DISSERTATION

Rapid Control Prototyping for Networked Smart Grid Systems Based on an Agile Development Process

Submitted at the Faculty of Electrical Engineering and Information Technology, TU Wien
in partial fulfilment of the requirements for the degree of
Doktor der technischen Wissenschaften (equals Ph.D.)

under supervision of

em. o. Univ. Prof. Dipl.-Ing. Dr. techn. Dietmar Dietrich
Institute of Computer Technology, Institute number: 384
TU Wien, Austria

and

Prof. Dr. Sebastian Lehnhoff
OFFIS – Institute for Information Technology
Carl von Ossietzky University in Oldenburg, Germany

by

Dipl.-Ing. Mario Faschang
Matr. No. 0525929
Krottenbachstraße 102/2/1
1190 Vienna, Austria

June 29, 2015

— *This page intentionally left blank* —

Putting a man on the moon was one of the greatest technological challenges of the 20th century.
In the 21st century we face an even greater test – tackling climate change.
In contrast to the space race, the solutions required today must encompass us all.
This is not just about one man walking on the moon, but about 7 or 8 billion people,
the population of 2020, living low carbon lifestyles in harmony with our climate.

Mr. Steve Howard (CEO the Climate Group) – 2008

— *This page intentionally left blank* —

Kurzfassung

Ein nachhaltiges Energieversorgungssystem ist unentbehrlich um dem fortschreitenden Klimawandel erfolgreich entgegenzuwirken. Im Zuge der Entwicklung hin zu solch einem Energieversorgungssystem werden zunehmend erneuerbare Energiequellen zur verteilten Stromerzeugung eingesetzt. Regelungssysteme sind nötig, um diese verteilten Erzeugungsanlagen stabil in die elektrischen Energieversorgungssysteme integrieren und dort effizient betreiben zu können. Aufgrund der Neuheit dieser Regelung von aktiven Komponenten in elektrischen Verteilernetzen fehlt es an Methoden und Werkzeugen für eine durchgängige und konsistente Regelungssystementwicklung. Um dem zu begegnen, wird in der vorliegenden Arbeit ein Prozess zum Rapid Prototyping von Regelungssystemen für vernetzte, intelligente Stromnetze entwickelt, angewandt und validiert. Dazu wird zuerst ein Konzept des Prozesses zur Verbindung der einzelnen Regelungssystem-Entwicklungsschritte erstellt. Der Fokus liegt dabei auf der schnellen und Risiko-minimalen Entwicklung und Inbetriebnahme des Regelungssystems im Stromnetz. Für die Umsetzung des Konzepts wird eine spezielle Integrationskomponente entwickelt und eingesetzt die sich durch den gesamten Prozess, vom Entwurf des Regelungssystems bis hin zu dessen Betrieb, zieht und eine nahtlose Inbetriebnahme ermöglicht. Der Beitrag dieser Arbeit, der den vorgestellten Prozess auszeichnet, ist seine agile und flexible Struktur, die daraus resultierende schnelle Entwicklung und nahtlose Inbetriebnahme der Regelungssysteme, sowie die Risikoreduktion die während des gesamten Entwicklungsprozesses im Fokus steht. Der vorgestellte und umgesetzte Entwicklungsprozess wird in einem industriellen Forschungs- und Entwicklungsprojekt zur Regelungssystementwicklung für intelligente Niederspannungsnetze ganzheitlich eingesetzt. Dies beinhaltet die gekoppelte Simulation des Strom- und Kommunikationsnetzes, die Controller-Hardware-in-the-Loop-Evaluierung, die nahtlose Feldintegration, sowie die einfache Verbesserung und Erweiterung der entworfenen Regelungssysteme. Mit dieser Methode wurden Regelungssysteme entwickelt mit dem Ziel der Beeinflussung aktiver Netzkomponenten zur Spannungsstabilisierung in drei österreichischen Niederspannungsnetzen mit hoher Dichte an verteilter Energieerzeugung aus erneuerbaren Ressourcen. Die Ergebnisse des Feldbetriebs dienen in dieser Arbeit zur Validierung des Entwicklungsprozesses. Die Beiträge dieser Arbeit sind in der Anwendung des Prozesses klar erkennbar. Der schnelle und nahtlose Entwicklungsprozess reduziert die Entwicklungszeit signifikant von sieben auf drei Jahre im Vergleich zur Regelungssystementwicklung eines vergleichbaren Projektes. Durch den Einsatz des Prozesses wurden kritische Fehler in den Regelungssystemen frühzeitig identifiziert. Somit wurde das Risiko eines fehlerhaften Regelungssystems im Feld deutlich reduziert und eine schnelle, sichere und erfolgreiche Regelungssystementwicklung ermöglicht.

— *This page intentionally left blank* —

Abstract

In order to counteract climate change, a more sustainable energy supply system is inevitable. Therefore, generation from renewable energy sources is currently being introduced into electric energy supply systems in a distributed manner. Automatic control systems are necessary to control these distributed generation units and to ensure a stable supply of electric energy. There is a lack in specific processes and tools for consistent control system engineering because automatic control is new compared to the currently used technology of passively operated distribution grids. In order to fill in this gap, a process for rapid control prototyping for networked smart grid systems is developed, implemented, and validated in this thesis. At first, a concept is elaborated that combines the single steps of control system development to produce a consistent development process, with a focus on rapidness and risk mitigation during the development and deployment process. This process is then implemented utilizing a specific middleware, which supports the whole development process from the first draft through to the final control system in the field. The major contributions of the developed rapid control prototyping process are its' agile structure, the rapid and seamless field deployment of the developed control algorithms, and the mitigation of risk along the development process. The developed process, together with its middleware, is used in an industrial research and development project to develop low voltage grid control systems by using the following: Co-simulation of power and communication network, controller-hardware-in-the-loop evaluation, seamless field deployment, and simple refactoring of the designed control systems. Several control systems have been developed and are successfully maintaining power quality in three rural Austrian low voltage grids with a high share of renewable energy sources. The field operation results are used for the validation of the developed process in this work. The characteristic contribution of the process is advantageously manifested also in the field application. The process's rapidness and seamless field deployment result in a remarkable reduction of the time-to-field, from seven to three years, in comparison to a related control development project. Further to this, the risk reduction-focused process identified various potentially detrimental malfunctions at an early stage before field deployment. Thus allowing for the successful, safe, rapid development and deployment of control systems.

— *This page intentionally left blank* —

Acknowledgement

Should a Ph.D. thesis contain an acknowledgement? — This is what a Ph.D. candidate might ask themselves at a certain time. As you are currently reading mine, you can see that I think it should be contained in my thesis. This acknowledgement is not only for expressing my gratitude for the support I received in the creation of the thesis. For me, the time as a Ph.D. candidate was a journey with many experiences that involved learning, discussions, doubt, exchange, change, success, and personal evolution. Most relevant along the way were the innumerable people that accompanied me with their motivation, support, and inspiration. For all these things, especially the companionship along the way, I want to express my gratitude.

Foremost, I want to thank my colleagues from TU Wien and especially Friederich Kupzog, who introduced me to the field of future energy systems by providing me with a topic for my master thesis and hiring me to join his research group at TU Wien over a big portion of ribs. Thank you, Friederich, for the chances you have given me, for your motivation and trust, and for being my boss, colleague, and mentor for all the years.
Furthermore, I want to thank my colleagues at Siemens CT Austria, especially Ralf, Alfred, Tobias, and Andreas, for the insights and opportunities that they have given me, their technical brilliance, their motivating words, and the ongoing cooperation.
Special thanks go to the Austrian Institute of Technology and my colleagues there for supporting me with my thesis and also to OFFIS in Germany. At OFFIS, I want to thank Sebastian Lehnhoff, Sebastian Rohjans, and their colleagues for hosting and supporting me and for introducing me to the North German traditions.
I also want to thank my Ph.D. supervisor, Dietmar Dietrich, for his trust in me and his experienced, unconstrained, and open guidance.

Finally, I want to thank my dear family and my beloved darling for supporting me in all of my decisions and for encouraging me wherever and whenever they could.

Mario Faschang
Vienna — June 29, 2015

— *This page intentionally left blank* —

Table of Contents

Nomenclature & Abbreviations

(N)ACK	(not) acknowledge
3D	Three dimensional
AC	Alternating current
ACE	Area control error
AFC	Automatic frequency control
AMI	Advanced metering infrastructure
API	Application programming interface
CAD	Computer-aided design
CAPEX	Capital expenditure
CHIL	Controller-hardware-in-the-loop
CHP	Combined heat and power
CIM	Common information model
CSV	Comma-separated values
DAQ	Data acquisition
DER	Distributed energy resources
DG	Distributed generation
DLL	Dynamic link library
DSL	DIgSILENT simulation language
DSM	Demand side management
DSP	Digital signal processor
DUT	Device under test
EC	European Commission

EGDA	Express grid data acquisition
EPRI	Electric Power Research Institute
EV	Electric vehicle
EVSE	Electric vehicle supply equipment
FACTS	Flexible alternating current transmission system
FPGA	Field programmable gate array
GHG	Greenhouse gas
GUI	Graphical user interface
HIL	Hardware-in-the-loop
ICT	Information and communication technology
IDE	Integrated development environment
IGBT	Insulated-gate bipolar transistor
ISS	Instruction set simulator
JSON	JavaScript object notation
LFC	Load frequency control
LV	Low voltage
MIMO	Multiple-input multiple-output
MV	Medium voltage
OLTC	on-load tap-changer
OPEX	Operational expenditure
PC	Personal computer
PCB	Printed circuit board
PHIL	Power-hardware-in-the-loop
PLC	Programmable logic controllers
PR	Process requirements
PRC	People's Republic of China
PS	Primary substation
PS	Problem statement
PV	Photovoltaic(s)

RES	Renewable energy sources
RMS	Root mean square
RTS	Real time digital simulator
SCADA	Supervisory control and data acquisition
SGAM	Smart grid architecture model
SIL	Software-in-the-loop
SS	Secondary substation
SSH	Secure shell
STATCOM	Static synchronous compensator
SVC	Static var compensator
TCP	Transmission control protocol
THD	Total harmonic distortion
TLS	Transport layer security
UK	United Kingdom
UML	Unified modeling language
USA	United States of America
XML	Extensible markup language
XP	Extreme programming
XSD	XML schema description

1 Introduction

Energy has always been a basic requirement of human life. This fact is also reflected in the definition of the word *energy* as “the ability of doing work” [Way13, p. 96]. As stated by the *conservation law of energy* [Way13, p. 97], energy cannot be created or destroyed and the amount of energy in a closed system always stays the same. There is only the possibility of converting energy from one form to another. Saying this leads to the need to distinguish between the basic forms of energy: Kinetic energy, on the one hand, is contained in moving objects, and potential energy, on the other hand, which is (chemically, thermally, electrically etc.) stored energy.

Since the late 19$^{\text{th}}$ century electric energy has become established as one of the most relevant forms of energy due to several benefits, listed in Section 1.1. This introduction with its following sections describes the increase of demand in, and the classical generation and distribution of electric energy. Subsequently the evolution in nowadays’ power supply systems by the introduction of distributed alternative energy sources is discussed. From that evolution, problems in grid stability and controllability emerged, that lead to the motivation of the present thesis. At last in this chapter the methodical approach for tackling the described problems is covered.

1.1 Classical Electric Energy Supply Systems

This section describes the growing trend of electric energy consumption, its generation from various sources, as well as the classical transmission and distribution via various voltage levels.

Electric energy is one of the most suitable forms of energy with outstanding importance for humanity due to its following benefits [Cra07].

- Convenient generation from many other primary energy sources
  (e. g. kinetic, thermic, fossil energy)
- Highly efficient transformation of voltage level, alternating, and direct current
- Convenient and highly efficient transformation into kinetic or thermal energy
- Highly efficient transmission among long distances
- Wide field of application for customers and industry
  (e. g. lighting, heating, kinetic energy, etc.)

All of these benefits have made electric energy an indispensable resource for our daily life since its introduction in the late 19th century. Modern information and communication technology, automation and control processes, as well as most of the industrial production processes all over the world would not be available without electric energy. Electric energy now is one of the world's major final energy forms. A share of about one third of the world's gross primary energy consumption is used to generate electric energy [OO11, p. 3].

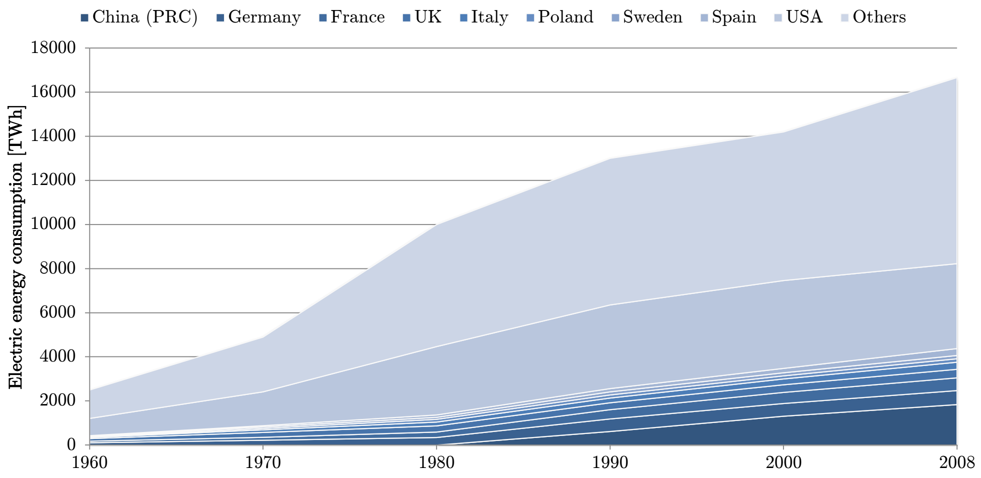


Figure 1.1: World gross electric energy consumption in TWh from 1960 until 2008 [OO11, p. 4].

Figure 1.1 shows the worldwide evolution of electric energy demand. Until the year 1980 the worldwide generation doubled every decade. Later its yearly growth rate reduced to about ten percent [OO11, p. 4]. Nevertheless, especially emerging countries in Asia and India with rapidly growing economies will have a strong demand for electric energy within the next decades.

To be able to cover the increasing demand in electric energy – which is depicted in Figure 1.1 – more and more power plants, as well as a reliable transmission and distribution grids have been installed. As there is no efficient way of storing electric energy in the power distribution system, generation has to follow the consumption of electric energy at any time. Therefore, various types of power plants with different characteristics concerning their generation flexibility are in use to be able to follow the customers' highly variable consumption curve.

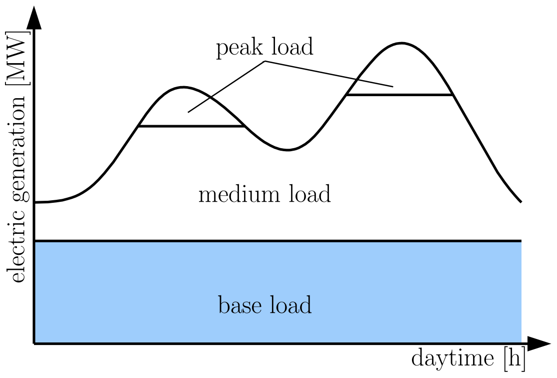


Figure 1.2: Schematic electric load curve with classification consisting of base load, medium load, and peak load.

Figure 1.2 shows such an exemplary electric load curve for one day together with a load classification. Such load curves typically show the total energy consumption of several (hundreds or thousands of) consumers. Depending on the required flexibility and other local parameters (e. g. fuel availability, geography, etc.), different types of power plants are used for electric power supply within the distinct load classes depicted in Figure 1.2 and listed in Table 1.1.

Table 1.1: Load classification and corresponding power plants

base load	**medium load**	**peak load**
• hydroelectric power plants • biomass power plants • nuclear power plants • coal power plants	• hydroelectric power plants • steam turbines based on natural gas or heavy fuel oil • coal power plants	• hydroelectric power plants • gas turbine power plants

This non-exhaustive classification is based on the duty of the power plant and its flexibility concerning run-up and run-down time. Base load power plants are optimized to generate electric energy at low cost and nearly constant output power. These power plants have typical run-up times between two hours to several days. In contrast there are peaking power plants which can provide electric energy to a system very quickly, but in general at much higher generation costs. In between these two types are the load following power plants providing power for the medium load band. These power plants are capable of varying their output power, but still provide high efficiency.

One interesting aspect of the listed power plants and classifications is, that hydroelectric power plants are capable of supplying both, base load and peaking power, as well as supplying energy in the medium power range. Another benefit of (pumped) hydroelectric power plants is their ability to consume electric energy in times of an surplus in the energy market.

The power plants listed in Table 1.1 have diverse energy sources and are optimized for supplying energy for various load characteristics. What they have in common is their feed in voltage level and current type, which generally uses to be alternating current (AC), either in the medium or high voltage level. This is due to the kind of generation (rotating three-phase generators driven by large turbines) and the convenience, that AC power can effortlessly be transformed to other voltage levels in a highly efficient way. The high voltage level is beneficial for the transmission of electric energy among long distances due to the reduction of losses ($P_{Loss} \propto \frac{1}{U^2}$) and thereby enabled savings in cables and lines (smaller diameters, reduced weight).

Figure 1.3 shows the classical hierarchical top-down energy distribution structure, which has established all over the world for efficiently supplying consumers with electric energy. According to [OO11, p. 219f.] in that classical structure electric energy is primarily generated by big power plants feeding in at very high voltage levels (380–1,000 kV). Other (medium sized) power plants are feeding in at reduced, but still high voltage level (110–220 kV). These different voltage levels are interlinked via transformers enabling energy transfer between them. Below the two high voltage levels – which are primarily connecting (high power) generation units and transmitting power among long distances – the medium (10–30 kV) and low voltage levels (0,4 kV) are located. On these two lower voltage levels power is distributed among industrial and private consumers and also some low power generation units may be connected.

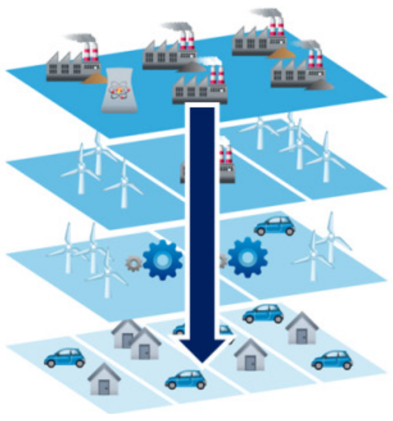


Figure 1.3: Hierarchical top-down energy distribution structure (with friendly permission of Manfried Kruska).

1.2 Paradigm Shift in Modern Electric Energy Supply Systems

In the previous section the classical power generation, transmission, and distribution system was described. This structure has evolved since the first days of electrification. Some improvements concerning efficiency and stability of the system were made, but generally the classical top down generation and distribution system remained the same.

Contrary to this conservative behaviour, in the past decade a revolution in nowadays electric power supply systems – concerning mainly the way of power generation and transmission – has been driven due to several issues, which are addressed in the following two subsections.

1.2.1 Changes in Power Generation Infrastructure

The world is suffering from air pollution, a high amount of greenhouse gas (GHG) emissions and global warming caused by these emissions. Approximately 25 % of the worldwide GHG emissions are related to heat and power production [Aic12, p. 10], primarily from coal and oil fired power plants. The main goal of the Kyoto Protocol of 1997 [18] is the reduction of worldwide emissions by 5,2 % (8,0 % in the European Union) compared to 1990 until 2012. Due to the burden sharing process the goal for Austria was a reduction by 14 %, which was not met by far [Eur13].

