Loose Coupling Architecture for Co-Simulation of Heterogeneous Components
Support of Controller Prototyping for Smart Grid Applications

Ralf Mosshammer∗, Friederich Kupzog†, Mario Faschang‡ and Matthias Stifter†

∗Siemens AG Austria - CT RTC NEC INN-AT
Siemensstrasse 90, 1210 Vienna, Austria
Email: ralf.mosshammer@siemens.com

†Austrian Institute of Technology - Energy Department
Giefinggasse 2, 1210 Vienna, Austria
Email: {friederich.kupzog.matthias.stifter}@ait.ac.at

‡Vienna University of Technology - Institute of Computer Technology
Gusshausstrasse 27-29/384, 1040 Vienna, Austria
Email: faschang@ict.tuwien.ac.at

Abstract—Co-simulation of different domains is the method of choice if specialised tools exist for each domain and no monolithic solution is available. Motivated by the need for simulation and emulation of networked smart grid systems with interlinked controller, communication and power hardware components, this paper proposes a solution for loose coupling of heterogeneous components (i.e. continuous and time-triggered subsystems). The focus is on a lightweight message bus, allowing multiple simulators to exchange the same messages like in the real system. The architecture allows fast configuration and easy switching between emulated and real-world components. The solution is applied and verified in the context of a Smart Grid application. Photovoltaic inverters in a distribution grid are coordinated with a central on-load tap change transformer in order to keep the grid voltages within the limits set by operation rules.

I. INTRODUCTION

The current development in the Smart Grid field shows that future smart grid systems will contain a heterogeneous agglomeration of active power electronics components and passive network components coupled via physical processes and dedicated communication connections. Developments such as the massive extension of photovoltaics (PV) installations in Germany [1] or small hydro plants in certain Austrian medium voltage grids [2] have resulted in the application of active control approaches in distribution systems such as coordinated voltage control [3], [4] or even local system balancing in case of a fault in the high voltage grid [5]. The verification of such networked smart grid systems will play a major role in future technology development, since these systems have a considerable complexity. Now, as the first demonstration projects for smart grid technology are running, verification and testing of the interaction of communication network with the energy network will be examined and evaluated in order to guarantee a stable service.

Exemplary areas in which co-simulation can be necessary:

- Voltage control (local or coordinated) in distribution systems
- Distribution network-supply shortfalls: Failures and critical network states
- Islanding (and network synchronisation) due to failure of higher-ranked networks
- Station-automation and safety devices (IEC 61850)
- Demand Response

For efficient testing of controller, communication and power hardware before deployment into the Smart Grid system, a real time simulation (emulation) of the system has to be realized. Only with an emulator it is possible to freely exchange simulated and real systems “on the fly” in system verification. A control unit can e.g. be developed against a simulated power grid and communication environment. The simulated environment can then be step by step replaced by the real system (open loop, later closed loop). The requirements for a platform aiming for system development and seamless verification are:

1) Threefold modelling: The elements that have to be adequately modelled by the framework are the power grid, the communication network, and the control algorithms connecting both infrastructures.

2) Scalability: The framework must allow simulating problems of different sizes from detailed analysis of a low-voltage network branch to a general fault analysis in a wide area of inter-networked medium voltage distribution grids. The size of simulated communication networks varies accordingly (but not necessarily fully correlated with the power network size).

3) Flexible model reduction: As a consequence of the above requirement, the utilised simulation models have to be flexible regarding their level of detail.
4) Flexible simulator coupling: Depending on the scope of the simulation, different coupling methods between the three models (power grid, communication network and control algorithms) must be possible. The coupling methods, besides the model granularity, have a major impact on simulation performance and result accuracy.

II. RELATED WORK

Related platforms and simulation interfaces for real-time simulation (emulation) of networked Smart Grid systems have been developed in recent years. A strong focus on internet-nesting of simulation components can be observed. Although general solutions for multi-domain simulations are existing that potentially allow to integrate all parts (controller models, communication models, power grid component models) into one simulation engine, the practicability of such solutions is restricted since existing models in specialised tools cannot be reused. Therefore the approach of a loose coupling architecture is chosen here. The following related approaches have influenced this work.

