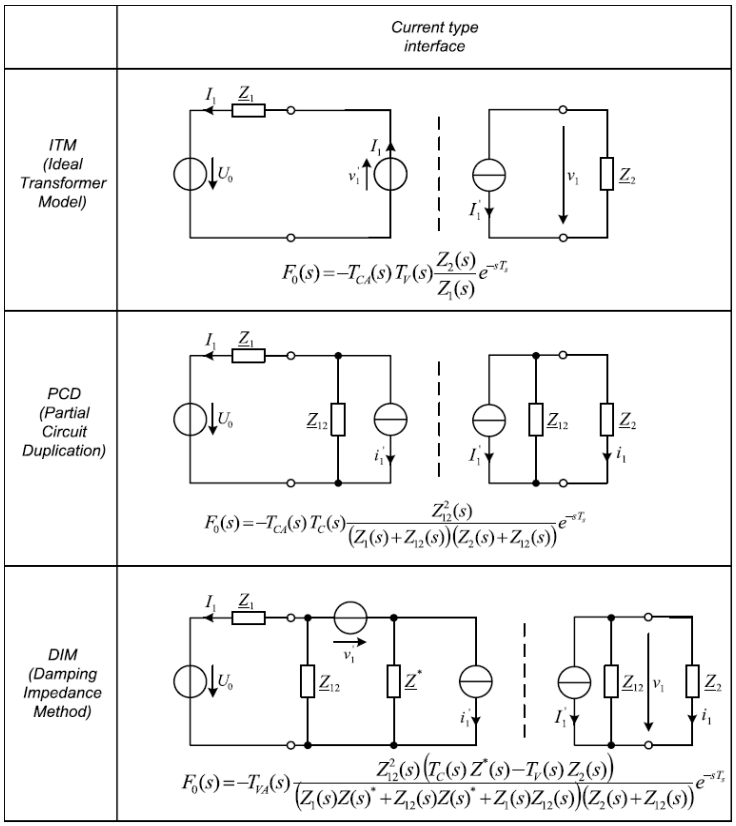

Figure 2-2: Current type Interface Algorithm [10]

Another IA is the Partial Circuit Duplication (PCD) in which we introduce in the software and hardware side the so called "linking impedance" Z_{12}. Through this impedance we solve the stability issues that we have in the ITM interface, but at the same time we increases the difficulty of the hardware implementation. In order to achieve a good accuracy, the value of Z_{12} must be higher than Z_1 and Z_2, but it is difficult to realize because we need to add more hardware in the system which add an extra impedance. For this reason this method is less used and with lower accuracy results. The Damping Impedance Method (DIM) is a generalization of the two described above IAs, where a damping impedance Z^* is introduced at the software side. Usually the linking impedance Z_{12} is so small that can be neglected. The DIM shows higher accuracy and also a higher region of stability than the ITM. The stability of the system increases if the value of the damping impedance matches the impedance of the HUT Z_2. If the two impedances are equals, the magnitude of the transfer function goes to zero and the system is completely stable. We cannot have an exact knowledge of the impedance of the HUT, but there are different techniques to have an estimation of it [12] [13]. Even if the error on the estimation of the "linking impedance" is not small, e.g. the damping impedance is three times the load impedance, the method gives a good system stability and accuracy. It has been proposed different ways to estimate the hardware impedance when the simulation is running [10, 12, 13].

The stability of the system is necessary to obtain accurate results of the simulation and to avoid the damage of the equipment. It is defined that "Accuracy relies on stability because it is necessary and sufficient condition for PHIL simulation" [11, 14]. Errors on the measurements must be low, otherwise the results will be meaningless. Ideally the power interface should not introduce time delay, but this does not happen in reality. This delay in conjunction with the computation and data acquisition time of the DRTS is the main source of instability for the system [11, 14, 15].

If we want to have a comparison among the ITM and the DIM in [10] it is proved as the DIM IA guarantees a much more stable region than the ITM IA. Moreover the ITM needs some modifications to be stable. The simplest modification in ITM usually consists in introducing a low-pass filter after the sensor's measurements, bringing more delay and an attenuation factor. However these two interfaces are the most used ones, because of their simple implementation and their stability. Looking at the accuracy both interfaces reach good results, however accuracy is improved if shorter is the delay between the ROS and the DUT.

2.3 Instability example

The instability of the HIL experiment is given by the closed loop, and it varies depending on the IAs. Due to the imperfection introduced by the power amplifier, a PHIL experiment is subject to instability.

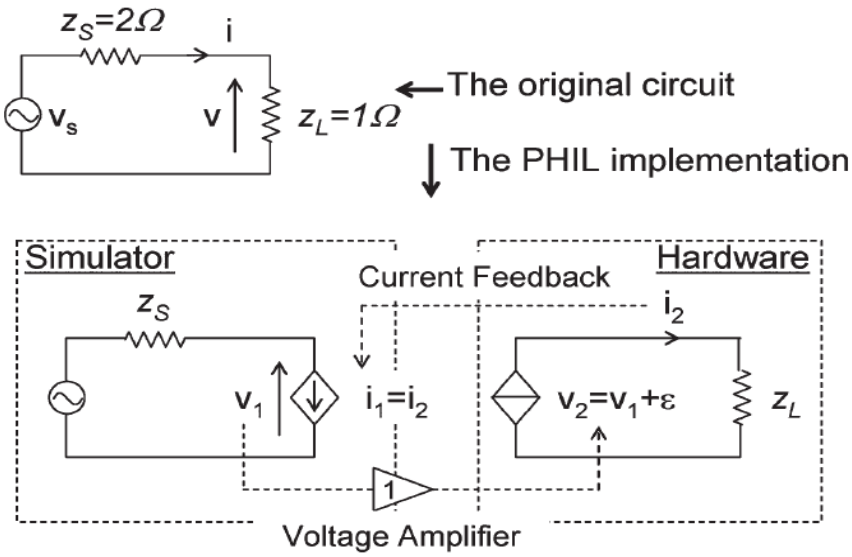

Figure 2-3: Example of ITM interface algorithm

In [14] the authors explain instability with a clear example which clarifies the concept. The original circuit is a voltage divider, which in nature it is always stable. If we want reproduce it through PHIL simulation, we can have serious instability problems. In the experiment v_S and z_S are the simulated voltage and impedance of the source generator; while z_L is the real hardware load impedance. The voltage v_1 is reproduced to the load, while the current i_2 is sent back and it controls a current generator. The reproduced voltage value v_2 contains errors due to the power amplifier; we represent the errors as ε. Imposing v_2 to the load, the current which flows is:

$$i_2 = \frac{v_1 + \varepsilon}{z_L}$$

so the error in the current value is:

$$i_2^{\varepsilon} = \frac{\varepsilon}{z_L}$$

When the current is measured and sent back to the simulator it introduces error on the value of v_1.

The value v_1 is:

$$v_1 = v_s - z_s * i_1$$

Where

$$i_1 = i_2 + i_2^{\varepsilon}$$

so the error introduced in the voltage is:

$$v_1^{\varepsilon} = -\frac{z_S}{z_L} * \varepsilon$$

At any time step the value of the voltage is amplified of a quantity $\frac{z_S}{z_L}$, so if $z_L < z_S$ the ratio is going to increase until the hardware limit. Usually to check the stability of the system we look at the open loop transfer function of the system and to check if it meets the Nyquist stability criterion. To be stable the open loop transfer function representation in the complex plane does not have to encircle the point (-1, 0) in the plane.

Stability does not imply accuracy. A possible way to evaluate the accuracy of the results is looking at the open loop transfer function influenced by the disturbance of the power interface. A smaller value for the transfer function, meaning that the magnitude is close to zero, attenuates the disturbance and so we have high accuracy [14].

2.4 Software safety level

It is always important to check the safety of the system. The first kind of software safety is the proof of the system stability. If it is possible, the model of the load should be modelled and represented in software. This mean that first of all is necessary to simulate the PHIL experiment. In this way we check the reaction of the system. If the simulated experiment gives positive response, we will be sure to implement it in the field. It is also important to protect the hardware that is used during the experiment. We have to check that the quantities under study are in between certain limits otherwise we have to react to stop the experiment. [15]. There are different techniques to implement overvoltage/overcurrent control, one is implemented in [16] and it will be used in the control algorithm of the experiment explained in section 3.5.

Chapter 3

3 Set-up and implementation of a real Power Hardware-In-the-Loop experiment

The chapter contains the description of all the experiment setup. It is described the topology of the hardware interconnection and the role of each component. A detailed description of all the devices and software used for the experiment is provided. We designed two Interface Algorithms: the Ideal Transformer Model (ITM) and the Damping Impedance Model (DIM). Before applying the real Hardware-In-the-Loop experiment, it is mandatory to check the stability of the experiment through simulation. We developed a simulated setup in Simulink, where the power interface and the hardware are modelled. Moreover, to guarantee safety operations, overvoltage and overcurrent control algorithms have been designed and implemented. The framework has been tested on real hardware with positive results.

The chapter includes the experiment setup, the different tests to prove the response of the Interface Algorithms to sudden changes.

The thesis work aims at integrating the Real Time Digital Simulator (RTDS) with the Power Amplifier (PA), and then at interfacing the virtual system with the real hardware. This is done to realize Power-Hardware-In-the-Loop (PHIL) experiments. The framework feasibility is tested on a simple circuit, voltage divider, because every network can be represented through Thevenin equivalent. In each bus the grid is seen as a voltage generator (V_S) with its equivalent impedance (R_S), while the load is just the equivalent impedance. The circuit is represented in Fig. 3-1 and it is called Natural Coupled System (NCS).

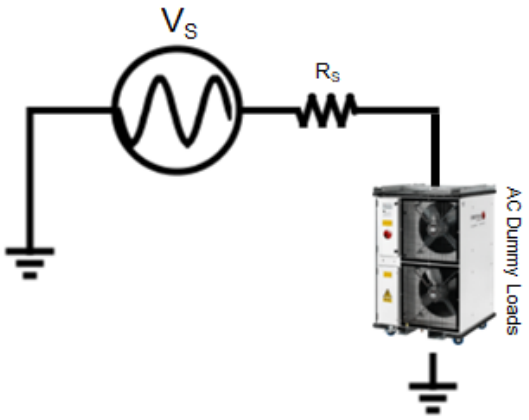

Figure 3-1: Natural Coupled System

3.1 Topology of the hardware interconnections and role of the components

In PHIL experiments we want to study the natural behaviour of a real piece of hardware when it is connected to the simulated network. In the Figure 3-1 above the Dummy Load is a three-phase 15kW fan; it represents the Hardware Under Test (HUT) and it is used only as a test for the framework. The rest of the circuit is simulated in the RTDS simulator and it is called Virtual Simulated System (VSS). The introduction of a power interface (power amplifier and sensors) is necessary for the Real-Time Simulator (RTDS) to generate/withdraw power to/from the HUT. The NCS topology changes and in between the VSS and the HUT we introduce the power interface which it is electrically connected to both systems. The voltage source and the impedance are substituted by the RTDS simulator, where they are simulated.

The new topology is represented in Figure 3-2 below.

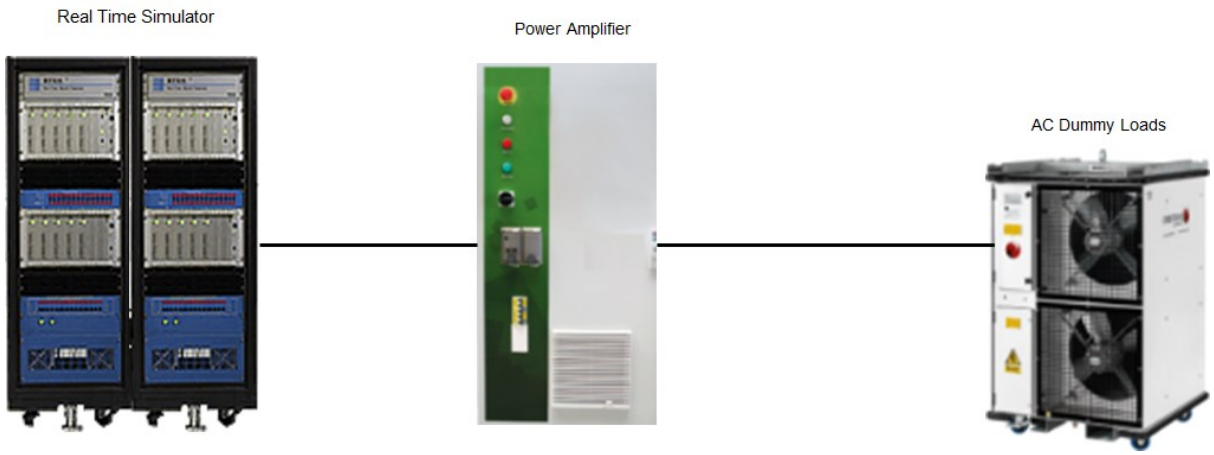

Figure 3-2: Logical topology

This is the logical topology, where are represented the hardware components and it shows how information flows from a device to another. So it means that the RTDS sends information to the PA, the PA supplies the load and vice versa.

The logical topology has a corresponding physical topology, so how the components are interconnected to each other in the field. Before explaining the new physical hardware it is reported the new topology in Figure 3-3.