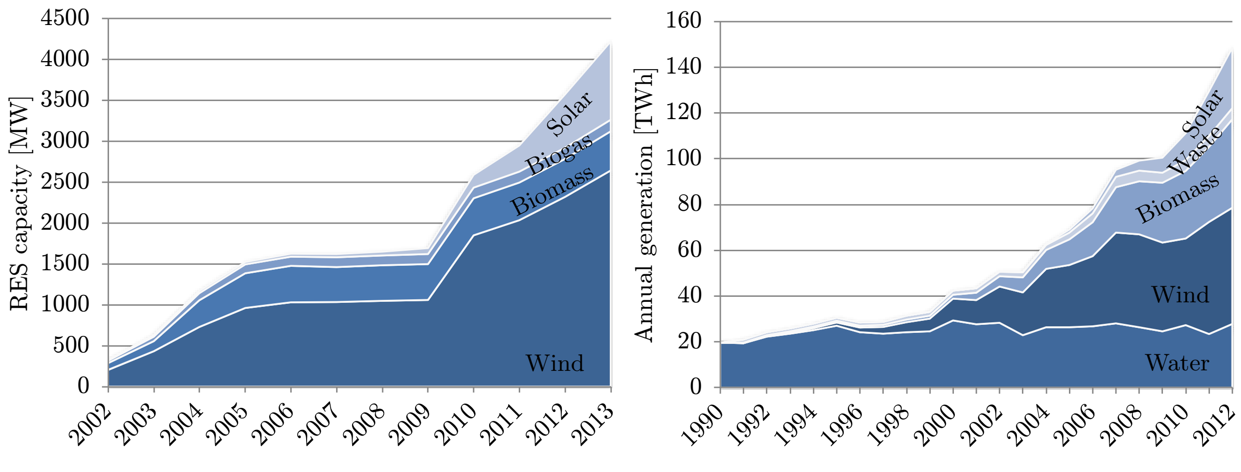
To counteract these problems and to facilitate a promising source of energy, from the middle of the 20th century on, nuclear power plants have been introduced to the power generation side. This seemed to be a suited solution to the emission problem, due to the nuclear power plants extremely low GHG emission rate and cheap generation costs. But, because of several devastating accidents in nuclear power plants – the most recent one in 2011 in Fukushima (Japan) – and the questionable solutions for the deposition of nuclear waste, criticism grew and eventuated in a nuclear power phase-out in several countries [HH11, p. 16ff.].

The main approach that has been pursued in recent years for the reduction of both, GHG emissions and the dependency on primary fossil energy imports – from potentially politically instable foreign countries – is harnessing of renewable energy sources (RES). These are unlimited or regrowing natural energy sources that can be utilized with reasonable effort and may only be available at specific times or locations. Such renewable energy sources are, for example, tidal energy, geothermal energy, as well as direct and indirect solar energy.

Despite the fact that indirect solar energy is already used as a high-yielding renewable source of energy in the form of biomass and hydro power, its utilization is still funded by national and international organizations. This makes sense due to the fact that the amount of solar energy that reaches the earth's surface is more than 10,000 times higher than the world's overall energy consumption [Cra12, p. 10f.]. Thus power generation from direct and indirect solar energy should be further expanded, as it is the only form of energy to use after the end of the fossil era.

In [Reb02, p. 293ff.] is noted, that in many regions of the world (e.g. Saudi Arabia, California, Andalusia) big solar power plants (solar dish-systems, parabolic troughs, heliostats) have been installed that are using concentrated solar power for electric power generation. This is highly efficient in regions of strong and constant solar irradiation. In Central Europe, however, this is not a suited solution due to the less and inconsistent solar irradiation power. Under these circumstances smaller distributed units serve best for power generation from solar irradiation.

Despite this disadvantage the potential for solar energy in Central Europe is high enough. To better use this immense potential for energy generation from solar power and thus to increase the yield from wind and solar radiation, many countries and nations set up road maps and action plans. For example, the European Parliament adopted a directive "on the promotion of the use of energy from renewable sources" in 2001, which was updated in 2003 and 2009 [EP09]. By this directive, all member states committed to increase their share of energy generation from RES to reach an average share of 20 % until the year 2020. This directive was transposed into local laws and road maps (e. g. the "Ökostromgesetz" in Austria [Bun12] or the "Erneuerbare-Energien-Gesetz" in Germany [Bun09]) and lead to an immense increase in generation from RES in these and also several other countries.



(a) Total installed RES capacity in Austria (2002–2013)

(b) Gross electricity generation from RES in Germany (1990–2012)

Figure 1.4: Increase of RES-utilisation in Austria and Germany between 1990 and 20013 [6, 4].

Figure 1.4 shows the significant increase of total installed RES capacity in Austria and the growth of gross electricity generation from RES in Germany since 2002 and 1990, respectively. In this figure a clear trend towards energy generation from renewable sources and especially from wind and photovoltaic (PV) power plants can be seen. Even though Austria lags a bit behind Germany concerning the installations of PV power, an immense increase in these renewable energy sources is anticipated.

It is obvious that the efforts to increase the share of RES are succeeding and leading to a re-structuring in the world's (electric) energy generation, which has already begun. Unfortunately, these new sources of energy require an adoption in the energy distribution system due to their characteristics in power generation and availability.

1.2.2 Changes in the Energy Distribution System

The previously mentioned changes in power generation infrastructure, which are induced by the need of GHG emission reduction and the increase in independence from fossil energy imports from foreign countries, are challenging the classical energy distribution systems. This is caused by the characteristics of RES that concern the three issues listed below.

1. **Fluctuation and availability**
   Contrary to nuclear or coal fired power plants that offer a constant and smooth generation curve, power from renewables is not available constantly during the day and not ready for delivery on demand at any time.
2. **Local nature and distribution**
   The location of the renewable power plants is predefined for many renewable energy sources by their primary energy source. It makes most sense to locate wind turbines and PV panels at places of high and constant wind or solar irradiation. Furthermore, renewable generation units are less likely centrally located but rather distributed over wide areas (e.g. in private households) with varying density.
3. **Efficiency**
   Transmission and transformation of electric energy always comes along with losses caused by the resistance of the transmission line, as well as magnetic and commutator losses. Therefore it is most efficient to have electric energy generated close to the location of consumption in order to avoid transmission among long distances and multiple voltage levels.

One of the major changes caused by these characteristics is a paradigm shift away from big centralized high-power power plants (like nuclear or coal fired power plants) towards several small or medium sized power plants for generation from RES (like wind turbines, PV panels, CHP units, and hydro-electric generators), which are regionally distributed. This distribution of renewable energy generation units dilutes the hierarchical, top-down energy distribution structure, which was described previously at the end of Section 1.1 and depicted in Figure 1.3. Through this a more flattened and peer-to-peer like energy generation and distribution system emerges.

This new energy distribution system, which is schematically depicted in Figure 1.5, builds up on the classical energy transmission and distribution system but contains several changes and additions that can be summarized as follows.

According to Kupzog et al. [KP11] generation shifts from comparatively few big power plants in high voltage levels towards several additional small and medium sized renewable power generation units in medium and low voltage levels (e.g. small wind turbines or private PV installations). As generation moves from the high voltage levels down to the lower voltage level where the classical consumers are connected to the low voltage distributed grid, these consumers change their role and – for example by the installation of PV panels or CHP units – become so called *prosumers*.

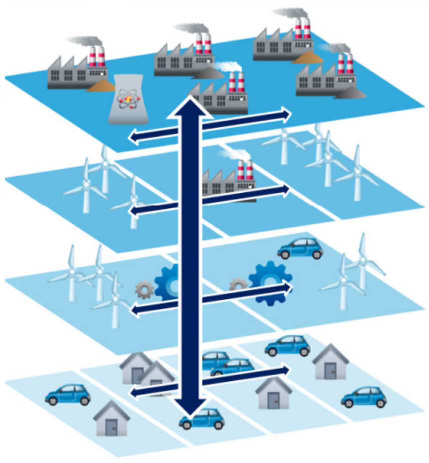


Figure 1.5: *Smart Grid* energy distribution structure (with friendly permission of Manfried Kruska).

This means, that they do no longer only demand electric energy from the distribution grid, but also feed in electric energy from their own renewable power generation unit, when their local generation exceeds their own consumption [PD11]. With an increasing number of *prosumers* also electric energy generation in the medium and low voltage level increases rapidly. Thus, electric energy is (depending on the amount of generation and consumption) distributed horizontally within the low voltage distribution grid or even fed backwards to higher voltage levels in times of high renewable generation.

These changes are mainly triggered by the integration of RES and their characteristics. For the efficient and successful integration of these new distributed energy sources some more challenges arise which are discussed in the following chapter.

1.3 Challenges and Scope

The task of demand and generation matching becomes more challenging due to the unpredictable and highly fluctuating behaviour of renewable energy and the distribution of their generation units. Unlike the classical power generation philosophy, where generation had to follow the consumption curve at any time, in the new energy distribution system also the demand side has to be integrated much more in the process of demand and generation matching. This process is called demand side management (DSM) and plays a crucial role concerning the integration of fluctuating RES, due to their often highly variable and unpredictable generation behaviour [PD11]. Not only the integration of RES is challenging the distribution grid, but also new high power consumption units like electric vehicles (EV) or electrical heating systems like heat pumps.

These new units in the power grid – both on the generation and on the consumption side – lead to two major challenges. The first can be referred to as the *grid bottleneck*. These are regional problems caused by high electric power, either fed in or consumed, regarding transformers, power lines, house connections, and protective devices on the distribution grid level (e.g. high density of PV power in south Germany – cf. Figure 1.6). The second major challenge concerns the supra-regional problem of balancing electric power generation among long distances. High local concentration of RES (e.g. offshore wind power in the North Sea – cf. Figure 1.6) leads to an

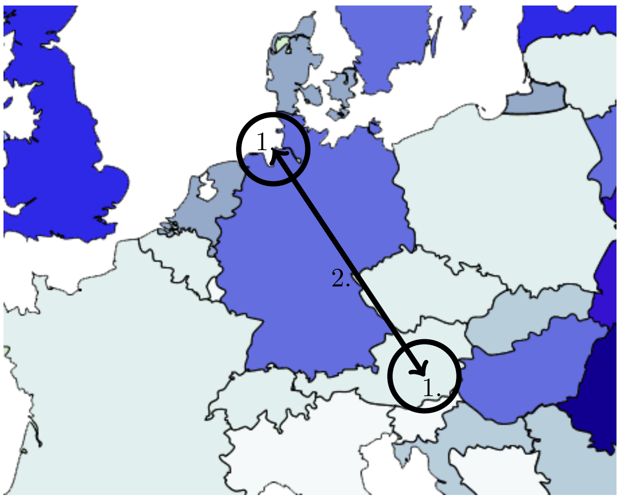


Figure 1.6: Examples for grid (1) and balancing bottlenecks (2) in the European transmission system.

immense local oversupply and thus to stress in the supply grid – referred to as the *balancing bottleneck*.

These problems could be overcome by an extensive expansion of the power distribution system with new power lines and transformers to increase transmission capacity and new standby power plants for the occasion of renewable energy shortcomings.

This way of facing the previously mentioned challenges has several drawbacks following [CW12, p. 586f.]. First condition to mention is the financial aspect. The proposed expansion in grid infrastructure would be extremely costly and would cause an enormous use of resources and labour force. Second is the people's commitment. Especially in populated areas there is high resistance of inhabitants against new power lines over their houses or gardens [GRSW12, p. 11ff.]. Third, and last argument to bring up is the efficient and economic use of grid infrastructure. The mentioned expansion would not only be very costly, it would also reduce the already very low utilization factor of power grid infrastructure even further. It is economical highly inefficient to build standby power plants only for the sake of compensating wind and PV power shortcomings, or to install further power lines only to be able to supply energy during a several minutes lasting load peak in the evening. Therefore optimization of existing grid infrastructure is preferable to the expansion (e.g. stated by the German Bundesnetzagentur in [2]).

Beside the challenges concerning both regional and supra-regional grid capacity, the broad introduction of RES and the related changes in the energy distribution system (cf. Section 1.2.2) result in necessary changes in grid planning, operations management, and grid connection. To allow for these changes and thus to be able to provide a functional, stable and save energy supply system, features like remote fault detection, automatic service restoration and advanced monitoring are of high interest [KP11, p. 36-5].

To enable these newly necessary features and to avoid the aforementioned extensive extensions, by still fostering the integration of RES, information and communication technology (ICT) is introduced to the power grid. Equipping renewable energy sources, loads, and grid infrastructure with ICT allows for all these new features and brings further benefits in this expanded kind of power grids, which are commonly referred to as *smart grid* and concretely introduced in Section 2.2.

To summarize, the challenges for the future low voltage power grids are the elaboration of applications and control strategies, and to find ways how the newly won opportunities could be used best by grid operators, utilities and customers. Because of this, it is necessary to develop new strategies, new roles may appear in the deregulated energy marked, and new business models will emerge [Aic12, p. VIII].

The focus of this thesis is therefore clearly set on the methods and tools for the development of grid control algorithms. These algorithms are the key element of any *Smart Grid* control application. Due to the high variability of control scenarios, the focus lies not on specific control strategies or algorithms, but rather on the process and methods for prototyping, testing, simulating, evaluating, deploying, and refactoring of distribution grid control systems.

1.4 Motivation and Problem Statement

After the preceding introduction and the description of the upcoming challenges, this section summarizes the motivation for this thesis , followed by the emerging research question. Deduced from the research question, the explicit problem statement – which is the basis for this work – is formed. Subsequently, the methodological approach is stated in the next section.

Based on the plans of the European Parliament for the promotion of the use of energy from renewable sources (cf. [EP09, EUD12]), many countries are making high efforts to reach the committed goals. Doing so, the share of electric energy from renewable sources increased rapidly since the beginning of the 21st century (cf. Figure 1.4 in Section 1.2.1) and is likely to rise even further, to reduce GHG emissions and to reach the goal of sustainable and independent power supply.

Due to the mainly fluctuating and intermittent characteristics of RES in Central Europe, power generation from these sources is most efficient either in concentrated form in regions of high and constant availability (e.g. offshore wind farms) or in distributed form among the countryside (e.g. solar or wind energy). From these kinds of energy generation from RES, two major challenges arise within the electric transmission and distribution grid that can be summarized as the grid bottleneck for local/regional power issues and the balancing bottleneck for supra-regional energy balancing issues.

These challenges can be faced by the utilization of the newly available grid entities – namely sensors and actuators like smart meters, transformers' OLTCs, inverters, EV charging equipment and several other. They became available due to the push in technology that has been forced by the introduction of RES. To be able to utilize the sensors and actuators via the ICT network, which is a core element of every *Smart Grid*, not only hardware but also software tools are needed.

So far, electrical industry has done a good job, keeping pace with the rapid evolution in power electronics and electric systems hardware. Affordable and practical sensors and actuators have been developed and also industrial communication systems and *Smart Grid*-related communication protocols (e.g., OpenADR [12] and IEC 61850 [8]) are available. Unfortunately this is not the fact with software tools for supporting industrial engineers in control system engineering. It is not an easy venture to bring newly designed control algorithms to the power grid, as they cannot easily be tested and evaluated without the risk of grid instabilities or even blackouts. This makes it even more relevant for industry to have reliable tools at hand for control development and deployment.

Due to these facts, the research question of this thesis is formed as follows.

> *Can control systems for low voltage distribution grids efficiently be designed, implemented, and tested in such a way that there is as little risk as possible and that the controlled power grid is not adversely affected?*

Derived from that research question, which itself results from the previously presented challenges for future power grids, several problem statements (PS) are formulated to be answered in this thesis.

PS 1 – Investigate the method of control system development, identify related stakeholders, and define relevant requirements.

- **PS 1a** – Research methods for control system development in other application domains.
- **PS 1b** – Analyse tools which can be used in a rapid control prototyping process.

PS 2 – Elaborate and define a concept for rapid control prototyping for smart low voltage grids, which fulfils the requirements defined in PS 1.

PS 3 – Realize a fully integrated implementation of the concept, which has been elaborated in PS 2.

PS 4 – Finally, discuss and evaluate the suitability of the created rapid control prototyping process.

1.5 Scientific Research Method

The methodology chosen for this work follows the inductive-hypothetical research strategy, which is a typical top-down approach. It works from the more general to the more specific and consists of five stages depicted in Figure 1.7. According to Bonin [Bon14, p. 86f.], several benefits (e.g. the permission of continual feedback and learning, the enabling of the generation of various of solutions for the problem at hand, etc. [Sol82]) make this research strategy very useful for new and emerging research fields.

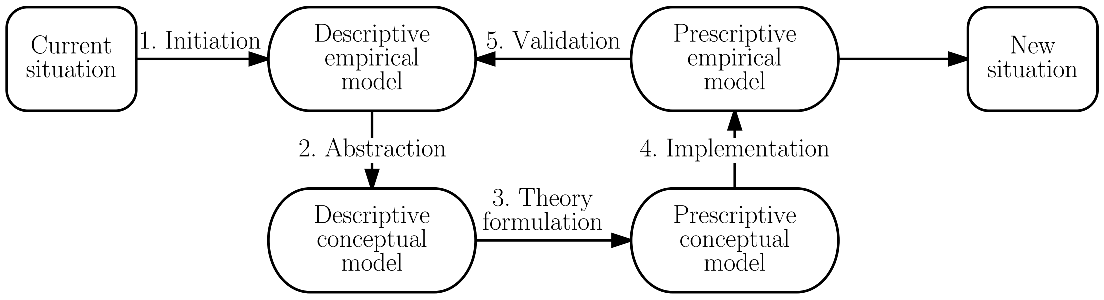

Current situation
1. Initiation
Descriptive empirical model
5. Validation
Prescriptive empirical model
New situation
2. Abstraction
4. Implementation
Descriptive conceptual model
3. Theory formulation
Prescriptive conceptual model

Figure 1.7: Inductive-hypothetical research strategy [Sol82].

Starting with the current situation, in the first research phase *Initiation*, observations of the current state of power distribution system in general and the concept of smart grids and the adjunct integration of RES in specific are made. This state-of-the-art is derived from literature research, talks to systems experts, within ongoing Austrian and international research projects and on (inter)national conferences, and summarized in Chapter 1 and Chapter 2 of this thesis.

From this empirical description, the requirements are extracted within the *Abstraction* phase on the basis of use case descriptions derived from the list of stakeholders, which are integrated in the prototyping process. This list of process requirements (Section 2.3.2) forms the conceptual description, that is used to formulate the concrete process for rapid control prototyping – in the *Theory formulation* phase. The process for rapid control prototyping, which is the main outcome of this work, is the prescriptive conceptual model of the inductive-hypothetical research strategy (and presented in Chapter 4). To allow for the final step, which is the *Validation* of the developed process, it is *implemented* (better said *applied*, as this work focus on a process) and one concrete example for low voltage grid control development (for an Austrian distribution grid) is presented.

Speaking about the validation directly leads to the question of the applied research methodology. In the field of computer science and related disciplines, Shaw [Sha02] analysed several hundreds of scientific publications and created a classification of the type of research question, the proposed solution, and the type of validation; summarized in Figure 1.8.