A. Mosaik

Mosaik aims to provide a framework for modular simulation of various Smart Grid components by means of composition of various existing, in general heterogeneous models realised in different simulation tools. It has been developed by Offis in Germany [6] and is build up of various layers for supporting scenario specification, simulation composition and execution, and finally result analysis. The scenario specification for large scale simulations and control strategies is defined in a meta-model approach. Mosaik models are based on discrete event simulation and integrates continuous simulation by coupling them via discrete event wrappers. Communication and control strategies are connected via standardized interface OPC Unified Architecture and the common information model (CIM). Semantic information is utilized for the mapping to standard conform interfaces.

B. FMI

An approach for coupled-simulation based applications is the open standardised interface specification Functional Mockup Interface (FMI) for model exchange and co-simulation [7]. Stand-alone, self-contained units are wrapped by an API, which provides functionality to create and simulate instances of models. Various commercial and open source simulation environments provide functionality for FMI export, like Dymola or OpenModelica. Support for other tools are currently developed, like GridLAB-D [8]. In [9] the authors propose a high-level wrapper for FMI’s fundamental functionalities, which are initially a C interface only. This library provides adequate numerical integrators and is suitable for e.g. discrete event simulation. It enables also continuous time-based simulation to used as co-simulation components.

C. GridLAB-D

GridLAB-D is a simulator for power distribution systems developed by the U.S. Department of Energy (DOE) at Pacific Northwest National Laboratory (PNNL) [10]. Its philosophy is to combine detailed simulation of a large number of individual processes connected to the grid infrastructure with flexible possibilities to interface the simulator with other tools. In GridLAB-D, each grid-connected resource can be modelled e.g. by a number if differential equations that are solved numerically during runtime. There is no need for aggregated models. In terms of simulation control, GridLAB-D works like a time-discrete simulator with an event cue, where each component can inform simulation control about the next point in time when it wants to be updated. Simulation control performs an update on all components when the earliest update request is pending. Due to its advanced interfacing features, it can be used as platform for co-simulations with other domains, such as communication, thermal systems or energy markets [11]. From its design however it is not a lean simulator coupler but rather an enriched distribution grid simulator with a special time-triggered simulation engine, that wakes up the simulation models when required. GridLAB-D typically is used for the combined simulation of generator and consumer behaviour in conjunction with different market models [12].

D. HLA

High Level Architecture (HLA) has been developed by the U.S. Department of Defense (DOD) and is an abstract architecture that supports distributed simulation. It is an international standard (IEEE-1516) since 2000. HLA consists of a central coordination unit named Runtime Infrastructure (RTI), as set of rules to govern the overall simulation as well as every single simulator, and an Object Model Template (OMT) that defines the form of data exchange. HLA uses publisher and subscriber mechanism and offers an object based API and several daemons (e.g. synchronisation, time management, data persistence) [13]. It is not capable of integrating hardware components and offers no user interfaces. Furthermore HLA is not suited for smooth transplantation of developed algorithms to the field hardware (as described in [14]).

III. SIMULATION MESSAGE BUS CONCEPT AND ARCHITECTURE

To meet all the previously mentioned requirements and to overcome the drawbacks of the existing coupling solutions a server software component – the Simulation Message Bus (SMB) – has been developed. The underlying philosophy and application domain, as well as the detailed architecture of the SMB are described in the following subsections.

A. Philosophy and Application Domain

Coupling of heterogeneous software components is a common task in both localized and distributed systems, and a plethora of off-the-shelf solutions are available. The requirements described in I, however, introduced additional constraints for the coupling software:

- The components offered only limited API access and communication capabilities, with the lowest common denominator being the plain TCP/IP socket.
- To enable hardware-in-the-loop emulation, and later field deployment, data distribution had to be similar to component behavior in actual field scenarios.
• Real-time emulation put latency constraints on overhead operation such as routing and translation.
• Very volatile requirements had to be accounted for by being able to tweak every aspect of the system.