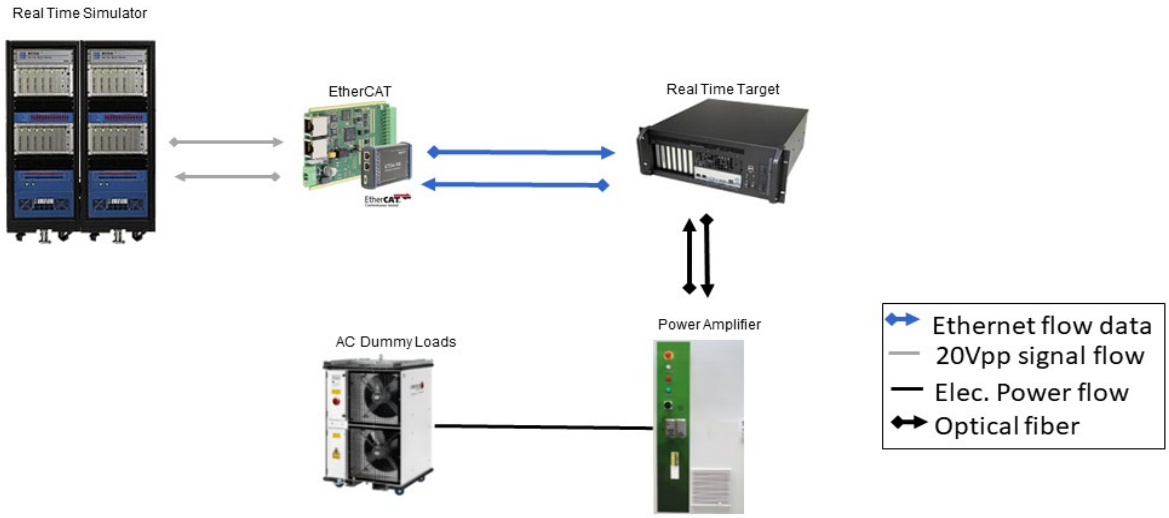

Figure 3-3: Physical topology

In the Figure 3-3 we show the hardware interconnections and also the kind of information exchanged. The power amplifier is switching mode and it is provided by Triphase. The Triphase system architecture includes:

- Power Amplifier (PA): it is defined as power module (PMx), and it is used to actuate on the HUT. It includes all the necessary sensors for measurements.
- Real Time Target (RTT): it is used for Triphase system management and control.
- EtherCAT: it is not a component made by Triphase, but it has been provided by them to translate electronic signals to digital. It acquires a $\pm 10 V_{PK}$ analogue signal and it sends the data through Ethernet cable to the RTT. The RTT will use this information to set the output voltage of the PA.

The PHIL experiment is well represented in the block scheme below. The simulated voltage or current (e.g. $400 V_{RMS}$ Line-Line) from the RTDS is scaled down by the digital to analogue output card to accepted electronic levels $\pm 10\ V_{PK}$.

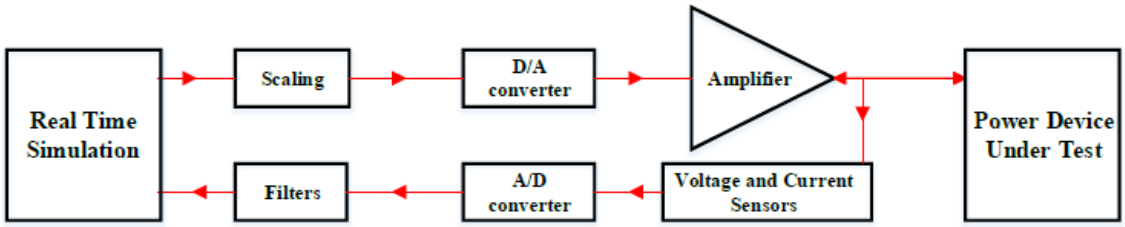

Figure 3-4: Block scheme of PHIL **[33]**

The power amplifier uses this control signal to generate an output voltage properly scaled at the operating voltage/current level of the HUT. The sensors acquire current or voltage and fed back these signals to the RTDS passing through an analogue to digital converter.

3.2 Detailed description of devices and software (RSCAD & Simulink) used in the lab for the experiment

The VSS has been implemented in RTDS Simulator. The RTDS Simulator has been designed to perform electromagnetic transient simulations in real time. Its operational frequency range is from DC to 3 kHz and this allows us to analyse very dynamic systems. The simulation time step can vary from short time interval values such as 1-4 μs to common values like 25-50 μs. The RTDS Simulator behaviour is programmed with an "all-in-one software" called RSCAD. RSCAD is a user-friendly code block software. The software contains modules that allow designing the simulation circuit and then a dedicated module with which we can create a Graphical User Interface (GUI) to run and to analyse the simulation. The two most important libraries in RSCAD are the Power System Component Library and the Control System Library. The Power System Component Library contains all the basic elements of an electric power system; while the other allows to create customized control system that interact with the model power system and/or with the external world. With the Control System Library we can configure the Input/Output card of the RTDS Simulator. The Gigabit Transceiver Analogue Output Card (GTAO) is used to interface analogue signals from the RTDS to external devices. The GTAO card has twelve output channels with an output range of $\pm 10\, V_{Pk}$ with a generation speed of $1 \mu s$. The circuitry introduces a high frequency noise with a $\pm 20\, mV_{Pk}$. From the RSCAD it is possible to set the scale factor for each output channel, so that each floating point input value is scaled down properly and sent in output. The Gigabit Transceiver Analogue Input Card (GTAI) is used to interface analogue signals from an external device to the RTDS Simulator. The GTAI card has twelve differential analogue input channels with an input range of $\pm 10\, V_{Pk}$. All the channels are sampled synchronously with sample time 6 μs. Each input channel has an anti-aliasing low pass filter with cut-off frequency 10.1 kHz or 84.2 kHz; they are setted manually in the GTAI card.

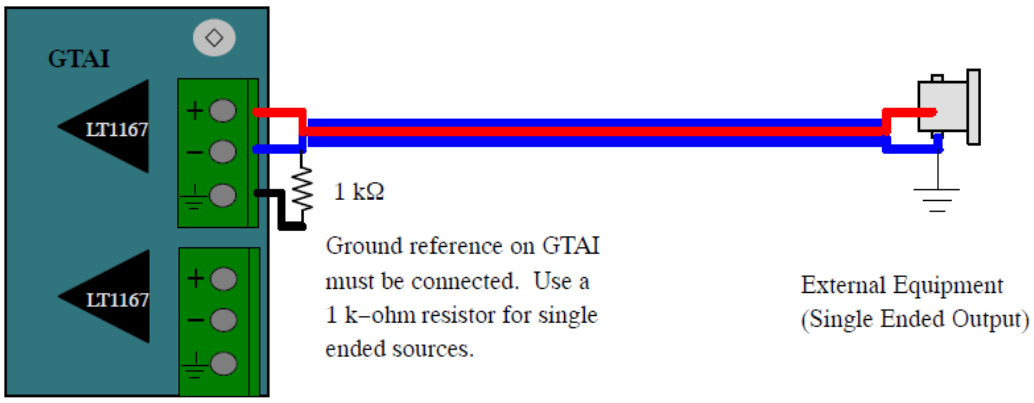

Figure 3-5: physical connection to GTAI card **[17]**

Each input channel requires three connections per channel (+ signal, - signal and ground reference). Due to the kind of input signal, we need to use a pull down resistance, such that the acquired signal is referenced by the GTAI card ground [17].

In Figure 3-6 we describe the software representation of the GTAI and GTAO cards in RSCAD software. For each card we can control up to twelve channels. Because the system under study is a three-phase only three channels are activated. In particular, each channel is one channel distance to each other so that to reduce the co-channel interference that introduces errors in the measurements.

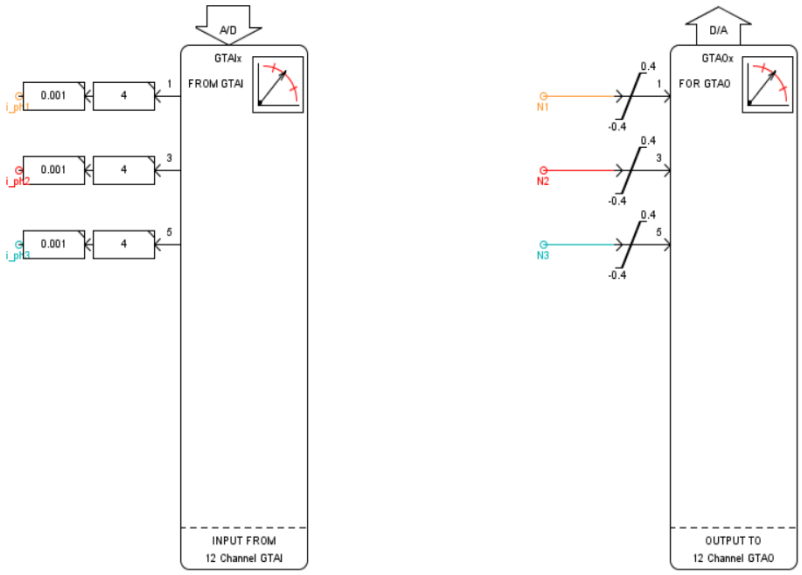

Figure 3-6: GTAI and GTAO card representation in RSCAD software

The GTAO software block in RSCAD writes input signals to a GTAO physical output channel, while the GTAI software block reads data from the analogue input channel of the GTAI card. The N1, N2 and N3 are the output of the voltage source extracted from the three-phase signal of the GTAO signal of the RSCAD. The voltage source can generate up to 0.4 kV (400 V), but a control limiter (-0.4 kV, 0.4 kV), before to send the signal in output, is used to increase safety of the hardware in case of instability. The scaling factor of the GTAO is set such that for values of 0.4 kV we have an electronic voltage value of 10 V in the RTDS. During the configuration of the GTAO card, we have to set for which value of the source voltage generator the output signal voltage is 5 V. So, we set that the value is 0.2 kV and the software automatically gets the scaling factor. With the GTAI card we want to acquire load current values for each phase. We use a scaling factor of 1, such that we have 1:1 ratio between input and output. The acquired value is scaled up by a value 4 and then we multiply by 0.001 in order to have a value in kA unit.

To summarize the GTAO send the voltage output source to the PA, then the closing loop signal which is coming back is a reference load current value acquired by the GTAI.

These Input/Output signals are received/transmitted from/to the EtherCAT terminals. The EtherCAT station, represented in Figure 3-7 below, consists in an EK1100 coupler and a group of EL3702 and EL3472 terminals. The coupler connects the terminals to the Ethernet network through its Ethernet interface. The EL3702 analogue input terminal is able to acquire signals in the range $\pm 10\ V_{Pk}$ with a maximum sample rate of 100 ksamples/s. The signal acquired from the input terminal is digitalised in an Ethernet data packet and it is sent to the network. The EL4732 analogue output terminal generates signals in the range $\pm 10\ V_{Pk}$. The output terminal receives the 16 bits digital signal coming from the Ethernet network and it is converted to the corresponding electronic analogue signal. For each terminal we have two independent channels. There is a distributed clock function which allows different terminals to be synchronized to each other.

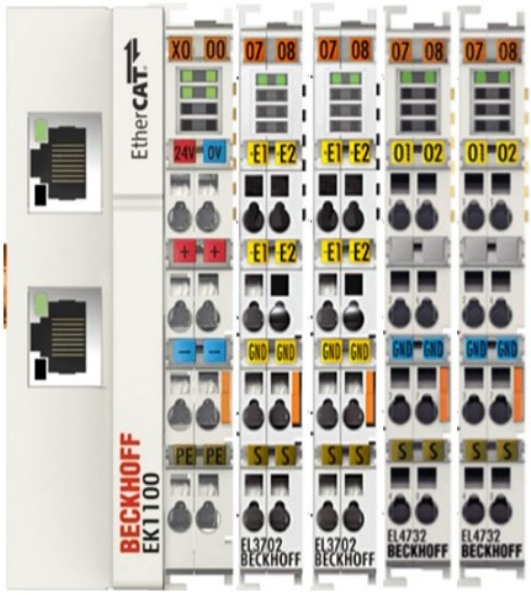

Figure 3-7: EtherCAT configuration in SGI lab **[18]**

The represented EtherCAT station shows the configuration we have at SGILAB in Ispra. In the first and second position there are the input terminals, while in third and fourth position there are the output terminals. The Triphase provided the Simulink model to configure the terminals and manage the data acquisition/transmission. The Simulink model for the EtherCAT is separated from the Power Module Simulink model. In order to allow exchanging of data in real-time among the two models we use buffers that are explained at the appendix A.1. The input/output terminals in Simulink are represented as blocks, where in each we have a couple of channels and we have to specify the position of each terminal in the EtherCAT station. In Figure 3-8 we show the blocks with the corresponding input (black) and output

(red) terminal channels. In particular the first block is the input terminal and we acquire a different signal per channel. The second block is the software representation of the output terminal, where in each channel we send a different signal.