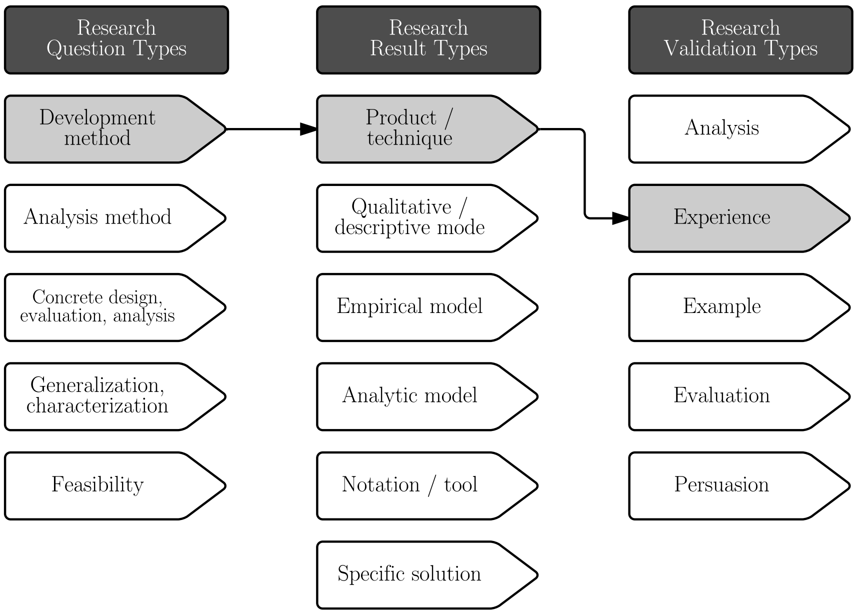


Figure 1.8: Categorization of research question, result, and validation; highlighted is the selected approach for this work (based on [Sha02]).

In this work the prosed framework is used as guidance for a clear methodology and to help both the author and the reader to stay focused on the research question, the adjunct result and the methodology for its validation. The research question of this thesis, which is presented in the previous Section 1.4 together with the resulting concrete problem statements, is categorized as *development of a method* (for rapid control prototyping for networked smart grid systems). Results of research can vary, depending on the novelty and complexity of the field of research. Research of modern power grid techniques is quite mature. Thus, currently ongoing research activities create clear and concrete results, which are solving concrete problems. This is also the case for this work that presents a *technique* (in the form of a clear process) according to the categorization of result types in Figure 1.8.

Categories of validation techniques are listed in the third column of the figure. For showing the soundness of the results presented in this work, validation is carried in a twofold way. Due to the direct usage of the process within an Austrian research project (cf. Section 2.2.3.2) by a set of domain experts and engineers, where a low voltage grid controller has been designed that operates up to three control algorithms in parallel in three rural distribution grids, much experience has been gained concerning the soundness of the presented process. Additionally, the process has been adopted and used for the creation of an EV evaluation system [Nö14] and will be extended to a validation system for smart grid equipment in the lab of AIT – Austrian Institute of Technology [NLS14]. All these actions can be categorized as empirical validation by *experience* with respect to the validation type categorization of Shaw [Sha02].

2 Context and Requirements for Rapid Control Prototyping

Control system and algorithm prototyping is relevant for a broad field of engineering disciplines, e.g. in automotive engineering, chip design, and building automation. As stated in the previous introduction, control development is relatively new to electric energy distribution systems and especially to low voltage distribution grids. In this chapter, the distribution grid related context for this novel task is analysed. Relevant context is the current state of power grid control and the power quality requirements defined by EN 50160. Furthermore, the general characterizing structure of networked smart grid systems with a focus on distribution networks is analysed on the basis of existing smart grid field installations. The third and last relevant contextual topic presented here, is the Smart Grid Architecture Model. Rapid control prototyping is embedded within this internationally acknowledged framework. From this related context, requirements for rapid control prototyping for networked smart low voltage grids are extracted by a requirements engineering methodology suggested in the IntelliGrid approach. These requirements serve as the basis for the concept presented in the next chapter.

2.1 Classical Power Grid Control

The main objectives of classical power grid control are to keep the electrical supply system stable by balancing supply and demand at any time and to guarantee defined quality parameters to the customers. An adequate definition of power system stability from a control system perspective is given by Kundur et al. as *"the ability of an electrical power system, for a given initial operating condition, to regain a state of operating equilibrium after being subjected to a physical disturbance, with most system variables bounded so that practically the entire system remains intact."* [KPA+04]

The classical power grid control tasks are primarily accomplished by cascaded frequency control and voltage control. Due to the direct relation of a power grid's alternating current's frequency and the (rotating) generators' inertia, the frequency is an universal, pervasive indicator for the (im)balance within an electric energy system. On the contrary there are the voltage levels, which are local values determined by the impedances of the power grid elements, such as loads, power lines, cables, transformers, etc. Within the following subsections the power quality objectives defined by the European norm EN 50160 are discussed, followed by a description of established control methods for frequency and voltage control.

2.1.1 Power Quality according to EN 50160

For safe and failure-free operation of electric appliances, network operators are legally bound to guarantee specific power quality criteria. For the European Union, these criteria were elaborated and applied by CENELEC within the European norm EN 50160 [CEN11]. The purpose of this norm is the definition, description, and determination of the power quality criteria concerning frequency, magnitude, waveform, and symmetry of the three phase system.

Table 2.1: Power quality criteria according to EN 50160 [CEN11, p. 12ff.].

Property	**Criteria limits**	**Remarks and description**
Frequency f	$49.5\text{ Hz} < \overline{f} < 50.5\text{ Hz}$	99.5 % of all 10 sec. average values
	$47.0\text{ Hz} < \overline{f} < 52.0\text{ Hz}$	100 % of all 10 sec. average values
Relative voltage u_r	$0.9 < u_r = \frac{U_{RMS}}{U_c} < 1.1$	95 % of all 10 min. average values within a week; U_c is U_n (230 V) in LV grids and any other contracted voltage U_c in MV grids
Unbalance of 3-phase voltage	$\frac{\overline{U}_{RMS,inverse}}{\overline{U}_{RMS,direct}} < 0.02$	95 % of all 10 min. average values of the root mean square (RMS) components of the inverse system in relation to the related direct system
Flicker P_{lt}	$P_{lt} < 1$	95 % of all 2 hr. integration intervals (long term) within a week
Voltage dips (10 ms $< t <$ 60 sec.)	$N_{VD} < N_{VD,max}$	Number of voltage dips N_{VD} ($u_r < 0.9$) with $N_{VD,max}$ is 10 to 1,000 per year
Short interruptions (t $<$ 3 min.)	$N_{SI} < N_{SI,max}$	Number of short interruptions N_{SI} ($u_r < 0.05$) with $N_{SI,max}$ is 10 to several 100s per year
Long interruptions (t $>$ 3 min.)	$N_{LI} < N_{LI,max}$	Number of long interruptions N_{LI} ($u_r < 0.05$) with $N_{LI,max}$ is several 10 to 50 per year
Temporary overvoltage	$U_{RMS} \leq U_{max}$	U_{max} shall not be exceeded at any time; with $U_{max} = \begin{cases} 1.5\,kV & \text{LV grids,} \\ 1.7 \cdot U_c & \text{MV grids (grounded),} \\ 2.0 \cdot U_c & \text{MV grids (isolated).} \end{cases}$
Total harmonic distortion (THD)	$THD < 8\%$	Relation of the summarized power of the 2nd to the 40th harmonic to the power of the fundamental frequency

Table 2.1 shows a summary of power quality criteria according to EN 50160. All of these criteria can be assigned to one of two types. Either they are specifically defined (e.g. frequency or voltage magnitude) or they are of rather indicative type (e.g. number of voltage dips and interruptions). The latter rather imprecise criteria have to be used because of the specific nature of the related measures or their strong local dependencies. These indicative values are, for example, the number of voltage dips or the amount of short and long supply interruptions. Voltage dips can be

distinguished from short or long supply interruptions by their duration and resulting voltage magnitude. Another type of nonconforming voltage situations is temporary and transient overvoltage, which may be caused by switching operation in connected grid elements and induction caused by lightning.

While these under- and overvoltages typically appear as sporadic non-repeating events, flicker describes repeating voltage deviations, which result in optically recognizable, disruptive variations of luminance [Cra07, p. 355]. IEC 868 [IEC86] and its updated version IEC 61000-4-15 [IEC10] distinguish between short and long term flicker (P_{st}, P_{lt}) and describe a method for the measurement of the short term flicker severity level. $P_{st} = 1pu$[1] is defined as the limit for human irritation focusing on 8.8 Hz, which might be dangerous for people prone to epilepsy. The severity level for long term flicker can then be derived from that value, following Equation (2.1), and is relevant for power quality assessment according to EN 50160.

$$P_{lt} = \sqrt[3]{\sum_{i=1}^{12} \frac{P_{st_i}^3}{12}} \tag{2.1}$$

Another measure of quality in power grids, which is of high relevance for this work, is the unbalance of 3-phase voltages. It is a specifically defined measure that can be determined by the method of symmetrical components. The unbalance of 3-phase voltages is the relation of the 10 minute average RMS values of the inverse system to the related direct system. To be conform to EN 50160 at the customer's point of supply, this relation has to be below 2 % during a one week observation period. The total harmonic distortion (THD) sets the summarized power of the harmonics (2nd to 40th) in relation to the power of the fundamental frequency (cf. Equation (2.2)).

$$THD = \sqrt{\sum_{h=2}^{40} {u_h}^2} \quad \text{with} \quad u_h = \frac{U_h}{U_1} \tag{2.2}$$

Further attention is also paid to specific amplitude limits for each harmonic component and limits for signalling voltage, for which the norm EN 50160 [CEN11] should be consulted.

2.1.2 Automatic Frequency Control

The primary goal of control systems in power grids is the balance of electricity generation and consumption. An (im)balance is directly mapped to the frequency (deviation) of the sinusoidal waveform of the grid voltage, which relates to the rotating mass (turbines, rotors) of the generators. To avoid too large frequency deviations – either caused by faults in the grid or at the generation side – in a power system, automatic frequency control (AFC) is used. The AFC mechanism consists of automatic primary and (mostly) automatic secondary frequency control mechanisms, which operate – beside other longer-time-scale control systems – in a specific temporal order; depicted in Figure 2.1.

[1]per unit-value

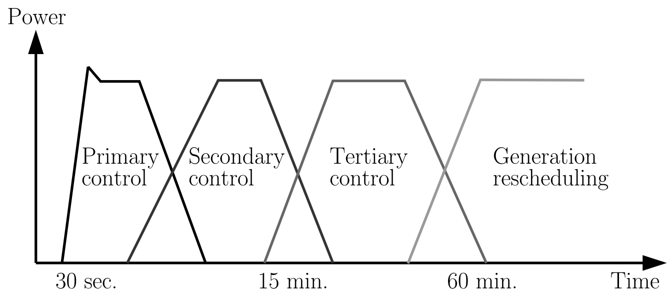


Figure 2.1: Frequency control time characteristics

2.1.2.1 Primary Frequency Control

Primary frequency control is done locally by each participating power plant to counteract frequency deviation instantaneously and to bring grid frequency back to short term acceptable values; with a remaining constant frequency error due to the applied purely proportional control law [Kun94, Chap. 11.1.5]. As each participating power plant counteracts frequency deviation independently and without a coordinating entity, it is not possible to use integrating frequency control components. This is due to the fact that integrators of different power plants may influence each other and thus lead to unreasonable power generation distribution among the single generation units.

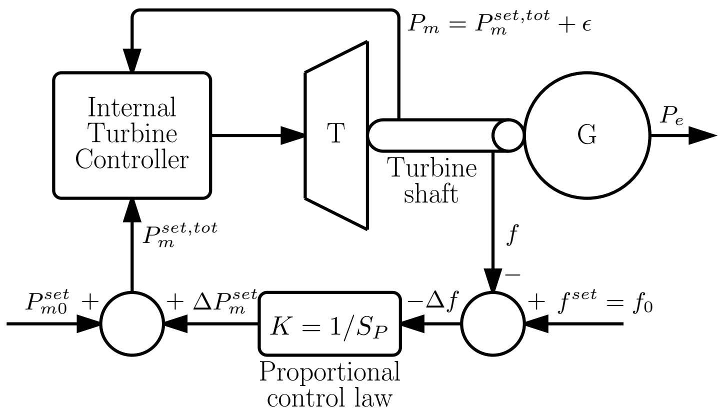


Figure 2.2: Block diagram describing the primary control law [And12].

A block diagram of a typical turbine driven power plant operated with a proportional feedback control is depicted in Figure 2.2. The generator G is driven by the turbine T, which is controlled by an internal turbine controller that controls the turbine to generate the desired mechanical output power $P_m^{set,tot}$. The deviation Δf of frequency f from the set frequency f^{set} is converted via the proportional control law (droop control) $K = 1/S_P$ with S_P being its frequency droop characteristic and K – the droop's inverse value – the stiffness of the generator [HM14, Sect. 4.2].

The frequency droop characteristic S_P is determined by steady state analysis of the control system. Equation (2.3) describes the primary frequency control law according to the nomenclature of Figure 2.2.

$$(P_{m0}^{set} - P_m^{set,tot}) + (f_0 - f) \cdot \frac{1}{S_P} = 0 \tag{2.3}$$

The frequency droop characteristic can be derived from Equation (2.3) to

$$S_P = -\frac{f - f_0}{P_m^{set,tot} - P_{m0}^{set}} > 0 \tag{2.4}$$

and is depicted in Figure 2.3.

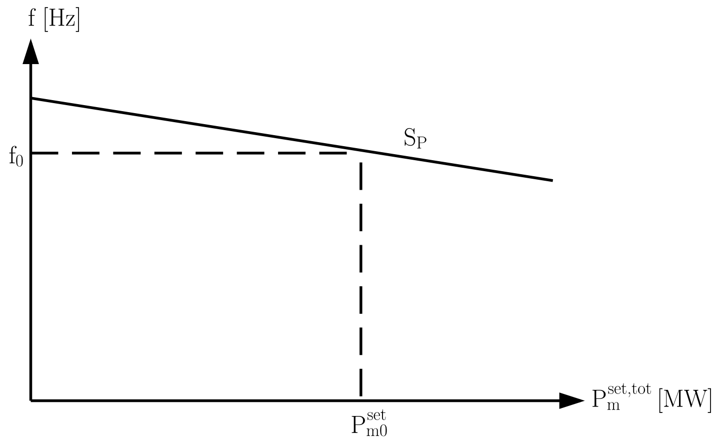


Figure 2.3: Frequency droop S_P as the relation of the generator's mechanical power $P_m^{set,tot}$ and grid frequency f; having f_0 as the nominal frequency and P_{m0}^{set} the set value of the generators output power [And12].

This frequency droop control allows for interaction of multiple generators. In this case an equivalent machine model, compared to the one depicted in Figure 2.2, can be compiled by summing generation power. This results in a new virtual frequency droop characteristic S'_P of the composed cooperating set of generators [Bev14, Chap. 2.3].

2.1.2.2 Secondary Frequency Control

Secondary frequency control – also known as load frequency control (LFC) – is a supplementary control strategy of AFC operating on an approximately one magnitude slower time base (cf. Figure 2.1) than primary frequency control [CW12, p. 62]. LFC is used to relieve the occupied generator's primary frequency control reserve and thus to eliminate the temporally constant frequency error. Also, the deviated scheduled interchanges among different areas and the scheduled power are restored by LFC after necessary control action taken by primary frequency control. Therefore, an LFC controller is used in each area maintaining the scheduled power among its related tie-lines and the frequency. Typically such controllers are of proportional-integral (PI) type.

LFC PI control operates on the basis of a so-called area control error (ACE), which is a linear combination of the steady frequency error Δf and the deviation of tie-line power P_{tie} and scheduled exchange power P_{sch}. One solution for such a linear combination is Equation (2.5) with the bias frequency constant B_f.

$$ACE = (P_{tie} - P_{sch}) + B_f \Delta f = \Delta P_{tie} + B_f \Delta f \tag{2.5}$$

$$\Delta P_{ref} = -K_i \cdot \int ACE \, \mathrm{d}t \tag{2.6}$$

The resulting ACE of Equation (2.5) is then integrated according to Equation (2.6) and dispatched to one or more participating turbine generator controllers (as additional input to vary the reference power for their primary frequency controller) [GSO07, Chap. 11.3].

2.1.3 Voltage Control

In the previous section, frequency control systems were presented, which showed to be in direct relation to the balance of active power in the grid. In this section the focus lies on voltage control, which is strongly related to reactive power control in the grid [DBBVdK$^+$07]. Unlike the grid frequency, which is a universal measure, voltage is a local value that may vary for every node in the respective power grid; and so is reactive power. Reactive power and thus voltage control is performed by various methods amongst all voltage levels and several entities like synchronous machines, (switched) capacitor banks and shunt reactors, or transformers [Kun94, Sec. 11.2]. Hereafter follows an overview of those voltage control mechanism.

On the high voltage generation side, automatic voltage regulation of the generators is performed in a dynamic manner. This means that the generator bus voltage is fed back to a control loop that uses the offset of the generator output from the set value to alter the excitation current of the generator. In a generator compound the voltage is controlled by all interconnected generators by a voltage droop control. Its linear behaviour is comparable to frequency droop control and is depicted in Figure 2.4.

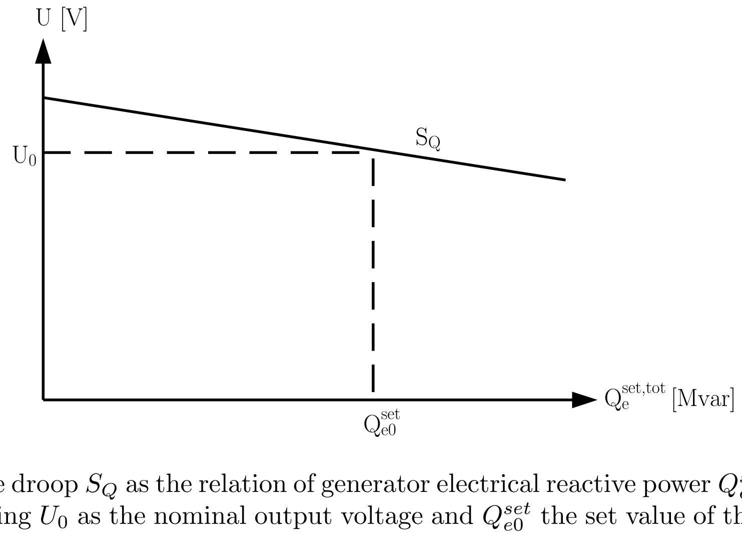


Figure 2.4: Voltage droop S_Q as the relation of generator electrical reactive power $Q_e^{set,tot}$ and bus voltage U; having U_0 as the nominal output voltage and Q_{e0}^{set} the set value of the generators reactive power.

Comparable to the frequency droop in Equation (2.4), the voltage droop can be stated as follows.

$$S_Q = -\frac{U - U_0}{Q_e^{set} - Q_{e0}^{set}} > 0 \tag{2.7}$$

S_Q in Equation (2.7) represents the primary proportional control element of a generator's automatic voltage regulator, which can also take into account voltage deviation caused by an output transformer that connects the generation set with the grid connection point. Q_{e0}^{set} is the set value for the generators reactive power for the nominal voltage U_0 and Q_e^{set} is the actual reactive power.