With a central component of the system – the Data Concentrator, responsible for aggregating field measurements – operating in push mode, it was natural to implement a source-defined data driven routing. Instead of handling distribution via a central data storage, a bus concept was implemented, where the server component – the SMB – is ignorant of the incoming data and simply performs routing. This approach has several advantages: there is no need for inventing a data model to accommodate all data types exchanged between clients; problems with concurrent access to data points are avoided; and the approach is very similar to existing data routing schemes in automation hardware. The data agonisticism of the SMB also facilitates component swapping, as interpretation responsibility is deferred to the clients. Thus, a component providing measurement data must take care that the data semantics are interpretable by consumer components, e.g., a control algorithm. No adaption of the server is necessary. The drawback of this “loose data coupling” is the need for a separate specification clarifying the actual semantics. With a limited amount of data classes, including control commands, Smart Meter readings and data from distributed components such as PV inverters, this was perceived as an acceptable cost for the gained flexibility. The principal deployment scenario of the SMB is coupled simulation: interlinking of heterogeneous system simulators to achieve an integrated simulation toolchain (see exemplary application in section IV-A). With the possibility of swapping software components for actual hardware, and providing real-time emulation, hardware-in-the-loop emulation becomes possible. In this scenario, the grid and communications simulator are exchanged for live data from the grid, allowing for verification of the developed models. With a successive transfer of models to real components, the usage scenario for the SMB approaches actual field deployment. Here, the SMB acts as data distribution centre between the Data Concentrator (DC) and control algorithms (see exemplary application in section IV-B).

B. Architecture

The SMB is designed as a stand-alone server component which will route messages between clients much like a network switch. It supports the connection of an arbitrary number of such clients, and acts as the central message hub for the entire simulation. The principal architecture, illustrating the components of the SMB, is shown in figure 1.

1) Channels and Proxies: The SMB holds a number of channels, which serve as I/O points for connecting clients. When configuring the SMB before startup, each channel is defined with a unique ID. This ID allows the SMB to route incoming packets.

Atop each channel sit zero or more proxies, which are pluggable components extending the channel functionality. A proxy (depicted in figure 2) has input and output queues on both ends (bottom, which is the end pointing towards the channel, and top) and processing functions for packets passing through either direction (upstream going from bottom to top and vice versa). To create custom functionality proxies can be overridden. Due to the fact that the processing functions are independent and free running processes, SMB packet processing is nondeterministic. Nevertheless packets within the single queues will be processed in the same order as they arrived.

2) XML Configuration: The SMB is configured via a XML file outlining the previously described architecture (the channels and proxies). An exemplary configuration file is found in listing 1. It represents a SMB with one channel. This channel holds two proxies, where every proxy has a parameter that can be used for the proxy’s initial configuration. The name of the XML configuration file has to be passed to the SMB instance on startup (e.g. via command line parameter).
3) Synchronisation and Simulation Control: The SMB – as it is described so far – can already be used for simulation and emulation tasks. The basic structure allows for heterogeneous simulation tools to be connected and exchange messages. Depending on the type of simulation a synchronizing unit is necessary to keep the simulation units synchronous and to avoid a spread of units during emulation. Furthermore to allow central configuration if the coupled simulation some control tasks have to taken over.

![Synchronisation Proxy state machine](image)

For these tasks of synchronisation and simulation control a proxy implementation, namely the SyncProxy, was introduced. This proxy relies on the previously described abstract proxy and has an internal state machine, depicted in figure 3 (simplified version). The SyncProxy is autonomously running directly within the SMB and offers no connections for other on-top-proxies or external clients. Its configuration comes within the SMB XML configuration and is a separate channel holding the SyncProxy and eventually a routing proxy.