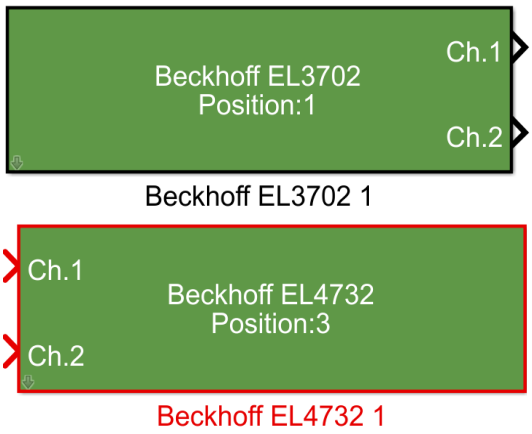

Figure 3-8: software representation of input/output terminals in Simulink

Because of the signals are in the range $\pm 10\ V_{Pk}$, we have to properly scale up or down the data, also to avoid to break the hardware. The sample time acquisition of the input terminal is $6.29 \cdot 10^{-4} s$ and it is has been set in the configuration file by the manufacturer. With the signals we acquire from the RTDS Simulator, we want to control the output voltage of the PA. The manufacturer provided a Simulink library to program the behaviour of the PM90I30F42 power amplifier. The PA can generate a power up to 90 kW. It is suitable for a bidirectional three phases energy flow between the supply grid and AC/DC loads and it has two outputs in DC and one output in AC. Depending on what the parameter that we want to control, current or voltage of the PA, we have to configure the output filter respectively LCL or LC filter.

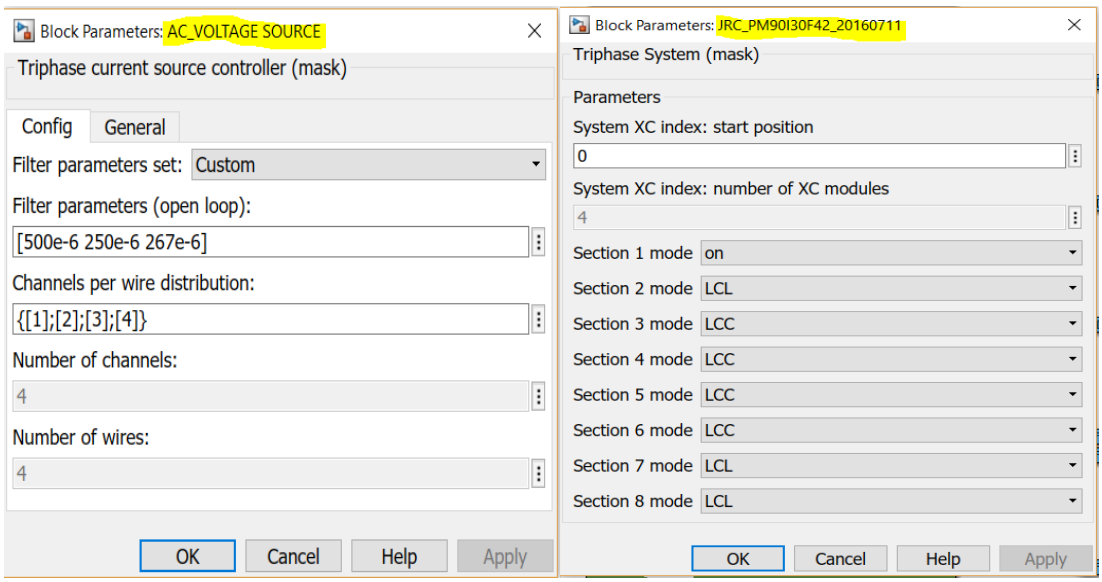

Figure 3-9: block parameter configuration

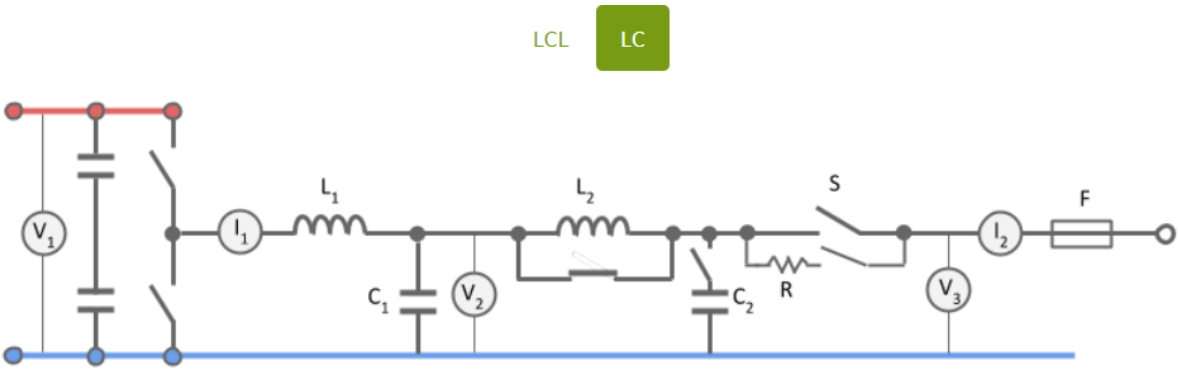

Figure 3-10: one phase output leg **[19]**

The values for the inductor, capacitor and resistor are in the "Schematics" datasheet provided by Triphase [19]. When we have the LC filter, the L_2 is bypassed and the C_1 and C_2 are summed up to each other. Moreover, each output leg has a set of sensors which allow to take measurement of voltage and currents. The filter is configured by the Simulink code. In the Figure below, the highlighted names are the blocks in the main screen of the Simulink models. Double clicking the blocks we can change the parameters. In one we have to change the section mode in LCC, in particular section 3, 4, 5 and 6, of Figure 3-9, are related to the AC output of the PA, if we want to control the output voltage; in the other we set the filter parameters. The Triphase Power Amplifier can be used in standalone mode, in the command center we assign the fundamental frequency and its RMS amplitude value, moreover harmonics can be added. The PA based on this value generates set points with which we are able to control its output. The PA can work in Phasor domain or in Time domain. For PHIL experiments we need to use set points coming from the RTDS, so the Simulink model has been adapted to this scope and its description will be later. We use set points in Time domain; all the information in terms of frequency, harmonics and amplitude are inside the signal in time. The usage of set points in Time Domain reduces the complexity, otherwise we have to add some processing delay to translate the signal from Time Domain to Phasor Domain. Both Simulink models of PMx and EtherCAT are uploaded on the RTT. The RTT synchronizes and controls a high speed optical ring network, in which PMx is connected [19]. The EtherCAT coupler is connected to the RTT with an Ethernet cable. Based on how the Simulink models are configured, the RTT acquires data from the EtherCAT and it sets the output voltage of the PA and vice versa. In particular, the RTT controls the switch of the inverter such that it generates the output voltage set by the RTDS.

3.3 Description of the implemented interfaces to connected the virtual system with the real hardware in the PHIL experiment

IA implements the physical and software topology to interconnect the HUT and the VSS. The VSS is realized in RSCAD, while the Simulink software is needed to control the PA. With the motivation given by the state of the art we implement the Ideal Transform Model (ITM) and the Damping Impedance Model (DIM) IAs.

The ITM is modelled as follow:

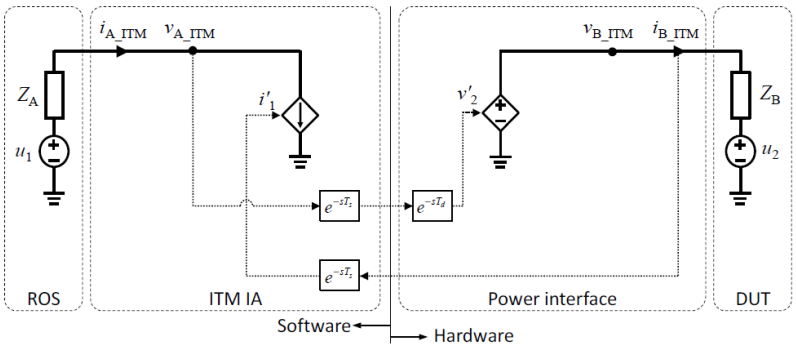

Figure 3-11: ITM voltage type IA **[10]**

The Figure 3-11 is self-explanatory and allows us to build it in the RSCAD software. The VSS or also called ROS (Rest Of the System), reproduced in RSCAD, generates the v_{A-ITM} voltage and through the ITM IA block this value is used to control the voltage output of the PA as can be seen in Figure above. Because of the time delay and other components which affect the voltage value v_{A-ITM}, the actual control value is v'_2, that is what we have in output of the power amplifier. Through the set of sensors in the Triphase Power Module, we can acquire the current value absorbed by the load. The current value is used to control a current generator, but due to the delay the value of the control current is i'_1.

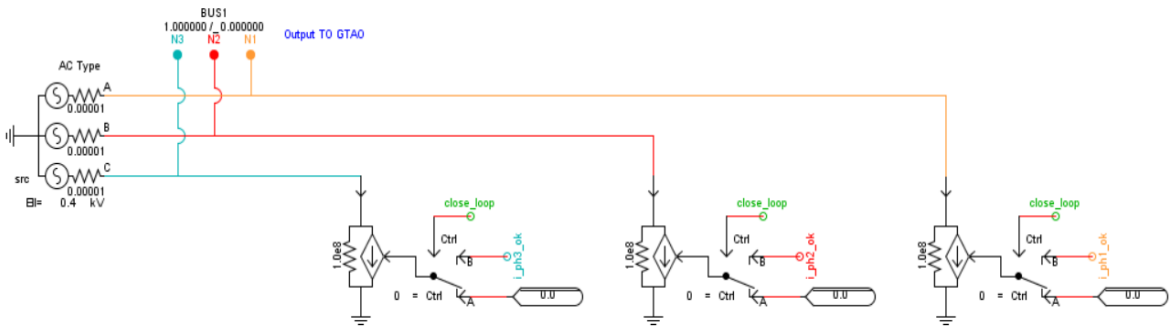

Figure 3-12: ROS implementation in RSCAD software of the ITM interface

Through a “control switch” we close the loop and the experiment can start. The “control switch” decides if the current generator has to generate the input current or it has to be 0 A. The “control switch” lets the signal A for all the three switches, coloured in red in Figure 3-12, to pass when the “Ctrl” is equal to 0; otherwise when “Ctrl” is 1 it passes the signal B for all the three switches. When the loop is closed we feed the current generator with the feedback current of the load; while if the loop is open the current value is 0 A. The current value i_ph1_ok, i_ph2_ok and i_ph3_ok are the one acquired from the GTAI card and they are “ok” because they were checked through an overcurrent control logic (this will be explained later in section 3.5). N1, N2 and N3 are the voltage values extracted from the bus for each phase and these values are sent to the GTAO card to control the output of the PA. The impedance of the voltage source is purely resistive and we set it to a very low value. The value of the source impedance is set really low with respect to the load impedance, in order to respect the stability relationship between VSS and HUT impedance in the voltage control ITM IA. Because the HUT is a fan powered at 400V, the voltage source generates 0.4 kV Line - Line RMS. The electronic signal generated by the GTAO card is acquired by the EtherCAT input analogue terminal and processed in the Simulink model. The signal is scaled up with a scale factor equal to 40, because in this way we have a voltage value up to 400 V when the electronic signal is 10 V. When the acquisition from RTDS is enabled, the PA will use the external signal as set points to generate the output voltage. In the command center in the Simulink model we set the set points of the external signal and then the signal will be routed till the PWM (Pulse Width Modulation) conversion. The PWM is the signal which controls the switches of the PA. The DIM is modelled as follow:

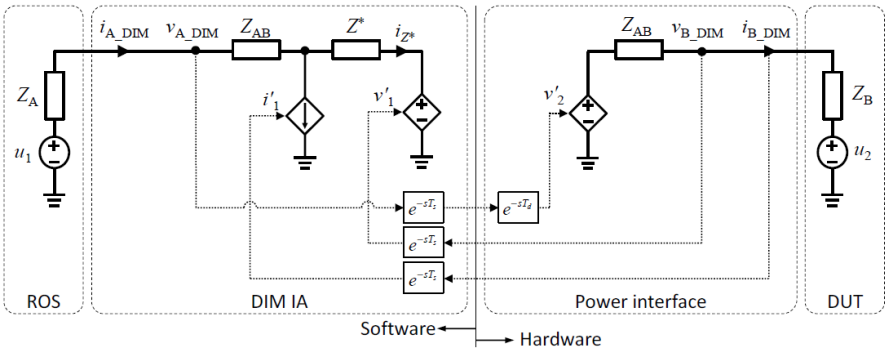

Figure 3-13: DIM voltage type implementation [10]

In RSCAD software we neglect the Z_{AB} impedance because it is usually close to 0. Differently from the ITM IA, here we add an impedance Z^* and not only the current absorbed by the load, but also the voltage given to the load is fed back to the ROS. The interface increases its stability if the impedance Z^* is close to the load impedance. As it is proved in [20], the system is stable even if the error of the impedance estimation is high. In RSCAD we reproduce the ROS as follow in Figure 3-14.