In case of coupling several generators, non-ideal states result from this autonomous primary voltage control, which manifest in unplanned reactive power flow among the coupled generators. Despite the fact that each generator above a specific power rating has to be capable of under- and over-excited operation to a specific amount, this unwanted reactive power flow should be avoided amongst coupled generators. Therefore, secondary voltage control mechanisms exist, comparable to secondary frequency control. They are realized as coordinating control entities using a reference point within their respective control area to modify the primary controllers' set points of each generator set [And12]. Furthermore, there is tertiary voltage control, which is, like tertiary frequency control, mostly manually performed or calculated via optimization algorithms. It operates another level above secondary voltage control – both in a temporal and regional perception – by controlling the set points of each control area to maintain trans-regional reactive power flow.

The grid connected operation of a synchronous machine without a prime mover for both reactive power generation and consumption (varied by modification of the excitation) is known as synchronous condenser. Their rotating masses provide support for grid stability (i.e. short-circuit capacity). Their reactive power output can be controlled in a continuous manner by the machines' closed voltage control loops. Due to the synchronous condensers' high costs, these voltage control elements have been largely substituted by static var compensators (SVC) [Kun94, p. 638f].

Besides synchronous machines, also power line impedances have a direct influence on the voltage levels. To vary the line impedance in order to reduce reactive power flow – primarily used for long HV lines –, breaker switched shunt capacitor banks and shunt reactors are used. They serve for the coarse reactive power compensation and free the synchronous condenser's and generators' reserves [And12]. A faster but more complex form of impedance variation is realized by thyristor controlled capacitors and reactors, also knowns as SVC, which are a subset of FACTS (flexible alternating current transmission system) devices [IEE97]. There are several different methods and combinations of connecting the switched reactors and capacitors in shunt or in series to the branch, depending on the intended use. As the SVC are thyristor switched, they are limited to a certain bandwidth, which typically is not sufficient for fast voltage compensation (e.g. voltage dips or flicker; cf. Section 2.1.1). Therefore, FACTS based on voltage source technologies have to be used [And12]. This subset of FACTS are called static synchronous compensators (STATCOM) and use current switching elements (e.g. IGBT) and voltage source converters for feeding reactive current into the grid for voltage control [CW12, Chap. 4.3.2].

Another means of voltage regulation, which is much slower than the FACTS solution, is the use of controllable transformers. This is done either by on-load tap-changers of transformers located with the transformer in the substation or by the use of step voltage regulators within the single feeder. The step voltage regulator is an autotransformer that is connected in series into the single feeder. Voltage variation is done by changing the series side mounted tap-changer. This either adds or subtracts the voltage inducted at the series side to the output voltage [Kun94]. Substation located tap-changers are typically mounted on bus level regulating all feeders on three phases at once. They are typically located at the transformer's high voltage side due to the lower current that has to be switched. Often, the tap-changers can be switched under loaded conditions by a motor which is controlled by an automatic tap-changer controller aiming for a specific set point on the transformer's secondary side. While this automatic voltage control mechanisms are a convenient way of maintaining voltage levels in the short term, off-load tap-changers are typically used to counteract long term (e.g. seasonal) effects.

As a conclusion one can say that a diverse set of methods for voltage regulation exists, which orient on different time scales and types of voltage regulation problems. They either use methods to actively modify the grids impedances (and thus the reactive power flow) or change transformer configurations in order to maintain voltage levels.

2.2 Networked Smart Grid Systems

As reflected in Section 1.3, the increased demand in electric energy, the extensive integration of RES into the ageing classical power grid infrastructure, and the adjunct intermittency of generation are challenging current grid infrastructure. To be able to cope with these challenges by avoiding large investments due to classical grid extension, information and communication technology (ICT) is introduced to the power grid. Equipping renewable energy sources, loads, and grid infrastructure with ICT allows for their coordinated operation, advanced monitoring, higher utilisation, and leads to further benefits in this expanded kind of power grids, which are commonly referred to as *Smart Grids* and schematically depicted in Figure 2.5.

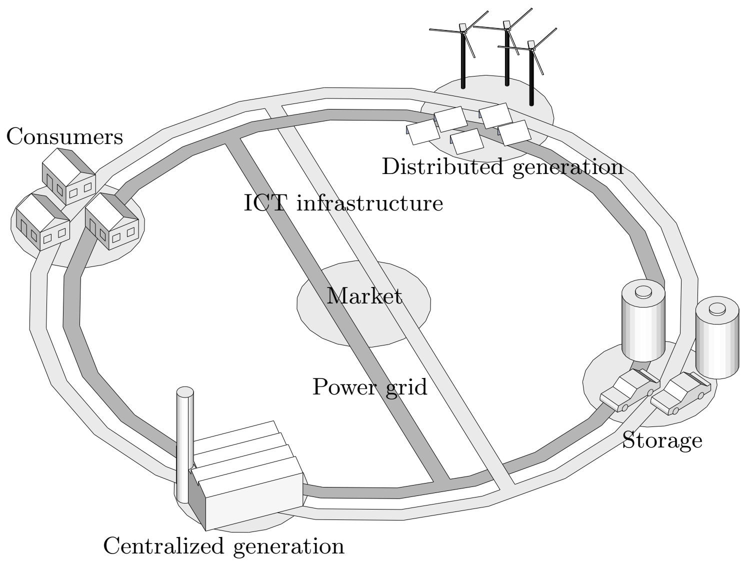


Figure 2.5: The *Smart Grid* concept – Power grid and ICT infrastructure integrating customers, distributed and centralized generation units, storage, and market in order to efficiently deliver sustainable, economic, and secure electricity supplies (with friendly permission of Klaus Pollhammer).

The European Technology Platform Smart Grid [16] defines *smart grid* as follows [SMB10]:

“A Smart Grid is an electricity network that can intelligently integrate the actions of all users connected to it – generators, consumers and those that do both – in order to efficiently deliver sustainable, economic, and secure electricity supplies.”

There are several definitions available for the term *smart grid*, depending on the related context and the point of view. This rather generic definition focuses on the actors and their intelligent interaction for sustainable, economic, and secure electricity supply. The following subsections will cover these users (not only humans but in a broader sense; referred to as active components) and ICT networks needed for the intelligent integration of their actions. To gain a holistic overview, the unified smart grid architecture model (SGAM) is presented subsequently and the thesis at hand is situated within the SGAM. Finally, research projects that are directly related to this thesis and the networked smart grid domain will be presented.

2.2.1 Smart Grid Architecture Model

For the step-wise evolution of classical power grids towards future smart grids, not only electrical engineering is necessary. Other technological, economical, and organisational disciplines are inevitable in order to successfully develop and operate the future energy systems. To be able to consolidate the different perspectives that utilities, developers, and other domain experts are likely to have, a common abstract model of the smart grid domain is necessary. This need has explicitly been formulated in the technical reference architecture part of the EC standardization mandate M/490 on smart grid standardization [eur11].

To address this need, in 2012 the Smart Grid Coordination Group, formed of CEN-CENELEC-ETSI members, created and introduced the smart grid architecture model (SGAM) [CEN12]. The SGAM and its adjunct use case mapping methodology have primarily been created for the identification of standardisation gaps. Due to its abstract representation of the smart grid, SGAM has been used beyond this range of use (e.g. for the suggestion of a common modelling language [ASRU13]; the extension towards a model-driven-architecture process [DNR$^+$14]; for smart grid agents' role definition [GH14]) and should also serve for the determination of this work's scope.

SGAM has been chosen for the classification of this work due to the fact that it is an established, generic, and future-oriented model of the whole smart grid domain that allows for the representation of both totally centralized and totally decentralized grids, and arbitrary intermediate stages. Another argument for its use is its explicit consideration of the DER domain (e.g. in contrast to the NIST Framework and Roadmap for Smart Grid Interoperability [GWP$^+$14]), which is the major target area for networked smart grid control systems [CEN12, Chap. 6.3.5].

The SGAM is a three dimensional layer-based model of the smart grid domain (depicted in Figure 2.6) consisting of the smart grid plane and interoperability dimensions atop. The smart grid plane is partitioned into physical *domains* representing the electric energy chain and into hierarchical *zones* for the management of electrical processes. These *zones* represent typical layers of classical automation systems, as e.g. given by Sauter et al. [SSKD11].

The first dimension of the smart grid plane is formed of six zones. The first and most granular one is the process zone. This one contains the primary equipment of the power system (e.g. lines, loads, and generators) and is the only one containing power hardware and equipment. All other zones are focusing on information management. The field zone manages and combines the process zone's primary elements and takes into account their interactions, protection, control, and monitoring. The station zone aggregates those field level sub-systems (e.g. on substation level, for data concentration, etc.) and hosts coordination systems on this level. The next abstraction zone is on the operational level. Here, high-level coordination and control action takes place

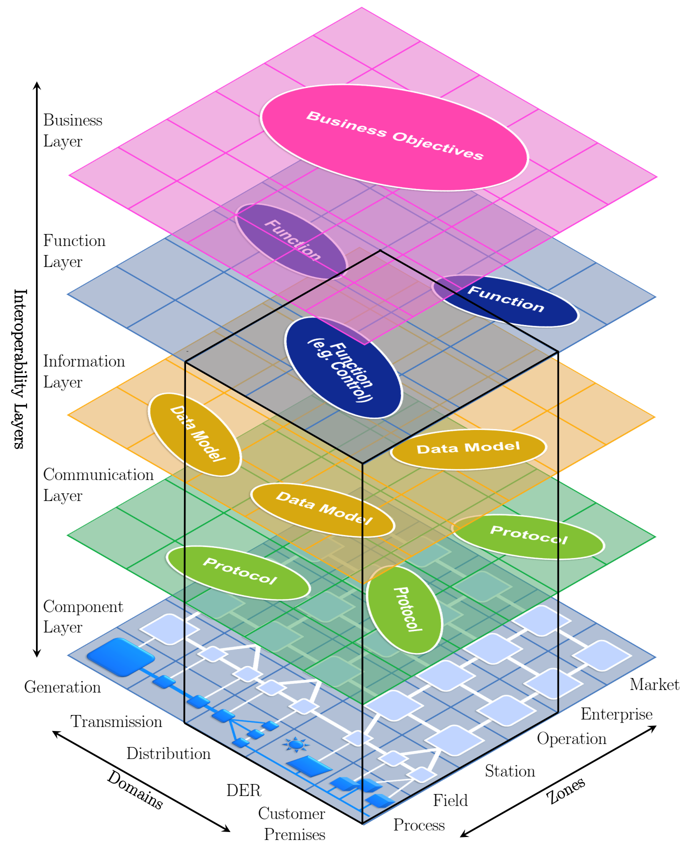


Figure 2.6: Smart Grid Architecture Model (SGAM) with the smart grid plane (spanned by zones and domains) as the bottom layer and further interoperability layers on top (extended from [CEN12]).

(e.g. EV fleet charging coordination or virtual power plant applications). The enterprise zone already contains non-technical aspects, such as asset management, commercial and organizational services, and billing. In the highest abstraction zone, which is the market zone, trading action is considered [CEN12, Chap. 7.2.5].

The second dimension of the SGAM's plane is formed by five domains representing the power generation, transmission, distribution, and consumption structure. It describes the classical power system structure, having bulk generation at the top (leftmost position in SGAM) with underlay transmission and distribution grid elements. Then the distributed energy resources (DER) domain is explicitly mentioned before the most granular one, representing customer premises.

This two dimensional layer (the smart grid plane) allows for the identification in which zone management actions between domains take place. This interaction can be specified by the smart grid plane for one system (e.g. business, utility, grid operator, area, country). For the discussion of interoperability among those systems, the interoperability layers are used within the SGAM. Those layers create the third dimension of the model and are structured as follows. In the component layer interoperability amongst power system equipment and related ICT components has to be ensured. The communication layer interoperability relates to protocols and standards, whereas in the interaction layer data models are used. The two topmost interoperability layers are handling functional and organisational issues.

To gain a clear and common view also for this work, rapid control prototyping is situated within the SGAM. This ensures to mention all relevant domains along the energy supply path, all technically relevant zones and the interoperability layers. The highlighted box in Figure 2.6 represent the affected subspace of the SGAM. The control prototyping concept that is presented in this thesis has a clear focus on the distribution level together with DER and customer premises. It is intended to use and effect entities from the process level up to the station and in some cases also operation level. Concerning the interoperability layers, all layers except the business layer are taken into account.

2.2.2 ICT in Networked Smart Grid Systems

For the intelligent integration of all actors, which are connected to the smart grid (as quoted from the European Technology Platform Smart Grid [16] in Section 2.2), ICT network integration of these actors is inevitable and a key aspect of smart grids. Even if the use of ICT is not that new to the electric energy grid, there are significant differences between ICT for classical power grid operation and the smart grid approach. These differences can nicely be identified and described by use of the SGAM (introduced in Section 2.2.1) and is performed as follows.

Classically, communication is used in the transmission and high level distribution grids for market negotiation, energy balancing, and long term power and voltage regulation. Thus, in the smart grid plane of the SGAM, the zones from market down to the station level are concerned with classical power grid ICT as well as the generation, transmission, and (partially) the distribution domains. Typically, human-to-human or human-to-machine interaction is part of this communication (e.g. via email, phone call, telecontrol, and fieldbus systems). A heterogeneous set of proprietary and legacy systems is used by grid operators and utilities, which typically is kept under lock and is thus only accessible for the respective technicians and domain experts [KLS$^+$14].

The introduction of ICT in networked smart grid systems immensely increases the number of communicating nodes, due to the fact that granularity is extended from the station zone to the field and process zones as well as amongst all domains (including especially nodes from the DER and customer premises domain). Furthermore, the once physically locked down ICT systems become exposed to a high number of potential attackers as the information endpoints are not located in a (sub)station any more but in the customers' premises.

These two facts (the high number of communication endpoints and the exposure of the ICT system) requires for an extension of the heterogeneous legacy ICT systems by standardized protocols and devices that allow for safe operation and an increased share of machine-to-machine communication. The joint project (SG)^2 – Smart Grid Security Guidance [SBKK12] used the SGAM to gain an overview over ICT technologies that are used in networked smart grid systems. The cumulative overview has been obtained by an analysis of eleven national and international smart grid research projects and is depicted in Figure 2.7.

The provided overview of ICT technologies used in networked smart grids focuses on the distribution, DER, and customer domain amongst all zones of the SGAM. It shows the already established ICT technology which is used in the distribution domain with its large share in the enterprise, operation, and station zones. Those technologies often set up on Ethernet and TCP/IP or field bus systems and they are increasingly applied also in the DER domain both in medium and low voltage distribution grids. The customer domain hosts three main pillars which are households, functional buildings, and electric mobility. It can be seen that here a more diverse set of technologies is listed, which is caused by the novelty of energy related ICT usage in this domain.

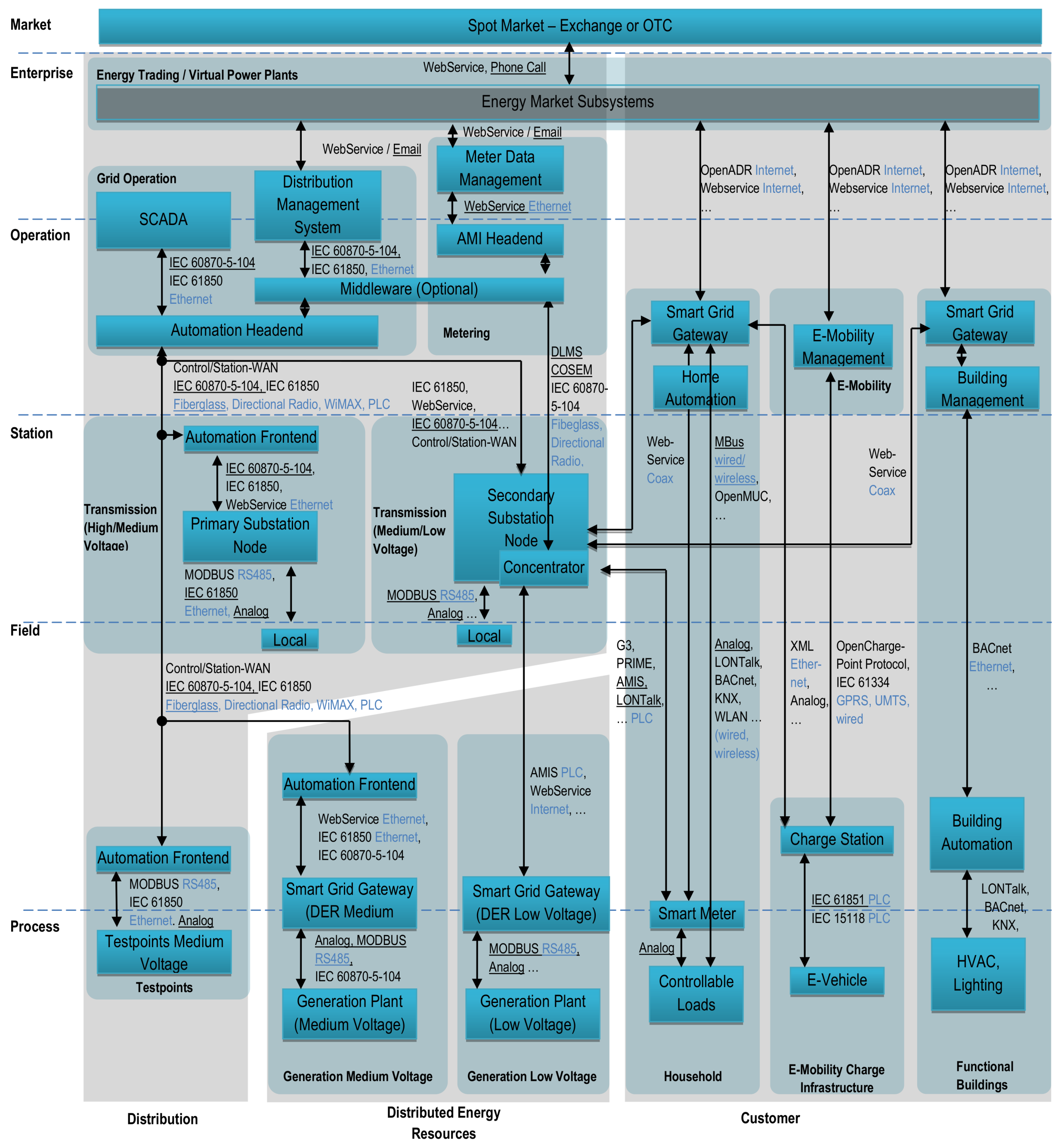

Market
Spot Market – Exchange or OTC
Enterprise
Energy Trading / Virtual Power Plants
WebService, Phone Call
Energy Market Subsystems
WebService / Email
WebService / Email
Grid Operation
Distribution Management System
Meter Data Management
SCADA
WebService Ethernet
AMI Headend
OpenADR Internet, Webservice Internet, ...
OpenADR Internet, Webservice Internet, ...
OpenADR Internet, Webservice Internet, ...
Operation
IEC 60870-5-104 IEC 61850 Ethernet
IEC 60870-5-104, IEC 61850, Ethernet
Middleware (Optional)
Automation Headend
Metering
DLMS COSEM IEC 60870-5-104 Fibeglass, Directional Radio,
Smart Grid Gateway
Home Automation
E-Mobility Management
E-Mobility
Smart Grid Gateway
Building Management
Control/Station-WAN IEC 60870-5-104, IEC 61850 Fiberglass, Directional Radio, WiMAX, PLC
IEC 61850, WebService, IEC 60870-5-104... Control/Station-WAN
Station
Automation Frontend
IEC 60870-5-104, IEC 61850, WebService Ethernet
Web-Service Coax
MBus wired/ wireless, OpenMUC, ...
Web-Service Coax
Transmission (High/Medium Voltage)
Primary Substation Node
Transmission (Medium/Low Voltage)
Secondary Substation Node
Concentrator
MODBUS RS485, IEC 61850 Ethernet, Analog
MODBUS RS485, Analog ...
Field
Local
Local
G3, PRIME, AMIS, LONTalk, ... PLC
Analog, LONTalk, BACnet, KNX, WLAN ... (wired, wireless)
XML Ether-net, Analog, ...
OpenCharge-Point Protocol, IEC 61334 GPRS, UMTS, wired
BACnet Ethernet, ...
Control/Station-WAN IEC 60870-5-104, IEC 61850 Fiberglass, Directional Radio, WiMAX, PLC
Automation Frontend
AMIS PLC, WebService Internet, ...
WebService Ethernet, IEC 61850 Ethernet, IEC 60870-5-104
Charge Station
Building Automation
Automation Frontend
MODBUS RS485, IEC 61850 Ethernet, Analog
Smart Grid Gateway (DER Medium
Smart Grid Gateway (DER Low Voltage)
Smart Meter
IEC 61851 PLC IEC 15118 PLC
LONTalk, BACnet, KNX,
Process
Testpoints Medium Voltage
Testpoints
Analog, MODBUS RS485, IEC 60870-5-104
MODBUS RS485, Analog ...
Analog
Controllable Loads
E-Vehicle
HVAC, Lighting
Generation Plant (Medium Voltage)
Generation Plant (Low Voltage)
Generation Medium Voltage
Generation Low Voltage
Household
E-Mobility Charge Infrastructure
Functional Buildings
Distribution
Distributed Energy Resources
Customer

Figure 2.7: Cumulative smart grid communication model [KLS+14] (with friendly permission of Friederich Kupzog).