The only two parameters that are needed for operation of the SyncProxy are the number of clients to connect and the location of a scenario file. The scenario file holds arbitrary parameters which will be forwarded to all clients during the <INITIATE>-state (see state machine in figure 3). This <INITIATE>-state is performed after all simulation clients were connected to the SyncProxy in its <LOAD>-state, where the current simulation scenario is loaded from the predefined file. After the acknowledge of all clients a <PREPARE>-state is passed, where all clients have time to load necessary data and to prepare for the upcoming start of the simulation / emulation. The <NEXT>-state is the place where the actual simulation step is performed. All clients are calculating the predefined constant time slice starting with a common start and end time. After finishing their calculations they report to the SyncProxy and therewith finish their operation until they receive the next <NEXT>-command from the SyncProxy. This discrete time slice simulation [15, p. 219 ff] is done until the – in the <INITIATE>-state briefed – simulation end time is reached. In case of emulation – where an emulation scale factor is communicated during initialisation – there is only one <NEXT>-step performed, where all clients start a (scaled) free running emulation until the emulation end time is reached. After finishing simulation / emulation, a <NEXT>-state is passed to await all clients stopping and unloading / processing their simulation results. Subsequently, the <LOAD>-state is reached, where either a next scenario is loaded and the cycle starts again, or a termination signal will be sent to the clients when no more scenarios are available. By this mechanism it is possible to configure all – potentially distributed – clients from a central point and to process a bunch of simulation and emulation scenarios automatically in a row.

4) Web Services and Interface: Even though the SMB is a flexible coupling entity there is no easy way to get access to the transmitted data, or to fake data and commands for testing purposes. Especially when using the SMB in C-HIL, closed- and open loop hardware operation is necessary to observe the internal states and to influence them. To face these challenges, a WebserviceProxy was introduced. This proxy is used like the previously described SyncProxy: If needed, it can be added to the SMB as a separate channel by adding the lines printed in listing 2 to the SMB's XML configuration.

![SMB XML configuration for webservice proxy channel](image)

Listing 1. minimalistic XML configuration describing SMB architecture

```xml
<Configuration>
  <Channel id="channel_1">
    <Proxy id="proxy1" type="proxyTypeA">
      <param name="par1" value="value1"/>
    </Proxy>
    <Proxy id="proxy1" type="proxyTypeB">
      <param name="par2" value="value2"/>
    </Proxy>
  </Channel>
</Configuration>
```

Listing 2. SMB XML configuration for webservice proxy channel

If configured, the SMB loads several RESTful web services at startup, providing access to the SMB data. Furthermore a web page is provided (either via http or https) where SMB status information and packet logs are available, and packet injection can be performed.

IV. EXAMPLE OF APPLICATION

As the SMB is a general and flexible packet router like software component it has a wide field of application. In general it can connect any socket based components for packet exchange purposes. Nevertheless, its primary field of application is coupled simulation for Smart Grids applications, where it can
be useful throughout the whole development process providing a seamless way from the control prototype, through testing, (P)HIL operation, towards the final controller in the field with monitoring and control capability. Two steps of the Smart Grid control deployment process – on the one hand the co-simulation, on the other hand the deployment in the field – and the role of the SMB thereby will be explained in the following subsections. For a detailed description of the whole seamless development and deploy process supported by the SMB please be referred to [14].

A. Example: Distributed Smart Grid Control Simulation

One scenario, where the SMB is in use, is the co-simulation for Smart Grid control and evaluation in the project DemoNet Smart LV Grid. The project aims to enable an efficient and cost effective use of existing grid infrastructure based on a three-step concept (intelligent planning, on-line monitoring and active grid management) to be capable of managing high penetration of distributed energy resources and electric vehicles in rural low voltage grids. To support the intelligent planning process and the development of control strategies co-simulation of power grid and communication is done.

![Diagram of SMB architecture for distributed Smart Grid control simulation](image)

The setup for this application can be seen in figure 4 where the SMB interconnects the following simulation clients:

**Power Grid Simulation**
For simulation of the low voltage power grid DlgsILENT PowerFactory was used.

**Communication Simulation**
To provide realistic power line message loss and delay behavior a communication simulator was used. This client uses statistical loss and delay information of each power grid to simulate packet interference.