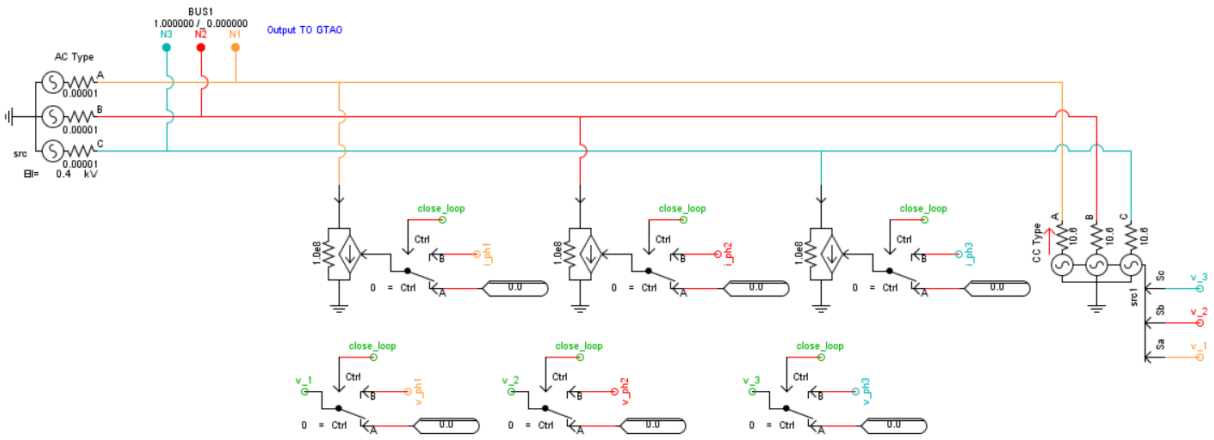

Figure 3-14: ROS implementation in RSCAD software of the DIM interface

Through a "control switch" we start the experiment by closing the loop. Because of the limited number of input channels in EtherCAT terminal, the v_{B_DIM} voltage is derived from the Ohm law, knowing the load impedance and the absorbed current.

The Z^* impedance is represented by three resistances. The load is a three phase fan. The fan is modelled in a simplified way as a resistance.

3.4 Development of new modules integrated in the simulator for first simulation trials

Before doing a PHIL experiment with a new load, it is a common way to analyse the stability of the system. This can be done via a Simulink simulation. The hardware block in each Simulink model, that controls the inverters and receives measurements, expects to have always the same kind of signals as inputs and sends back always the same data. The idea of simulation is not only to mimic this behaviour, but also to measure the delays in transmission and processing. Software simulation is a technique that helps to study the limits of the power interface, having knowledge of its stability region, but we are not able to evaluate the accuracy because it cannot be evaluated during the simulation. It is possible to obtain the accuracy of the results only when we are executing the experiment with HUT. The simulation guarantees the safety of the application and prevent possible damages to the devices that will be tested. The Triphase manufacturer provides a subsystem which has the same inputs and outputs of the real hardware block. The Simulation library provides a general purpose block. The simulated PMx block must be adapted to the Power Amplifier you have in the field and to your specific application. The model of the load must be implemented and introduced in the simulation hardware block. Currently the load is a fan, so its behaviour can be approximated to a resistance. The system is balanced and it requires the same power for each phase. In the future application the fan would be substitute with a charging column.

The charging column has the same behaviour of the fan during the charging process. When a vehicle is charging the charging column has a power factor close to one and the system is balanced.

The simulation block has been configured to contain all the pieces useful for our application, like the DC bus and the voltage source controller of the Power Module. The input of the simulated hardware contains the PWM sequence to control the inverter. The inputs are delayed spontaneously to simulate the communication between the RTT and the PA. The power module has a DC bus at 720 V, through which the inverter takes the voltage. The most important simulated block is the one which emulates the inverter behaviour.

The inverter Simulink block is the one presented here in Figure 3-15, which is a part of the annex A.2:

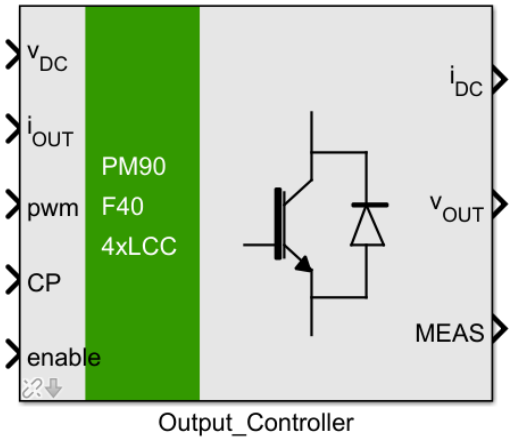

Figure 3-15: Power Amplifier inverter Simulink software implementation

Double clicking on the block we open a configuration panel where it needs to know the kind of filter configuration, the kind of PMx module and the number of output legs. The power module in the lab is a PM90, we want to use a LCC configuration for the voltage control topology and it has four output legs.

The inputs data are:

- v_{DC}: the voltage taken by the inverter from the DC bus;
- i_{OUT}: the current in output from the load;
- pwm: the PWM sequence to control the inverter;
- CP and enable: boolean data to connect and to enable the voltage output of the PM90.

The outputs are:

- v_{OUT}: the voltage to be applied to the load;
- MEAS: the bus containing all the measurements provided by the real power modules.

3.5 Implementation of control boxes for ensuring load safety

Ensuring a safe PHIL experiment operation is really important. The main risk is the instability case of the PHIL experiment. During an unstable condition the voltage and current levels increase a lot and it is dangerous for the HUT, the Power Amplifier and all the Input/Output cards, if the electronic signal exceeds their threshold. So, it becomes of great importance to enrich the hardware with software protection. The software control blocks are realized in RTDS and also in the Power Amplifier Simulink models, they sense when the value is over a maximum limit and they avoid sending in output unstable quantities. The control algorithm implemented in RSCAD decides the current values of the variables i_ph1_ok, i_ph2_ok and i_ph3_ok that are presented in the Figures 3-12 and 3-14.

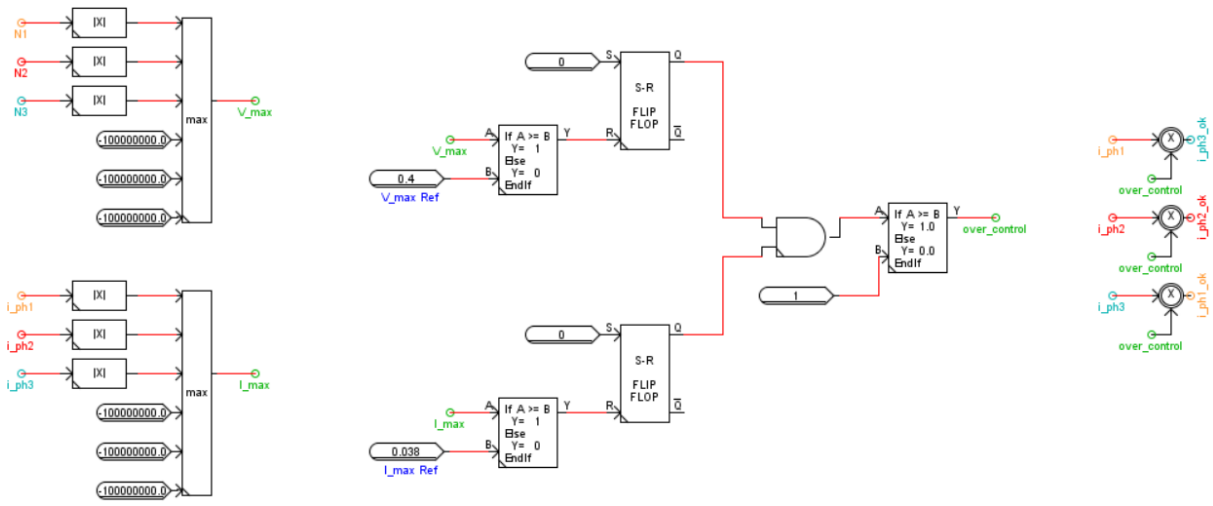

Figure 3-16: Software protection in RSCAD software

The parameter "over_control" is a control variable which can assume a value of "1" or "0" coming out after a decision. The variables i_ph1, i_ph2 and i_ph3 are the current values acquired by the GTAI card and they are multiplied by the control variable. If the "over_control" variable is "0" we do not let the unfeasible current value to go through and to feed the current generator in the Interface Algorithm. The "over_control" value is decided by an if-else condition. The if-else condition depends on its inputs A and B. The input B is always "1", because it sets a threshold. If input A is "1" the variable "over_control" is "1" and the current in output is what we are acquiring from our HUT; otherwise if A is "0" the variable "over_control" is "0" and we filter out the false input value. The value of A is given

by the output of an AND logic port. The AND port inputs are given by the results of an Set-Reset Flip-Flop. The Flip-Flop is a register, it is used to maintain the status “0” or “1” of the output variable Q. We set the initial condition Q_0 equal to “1”, such that initially the inputs of the AND port are both “1” and the system is in normal condition. We need to use only a specific condition of the Truth-Table of the Flip-Flop, in particular when the S is always equal to “0”.

Table 3-1: Truth-Table of the S-R Flip-Flop

Set	Reset	Q
0	0	Q_{i-1}
0	1	0

The status of the input R depends on the result of an if-else condition. The inputs of the if-else condition are the value of current and voltage under study and they are compared with a reference value. The reference value of voltage and current are the one at which the system can be subject to. The working idea of the combination of if-else condition and flip-flop circuit is the same for both voltage and current value. We describe only the case for voltage. The “V_max” variable is the maximum negative or positive voltage value among the three nodes that we probe at the Bus 1 as seen in Figure 3-12 and 3-14. Till the “V_max” is under the maximum threshold the flip-flop gives as output the initial state “1”. When the “V_max” exceeds the threshold the input R gets the “1” status and the output Q becomes “0”. Becoming “0”, the output of the AND port is also “0” and the “over_control” control variable will be set to “0”. In this way the current generator of the Interface Algorithm in RSCAD will be fed with a current of 0 A. The flip-flop is used because it maintains the error condition even if the voltage is within the limits immediately after. If the voltage is within the limits, R has “0” status and the Q output has the precedent value that was set to 0. In this way we have to stop the experiment and restart it again. We have to remind that we are studying a three-phase system, so we need to find the maximum value, in absolute value, among three quantities. Unfortunately in RSCAD there is no the way to compare only three elements, but six contemporary. We skip the problem by setting three out of six with a negative values such that it cannot influence the maximum comparison with absolute value results. For the power interface it has been modified the Simulink model provided by Triphase. The PA has already some basic safety checks. In particular we can set in the command center the current limit provided by the inverter, but it is only a protection for the Power Amplifier. It is up to the user to introduce a control logic to protect the used load. In particular it has been developed an overcurrent and an overvoltage control. The overcurrent control algorithm in Simulink is represented in Figure 3-17.

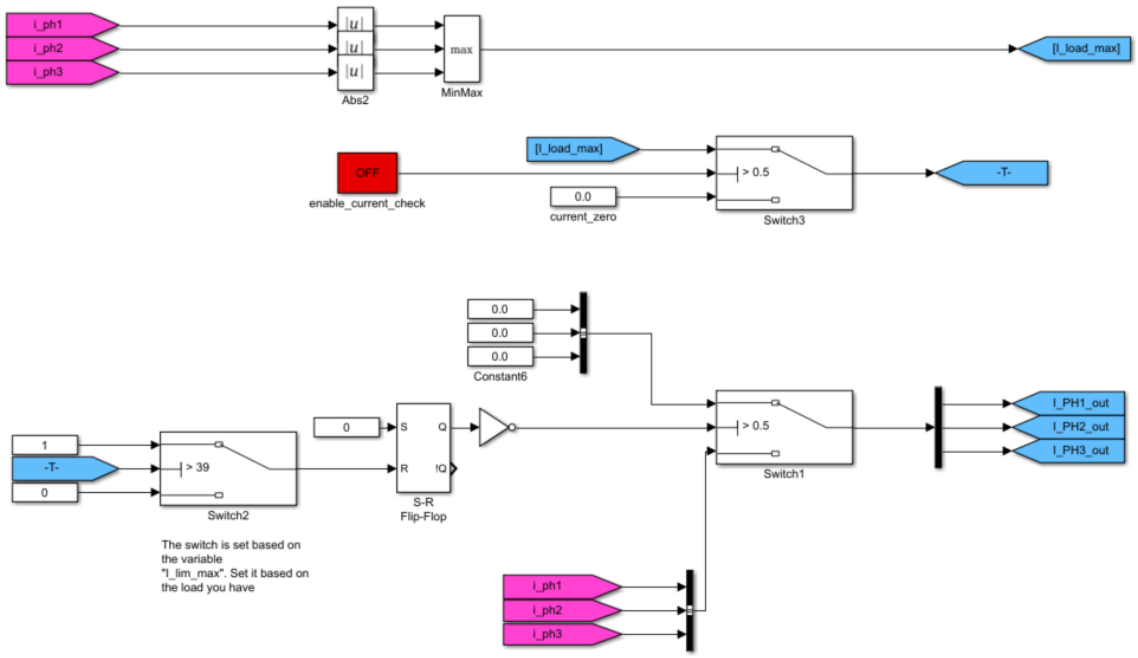

Figure 3-17: Overcurrent control algorithm

Out2, Out3 and Out4 are the current values measured by the Power Amplifier which corresponds to the current absorbed by the load. We take the maximum current absolute value and we set the flag “I_load_max”. We compare this value with the current hardware limit, in this case 39 A, we suppose that our load can support up to this value. If one of the measured values is higher than the threshold the output switch is set to “1”; otherwise it is set to “0”. The flip-flop circuit follows the same rule of the one described before and the initial value Q_0 is set to “1”. When Q is set to “1” the current in output is the one acquired from the sensors, otherwise we send a current of 0 A. I_PH1, I_PH2 and I_PH3 are the values of current sent in output to the RTDS. The values of the thresholds for the switch and for the limiters have been set in a file called “variables.m” in Matlab. In this way the code does not need to be changed and we modify the file, variables.m, depending on load we are using. The new variables added in the file are: “I_lim_max”, “V_lim_max” and “lim_out_Eth”. The first two variables are the maximum current and voltage supported by the load. The third variable is a limit of current associated to the maximum value of the EtherCAT output terminal. Before closing the loop we have to enable the button “enable_current_check”. This button enables the described control algorithm and it is needed to prevent the inrush current (load start up current).