2.2.3 Relevant Research Projects

Relevant research projects related to this thesis is presented hereafter. The focus of the presentation clearly lies on the Austrian project chain *DG DemoNet*, which focuses on the integration of renewable generation units in conventional distribution grids without grid reinforcement and aims to solve the resulting challenges. The project *Smart LV Grid* of this project chain is also included in the subsequent section, as a major share of this thesis' research has been conducted during the work in that project.

2.2.3.1 DG DemoNet Concept, DG DemoNet Validation, and BAVIS

The Austrian research projects *DG DemoNet Concept* [Bun08], *DG DemoNet Validation* [FFG13a], and *BAVIS* [FFG12a] aim to maximized power generation from renewable energy sources (RES) in existing power grid infrastructure by avoiding classical, costly power grid reinforcement. These three projects focus on the medium voltage level and *DG DemoNet Validation* builds up on the two other preceding projects. In *DG DemoNet Concept*, typical Austrian distribution grids have been identified and analysed for active grid operation to reach the goal of a better utilization of existing grid infrastructure. The identified problems for maximization of distributed renewable generation in Austria's power grids are twofold.

On the one hand, in typical Austrian distribution grids, relatively long MV power lines are feeding several rural LV distribution networks by also connecting MV infeed from various sources (e.g. wind generators, small hydro power plants, photovoltaic farms). On those rural MV distribution grids – which form the majority of Austrian distribution grids – voltage problems (according to EN 50160 [CEN11]) are caused among the power lines by an extended integration of distributed renewable generation. Without active operation of these distribution grids, they have to be dimensioned for the worst case scenarios of either very high generation at times of low consumption or very little or no generation at times of maximum consumption. This typical operation and dimensioning strategy is characterized by a very ineffective usage of existing infrastructure and high costs for the integration of new generation units.

On the other hand in urban areas the distribution grids are characterized by much higher short-circuit power due to reduced impedances. In these distribution grids there is less risk of voltage problems than risk of power or thermal problems. Due to the high share of distributed generation and intensive loads in urban distribution grids, the maximum power allowed for specific grid elements may exceed – probably without even mentioning this at the transformer station.

As the former described problem is more common in Austria and thus offers more potential for extending DG hosting capacity, the projects focus on active grid operation for voltage control. Furthermore, voltage control in distribution grids is of special interest for distribution grid operators due to their duty of guaranteeing grid voltages to stay within EN 50160 boundaries. This is of especial interest because since deregulation of the electricity market they usually do not have direct access to the generation units.

Within the project *DG DemoNet Concept* a first set of voltage control concepts has been worked out, which was concreted in the successor project *BAVIS* and compiled to a toolbox of mature measures for active voltage control. In detail, the following measures were proposed.

- Local voltage control by decentralized generation or consumers
- Coordinated voltage control (by a centralized control system)
- Smart control of the transformers' on-load tap-changer

Furthermore, a methodology for assessing the acuteness of the voltage problem in distribution grids and for estimating the effectiveness of the previously proposed voltage control measures has been presented in the project *BAVIS*.

The following concrete control strategies for rural MV distribution grids have been elaborated within the joint research project *BAVIS* [FFG12a].

Uncoupling of branches/sections – When the local tap-changer controller in the secondary substation is configured to a fixed voltage set point for the local bus bar voltages, single branches may violate voltage limits on their ends. This is caused either by strong loads or high distributed generation connected to those branches. To fix this issue, in *BAVIS* autotransformers with local tap-change controllers have been used. They are installed at the position where half of the maximum voltage deviation is located along the branch. This creates an uncoupling of those long branches. The ideal positions for the placement of the autotransformers can be found by a preliminary offline simulation of the respective grid. Still the grid is passively operated in general [FFG12a, p. 15].

Local control – This control approach also operates with a fixed set point for local tap-changer controller. In addition, selected DER also operate local control loops that control active and reactive power to keep the local bus voltages at a specific level. Due to the higher R/X ratio of distribution grids (compared to transmission grids), it might be inevitable to also curtail active power besides reactive power to avoid voltage limit violations. Due to the fact that with reactive power control all potential (renewable) energy can still be used, this type of voltage control is preferred. The selection of the relevant nodes depends on their connection point within the distribution grid and their parameters which are

- power rating (effectiveness of controlling the generation unit for voltage regulation),
- controllability (ability of the generation unit to vary active and reactive power),
- connection point within the distribution grid.

Furthermore, also loads are foreseen to be used to shift or shed their demand based on their local voltage measurements to avoid voltage band violation.

Remote control – This control approach applies exclusively to the transformers tap-changer controller, which is controlled on the basis of remote voltage measurements from critical grid nodes. Due to the utilization of remote measurements, there arises the need for a highly available overlay communication system with this control approach. In case that any of the critical nodes indicates a voltage band violation, the remote control algorithms seeks to change the tap position to avoid this violation. The critical nodes are identified in a preliminary offline simulation of the power grid over the whole year. The nodes that are exposed to the minimum or maximum grid voltages, over a whole year, are defined as critical nodes. This identification should ensure that if all critical nodes are within limits, no unmonitored node is exposed to voltage limit violations. This is a critical requirement for the remote control approach that has to take topology changes and loss of communication into consideration. The effectiveness of this approach highly depends on the topology of the MV distribution grid and its related branches with their load characteristics.

Coordinated control – The coordinated control approach developed in *BAVIS* is the most complex control approach of this project. It combines the previously described remote control approach with the capability of the generation units to vary active and reactive power. To avoid mutual influence of the controlled generation units by the centralized controller, a contribution matrix has been designed, which represents the impact of generation units on critical nodes. This matrix allows for the prioritization of generation units for each critical node. Due to the complexity of this control approach, which uses both remote readings and remote control commands, the requirements for the communication system are higher than for the other approaches (e.g. bidirectional communication, reliable set point transmission, etc.).

The previously described developed voltage control strategies have been implemented and validated in field tests in the course of the project *DG DemoNet Validation*. The field tests are located in medium voltage distribution grids in the Austrian provinces Vorarlberg (in the valley Großes Walsertal) and Salzburg (in Lungau) [SBB^{+}12].

To generate a need for active voltage grid control according to EN 50160, additional generation units and loads would have been necessary within the field test regions. To overcome these expensive and time consuming installations of extra assets, a voltage band which is narrower than the one defined in the norm has been used. This new definition of voltage limits directly creates an artificial need for voltage control without changing the distribution grid setup in the demo regions.

The validation of the developed control concepts was done during different grid situations of the year (maximum of hydro generation in spring, maximum load in winter caused e.g. by skiing lifts), under varying topological situations (internal topology changes and bypass/backup supply). Furthermore, the novel control approaches were activated in an alternating manner with the business as usual operation and (only in Lungau) the *ZUQDE*[2] control approach for long term validation [FFG12b]. For short term validation, all generation units were driven consecutively once from under-excited to over-excited mode under stable conditions, to evaluate their capabilities and their impact to the voltage levels.

For the communication infrastructure, the use of narrow-band PLC was planned because of the idea to use the “path of energy” also as the “path of communication”. This would implicitly provide necessary information about the topology and thus solve the issue of reaching the correct generation units with the control commands independent of topology changes. Extensive tests in the MV distribution grid in Vorarlberg’s field test region showed that narrow-band PLC with primarily inductive couplers is principally usable for point-to-point communication. Nevertheless, it was not possible to use the PLC solution in a dynamically interlinked and meshed network configuration due to its fixed structure. The suggested PLC solution is not capable of changing the communication path according to topology changes because such a feature would use further bandwidth, which is not available in the narrow-band system. Thus, this essential feature is not implemented in the PLC protocol and so the topology information cannot be obtained from the communication system. The result of the field tests for the ICT infrastructure was the usage

[2]*ZUQDE* is another Austrian research project focusing on centralized control systems for voltage and reactive power control in MV distribution grids [FFG12b]

of a heterogeneous ICT structure on the basis of existing WiMax and fiber-optic systems in combination with the investigated point-to-point narrow-band PLC system [FFG13b].

The following problems and learnings can be summarized from the three interrelated research projects.

A first task for (remote) control of distribution grids is the identification of critical nodes and the equipment of those nodes with reliable measurement units and communication technology. The identification of these relevant nodes can be done by offline power grid simulation over a whole year. The creation of a representative model for such simulations may be effortful because of the modelling of each grid element and potential manual creation of the power grid structure in the simulation software. Furthermore, changes of these critical nodes by topology changes must be considered.

Concerning the topology of the power grid, a representation of this information is inevitable for the proper operation of suggested grid control algorithms. These information cannot easily be automatically obtained, but has to be extracted from the existing SCADA system. One has to distinguish between temporary topology changes, which might be caused by switches, failures, bypass and backup supply, and permanent topology changes, caused by construction and grid extension.

One disadvantage for the developed system is that several existing generation units are not well suited for remote (re)active power control because of a lack in accessibility and controllability, their unknown power factor/PQ diagram and risk of instability in the empirical exploration of excitation and insufficient dimensioning of the generator rated current for the necessary under- and over-excitation. As a conclusion a need for model independent, self-learning, and simple to install systems was stated.

2.2.3.2 DG DemoNet – Smart LV Grid

The national joint research project *Smart LV Grid* is another successor project within the Austrian *DG DemoNet* project group. It lays the basis for the work presented in this thesis. The primary goal of this project is the maximization of distributed generation (DG) in rural low voltage (LV) distribution grids without cost intensive extension of the grid infrastructure. The focus clearly lies on rooftop mounted photovoltaic (PV) generation units combined with controllable electric vehicle (EV) charging stations. The methodology in this project is the active control of grid components (e.g. PV inverters, EV charging stations, and tap-changers in the secondary substations) to keep voltage levels within EN 50160 [CEN11] limits. The monitoring and control features implemented within the project allow to operate the grid infrastructure closer to its operational limits, which consequently increases DG hosting capacity.

Within the project, three paradigms are applied, which are listed below.

Intelligent planning – Usage of new planning methods to allow for higher DG density.

Intelligent monitoring – Incorporation of new monitoring solutions to support intelligent planning and live operation.

Active measurement and control – Usage of new ICT infrastructure with restricted bandwidth and availability for enhancing DG hosting capacity by new and cost-effective active control solutions.

These paradigms are developed and applied in three field test regions in Upper Austria and Salzburg. In this field test, approximately 70 rooftop mounted PV panels were installed with an accumulated maximum generation of around 400 kWp. In the demo region of Salzburg AG approximately 35 PV panels with a total maximum generation of 125 kWp and another 30 EVs were distributed within the respective LV grid [Bru12].

The control approach uses both a centralized control system located in the secondary substations for the control of the central on-load tap-changer, the distributed inverters, and the charging stations and a non-communicating distributed control system as fall-back option. This distributed control system consists of local control loops in the inverters, which control their active and reactive output power and the tap-changer controller, which controls the local bus bar voltages.

As underlying ICT system, a narrow-band PLC solution by SIEMENS has been chosen. It uses a multi hop routing protocol to also reach components which are further away from the central data concentrator located in the secondary substation. The smart meter infrastructure that relies on this PLC communication system was used as measurement system to obtain the voltage status in the LV distribution grid and to forward control commands towards the actively controllable PV inverters. In the planning phase, it was also intended to send set values to the EV charging stations. Due to the PLC communication system's relatively high latency and low bandwidth this was not feasible. To remedy this, a TCP/IP connection has been installed and was used to control the charging equipment in each household.

The control approach, which is used for the field test regions in this project, is derived from prior projects of the project chain *DG DemoNet* and focus on voltage control by tap-changer control, reactive power control, and demand side management. Concretely spoken, besides the tap-changer in the secondary substation, also the PV inverters and EV charging stations are remotely controlled by a multi-stage set of control entities.

From the analysis of this LV distribution grid related research project with its demo regions, where centralized together with non-agent based local control approaches were engaged for maximizing DG hosting capacity, the following conclusion can be drawn. It has been shown that by coordinated intelligent planning, intelligent operation, and active operation of consumers and DG units it is possible to significantly increase DG hosting capacity and the distribution grid utilization factor in rural LV grids. The pre-installed advanced metering infrastructure (relying on PLC) proved its suitability as sensor network for coordinated control applications. In contrast to the sensor application, the narrow-band PLC channel showed significant disadvantages for more complex applications, which require for higher bandwidth and reduced transmission delay. Therefore, a dedicated TCP/IP solution has been chosen. Concerning the ICT infrastructure, another conclusion is that the operating control system always has to be able to cope with reduced communication performance or even losses. Therefore, the ability for safe (fall-back) operation of the control system has to be guaranteed. This requires a sophisticated control development, testing, and deployment procedure, which is presented in the course of this thesis.

2.3 Requirements for Control System Prototyping

As stated earlier and in particular described in Section 2.2, smart grids are complex distributed systems of systems. Thus, the discipline of control system prototyping is a form of system engineering. According to Hull et al. [HJD10, p. 8], requirements engineering is an essential building block of system engineering and is of prime importance for the success of the engineering process. The IntelliGrid Smart Grid Roadmap [EPR12] published by the Electric Power Research Institute EPRI in 2012 clearly suggests the identification of stakeholders and the adjunct requirements extraction in IEC/PAS 62559 [IEC08].

Following this suggestion, in the next subsections the stakeholders for the control system prototyping process are identified and adjunct concrete use cases are described. From those use cases non-functional requirements are derived. They are the basis for the development of the rapid control prototyping process and are described in the last part of this section.

2.3.1 Stakeholders and Use Case Definition

Figure 2.8 shows a set of identified stakeholders. Not all of them are actors in the sense of their direct interaction with the control prototyping process, but all of them care about how the system works and its outcome. Further, it has to be mentioned that in systems engineering use cases are understood in a more abstract way than within software engineering [HJD10, p. 26]. Subsequently, they are used to describe stakeholders' goals and interests, also referred to as *stakeholder requirements*. In the following description, immediately after each of those generic *stakeholder requirements*, the adjunct *system requirements* are derived. In case of this work, where a process for rapid control prototyping is being developed, they are referred to as *process requirements* (PR).

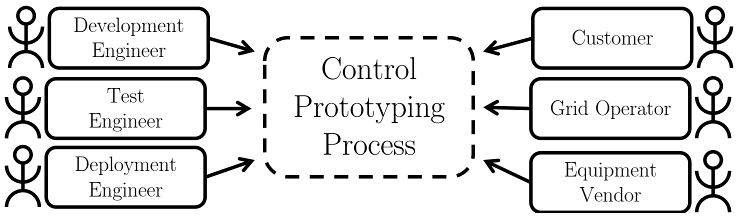


Figure 2.8: Stakeholders for the rapid control prototyping process.

Development engineer aims to develop effective, high-performance control systems for controlling the actors within a (distribution) power grid. Such a control system typically consists of one or more control algorithms and controller hardware to operate the control algorithm(s) in the field.

To fulfil this tasks, the development engineer requires a concrete specification of the destined control behaviour, which has to be defined at the beginning of the process (`PR1 - Definition of specification`). On the basis of this specification the developer needs to create a conceptual design of the algorithms (`PR2 - Creation of the conceptual design`). For this the development engineer needs to have knowledge about the system boundaries and its interfaces. This information also has to be included in the specification. The final requirements for the rapid control prototyping process, which are related to the development engineer, are the concrete programmatical implementation of the control algorithm (`PR3 - Implementation of control algorithm`) and the later implementation of these algorithm(s) on the target hardware (`PR4 - Implementation of control system`).

Test engineer aims to test the developed algorithms and the control system on the basis of its specification (cf. `PR1`). By doing this, the test engineer aims to reach 100 % test coverage through the application of multiple test methods. Not only wants the test engineer to evaluate the behaviour of the software algorithms; he also needs to perform hardware tests of the control hardware component hosting the algorithm(s), its interfaces, and interaction with other components.

Due to the characteristic of the controlled system (i.e. the energy distribution system), the developed control algorithms cannot directly be tested within the system. Thus, a simulative testing approach with a modelled energy system representation has to be chosen for the testing phase of the prototyping process (`PR5 - Simulative control algorithm testing`). The same applies to the final control system. Thus, also hardware testing (i.e. controller hardware-in-the-loop (CHIL) tests) has to be performed off the real power grid; i.e. in the lab (`PR6 - Off-grid control system testing`).

Deployment engineer aims to integrate the developed and tested control system into the destined power grid in coordination with the grid operator. The development engineer has to ensure the control system's proper installation and operation in the field and its interaction with all relevant sensors and actuators.

To allow for these actions, the deployment process has to support field deployment where the deployment engineer is support in the installation and start-up phase of the control system (`PR8 - Seamless field deployment`).

Customers want to have guaranteed power quality at their connection points (i.e. defined by EN 50160 or specially contracted in case of industrial customers). Furthermore, they want to be able to connect and operate modern power grid components (e.g. PV inverters, EV charging equipment, CHP units, etc.) to their full extent to save money and to maximize revenues from (governmental) funding programs (e.g. for integration of renewable energy sources (RES)).

The customer does not have any direct relation to the development process but is affected by the outcome of it. Thus, the customers and especially their appliances have to be considered in the development process as active elements in the electrical power grid (`PR11 - Consideration of customers`).