**EGDA Client**
This Express Grid Data Acquisition (EGDA) client represents the EGDA functionality of the SIEMENS AMIS Data Concentrator (AMIS-DC), which is a hardware unit located in the grid, collecting and forwarding data from the AMIS smart meters towards a control unit.

**Modbus Client**
This client models the cyclic modbus communication between the AMIS DC and the modbus module in the transformer tap changer.

**Simple Control Unit**
Here a simple control unit is used to process all measurement data and to create tap change commands, which will be sent to the grid simulation in this setup.

**Sync Proxy**
The Sync Proxy is, like described in chapter III-B3, responsible for simulation configuration and control, and client synchronisation.

**Web Services and Interface**
The web services and the web interface are used for simulation monitoring and data injection.

There are some proxy implementations used in figure 4, which are described in the following listing.

(R) **Routing Proxy** applies a rule set to all received packets to determine the channels where the packets should be routed to.

(S) **Serialisation Proxy** converts any (binary) element to its string serialisation (e.g. JSON) to be easily readable for clients which are not capable of handling binary Java objects.

(SC) **Socket Connector Proxy** provides an open TCP/IP socket (on a pre-configured port) for clients to connect.

The power grid simulator uses representative models of the rural low voltage grids including all newly installed components, like PV inverters and electric vehicles, as well as transformers with remotely controllable tap changers. All smart meters and PV inverters are connected via the SIEMENS Automated Metering and Information System (AMIS), which uses power line communication (PLC). The PLC channel and the AMIS system behaviour can not be modelled within the grid simulation of DlgsILENT PowerFactory and thus has to be modelled separately. Therefore the communication simulation, the EGDA client, and the Modbus client were introduced into co-simulation. The Sync Proxy was used to synchronise all simulation components. Furthermore the Sync Proxy is capable of automatic simulation scenario processing, so it was possible to simulate many different scenarios (PV generation profiles, load profiles, communication channel behaviour) in a row without interaction. Figure 5 shows one exemplary result of the co-simulation. The upper chart shows that the transformers tap changer is on tap position +1, which means that the transformer secondary side provides slight over voltage (4.8 %). Due to the increase of PV power generation the upper voltage band limit is exceeded after a while. The simple control unit – which constantly receives AMIS EGDA information from distinct meters in the grid – reacts to the voltage band violation by tapping down the tap changer one level to the neutral position.

B. Example: Control Deployment in Low Voltage Grid

By the consequent replacement of simulated elements through real grid components the SMB evolves from a (co-)simulation coupling and control unit towards a data distribution center for field deployment. Figure 6 depicts an exemplary application of a low voltage grid control unit that is deployed in rural substations in the course of the research project DemoNet Smart LV Grid.
hardware-in-the-loop validation tests. The SMB is successfully exchange of simulated by real components, allowing controller-components. The message-bus approach supports the simple also to model variable network topologies of the real-world bus presented in this paper allows not only to couple sub-a wide field of applications. The modular setup of the message simulation environment. The resulting platform can be used for the approach of coupling specialised simulation tools with a simulator for networked Smart Grid systems, this work followed and remote control.

V. Conclusion

In order to achieve a precise multi-domain simulator/emulator for networked Smart Grid systems, this work followed the approach of coupling specialised simulation tools with a lightweight Simulation Message Bus (SMB) realising a co-simulation environment. The resulting platform can be used for a wide field of applications. The modular setup of the message bus presented in this paper allows not only to couple sub-simulations in the continuous and time-triggered domain, but also to model variable network topologies of the real-world components. The message-bus approach supports the simple exchange of simulated by real components, allowing controller-hardware-in-the-loop validation tests. The SMB is successfully used for the evaluation and development of voltage control concepts for a smart distribution power grid prototype in Austria [14], [16].

Acknowledgement

The research project DG DemoNet Smart LV Grid is funded within the program Neue Energien 2020 by the Austrian Klima- und Energiefonds under project number 829867.

References