The last implemented control algorithm implemented is the overvoltage one. It is represented in Figure 3-18.

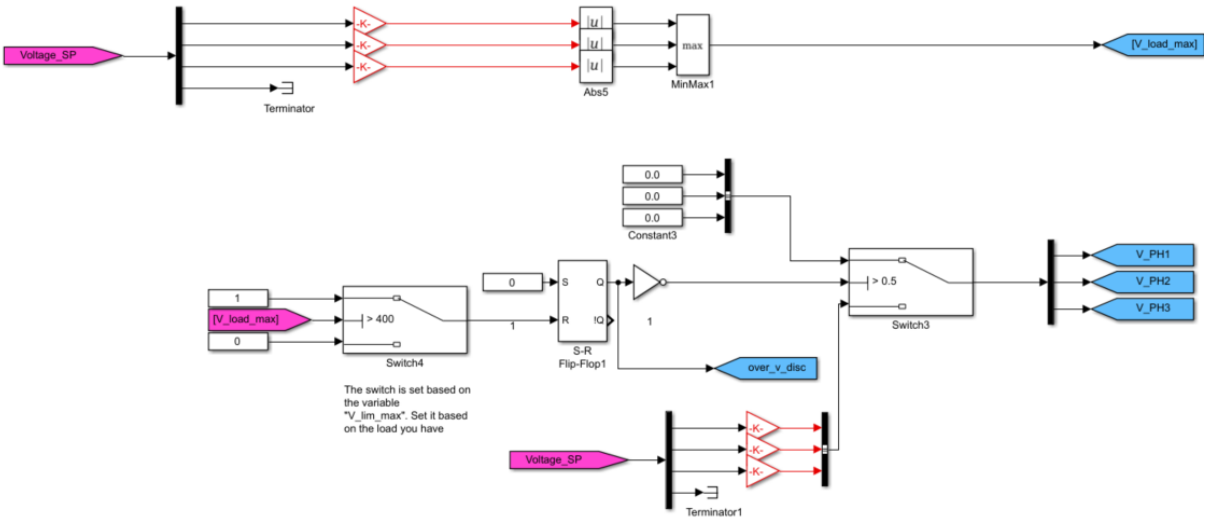

Figure 3-18: Overvoltage control algorithm

In1, In2 and In3 are the voltage values acquired from the RTDS and they are used to control the output voltage of the power amplifier. The three values are electronic signals, so they are scaled up to reach the right operational level. The applied control logic is the same as described for the overcurrent control, with the difference for the values of the thresholds. In case of instability we disconnect the hardware from the power amplifier's output.

3.6 Stability analysis and verification of the well-functioning of the built interfaces

The well-functioning of the developed Interface Algorithms has been tested in several simulations at the SGI lab. The important thing to check is that the behaviour of the interfaces respects what we already know from the state of the art [20]. The simulation is the proof that the framework is valid and it can be used in the next laboratory projects. For each simulation experiment we simulate a PHIL test. The load and the interconnection between the Power Amplifier and the load are emulated in Simulink. Once again the load consider is the fan load, observable in Figure 3-2, with a resistance of 10.6 Ω. The choice of the load derives from the HUT available in the laboratory for this kind of experiment. The HUT is a fan that can be approximated as a resistance. Moreover we are simulating a case in which the system is balanced, so the load requires the same power for each phase. The Triphase system architecture is the power interface, where the PA acts as a voltage generator and we use the internal current sensor to provide the external output signals. This power interface is simulated in Simulink. Both the interfaces, the ITM and the DIM, are tested and it is shown their behaviour in normal condition and in unstable condition. The simulators run at the same time step for each simulation. The RTDS has a simulation time step of 50 µs, while the Triphase architecture has a time step of 62.5 µs. The entire simulation is constrained by the slower time step, but it is suitable for real time applications.

3.6.1 ITM interface in stable condition

The interface is realized as in Figure 3-11. It is a voltage type and the stability condition requires that the load impedance is higher than the software impedance. The working operation is represented in the command center in RSCAD, in Figure 3-19.

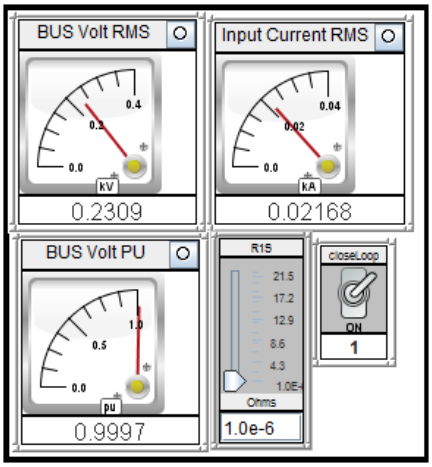

Figure 3-19: GUI in RunTime

In the front panel we have the switch and it is set to "ON" mode, so the loop is closed and the simulation can start. The values shown in the meters are the ones reached by the system in steady state in a stable situation. The bus voltage is the one generated in RSCAD and used to control the PA; the value shown at the "BUS Volt RMS" meter is a voltage Line-Neutral RMS equal to 231 V. The input current, observed in "Input Current RMS" meter, is the one absorbed by the load that we are acquiring to control the current generator in RSCAD. The meter "BUS Volt PU" for the bus voltage in p.u. unit is added to check any bus fluctuation when the load is connected to the generator. The stability is guaranteed because the software impedance (R1S) is set to 10^{-6} Ω and it can be controlled through a slider. The other Figures 3-20 and 3-21 are shown to prove the stability of the experiment. The Figure 3-20 shows the current acquired from the load. The "i_ph1_ok" is the current acquired from the GTAI card, scaled up and it has been checked from the overcurrent control algorithm. "INORT1A" is the actual current which flows in the current generator when we close the loop. The current generator is fed by the "i_ph1_ok" current. The Figure shows the moment at which we close the loop and the current starts flowing in the circuit. When we close the loop the current flowing in the current generator in red reaches the current we are acquiring in black. The loop system is stable and it is proved by the Figure because the current does not grow, maintaining its normal working operation. The Figure 3-21 shows the node voltage values. We can see that the system is stable, it is in its working operation range and the system is balanced.

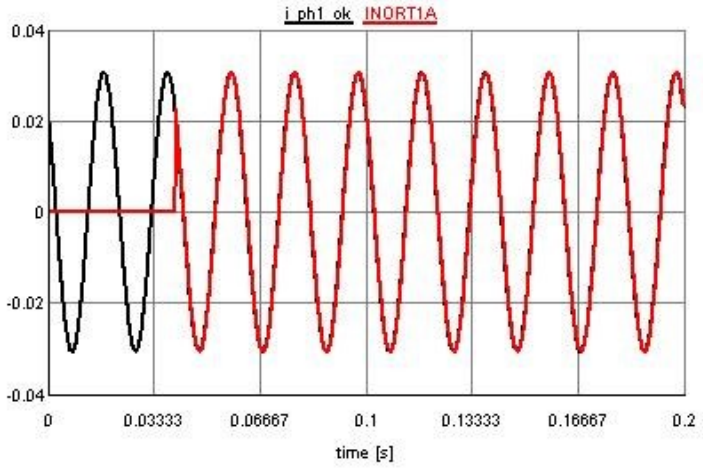

Figure 3-20: Feedback current in current generator

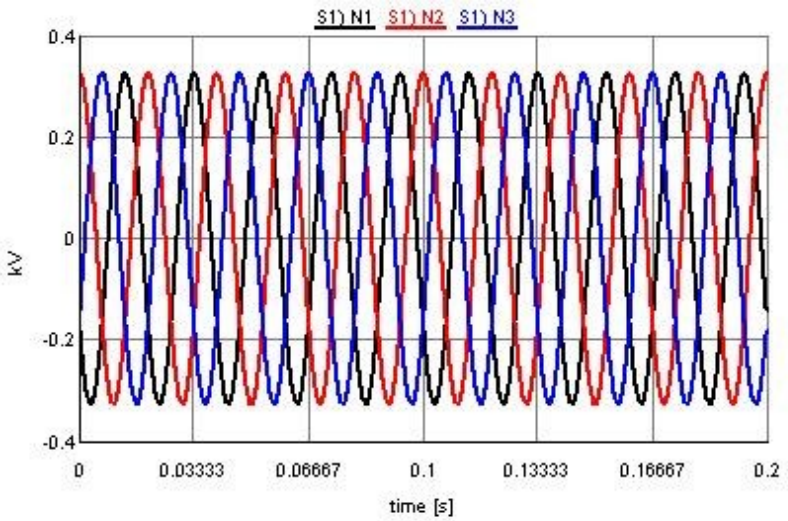

Figure 3-21: Three-phase voltage bus sent to the GTAO card

3.6.2 ITM interface in unstable condition

The instability condition is reached as soon as the load impedance is lower than the VSS impedance. In the front panel of Figure 3-22 we can see the impedance setting and the meters to check the voltage and current RMS values.

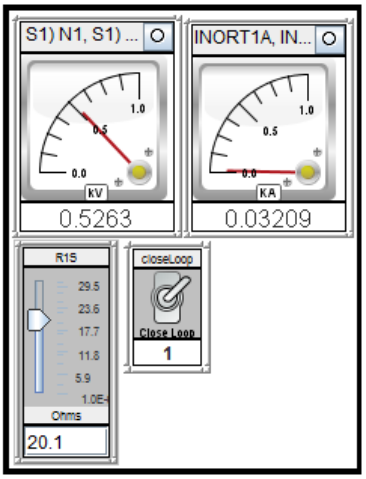

Figure 3-22: Front panel in unstable condition

The software impedance (R1S) is set to 20.1 Ω, which is almost double of the simulated load impedance. The switch shows that the loop is closed and that the interface is running. From the measurements plotted in Figure 3-23 and 3-24 we can see that the voltage and current RMS value is outside the normal range and they are continuously increasing their amplitude value. It is clear that in such an unstable situation if we would have test a real load connected to the Power Amplifier the load would have been burnt. At an initial step the source generator software impedance is set at the lowest value and the switch is open. Then we close the loop and we let the system reaching its steady-state condition with stability. At a certain point in time, we increase suddenly the software impedance and we are in an unstable condition. As shown in the Figure 3-23 and 3-24 this sudden unstable condition is translated in a sudden and continuous growth of the current and voltage value. This behaviour is explained and demonstrated at section 2.3 of the previous chapter. These two tests have been done to check the well-functioning of the ITM interface implemented in RSCAD presented in the section 3.6.1. They work perfectly; they respect the results and the operational ranges as demonstrated in the article [20].

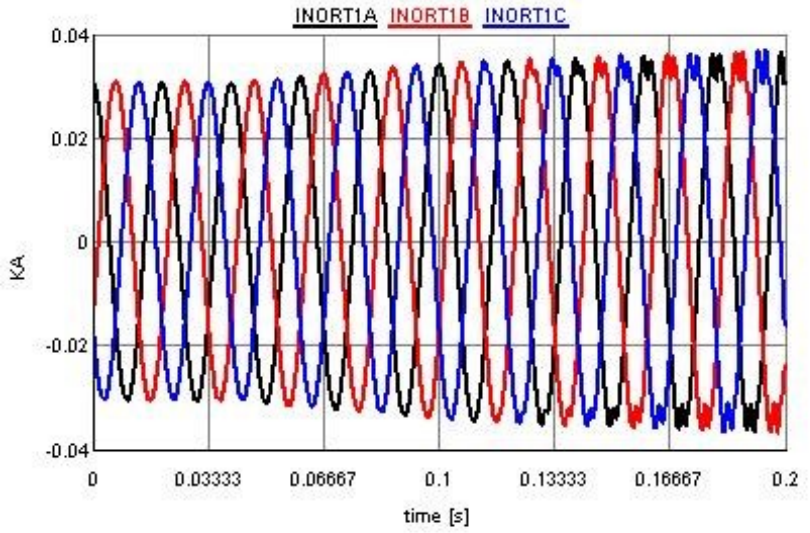

Figure 3-23: Increasing load current

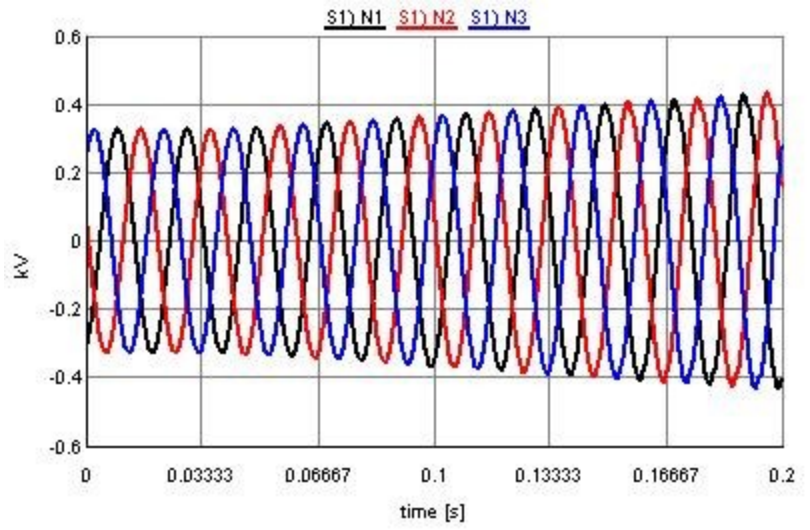

Figure 3-24: Increasing voltage bus generator

3.6.3 DIM interface stability test

The DIM interface introduces a damping impedance at software level and its implementation follows the Figure 3-13. It is more stable than the ITM interface. The interface is absolutely stable when the damping impedance matches perfectly the load impedance. Usually the knowledge of the load impedance is difficult to know it before the simulation can start, but if it is known or it can be derived, it is a very strong tool to counteract the presence of unstable condition due to load impedance fluctuations. In RSCAD we built a panel through which we can control the software generator impedance and the damping impedance. The panel is shown in Figure 3-25.