Grid operator wants to keep the (distribution) grids in stable conditions despite the increasing amount of additional loads and new generation units integrating RES. Therefore, grid operators want to have smart control systems operating their distribution grids in a stable and cost effective way and that contracted power quality is ensured to their customers. They do absolutely have to avoid failures and blackouts and want to keep the risk of such events as low as possible.

Besides the fact that the grid operator wants to be in charge of the behaviour of the developed algorithm and thus of its specification (cf. `PR1`), he is very concerned about the costs of control system development (accounted as capital expenditure – CAPEX) and operation (operational expenditure – OPEX). Development costs are closely related to the development time and the success rate (`PR9 - Rapidness and high success rate`).

As grid operators are by nature very concerned about the operation of their assets, they have to be integrated in the development process. Due to the fact that control of networked smart grid systems is new to the energy domain, the grid operator's requirements are likely to change during the development process (`PR10 - Grid operator integration and ability to handle changing requirements`). Finally, but most relevant is that the process should not evoke any risk of malfunction within the grid (`PR7 - Mitigation of risk`).

Equipment vendor wants to provide power grid equipment (i.e. sensors, actuators, and passive grid components) for grid operators and their customers. This equipment has to be compatible with the grids' electrical and communicational specification and especially should also support novel control systems. Two examples are the usage of advanced metering infrastructure (AMI) as sensors and remotely controllable EV charging stations as actuators in low voltage power grids.

Equipment vendors are not directly affected by the development process, but indirectly by the equipment that they produce. So these products, which are building elements of the future smart grids, together with their capabilities have to be considered within the prototyping process (`PR12 - Consideration of power grid equipment`).

2.3.2 Process Requirements

In the following Table 2.2 all process requirements that have been identified in the previous use cases descriptions are collected. These non-functional requirements are described in detail and further refined to identify relevant related work in Chapter 3 and to serve as basis for the concrete definition of the control prototyping process within Chapter 4.

Table 2.2: Requirements for the control prototyping process.

#	Requirement `name` and detailed description
`PR1`	`Definition of specification` The control specification is a functional specification that clearly defines the behaviour of the control system. Besides those behavioural definitions of the control system the specification shall define syntactic and semantic details concerning the control system's communication interfaces and related requirements, which are given by the environment. Summarizing, it contains requirements, which the developed control system has to fulfil in order to comply with the specification in a later evaluation.
`PR2`	`Creation of the conceptual design` Within the rapid prototyping process, a conceptual design has to be created on the basis of the specification. While a specification typically does not contain any information about the concrete realization (how to build the system), this is indeed addressed in the conceptual design. The design has to propose a programmatical concept for fulfilling all the given control system requirements.

PR3 `Implementation of control algorithm`
This is a very obvious requirement that has to be fulfilled by the prototyping process. The implementation is the operationalization of the theoretical algorithm specification and design in software. This requirement is the core requirement of the whole process, as it implies the creation of the core control algorithm(s).

PR4 `Implementation of control system`
The implementation of the control system is the first processual requirement that goes further than classical software development processes. Here it is necessary to port the control algorithm to suited controller hardware in order to complete the projected control system, which consists of the control algorithms and controller hardware.

PR5 `Simulative control algorithm testing`
Modelling and simulation of real world systems is a common technique for analysis and evaluation in cases where the real system cannot directly be used (e.g. because it is too dangerous, too expensive, not available, etc.) [RLS+14]. This is also the fact with power grids and thus a simulative approach is the way to go for the evaluation of the control algorithms.

PR6 `Off-grid control system testing`
This requirement results from the need to evaluate the complete control system that consists of the field controller hardware operating the control algorithms. To be able to perform these tests, an off-grid test environment has to be used that emulates the real field conditions.

PR7 `Mitigation of risk`
As the electric power grid is critical infrastructure, grid operators do absolutely aim for a stable, save, and cost efficient operation. Their need for control algorithms arose in order to maintain stability but this must not come at the expense of reliability. Thus, the control development process has to aim for absolute risk mitigation.

PR8 `Seamless field deployment`
To operate the developed control system in the field, it needs to be deployed. This requirement contains the physical installation of the control system in the substation or control room of grid operators, but furthermore also necessary adoptions to the real field infrastructure. To avoid risk of malfunctions and guarantee a rapid development process, this task should be kept as simple as possible and thus goes along with the following process requirements: `PR12 - Consideration of power grid equipment` and `PR9 - Rapidness and high success rate`.

PR9 `Rapidness and high success rate`
Control of networked smart grid systems is a novel and interdisciplinary task. Thus, development life-cycles are rather short and new control approaches and controllable grid elements appear quickly. Due to this fact and the CAPEX that are directly related to the duration of the development process, it is of high interest to rapidly come from the first specification to the operating control system in the field.

PR10	`Grid operator integration and ability to handle changing requirements` A prototyping process has to be found that allows for the constant integration of the grid operator in the development process. This has two major reasons. First is that due to the novelty of the control of networked smart grid systems, requirements given by the grid operators are likely to change during the development process. Changed requirements have to be directly considered to avoid failure of the development process. Second is the gain of acceptance and trust in the developed control system resulting from a constant integration of the grid operator in the development process.
PR11	`Consideration of customers` Grid operators are supplying private, institutional, and industrial customers who are having different behaviours and needs that have to be taken into consideration in the prototyping process. Primarily, customers can be characterized by their consumption and generation of electric energy (i.e. consumed electric energy and power capability of their connection point) which is summarized as demand side characteristic.
PR12	`Consideration of power grid equipment` One necessary feature for networked smart grid control, besides the existence of ICT infrastructure, is the availability of remotely controllable active grid components, which might be property of the grid operator or the customers. To benefit from the flexibilities that those components offer, they have to be considered from the beginning of the development process.

3 Existing Tools and Methods for Control System Prototyping

Due to the fact that control system prototyping for networked smart grid systems is a relatively new discipline, currently no monolithic development tools are available to cover the complete prototyping process. Thus, control prototyping is a novel process that requires a set of dedicated tools (e.g. for testing, deployment, etc.). For this process, a complex set of such dedicated methods and tools from other application domains are combined and adapted in order to create a suited solution for rapid control prototyping. At first, well established software engineering processes are analysed. Then, simulation software is analysed, as it is the core tool for the evaluation of developed algorithms. Thereby, a clear focus is put on the combination of simulation software and tools for the co-simulation of different physical domains. In this field it is necessary to have a look at the different simulation approaches for continuous system and discrete event simulation, the modelling of real-world components, and the respective level of abstraction and granularity. The last topic covered in this chapter are hardware-in-the-loop (HIL) development and evaluation systems. They are relevant because HIL evaluation is one of the final stages of the control prototyping process, which allows for dynamic system emulation at a very subtle level of granularity.

3.1 Software and System Engineering Methodologies

From its first days, software and system engineering has evolved to a complex profession with distinct methods and solutions for specific tasks. As control system development for networked smart grid systems is in large parts nothing else but an extended software engineering task, this section focuses on potential software engineering methods to accomplish this complex task.

The engineering methods can be divided into two large groups (e.g. published by Boehm et al. [BT04]), which are traditional plan-driven and modern agile methods. These methods are not strictly bound to software engineering processes but rather to software related project management (cf. [Cob11]) and thus are well suited for the problem at hand.

Due to the wide variety of software and system development methods and their complex nature, a direct comparison of plan-driven and agile methods is not reasonable. Nevertheless, to get an idea of the wide range of methods and to be able to apply the best suited ones, or at least parts of them, the most established methods of both plan-driven and agile nature are presented subsequently.

3.1.1 Plan-Driven Methods

Plan-driven methods are typically used when all requirements, as well as the plan for the realization of a product, can be defined to a certain extent prior to the actual implementation. The prior planning and creation of certifications, together with comprehensive documentation, is often required to meet certification standards [BT04].

3.1.1.1 Waterfall Model

The waterfall model is one of the first software development process models that were used to bring a structure into the software development process. The structural procedure of the iterative waterfall model is one representative for plan-driven, sequential software engineering methods generally referred to as waterfall models.

What all the slightly different forms of those sequential methods have in common is the structured progression through defined phases. Each of these phases contains specific tasks and deliverables that have to be accomplished in order to be allowed to progress to the next phase [MAL09].

According to Mall [MAL09], the phases from the planning phase to the verification phase are the development phases. The first phase, which is the planning phase, contains a general problem description and feasibility studies in order to determine whether the project is feasible in financial and technical means. Further, different strategies of how the problem can be solved are elaborated on the basis of the abstract problem description. The final step of the planning phase is the evaluation of the different solution strategies and the selection of the most suited one; all presented in a comprehensive documentation.

In the requirements phase, all customer requirements are collected and documented in a detailed way. This phase consists of the requirements gathering and analysis process and the concrete requirements specification. The goal of the subsequent design phase is the creation of a software architecture to fulfil all specified requirements. Each function required by the customer is analysed and decompiled into sub-functions. Data flows are analysed and specified in a structured way. The structured design resulting of this phase consists of an architectural design (representing the system as a set of modules and interfaces) and a detailed design, where each module is described in great detail.

The next implementation phase is the part of the process model where the functional design is translated to source code. Each module of the architecture is implemented and tested individually (e.g. by unit tests), isolated from other (probably unfinished) modules. Finally, each module and its interactions are documented. The following and last phase of the development phases is the verification phase. In this phase the developed software is verified against the software requirements specification created in the second phase. This also involves system testing by the development team and the customer.

As the successful verification through the customer typically goes along with its contractual acceptance, the final maintenance phase is goes on beyond this point. This task is likely to exceed development efforts and can involve corrections of errors identified during operation, perfective maintenance, which is the improvement of functions, and adaptive maintenance, which includes porting of the software to new environments or conditions [MAL09].

The waterfall model is well-suited for small to large and complex projects, with a problem that can be described well and provokes non-changing requirements. Its strict phase structure with the required deliverables causes a huge amount of documentation. This allows for different teams to work on different phases in an exchangeable way.

3.1.1.2 V-shaped Model

The V-shaped model is a process model not only for software development but also applied in systems engineering. It consists of the planning axis and the testing axis surrounding the actual implementation phase.

The V-shaped software development process model is a variation of the waterfall model with consecutive planning, implementation, and testing phases. It puts more weight on the testing phases, so that the weights of planning and testing are approximately equal. The phases of the model are similar to the waterfall model's phases. By comparing the waterfall model and V-shaped model, one can see that besides the different form of representation, also the testing and evaluation phase of the waterfall model is much more detailed in the V-shape model, where it explicitly consists of unit tests, integration tests, and system and acceptance tests.

Thus, the benefit of the V-shape model is its finer granularity and explicit focus on testing. Each phase of the planning axis has a corresponding testing phase. A testing document has to be created directly at the end of each planning phase, which lays the basis for comprehensive system testing on each development level by the early generation of test plans.

While the V-shape model has the same benefits as the waterfall model, which are its simplicity and straight forward processing, it also has some drawbacks. Two of the most prominent points of criticism are the rigid structure and the inability to react to changes of the requirements. Armour [Arm04, p. 98] states

> *"Change is Expected (...) Significant scope change is simply part of the environment today. It cannot be avoided, and therefore legitimate processes should not try to avoid it."*

This criticism is countered by the introduction of flexible agile software and system development process methods discussed subsequently.

3.1.2 Agile Methods

Agile methods for software engineering have been well-established and widely used since the turn of the millennium, building up on so-called lightweight programming approaches. One of the first popular publications was about Extreme Programming (XP) by Kent Beck [Bec03]. Since then, several adaptations and extensions of these methods have been introduced (e.g. Scrum), which all aim for a rapid and flexible solution-oriented development process. They orient themselves by the principles defined in the Agile Manifesto [1] in 2001. These principles are stated in [1] as follows.

> *"Individuals and interactions over processes and tools*
> *Working software over comprehensive documentation*
> *Customer collaboration over contract negotiation*
> *Responding to change over following a plan"*

Two representatives for agile software development methods – XP and Scrum – are presented subsequently and their benefits and drawbacks in comparison to plan-driven development processes are highlighted.

3.1.2.1 Extreme Programming

Extreme Programming (XP) is one of the first software development process models that is based on the principles of agile or lightweight development. It is described as "…a style of software development focusing on excellent application of programming techniques, clear communication, and teamwork,…" [BA04, p. 2] and relies on the values of communication, feedback, simplicity, courage, and respect.

This development process model constantly integrates the customer into the development process. The customer's requirements are defined by so-called story cards where the exact needs are defined in a use-case-like way. Unlike plan-driven development, in XP the customer does not define a complete set of specifications prior to the actual implementation phase. In contrast, very short development cycles are used in XP to be able to release functional but incomplete software products in a very early stage; e.g. after two months of a project with a twelve month planning horizon [Bec03].

The customer is able to use and test this early functional versions and refine the requirements or add more story cards to request further features. This approach allows for very flexible adaptation of the development process to changing customer requirements and avoids the development of a product that does not satisfy the customer's needs. Also in contrast to classical plan-driven methods and caused by the agile incremental development method, there is no comprehensive specification and documentation required before the beginning of the implementation phase. This would not even make sense because many details and features are not specified at the beginning of the project but rather emerge along the way. The extensive documentation is substituted by simple, modular, and proper written and documented source code [Bec03].

XP focuses very much on test-driven development. Tests are written before and during the implementation phase as soon as a module and its features are clear. Then the respective module is implemented ideally by another programmer in order to fulfil the tests. Also when an error is found, tests are written to clearly identify them and to avoid their reoccurrence. Tests are done in a highly automated manner and are a substantial element of XP as the idea is that errors (their impact and their correction) cause higher costs the later they are identified [Bec03].

XP is also known for its great importance of teamwork. It was one of the first process models that introduced pair programming, where two team members work jointly on a problem. This method is very beneficial in order to be able to concentrate on the program structure, to discuss with colleagues, to quickly identify misleading ideas, and to foster knowledge exchange. Close cooperation, both in spatial and human fashion is highly recommended. Daily stand-up meetings ensure fast interaction and little reaction time [Bec03].

Due to the high amount of information exchange and interaction, XP was intended to be used for small to medium size projects and not for big ones. The usage of XP within recent years, where it has also been used for large-scale projects, drew a different picture, even though some organisational adaptations had to be made [BA04, p. 3].

Concluding the analysis of XP, the benefits seem to predominate some drawbacks. Its clear strength lies in the short development cycles and constant customer integration and the adjunct ability to adapt to changing requirements in a flexible way. Thus, it is a process model that is well-suited for a specific set of problems where flexibility is one of the main requirements.

3.1.2.2 Scrum

Scrum is, other than XP, not only an agile process model for software development, but rather an agile product development framework as it only consists of some elements and adjunct rules. The elements of the Scrum framework consist of three roles, five activities, and three artefacts.

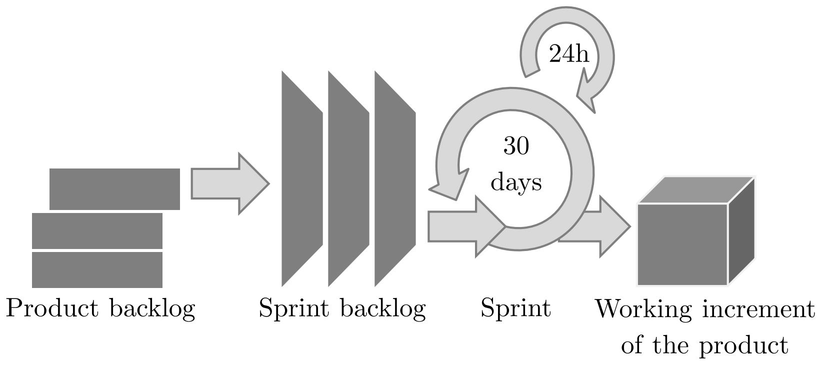


Figure 3.1: The Scrum software and product development framework.

The three artefacts are the product backlog, the sprint backlog, and the working increment. They are part of the Scrum illustration in Figure 3.1. The roles are the product owner, the development team, and the Scrum master. The former represents the customer and the customer's needs. These needs are typically described in user stories, which the product owner organizes and prioritizes in the product backlog. Each development team typically has one product owner, who can optionally also be a member of the development team. As the product owner is responsible for the product backlog, he ensures its transparency and accurateness, enrols user stories to the product backlog, and acts as a representative for the development team towards the stakeholders [Rub12, Cha. 2].

The Scrum master has a mediator role and is responsible for the application of the Scrum framework. He supports the single actors and the company in a potential change process when applying Scrum. Furthermore, the Scrum master takes a leader role and removes barriers between team members and protects them from influences from outside.

The third role is the one of the development team. This is a self-organizing team that does the actual work, which is analysis, design, implementation, testing, etc. Typically such teams have five to nine members of diverse skill levels that are all necessary for proper software or product development. Their major task is the creation of potentially deliverable products (that form the working increment) at the end of each sprint period, which typically lasts for up to 30 days and focuses on feature realization listed in the product backlog [Rub12, Cha. 2].

Besides the product backlog that holds the necessary features that should be developed for the customer in a prioritized order, the sprint backlog holds a subset (often also just one) of these features with detailed sub-tasks that were defined within the sprint planning phase at the beginning of each sprint. These tasks are planned to be realized by the development team within the next sprint interval and are coordinated by the product owner.

Besides the sprint planning, which is the first activity, the daily Scrum (or daily stand-up) is a short regular meeting where each team member reflects on his recently conducted work and gives an overview of the current and an outlook on the upcoming work, highlighting some potential obstacles or other issues.

At the end of a successful sprint phase, the sprint review is performed. Together with the customer and other stakeholders, the developed features are checked to fulfil the requirements defined in the user stories. This is one iteration interval after which new features might be added as new user stories to the product backlog. The product owner typically also collects the stakeholders' remarks during this activity.

Also at the end of the sprint, a retrospective is performed. In this activity, all members of the Scrum team reflects their working methods, which they applied to solve the accomplished sprint together with the Scrum master. This is done in order to constantly improve the methods and the Scrum development process for the next sprints.

As a last action, the product backlog refinement is performed by the product owner on the basis of the feedback obtained from the stakeholders at the sprint review. Records may be added, removed, merged, split, or reordered in the product backlog and new priorities might be set in order to select the best one for the next sprint interval [Rub12].

To summarize, agile methods are a necessary adaptation of classical rigid plan-driven development methods building up on the waterfall method. They show little benefits for projects where it is totally clear what to build and how to do so and where change is unlikely. For any other type of projects – especially in novel disciplines – an agile approach might be the method of choice, but it is also not a guarantor for success. Which method exactly is the best choice depends on many issues, like concrete problem area, size and type of the development team(s), contractual issues, and company philosophy. Also, institutions choose to adapt agile methods to fit their specific needs.

3.2 Simulation, Software, and Tools

Simulation is the method of choice for the evaluation and analysis of systems that are either too expensive or too dangerous to use, do not exist, or are not accessible at the moment of interest. On the market, a wide range of simulation software is available for an extensive field of (engineering) applications, like flight simulators, power grid simulators, or simulation software for power electronics design. Different simulation approaches and abstraction levels are used to serve the diverse field of application for simulation in the best way. These approaches are discussed in the next section, followed by a focused analysis of power and communication system simulators. Subsequently, frameworks and tools for co-simulation are presented because of their high relevance for this work.

3.2.1 Simulation Paradigms and Methods

In literature, several solutions for simulation of physical systems are presented, which all basically rely on the following same principals. The modules of the complex systems are typically mathematically modelled by the use of differential equations. These models are compiled to a set of equations that are discretized and solved by domain specific solvers after a parametrization of the model by the developer. To support the developer in this task, several domain specific modelling and simulation tools exist.

3.2.1.1 Classification of Simulation Paradigms

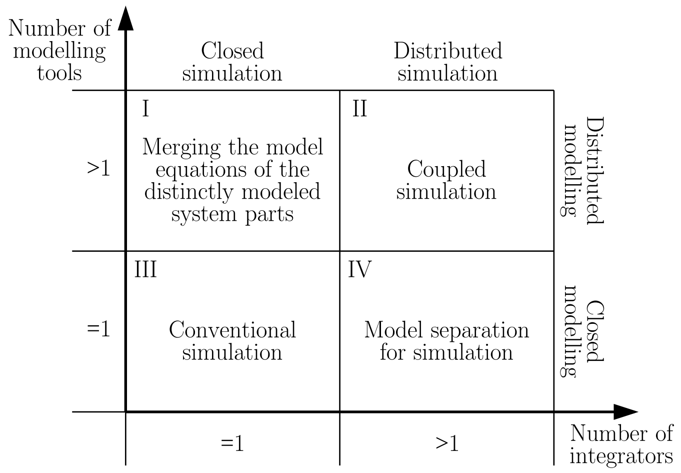


Figure 3.2: Classification of modelling and simulation paradigms [Dro04].

A classification of simulation approaches, e.g. applied by Rohjans et al. [RLS+14], is used to distinguish between single- and multi-domain simulations, according to the number of simultaneously simulated physical domains. Figure 3.2 shows an even more detailed classification based on the number of simulators/solvers and the number of used different modelling tools. The classification also considers modelling and simulation paradigms that arise from the simulation of dynamic systems amongst various (physical) domain boundaries [Dro04, GKL06].

If a dynamic system is modelled in only one physical domain and simulated by one specialized tool or solver, this approach is referred to as single-domain simulation (III in Figure 3.2). In the case of multi-domain simulation, different approaches exist according to the number of modelling tools and solvers that are employed. For the modelling of complex dynamic systems among multiple physical domains, multiple specialized (commercial off-the-shelf) modelling tools might be used to reach the best solution. These models can be compiled and transferred to one of the simulators and are solved by this solver (known as “strongly coupled” [SB09]). This combination of the model equations (I in Figure 3.2) allows for highly accurate modelling with the potential drawback of an inappropriate solver. The inverse method is the model-separation within one modelling tool but by using multiple specific solvers (IV in Figure 3.2). Besides the benefit of having an appropriate solver for each modelled domain, the model itself may not be accurate enough.

While the single simulation approach (Sector III in Figure 3.2) is highly sophisticated and accurate for the modelling and simulation of single (physical) domain systems, these benefits cannot directly be used for multiple domain simulations. Either the loss of the modelling or the solver accuracy has to be taken into account [GKL06].

To overcome these drawbacks and to be able to use the high quality of domain specific simulators, a coupled simulation or co-simulation is used (Sector II in Figure 3.2 – also referred to as “loosely coupled" simulation [SB09]). In this simulation paradigm, specialized solvers and modelling tools are employed. To be able to simulate a complex multi-domain system, state variables of the interacting systems are exchanged between the simulators at specific communication or synchronisation intervals. Each of the coupled simulators uses its domain-specific modelling tool and solver and is therefore able to conduct highly accurate simulation.

3.2.1.2 Continuous and Discrete Event Simulation

Continuous systems are typically represented by states $\mathbf{y}$ and their derivatives $\mathbf{y}'$ (represented in Equation (3.1)). These states are changing constantly over time due to the inputs $\mathbf{u}$ and the characteristics of the system described by ordinary or partial differential equations, and auxiliary algebraic variables $\mathbf{x}$ [CK06].

$$y' \equiv \frac{dy}{dt} = f(x, y, u) \quad \text{and} \quad x = g(x, y, y', u) \tag{3.1}$$

In computational simulation, solvers (using numerical integration algorithms) are used to compute approximations of these mathematical models. Therefore, time does not progress constantly but is partitioned into discretisation intervals. Given a system like Equation (3.1), leads to the calculation for the states of the next interval $t + \Delta t$ in the form of

$$y(t + \Delta t) = f(x(t), y(t), u(t)) \tag{3.2}$$

and thus depends on the current state, on inputs, and on other system variables. Often, this system can be expanded into terms equal to the Tailor-series, that can be accordingly written as

$$y(t + \Delta t) = y(t) + \Delta t \cdot y'(t) + (\frac{\Delta t^2}{2!}) \cdot y''(t) + (\frac{\Delta t^3}{3!}) \cdot y'''(t) + ... \quad . \tag{3.3}$$

As an example, the Euler integration method can solve Equation (3.1) by

$$y(t + \Delta t) = y(t) + \Delta t \cdot y'(t) \quad . \tag{3.4}$$

This is a very intuitive way, as simply the derivatives are assumed to be constant for the discretisation interval. The state variables for the next interval are then extrapolated, starting from the current state with the constant derivative [PM12].

A comparison of continuous system and discrete event simulation is depicted in Figure 3.3. The combination of power system simulation and ICT, which is presented in Figure 3.3, is a good example for the simulative combination of a continuous and a discrete system. In discrete event simulation, system states are defined at specific points in time represented by an event. Between these events they are not defined. The events originate either from other events or from outside of the discrete event simulator. They are typically organized in an ordered event queue within the simulator. By taking the next event from the queue, the simulation progresses. This behaviour is shown on the ICT system simulation time line illustrated in the upper section of Figure 3.3.

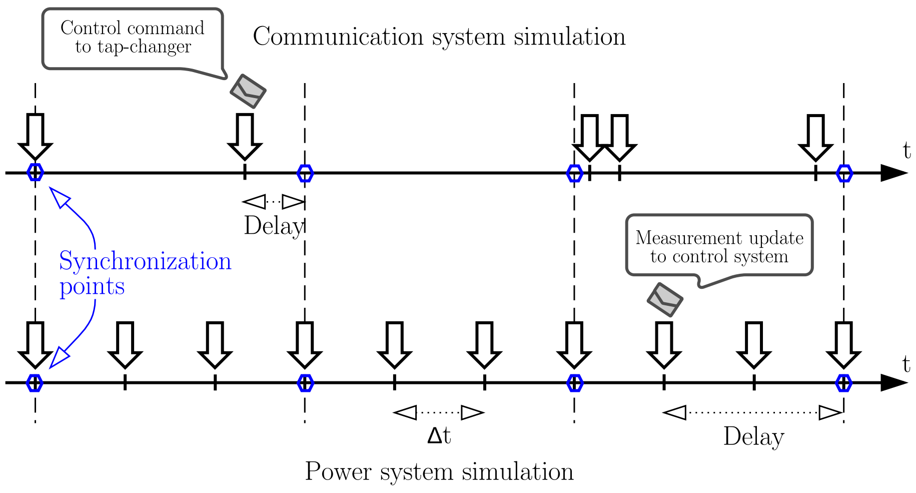

Control command to tap-changer
Communication system simulation
t
Delay
Synchronization points
Measurement update to control system
t
Δt
Delay
Power system simulation

Figure 3.3: Continuous system simulation with a constant time step interval and discrete event simulation on the example of power grid and communication system simulation [MOD14]. State variables are exchanged at constant communication intervals (synchronization points).

When coupling these two fundamentally different simulation approaches, their synchronization is of high relevance. In order to keep the quality standards provided by the dedicated system simulators, this synchronization and especially its synchronization interval has to be chosen properly. Due to the nature of the continuous system simulators, it makes no sense to choose the duration of the synchronisation interval shorter than the discretisation interval of the continuous system simulator's solver. Rather it should be a multiple of this interval (Δt), as illustrated in the power system simulation time line in the lower part of Figure 3.3.

The definition of the synchronisation interval, where state values of the single simulators are exchanged, is comparable to a sampling of these systems. This can be done according to two approaches (following [PM12, Chap. 4]), always orienting themselves on the fastest acting subsystem that is simulated. The first approach is aligned to the Nyquist-Shannon sampling theorem. This approach follows the Nyquist-Shannon theory by setting the synchronisation interval to at least the double of the characteristic frequency of the system. It is highly anticipated to set the synchronisation frequency even higher by the factor six to twenty to ensure proper command action and interference suppression [Lun04, Chap. 10.2.3]. Due to the high effort for analysing the simulated systems to identify their characteristic frequency – which might also vary depending on different simulation scenarios – the second approach is rather used than the first one.

In the second approach, the selection of the synchronisation interval is related to the local quantisation errors of the single solvers. This means that a specific relative or absolute threshold for the state variables is defined that shall not be exceeded. The maximum derivatives of each simulation's system states is determined from a prior simulation run and the synchronization time interval is chosen in such a way that the fastest simulated system does not exceed the predefined threshold. This is a convenient method that can also be extended to dynamically adapt the synchronisation interval in order to be compliant with the quality criteria and reach maximum co-simulation performance.

3.2.2 Power System Simulation

For the power system analysis, a wide range of distinct sophisticated simulators exist. These simulators are used for network planning, forecasting, short-circuit analysis, transient stability, power flow and optimal load flow analysis, and further applications. A classification can be made concerning the time-scale of the analysed problems in transient dynamics simulators and steady state simulators.

Transient dynamic simulation studies dynamic processes in power grids between stable states (e.g. after switching activities or starting of a generator). Therefore, the power system is modelled to its very physical detail to be able to analyse the impact and the distribution of oscillation and harmonics. This is typically done by the use of ordinary or partial differential equations and at very dense discretisation intervals of the solver (i.e. micro- to nanoseconds) in order to cover very fast effects like lightning and harmonic oscillations up to a very high level [KML15, p. 268 f].

For this thesis the transient dynamic simulation is not relevant because of the scope on control prototyping for networked smart grid systems, which are coupled via communication systems that typically are not fast enough to allow for the control of dynamic effects. Thus, the focus is set to steady state power system simulation. Nevertheless, several commercial and open-source simulators support both types of problem analysis. Mets et al. [MOD14] compiled an actual list of commercial and open source power system simulators, reworked in Table 3.1.

Table 3.1: Commercial and open source power system simulators (extended from [MOD14]).

Simulator	Simulation type	Subsystem domain	License
Cymdist	steady state	generation, distribution	comm.
DS PowerFactory	transient, steady state	generation, transmission, distribution	comm.
EMTP-RV	transient	transmission, distribution	comm.
ETAP PSMS	transient, steady state	generation, transmission, distribution	comm.
EuroStag	steady state	generation, transmission, distribution	comm.
OpenDSS	transient, steady state	generation, distribution	open
PyPower	steady state	generation, distribution	open
GridLAB-D	steady state	generation, distribution	open
PSCAD/EMTDC	transient	transmission, distribution	comm.
PSS® E	transient, steady state	generation, transmission	comm.
PSS® Sincal	transient, steady state	generation, distribution	comm.

Besides the listed power system simulators, also general purpose tools such as Matlab offer a wide set of packages that support power system analysis. A comprehensive list of Matlab packages for that purpose is presented in [MOD14]. All of the simulators presented in Table 3.1 are highly sophisticated and established. They provide a rich set of options for both steady state and transient simulation amongst the whole generation, transmission, distribution, and consumption chain. Most of these simulators also provide a modular extensible structure or interfaces to allow data exchange with other tools, e.g. on the basis of proprietary APIs and programming or scripting languages, IEC 61850 support, CIM, COM interface, database access, etc.

3.2.3 Communication System Simulation

The simulation of the communication system is, beside the actual power system simulation, the second pillar of smart grid co-simulation. Due to the heterogeneous set of communication types and systems (wired, radio, connection-oriented, connection less, etc.) as well as paradigms (point-to-point, point-to-multipoint, master-slave, peer-to-peer) also a wide range of simulators exists with different levels of detail.

Besides a very detailed consideration of physical parameters of the communication channel (e.g. relevant for PLC communication [KHV^{+}11]) also the creation of stochastic models of the communication channel is a practicable way (e.g. for MIMO wireless radio communication [WHOB06]). Statistical methods rely on experienced or concretely measured data from the communication channel in the field.

Very often packet-based communication systems are simulated by discrete event simulators such as OMNeT++ [VH08]. This simulator provides several extensions to support various protocols. Due to its convenient interfaces, it supports smart grid related problem analysis (e.g. the integration of IEC 61850 protocol presented by Juárez et al. [JRMRM12]).

Another widely used communication system simulator is the Network Simulator version 2 (ns-2) and its successor ns-3 [9]. They are discrete event network simulators created for research and education purposes with a strong focus on internet communication systems. ns-2 provides its own scripting language for the scenario definition (e.g. for the network topology), which is replaced by the Python programming language in the newer version. ns-2 is very popular and has been used in several co-simulation approaches (e.g. [NKM^{+}07, ASHL13, LAH11]).

While the presented network simulators primarily focus on the simulation of the communication channel and the impact of their characteristics to networked smart grid systems, another set of simulators exists, which focuses on the security aspect of communication systems. One representative of this set is the Network Security Simulator NeSSi [3], which is also open source software. This network simulator allows for modelling of packet network components, such as routers, client and server nodes, and provides a rich user interface for the creation of the topology model. The simulator emulates IP layer network traffic and provides methods for packet tracing, sniffing, and logging [AB10, Chap. 4.4].

3.2.4 Smart Grid Simulator Frameworks and Tools

In order to couple the heterogeneous types of simulators consisting of models, solvers, and further auxiliary tools, frameworks have been developed in academia and research. Hereafter such frameworks and tools that support “loosely coupling” (cf. Section 3.2.1.1) of simulators are described. Also efforts have been undertaken to support uniform interfacing, reuse, changeability, and compatibility of simulators and models. These efforts and hence originated tools are also covered subsequently.

3.2.4.1 GridLAB-D

GridLAB-D is a widely used open source software simulator for smart grid simulation, which has been developed by the U.S. Department of Energy at Pacific Northwest National Laboratory. Besides its functions as a distribution grid simulator, it provides extensive functions for smart grid simulation and analysis, which makes it count rather as a framework than as a power grid simulator [19].

In general, GridLAB-D is a power grid simulator for a three-phase unbalanced power flow simulation that provides the extra set of features that is necessary to analyse and simulate networked smart grid systems with all their relevant actors and components. It allows for detailed modelling of customer (load) behaviour to be able to evaluate demand side and load management concepts and related effects (e.g. the customer rebound effect after peak-shaving actions). Besides customer issues, also retail market related issues are well covered by this framework by an extra market module, taking into account, e.g. variable (dynamic) pricing structures. It supports the modelling of time discrete controllers (e.g. for voltage and reactive power control) and renewable energy sources.

The core of GridLab-D is an advanced algorithm that allows for the determination of the states of millions of independent devices of a power grid. These devices are represented by physical models mostly created by differential equations. This detailed modelling provides the following three advantages compared to reduced-order models according to [Pac12].

- The simulator is capable of handling uncommon situations accurately that might not be considered by reduced-order models.
- Handling of widely disparate time scales of different models can be done in an accurate and performant way.
- Integration of the existing models with new ones and third-party systems can be done easily.

The internal structure of the simulator bases on a time discrete structure, which uses an event queue where each simulator can enrol its next events [WdWP$^+$12]. By this flexible structure an automatic negotiation of the next synchronisation time slot is accomplished, which increases the co-simulation performance in comparison to a fixed synchronisation interval described in Section 3.2.1.1.

The advanced interfacing features, which are announced for the release version 3, allow for an even easier interaction with other simulators (e.g. transmission grid simulators or communication system simulators) and thus provide the full and convenient set of features necessary for a co-simulation framework [Pac12].

3.2.4.2 OFFIS Mosaik

Mosaik is a simulator coupling framework that has been developed at the OFFIS institute in Germany. Unlike e.g. GridLab-D it does not provide physical simulation capabilities but is a pure modular simulator coupling and model orchestration tool [SST11]. As its core element, Mosaik uses SimPy [17], a discrete event simulator that is, like Mosaik, written in Python. SimPy uses the same negotiation mechanism as GridLab-D does for the determination of the next synchronisation point. Basically, all simulators which are involved in the discrete event simulation propose the simulation time of their next event to the simulation core that is then able to announce the earliest event as the next synchronisation point among all simulators. For the integration of continuous time simulators, a wrapper entity is available that integrates them into the discrete event simulation.

For the compilation of a heterogeneous set of simulators to one seamless co-simulation, Mosaik provides several features on a layer-based structure. For the simulator coupling on the syntactic layer, two APIs are available (referred to as SimAPI). While the more generic low-layer API can directly be accessed on the basis of ZeroMQ messages carrying JSON data, the high-level API provides a base-class to inherit from but is only available for a few platforms [SSS12, SS12a].

A domain specific language (MOSL – MOsaik Specification Language) has been introduced in order to be able to describe the data semantics and the structure of simulators that implement the SimAPI. This allows for a direct integration of individual simulators into the Mosaik structure and for automated scenario creation [SS12b]. Mosaik uses the standardized interface OPC Unified Architecture and the Common Information Model (CIM) to incorporate communication and control strategies. Scenarios which are a defined combination of simulators, models, and their respective configurations can be defined on entity level by a scalable scenario definition proposed in [SSS12].

At the present time, Mosaik is being reworked by the development team and Mosaik 2.0 is about to be released. Its purpose is to provide an even leaner architecture and a wider range of supported commercial and open simulation software by the means of porting the high-level API to other well-established programming languages (e.g. Java) [10].

3.2.4.3 Other Relevant Tools

High Level Architecture (HLA), which has been developed by the U.S. Department of Defense, provides an abstract architecture to support distributed simulation. Since 2000, HLA has been available as the IEEE-1516 international standard [IEE10].

HLA provides a set of rules for the federation of simulators (federates) that each federate has to follow for distributed simulation. These rules define that each simulator has to implement a simulator object model in accordance to a provided object model template. Such an object model contains all modelled instances of a simulator and their attributes in a class structure. There are further rules concerning data/attribute exchange in a publication/subscription mechanism, for ownership transfer among simulators and others. For data exchange, each federate has to implement an interface specification that allows for the connection of the federate to the run-time interface (RTI), which is basically a middleware [DFW97, BO04]. An RTI is an implementation of the HLA interface specification that serves as a middleware for the federates and should be conform to IEEE-1516 API specification [IEE10]. There is a wide range of several concrete implementations available, e.g. Open HLA in Java and OpenRTI in C++.

The Functional Mockup Interface (FMI) is an open standard interface specification for model exchange and tool coupling [7]. A component that implements the FMI (a so-called functional mockup unit – FMU) can be either a model or a simulator containing several models and solvers. In the former case, the imported model is about to be solved within another simulator connected as an FMU. XML is used for the semantic description of interface data. The FMU itself can be either C code or compiled binaries, which ensures preservation of intellectual property. At the time being, FMI is supported by a comprehensive list of 63 simulation and modelling tools [7]. At AIT – Austrian Institute of Technology, FMI has been extended to a high-level utility package for model exchange [Wea13] and has been used for co-simulation [SWA$^+$13].

3.3 Hardware-in-the-Loop Integration

For power system analysis, components testing, controller prototyping, and related disciplines, different stages of hardware-in-the-loop (HIL) integration into simulation can be applied. The integration stage depends on the time scale and power rating of the devices under test and can be applied to both, power hardware (PHIL) as well as controllers (CHIL). Subsequently, the different stages and their relations to power and controller hardware are presented, followed by a discussion of real time digital simulators for HIL integration.