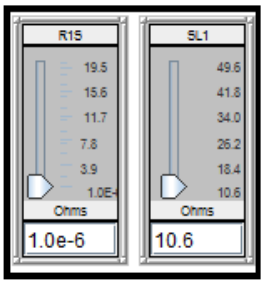

Figure 3-25: Impedance controllers

The damping impedance is controlled by the slider "SL1" and it is initially set equal to the load impedance of the fan which corresponds to 10.6 Ω.

The first test in Figure 3-26 is done by setting the software voltage generator impedance to 20 Ω to create instability in the system. This should bring to an unstable condition, but thanks to the damping impedance the system maintain its stability. Of course, when the R1S impedance is increased suddenly we have a reduction in voltage level as shown in Figure 3-27. Thanks to the damping impedance we may have simulations with an interface in voltage-type where the software impedance can be higher than the load impedance, but even this ratio as a limit. As proved by [20] the software impedance can be at maximum six times higher than the hardware impedance, when the damping impedance is not perfectly equal to the load impedance.

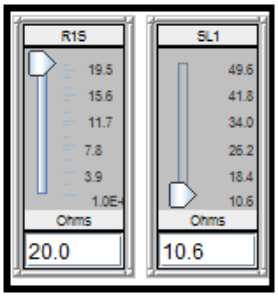

Figure 3-26: Increasing software impedance

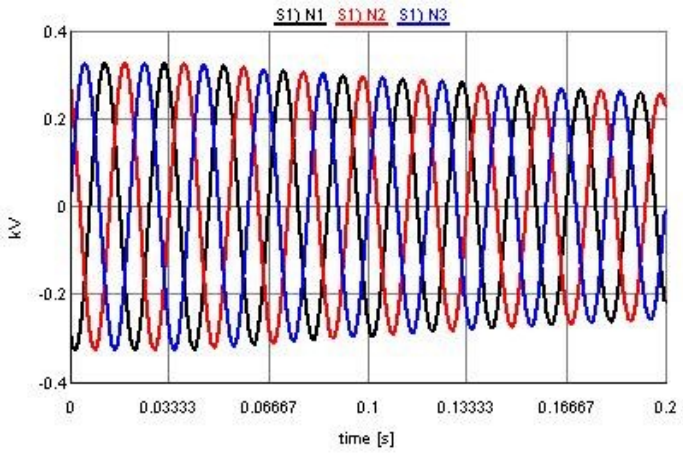

Figure 3-27: Voltage drop due to the increasing of the software impedance

The last test in Figure 3-28 wants to verify that the system is not stable anymore when the damping impedance is very far from the actual (10.6 Ω) value. The damping impedance is suddenly increased to a value five times higher than the real value.

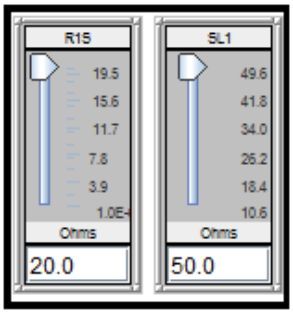

Figure 3-28: Increasing damping impedance for instability

In this condition the system is inside the instability region and the voltage level grows as we expect.

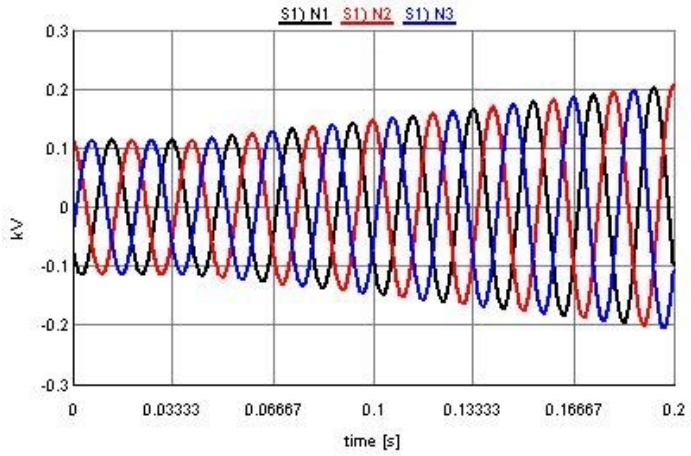

Figure 3-29 Unstable behaviour of the DIM interface

3.7 Implementation on real hardware

The setup stability has been tested on a TROTEC TDS 75 Electric Heater. The heater acts as a variable load with active power requirement from 5 to 15 kW. The load is supplied by the power amplifier which is controlled by the RTDS. The power hardware-in-the-loop test experiment has been successfully achieved. The close loop system is stable and the control algorithm manage the inrush current.

Chapter 4

4 Communication challenges and requirements for multi-site real-time co-simulation and remote hardware-in-the-loop

The electric power system is becoming smart and integrated with renewable generation and storage. Testing such a complex system requires large computational capabilities, availability of control algorithms, models of the different actors in the grid and real hardware, which impacts on the simulated environment to be tested. Different setups should be put together such that it is possible to get the best performance from each one. To make feasible such a kind of complex and large system setup, geographically distributed real-time simulation has been under studying for many years to enable real-time connection of laboratories. Internet-Distributed HIL has to face up two challenges: Interface Algorithm and communication issues introduced by Internet.

After a brief introduction of the two important challenges, the chapter focuses on the communication problem. Internet is the medium to transmit the information among the labs and I explained the Internet network and its characteristic and issues in a dedicated section. A literature review on distributed co-simulation was conducted containing examples from game applications to PHIL for power electric applications. Then in order to interconnect laboratories, an interface platform must be developed. I described which are all the requirements and open challenges that this platform has to face.

4.1 Introduction to geographically distributed co-simulation challenges

The importance of geographically distributed co-simulation is explained in section 4.4. In this section we want to list all the different challenges to face with this kind of application. The challenges are:

- System decoupling
- Interface Algorithm in Power Hardware-In-the-Loop case
- Communication Interface

The whole system has to be decoupled and each subsystem runs on a different simulator. The actual difficulty is to find out the right decoupling point. The point where we separate the system can affect the simulation quality results.

Whenever we perform Power Hardware-In-the-Loop experiment in a co-simulation setup we have to take into account the power interface. The Interface Algorithm manages the issues introduced by the power interface as already described in the chapter 2. The choice of the Interface Algorithm is closely linked with the application and a wrong choice can lead to instability of the system even if the natural coupled system is stable. Moreover, if the system is stable it does not guarantee accuracy of the results. Study on the Interface Algorithm becomes very important for the quality of the simulation results.

The communication interface must solve all the problems given by the Internet network. Internet is used as a medium to let the different subsystems to talk to each other. The rest of the chapter is focused on this challenge. After a description of Internet and its specification, we want to provide a literature review on Internet-Distributed simulation and on the last section we list the all sets of requirements that the communication interface has to satisfy.

4.2 Concepts over Internet

Internet is the most used communication media which carries information and through which it is possible to reach resources and services. Internet is the so called network of the networks, because it is based on connected networks that can be private or public. The Internet data is split and each piece of data is stored and sent in packets. The routing of the packets over Internet is managed by a network protocol called IP (Internet Protocol)

Two entities can be connected to each other and the data packets between sender and receiver are routed in many hops within the network. Packets experience delay when they flow on the Internet. The delay between sender and receiver, in the same communication session, is not constant, but it is variable. The delay variability is called jitter. Depending on the kind of application, such as voice call over Internet for example, the user's experience can be affected by the jitter.

The main data packet delay derives from four causes and they are accounted in the total delay:

- **Propagation Delay**: this is the main source on the total delay to be taken in account. It represents the amount of time for the packet to travel from source to destination at the speed of the light ($3.00 \cdot 10^8\ m/s$). It depends only from the physical distance length between sender and the receiver and this delay cannot be reduced. Usually this the delay is in the order of the milliseconds.
- **Serialization Delay**: it depends by the speed of the computer in the network at which it stores and sends the data packets to the next hop. It is the time to convert a certain byte of data into a serial bit stream and to be sent on the channel. The order of

magnitude of this contribution depends on the transmission rate of the device and typically it is in milliseconds or microseconds.

- **Routing and Switching Latencies**: from sender to receiver the data packets hop from a router or switch to another. Every time a new packet arrives, these devices have to find the best route to forward the packet to the next hop. Within the same connection the next hop it is not always the same router or switch, but it depends on the channel condition among the two devices. This kind of delays is in the order of microseconds.
- **Queuing Latency**: it is the time for the packet waiting in the buffers of the router's output port. Usually it is the time elapsed from when the packet arrives to the time at which it is sent in output in the channel. This quantity is variable in time and it depends on the congestion in the outgoing link. It is the main contributor of the delay variability and it causes the jitter. [21]

Part of the communication delay is also introduced by mechanisms of the protocols used over the IP protocol. The mechanisms guarantee an ordered and lossless communication and this can introduce ulterior delays due to retransmissions and/or reordering packets.

When establishing a communication among two nodes in the network, it is important to know if the channel we are using has enough bandwidth to satisfy our application. The bandwidth defines the channel capacity, so the maximum flow of data in a fixed interval of time. Protocols over IP not only increase the delay, but they can also reduce the available channel bandwidth. The bandwidth reduction given by the protocols is due to guarantee fairness in transmission among all the users that are using that channel, for an equity share of the medium. Moreover those protocols, that want to give a lossless communication, retransmit the missed packet and so there is a waste of resources just to resend duplicates. [21]

Companies, institutes and other entities want to use the already established Internet network for their working scopes, instead of building an own private network. They can save money, but they need a way to solve privacy issues in a public network as Internet. Internet-based Virtual Private Networks (VPNs) are the means to provide secure connectivity in a cost-effective way. VPNs are a kind of networks which use IP tunnel creating a private network over Internet. Cryptography is the technique to provide robust security and privacy over the IP tunnels. The [22] is a well-done article on VPN, it focuses on Open-Source Linux-Based VPN solutions (OSLV) giving their network performances. It contains also a deep explanation on software architecture for OSLV routers, VPN characteristics and comparisons among proprietary and open source VPNs.

OSLV belongs to open source VPN solutions and it is a service using a desktop computer running Linux. Network performances of a OLSVs are measured in terms of overhead (ratio between the useful data application and the length of the packet in bytes), bandwidth utilization and latency/jitter. Usually VPN solutions have worst network performances than networks without VPN utilization. The article [22] shows also some experimental results on OSLV using UDP or TCP-based tunnels. UDP and TCP are protocols belonging to the

transport layer of the OSI model protocols stack. From their experiments they observed that UDP-based tunnels have 50% lower overhead, 80% higher bandwidth utilization, and 40-60% lower latency/jitter than those using TCP. [22]

4.3 Introduction on Distributed Co-simulation

Since 1999 and even before, there was the interest on designing distributed architecture for applications on the Internet which includes distributed interactive applications (DIA) distributed interactive simulation (DIS), cooperative tools and so on.

In [23] is described the design of a transmission control mechanism for a distributed multiplayer game on the Internet called MiMaze. The game uses a communication system based on Real-Time Transport Protocol (RTP) over User Datagram Protocol/Internet Protocol (UDP/IP) multicast. Realizing such distributed and interactive application, each entity belonging to the application does not communicate with a central node, usually called server, but with the other nodes of the network. The central server would introduce only delay in the communication and it is not suitable for real-time applications. Distributed solution is the one adapt for real-time purposes. The central server becomes useful only for non-real-time tasks.

The article is based focused on the way to synchronize the users, because synchronization among the user is the minimum constraint to be satisfied for the feasibility of the application. In this application synchronization is guaranteed by using Network Time Protocol (NTP). NTP is used to provide a global reference clock for the entire network.

The delay on the Internet and its variability is something to take into account. Each application has its own requirements for the delay, for example MiMaze requires at maximum a delay of 100 ms, otherwise there is no consistency among the gamers. It becomes fundamental to have strategies and algorithms to recover lost or late packets.

Data packet length is something to control to reduce the traffic network. Part of the packet has to contain mandatory information required by the protocols used, like RTP, UDP and IP. It could be possible to compress the headers, but compression and decompression increase time delay. The payload is the actual information needed for the application and it is what we can control and reduced for the functional operation of the application.

Depending on the application, the transmission packet frequency should be tuned. For the game MiMaze the transmission frequency is 25 times per seconds and this is enough to take into account human reaction. The network congestion is something to monitor constantly and the transmission rate should vary depending on it. [23]

Distributed architectures are not only for interactive applications, but also for interactive simulations. Distributed simulations help to reduce the load of a simulation splitting the work load over many computers. Not only local, but also geographically distributed. The idea was born among research institutes and universities to give the possibility to share resources, so

increasing the computational power, facilities and knowledge while preserving the internal privacy.

In [24] the authors say “Distributed simulation, which aims to separately and concurrently compute the dynamics of several parts of a system, provides a way to share the computational load by multiple computers and, thus, effectively reduce the simulation time”. It means that through this technique it is possible to solve much more complex systems, splitting it in subsystems, solving them separately and then sharing the results. Sharing the resources we have much more computational power available. Each subsystem needs to share its results at each time step with the other subsystems spread in the distributed network, because the results are combined to obtain the overall system solution. Decoupling the system is not trial. Find a good method and the right point to decouple the entire system in subsystems, otherwise it affects the quality of the results. In literature it is called fidelity of the system, how much close the dynamics of the distributed system are to the dynamics of the natural coupled system [25, 24]. In each subsystem, e.g. distributed simulation for power system application, the neighbouring subsystems are represented through their equivalent circuits (Thevenin equivalence), so at each time step the subsystem uses the last solution of the neighbouring subsystems to find out its own solution. In Figure 3-1 the concept is represented. There are two subsystems geographically distributed and they are integrated over the network through coupling variables. At each simulation time step the subsystems exchange their coupling variables that actually are affected by errors introduced by the network. The fidelity of the system is higher when the errors among the all transmitted coupled variables and the received variables is lower. [25]

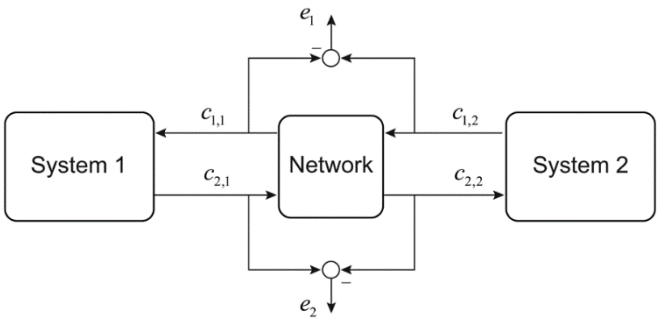

Figure 4-1: fidelity problems in distributed simulation **[25]**

To realize this kind of simulations we need a mean to access the network, able to communicate and also to recover from the latency based errors introduced by the network [26]. It becomes necessary a software-to-software framework for subsystems communication [24]. Distributed simulation over the Internet has been of interest of many different research domains such as power system, automotive, aerospace, telemedicine, robotics and army. The

thesis will be addressed in particular on distributed simulation examples on the power system domain, but will be also given some examples on different domains to understand how this kind of simulations can vary depending on the application and does not exist a methodology to do it.

In [26], they exploit a way to mitigate latency errors in such distributed system. Communication latency influences the desired bandwidth of the experiment, so the dynamicity of the system. The technique is based on the Observer Delay Compensation (ODC) approach. Each simulation subsystem is equipped with an observer. The observer knows the electrical model of the neighbouring subsystem and based on the delayed version result that it receives, it is able to predict the other simulation system state.

Then the subsystem can use the estimated state to estimate its own state of the portion of the electrical system. The output of the observer tries to be as much closer as possible to the actual system output (the neighbour system). It is like the two remote systems are operating closely to each other and it increases the accuracy of the results.

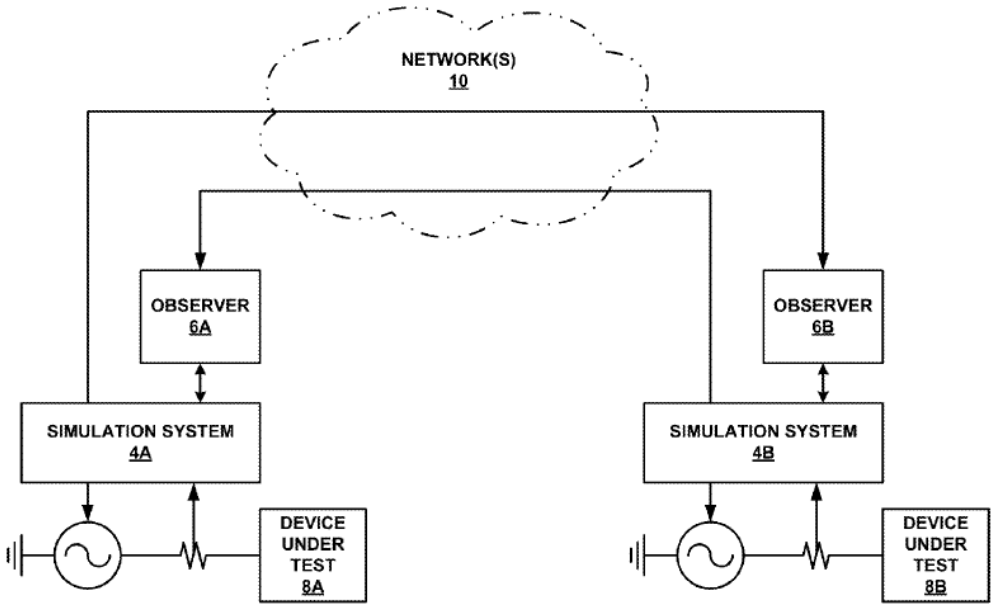

Figure 4-2: ID-HIL experiment with observer to mitigate latency errors **[26]**

The feasibility of introducing an observer depends if it is possible to recreate an accurate model and that the model does not require much computational time to be solved, otherwise it cannot be run in real time. Not running in real-time means that the simulation cannot follow the dynamicity of the system. In [27] the authors present an observer-free solution and they

propose for the first time an event-based solution in the automotive application. Sometimes, in the automotive HIL applications, observer solutions are unfeasible.

4.4 ICT requirements and methods to enable Internet-Distributed HIL

The importance on Hardware-In-the-Loop has been further explained in Chapter 1. “HILS provides a bridge between physical prototyping and virtual experiments by uniquely combining their advantages and allowing for cost-effective, high fidelity experiments” [27]. HIL test experiments are applied in many research fields, and as seen from Chapter 2 we want to apply it on electric power system domain.

The electric power system is becoming smart and integrated with renewable generation and storage. Testing such a complex system requires large computational resources, availability of control algorithms, models of the different actors in the grid and real hardware which impacts on the simulated environment must be tested. Putting together different setups such that it is possible to get the best from each one. To make feasible such a kind complex and large system setup, geographically distributed real-time simulation is under study from many years to enable real-time connection of laboratories. Connecting laboratories enables sharing of resources to satisfy the computational requirements, but also man power knowledge preserving privacy. The high-capacity and secure network, such as GÉANT, that interconnects research and educational networks over Internet can be one of the feasible solutions to interconnect laboratories over long distance in Europe [28].

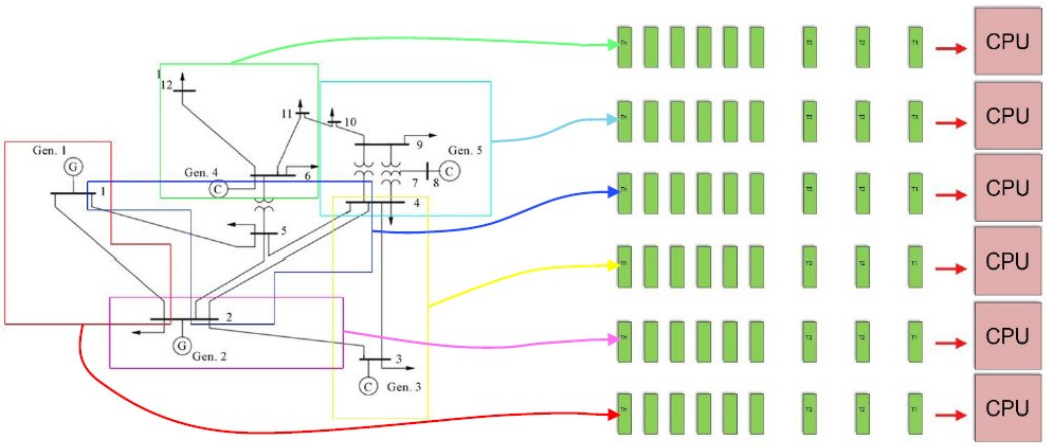

Figure 4-3: spread parallel computation **[29]**

Internet-Distributed HIL has attracted researchers since last twenty years.

Internet-Distributed HIL has two challenges to be faced up together. The first is to find a suitable interface algorithm to guarantee the matching among the two systems geographically separated and the second is the technology challenge to overcome the characteristic problems over Internet (latency/jitter, loss of data) to let the two systems exchanging information in a proper way for the application. The chapter focuses on the second issue. The impact of the Internet characteristics may affect the stability, robustness and fidelity of the distributed system, so guaranteeing those properties is a really challenge to overcome with the current ICT technologies [27].

From the technical report of the JRC in [29] ca be extracted some important questions for co-simulation laboratories that have to be taken into account in this kind of work:

- How can be compensated in real time the delay introduced by the Internet medium?
- How does the system running independently in different sites maintain its stability and accuracy?
- What kind of information each entity needs to share?
- Which algorithm should be used by the simulator to be interfaced?
- What is the right time step for each system running independently in each sites to guarantee the right level of dynamicity and interaction?

These questions are challenges related to the modelling part of the system, but they have to be known at the networking engineers when they have to implement an architecture.

From the ICT side the challenges and requirements for a framework to face up the troubles over Internet are described as follow:

- Each laboratory or each entity participating in the distributed architecture is not interfaced directly to the network. Usually the connection with the external world is done through a dedicated computer. The computer should be equipped with an operating system optimised for real time application, letting prioritization and parallelization processing. The most used is the Fedora distribution with Real Time patch [30]. The real time simulators are connected through their own network card to the interface PC.

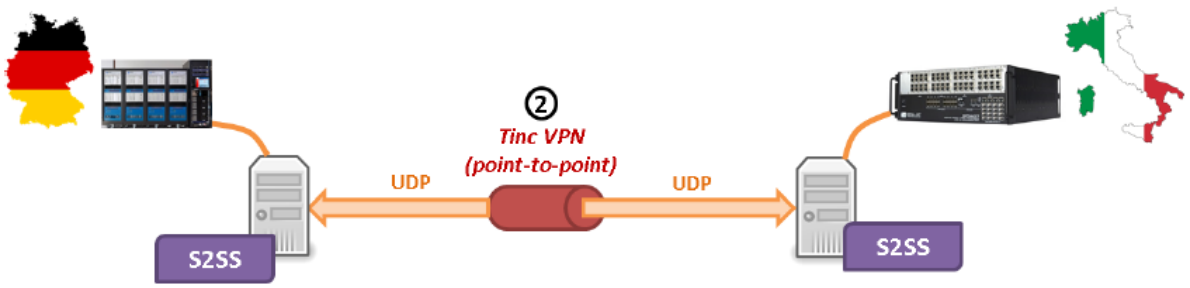

Figure 4-4: point-to-point connection through tunnelling VPN **[29]**

- The network interface must be tuneable and flexible. The interface, based on the feedback it receives of the connection state, can tune its sending rate at runtime. Flexibility means scalability, so the interface can open and manage parallel connections with other entities at the same time.
- Latency and its variability over internet are the main problems. Delay can impact the stability and the accuracy of the system's results. Better routing strategies can choose the shortest path in time.
- The architecture can be bases on two kind of connections: client-server or point-to-point. In the client-server connection there is a central entity (Server) in the system and it manages the packet routing and the synchronization among all the actors. This kind of architecture can influence the scalability of the system, when the number of actors increases and it requires a higher computational effort for the server. Moreover the server could be a bottleneck for the data routing. A point-to-point connection is complementary of the server-client, it solves all the negative aspect. In a point-to-point connection, the NTP protocol can be used for synchronization. Some framework can use both, but in particular the client-server is used for non real-time purposes such as storing or data visualization.

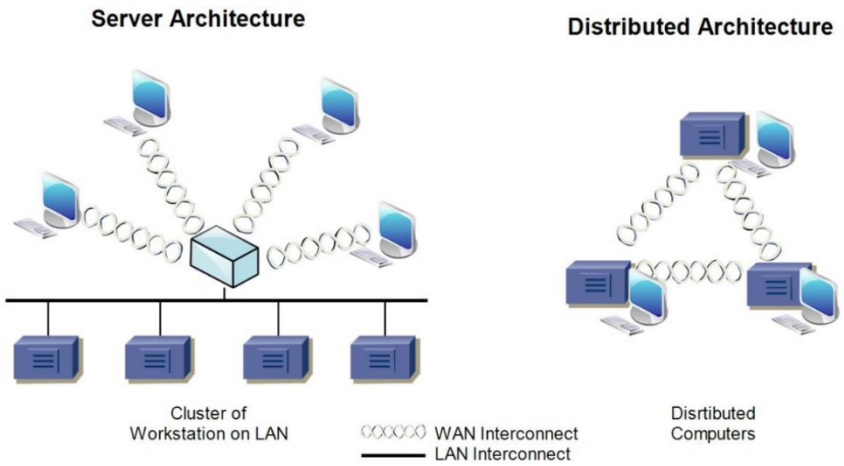

Figure 4-5: client-server and point-point architectures

- Quality and robustness of the architecture must be guaranteed. It means that the architecture should be secure from outside attacks and it should be reliable to give data integrity. VPN solutions are the most cost-effective. VPN can be based on a central node or can be point-to-point. The utilization of the VPN can increase the delay due to the encryption and decryption, but it does not influence the overall performance as has been demonstrated in [31, 32].

- The software interface must be equipped with the so called Data Distribution Management. Distributing the data only to subsystems that need the state of the system for energy conservation purposes. Usually the data packet is designed to be with a small overhead and payload length can vary. In [31] the authors propose a custom UDP packet with a negligible overhead for an Internet-Distributed co-simulation scenario. Data sequence and timestamp are important features to maintain in the packet structure. They are useful for processing data in order and accepting only the most updated information.
- The network Quality of Service (QoS) is described base on some key indicators:
    - Round Trip Time: the RTT statistic evaluation are made in comparison with and without VPN solutions, and also varying the sending rate and the payload length.
    - Bit rate
    - Error rate
    - Loss rate

Conclusions

Power Hardware-In-the-Loop experiments are very useful in order to test hardware prototype during its developing stage. Thanks to the Real-Time Simulator it is possible to analyse the hardware response at "wall clock" time. Its importance is not only in Electromagnetic transient's analyses, but also in hardware steady state behaviour. Building a distribution Reference Network Model on the RTDS Simulator makes possible the study of the introduction of a new hardware, such as AC/DC inverter, relay, etc., inside the distribution network. It is useful for the hardware and other control strategies validation.

The thesis work has been conducted at the Smart Grid Interoperability Laboratory (SGI lab) of the European Commission in Ispra (JRC). The main focus of the work has been on building the whole framework to guarantee stable Power Hardware-in-The-Loop experiments. The framework needed to be built from scratch due to the fact that this kind of experiment has never been performed at the SGI lab. The hardware and the software equipment needed to be investigated in depth since they have never been used with this purpose. The objective is to give the possibility to other researchers at the JRC and at the Politecnico di Torino to use this framework. This step is a very important starting point if one wants to reach the goal of the more ambitious project aiming at interconnecting the Ispra laboratory with the Politecnico di Torino's laboratory for co-simulation activities. Without this framework the SGI lab in Ispra cannot be a useful node on such a Distributed Laboratory project.

From a practical point of view, the work aims at integrating the Real Time Digital Simulator with the Power Amplifier, and then at interfacing, through Interface Algorithms, the virtual system with the real hardware. Moreover, when linking several dynamic systems, stability issues can seriously challenge the experiments, in particular, the component under test but also the human beings performing the experiments. For this reason, a stability study has been undertaken. It is remarkable to mention also that for each given setup in an experiment, the impact of the delay needs to be estimated. The delay's knowledge is fundamental for the working and the success of the experiment.

The setup infrastructure can be used to perform Power Hardware-In-the-Loop experiment. The all laboratory's equipment have been integrated to each other working at 62 μs. In order to interface the Virtual System with the physical hardware, it has been studied and implemented two interface algorithms. The Interface Algorithms are the Ideal Transformer Model (ITM) and the Damping Impedance Model (DIM). The stability of the two interfaces has been analysed and it has been proved their behaviour as described in

literature review. The DIM interface shows a higher close loop system stability than ITM. The ITM comes with an easier implementation.

Before performing real Power Hardware-In-the-Loop experiment, the system stability must be proved in simulation. So a simulation environment has been built in Simulink. Having a knowledge of the hardware's model is important to be replicated in simulation. Through simulation we can tune the interface algorithm parameters based on the hardware reaction. Moreover, safety control algorithms have been implemented to prevent conditions of overcurrent and overvoltage that could damage the hardware under test.

As future work it is really important to make an accuracy evaluation of the two interfaces when connected with a hardware.

Annex

A.1 Buffers

A Circular InterProcess Communication (CIPC) buffer is a shared memory strategy based on ring-buffers to allow Matlab Simulink models to communicate with each other. The EtherCAT and the Power Module Simulink models need to communicate to each other and they exchange data thanks to these buffers.

Each buffer has one writer that writes data into the buffer, from which multiple readers can read out the data. The usage of these buffers allow the Simulink models to run at 16 kHz without overloading the CPU of the Real-Time Target PC.

A.2 Simulated TriPhase Power Module

The Figure A-1 shows the blocks through which we emulate the Power Module's behaviour.

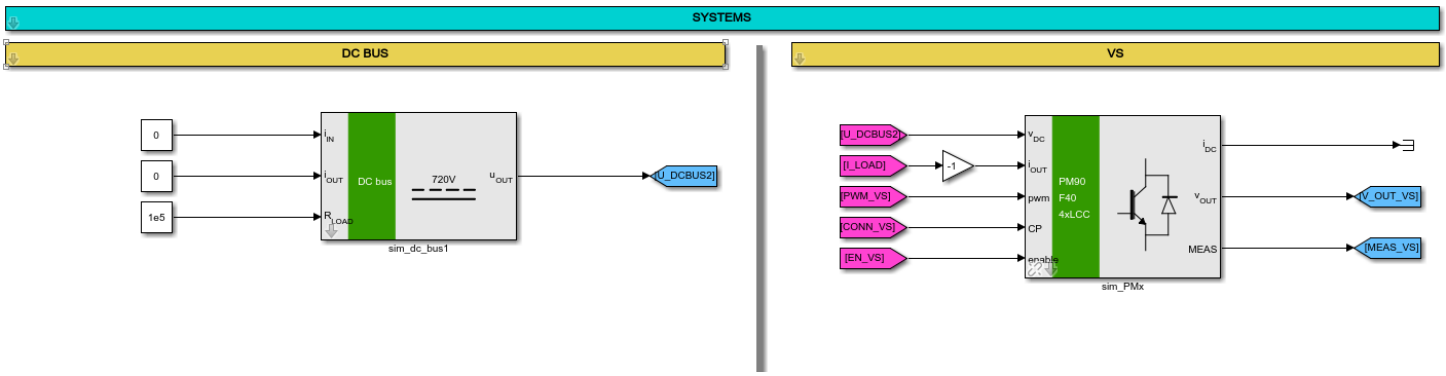

Figure A-0-1: Power Module simulation blocks

Bibliography

[1] C. Dufour and J. Bélanger, "On the Use of Real-Time Simulation Technology in Smart Grid Research and Development," *IEEE,* 2013.

[2] V. H. Nguyen, Y. Besanger, Q. Tran, C. Boudinnet, T. L. Nguyen, R. Brandl and T. Strasser, "Using Power-Hardware-In-the-Loop Experiments together with Co-simulation for the Holistic Validation of Cyber-Physical Energy systems," *HAL archives-ouvertes,* 2017.

[3] Bian D., Kuzlu M., Pipattanasomporn M., Rahman S. and Wu Y., "Real-time Co-simulation Platform using OPAL-RT and OPNET for Analyzing Smart Grid Performance," *IEEE,* 2015.

[4] W. Li, X. Zhang and H. Li, "Co-simulation platforms for co-design of networked control systems: An overview," *Elsevier,* vol. Control Engineering Practice, 2014.

[5] C. Washington and S. Delgado, "Improve Design Efficiency and Test Capabilities with HIL Simulation," *IEEE,* 2008.

[6] D. Ocnasu, C. Gombert, S. Bacha, F. Blache and S. Mekhtoub, "Real-time hybrid facility for the study of distributed power generation systems," *Revue des Energies Renouveblables,* 2008.

[7] T. Strassers, "Real-Time Simulation Technologies for Power System Design, Testing, and Analysis," *IEEE Power and Energy Tecnology System Journal,* 2015.

[8] X. Guillaud, O. Faruque and A. Teninge, "Applications of Real-Time Simulation Technologies in Power and Energy Systems," *IEEE Power and Energy Technology System Journal,* 2015.

[9] P. Forsyth and R. Kuffel, "Utility applications of a RTDS Simulator," in *The 8th International Power Engineering Conference*, 2007.

[10] T. Hatakeyama, A. Riccobono and A. Monti, "Stability and Accuracy Analysis of Power Hardware-In-the-Loop System with Different Interface Algorithms," *IEEE,* 2016.

[11] G. Lauss, F. Lehfuss, P. Kotsampopoulos, N. Hatziargyriou, P. Crolla and A. Roscoe, "Comparison of multiple Power Amplification types for Power Hardware-in-the-Loop Applications," 2013.

[12] A. Aguirre, M. Davila, P. Zuniga and E. Barocio, "Improvement of Damping Impedance

Method for Power Hardware-In-the-Loop Simulations," *IEEE,* 2016.

[13] S. Paran and C. Edrington, "Improved Power Hardware-In-the-Loop Interface Methods via Impedance Matching," *IEEE,* 2013.

[14] W. Ren, M. Steurer and T. Baldwin, "Improve the Stability and the Accuracy of Power Hardware-in-the-Loop Simulation by Selecting Appropriate Interface Algorithms," *IEEE Transactions on industry applications,* 2008.

[15] G. Lauss, O. Faruque, K. Schoder and C. Dufour, "Characteristics and Design of Power Hardware-in-the-Loop Simulations for Electrical Power Systems," *IEEE transactions on industrial electronics,* 2016.

[16] P. Kotsampopoulos, "Design, development and operation of a PHIL environment for Distributed Energy Resources," *IEEE,* 2012.

[17] RTDS Technologies, "Real Time Digital Simulationfor the Power Industry Manual set".

[18] https://www.beckhoff.com/.

[19] https://www.triphase.com.

[20] R. Brandl, "Operational Range of Several Interface Algorithms for Different Power Hardware-In-The-Loop Setups," *Energies,* 2017.

[21] 03b Networks, "What is Network Latency and Why Does It Matter?".

[22] S. Khanvilkar and A. Khokhar, "Virtual Private Networks: An Overview with Performance Evaluation," *IEEE Communication Magazine,* 2004.

[23] C. Diot and L. Gautier, "A Distributed Architecture for Multiplayer Interactive Applications on the Internet," *IEEE Network,* 1999.

[24] Q. Huang, J. Wu, J. Bastos and N. Schulz, "Distributed Simulation Applied to Shipboard Power Systems," *IEEE,* 2007.

[25] T. Ersal, M. Brudnak and J. Stein, "An iterative learning control approach to improving fedelity in internet-distributed hardware-in-the-loop simulation," 2012.

[26] J. Cale and E. Dall'anese, "Mitigating latency errors in distributed systems," 2017.

[27] T. Ersal, M. Brudnak, A. Salvi, J. Stein, Z. Filipi and H. Fathy, "Development and model-based transparency analysis of an Internet-distributed hardware-in-the-loop simulation platform," *Elsevier,* vol. Mechatronics, 2011.

[28] M. Stevic, S. Vogel, A. Monti and S. D'Arco, "Feasibility of geographically distributed real-time simulation of HVDC system interconnected with AC netowroks".

[29] C. F. Covrig, G. De Santi, G. Fulli, M. Marcelo, M. Olariaga, E. Bompard, A. Estebsari, A. Monti, M. Stevic and S. Vogel, "A European Platform for Distributed Real Time Modelling & Simulation of Emerging Electricity Systems," JRC Technical Reports, 2016.

[30] https://rt.wiki.kernel.org/index.php/Main_Page, "Linux RT-patch".

[31] M. Stevic, A. Estebsari, S. Vogel, E. Pons, E. Bompard, M. Masera and A. Monti, "Multi-site European framework for real-time co-simulation of power systems," *IET,* 2017.

[32] C. Wiezorek, A. Parisio, T. Kintaja, J. Elo, M. Gronau, K. Johannson and K. Strunz, "Multi-location virtual smart grid laboratory with testbed for analysis of secure communication and remote co-simulation: concept and application to integration of Berlin, Stockholm, Helsinki," *IET Generation, Transmission and Distribution,* 2017.

[33] RTDS Technologies, "Power Hardware-In-the-Loop Simulation".

[34] www.geant.org.

[35] L. Wang and E. Gelenbe, "Real-Time Traffic over Cognitive Packet Network".

[36] K. Ravikumar, N. Schulz and A. Srivastava, "Distributed Simulation of Power Systems using Real-time Digital Simulator," *IEEE,* 2009.

[37] N. Schulz and A. Monti, "Distributed simulation using the Virtual Test Bed and its real-time extension," 2007.

[38] A. Monti, M. Stevic and A. Benigni, "Development of a simulator-to-simulator interface for geographically distributed simulation of power system in real time," *IEEE,* 2015.

[39] A. Estebsari, E. Pons, E. Patti, M. Mengistu, E. Bompard, A. Bahmanyar and S. Jamali, "An IOT Realization in an Interdepartmental Real Time Simulation Lab for Distribution System Control and Management Studies," *IEEE,* 2016.

[40] R. Liu, M. Mohanpurkar, M. Panwar, R. Hovsapian, A. Srivastava and S. Suryanarayanan, "Geographically distributed real-time digital simulation using linear prediction," *Elsevier,* vol. Electrical Power and Energy Systems, 2017.

[41] I. k. Park, P. Forsyth, H. Kim and K. Hur, "A Study on Synchronizing Two Separate RTDS Simulation Instances," *IEEE,* 2016.

[42] M. Mirz, A. Estebsari, F. Arrigo, E. Bompard and A. Monti, "Dynamic Phasors to Enable Distributed Real-time Simulation," *IEEE,* 2017.

[43] M. Mirz, S. Vogel and A. Monti, "First Interconnection test of the nodes in pan-

European simulation platform,” 2017.