3.3.1 HIL Integration Stages

Figure 3.4 shows four stages of HIL integration with increasing complexity as a simplified block diagram. Integration stage (a) is the most basic one where the hardware device under test (DUT) is directly connected (e.g. via Ethernet) to the simulation environment, which typically is a developer's PC. Due to the fact that PCs only offer standard interfaces, this stage is useful for digital control system evaluation and development.

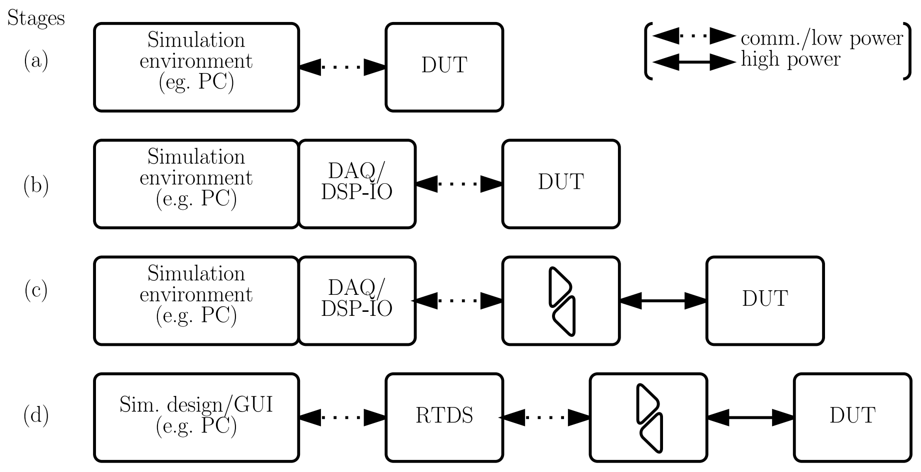

Stages
(a)
Simulation environment (eg. PC)
DUT
comm./low power
high power
(b)
Simulation environment (e.g. PC)
DAQ/ DSP-IO
DUT
(c)
Simulation environment (e.g. PC)
DAQ/ DSP-IO
DUT
(d)
Sim. design/GUI (e.g. PC)
RTDS
DUT

Figure 3.4: Four stages (a-d) of hardware-in-the-loop integration.

In case of analogue hardware or controller evaluation, extension hardware (e.g. PCI-based data acquisition (DAQ) cards) with digital and analogue input/output channels is connected to the simulation environment. Such hardware bases on digital signal processors (DSP) and thus allows for high sampling rates and accurate signal generation and acquisition with a limited power level. Solutions are provided, for example, by National Instruments (LabView with hardware integration [Nat13]), or Matlab in combination with FPGAs [CWN$^+$12].

For hardware evaluation with higher power requirements (generally referred to as power-hardware-in-the-loop, or PHIL) power amplifiers are used. The extension of the integration stage (b) by power amplifiers to stage (c) allow for high power evaluation of DUT up to several tens of kilowatts and even higher. Solutions for such amplifiers are provided, e.g. by Regatron [14] and successfully used, for example, for the implementation of a battery emulator [Sei14]. When using power amplifiers, their dynamic capabilities have to be considered. Chances are that the amplifier cannot reproduce the output value as dynamically as it is provided by the DSP.

Due to limited computational capacities of the simulation PC or the DAQ/DSP extension hardware, for highly accurate HIL evaluation of dynamic systems, real time digital simulators (RTS) are employed (which is represented as integration stage (d) in Figure 3.4). At this integration stage, the developer's PC is just used as a GUI and for the design of the simulation before deploying it to the RTS system. These highly accurate and powerful systems are described in the next section.

3.3.2 Real Time Digital Simulators

Real time digital simulators are a combination of high-performance computational cores with highly accurate digital and analogue input/output hardware interfaces. The core element typically is a real-time operating system. This allows, in combination with their high computational performance, for the capability to solve even a high number of differential algebraic equations (DAE) of large power system simulations within very short time (in the range of several microseconds). Through the RTS' input and output terminals, these highly accurate signals can be mapped to real physical signals in the range of some volts. Because of the extremely short calculation period and the mapping of the signals to physical voltages, RTS systems can be used for the dynamic hardware-in-the-loop evaluation of power and controller hardware.

Opal-RT [11] and RTDS Technologies [15] are the two biggest providers for commercial RTS systems consisting of real-time simulators and appropriate hardware testing equipment. Opal-RT offers the real-time simulator eMEGAsim, which is fully integrated with Matlab-Simulink. The system is capable of real-time analysis of electromagnetic transients with an update interval of ten microseconds. Furthermore, Opal-RT has SimPowerSystems for the simulation of power systems and machine drives in their portfolio. SimPowerSystems is a Simulink toolbox that provides multiple model components, all based on electromechanical and electromagnetic equations [SBN10].

eMEGAsim uses a specific real-time solver, the ARTEMiS Order-5 solver, to allow for fast and highly accurate real-time simulation. This solver performs a pre-calculation of the system equations for state-space model parameters. The solver comes with a set of decoupling elements for the distributed simulation of system state-space equations. This allows for the use of multi-core processors and distributed simulators on PC clusters [SBN10].

3.4 Prototyping and Co-Simulation in Other Application Domains

Prototyping per se has been used in many technical fields since the mid 20th century (e.g. in the automotive and aviation industry, or the power electronics design). It has evolved from the creation of simple mechanical drafts to a complex process that helps in the estimation of how a product or system that is under developmental shall look or function in its final version [CLL10]. Prototyping is a process that uses several different tools, depending on the field of application. The use of CAD systems and 3D printers has reached a prominent status in the recent century in prototyping of mechanical systems or constructive applications [Rei99]. For rapid control prototyping, co-simulation, automatic code generation, and HIL are three of the most relevant tools. They are also prominent in the following other domains.

In [PM12] real-time simulation and co-simulation are discussed for the progressive simulation-based design of embedded systems. The simulation is proposed for various levels of abstraction. Simulation at gate level is proposed, which is stated as the most accurate one for embedded systems design and validation, while also the most computation-intensive one. The author also states that several approaches use instruction set simulators (ISS) in order to obtain correct timing information. Because of ISS' fine granularity, this detailed simulation level also leads to slow simulation performance. This issue is faced by caching techniques and distributed simulation.

Adhikari, Schupfer, and Grimm [ASG12] created a co-simulation framework for the validation of radio frequency transceivers. In their approach they created models on the electronic system level by the use of SystemC AMS, which is an extension of the C++ based SystemC modelling library. This library allows for analogue and mixed signals modelling and simulation.

Jonke et al. [JSA+14] proposed a rapid prototyping approach for distributed energy resources (DER) integrating ICT and power electronics design. In their work they focus on the design and development of inverter-based DER devices. The presented approach contains hardware (HIL) and software-in-the-loop (SIL) validation with co-simulation of communication and power electronics components. They propose a realisation with the use of C-code generation and automated PCB design, which are supported by the proposed software tools Simulink, PLECS Toolbox, SimPowerSystems Toolbox, and 4DIAC.

4 An Agile Development Process for Rapid Control Prototyping

The electrical power supply infrastructure counts as critical infrastructure. Due to this fact, the introduction of active grid control systems into distribution grids shall not harness grid operation and safety of supply in any way. Thus, it is absolutely necessary to mitigate the risk that is caused by the development and deployment of active control systems to the power grid to its minimal extent. Besides this hard requirement, the process for rapid control prototyping has to fulfil various additional diverse requirements, elaborated in Section 2.3.2. To satisfy these requirements, the process is elaborated and designed in an abstract way, within the hereafter following subsections, focusing on the involved actors. Therefore, at first the method for the prototyping process is selected on the basis of established software and system engineering methods which have previously been discussed in Section 3.1.

The rapid control prototyping process is created according to the selected engineering method and consists of three major phases. These phases are the implementation phase, the evaluation phase, and the field deployment and operation phase. The three phases and their subordinated prototyping stages are elaborated and presented within the present chapter. Thereby, high priority is given to the evaluation phase because it is the main driver for risk mitigation prior to field operation.

During the following elaboration of the process concept, a deeper look is taken at the necessary data paths and data classification for the rapid control prototyping process. Furthermore within the concept, developer convenience is considered by means of observability of internal states as well as (user) interfaces.

4.1 Method Selection

In Section 3.1 the current state-of-the-art for well-established software and system engineering methods is presented due to the fact that control system prototyping for networked smart grid systems is in large parts software engineering. To find a suited approach for this specific task, Boehm and Turner [BT04] presented a sophisticated approach taking into account five relevant aspects of the system that is about to be developed.

Figure 4.1 depicts these five aspects which are the criticality, the size, the personnel situation, the dynamism, and the culture of the development project, all of them aligned along one out of five axes of the diagram. The evaluation of the rapid control prototyping process, on the basis of these five aspects, helps in the selection whether the process should follow an agile or a plan-driven software engineering method.

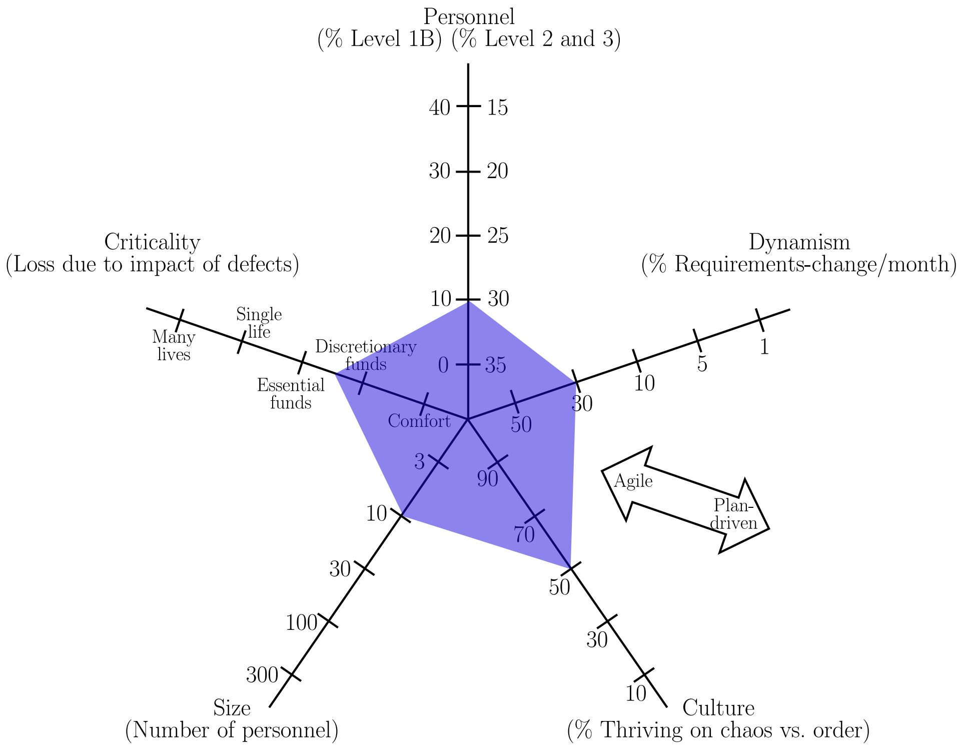


Figure 4.1: Dimensions of method selection (from [BT04]) applied to rapid control prototyping.

The spanned plane in the middle of Figure 4.1 represents the evaluation of the rapid control prototyping process and results from the following analysis, beginning with the aspect of size and following the axes in Figure 4.1 in clockwise order.

The number of personnel involved in the prototyping process is likely to vary from situation to situation and depends on the experience of the involved engineers. In the stakeholder discussion in Section 2.3.1 and depicted in Figure 2.8 (p. 30), six groups of stakeholders are identified. Under the assumption that ,on average, each group is represented by two persons, a maximum of a dozen people are involved in the development process. This estimated number is also consistent with experience values from similar research projects (cf. Section 2.2.3). Due to this fact and the issue that control prototyping is considered to be related to a specific (type of) distribution grid, it is seen as a limited size project.

The criticality describes the loss due to impact of defects on a range from loss of comfort to the loss of human lives. As the electric power grid is critical infrastructure, this aspect is of high interest. Nevertheless, a failure of development of a control system does not directly result in a breakdown of a power system as long as the defect is recognized before field deployment, which is anticipated here. On this scale, the criticality is set between the loss of discretionary and essential funds, due to the fact that long lasting or unsuccessful grid control system development attempts can be very costly for grid operators.

The personnel aspect relates to the required share of persons needed with a specific software method understanding. Therefore, levels of software method understanding and use are employed after Cockburn's definition, which is cited in an extended version in Table 4.1. Due to the complexity and novelty of the prototyping task and the adjunct high probability for changing requirements, it is absolutely necessary to have a high share of level 2 and level 3 experts working on the development task.

Table 4.1: Levels of software method understanding and use (after Cockburn; adapted by and quoted from [MW03, p. 3]).

Level	Characteristics
3	Able to revise a method (break its rules) to fit an unprecedented new situation
2	Able to tailor a method to fit a precedented new situation
1A	With training, able to perform discretionary method steps (e.g. sizing stories to fit increments, composing patterns, compound refactoring, complex Components-off-the-shelf integration). With experience can become level 2.
1B	With training, able to perform procedural method steps (e.g. coding a simple method, simple refactoring, following coding standards and configuration/change management procedures, running tests). With experience can master some level 1A skills.
-1	May have technical skills, but unable or unwilling to collaborate or follow shared methods.

The combination of the smart grids' highly dynamic environment, together with its novelty, the introduction of new flexible generation and consumption units, ICT paths, and control approaches, sets the dynamism of rapid control prototyping close to the maximum level. This might tone down with time and increasing maturity of tools and methods but currently is not foreseeable. It is more likely, that the requirements of the stakeholders, especially the requirements of grid operators and customers, are changing during the development phase and after the first learnings that are gained from the operation of first control system instances in the field.

A culture with a high share of chaos means a high degree of freedom and different possibilities for solving problems without a given, predefined single solution. The opposite is a culture where people are comfortable with defined roles and clear procedures and policies. While the electric energy domain has since known to be a strict and regulated business, for the smart grid control development task, a specific share of chaos is seen as fruitful for finding innovative and effective solutions. Thus, for this aspect of the diagram depicted in Figure 4.1, a neutral position is chosen.

From the method selection process of Boehm and Turner [BT04], after the performed analysis and categorisation of rapid control prototyping on the basis of the five aspects, the plane depicted in Figure 4.1 results. The closer this plane is situated to the centre of the five axes ground, the more the problem is situated in an agile method territory. Looking at the picture one can see that this is clearly the case for rapid control prototyping, especially in this early stage of smart grid related development.

According to the 2nd and 4th principle of the Manifesto for Agile Software Development [1], agile development takes the customer (which, in this case, is the grid operator) into consideration and welcomes changing requirements even late in development. This directly supports the satisfaction of `PR10 - Grid operator integration and ability to handle changing requirements`. Furthermore, following the first principle of the Manifesto which sets the highest priority to *"... satisfy* *the customer through early and continuous delivery of valuable software"* [1], directly supports `PR9 - Rapidness and high success rate`.

Due to this analysis of the rapid prototyping process and the resulting situation depicted by the prototyping-related plane in Figure 4.1, an agile approach is chosen for further proceeding.

4.2 Concept for Rapid Control Prototyping

According to the previously chosen agile development method, a concept for rapid control prototyping has been elaborated and is presented subsequently. The concept focuses on the satisfaction of the requirements derived from the stakeholder based requirements engineering process (suggested by EPRI's IntelliGrid Approach [EPR12]) and on the principles of the Manifesto for Agile Software Development [1].

The holistic prototyping concept is depicted in Figure 4.2 and has an intuitive and implementation-oriented structure. It consists of three main consecutive phases which are listed below together with their subordinated prototyping stages. The agility in this concept, meaning the free progression both in forward and backward direction between the development phases and stages, is illustrated by the arrows in Figure 4.2.

Phase I – Implementation

- I (a) Specification of the Control Algorithm
- I (b) Conceptual Design of the Control Algorithm
- I (c) Implementation of the Control Algorithm

Phase II – Evaluation

- II (a) Evaluation by Power Grid Simulation
- II (b) Evaluation by Coupled Simulation
- II (c) Evaluation by CHIL Simulation

Phase III – Field Operation

- III (a) Open Loop Field Operation
- III (b) Closed Loop Field Operation

In Figure 4.2, Phase I and Phase III are depicted in a summarized way (with omitted prototyping stages) due to two reasons. The first, less relevant and rather pragmatic one, is the better presentability. The second reason is to clearly show and emphasise the importance of Phase II (evaluation) and its subordinated stages. The evaluation phase is of such high importance because this phase is the one where a highly relevant process requirements – `PR11 - Mitigation of risk` – is mainly satisfied [CL13, Chap. 1.1.4].

Risk mitigation of malfunction of the developed control system along the rapid control development and deployment process is one of the main contributions (beside the seamless field deployment) of this work. This fact is also schematically illustrated below the graphical representation of the process concept, presented in Figure 4.2.

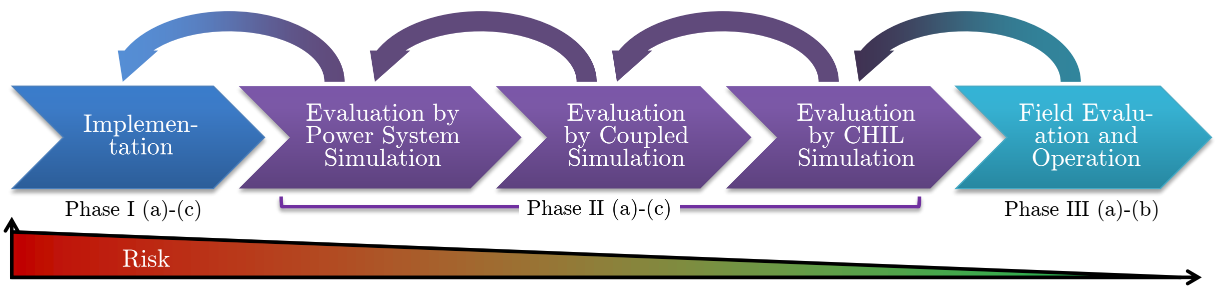

Implemen-tation
Evaluation by Power System Simulation
Evaluation by Coupled Simulation
Evaluation by CHIL Simulation
Field Evalu-ation and Operation
Phase I (a)-(c)
Phase II (a)-(c)
Phase III (a)-(b)
Risk

Figure 4.2: Agile concept for rapid control prototyping with a focus on risk mitigation along the process.

The technology related definition of risk is commonly known as the (negative) consequences of an event in combination with the likelihood of its occurrence (e.g. stated by ISO 31000 [ISO09]). The event, in case of this work, is the malfunction of the developed control system or directly aligned the failure of the development process per se. The consequences resulting from the occasion of such an event can be manifold, depending on the kind of control distribution grid and appliances, the time of recognition, and the adjunct cascaded effects. What is certain, is that by a malfunctioning control system, proper distribution grid operation cannot be ensured and the stability of the critical power grid infrastructure is endangered.

To reduce the risk of such an unwanted event, either the consequences have to be mitigated or the likelihood of the event's occurrence must be reduced. While the former is difficult to influence by a proper prototyping process, the latter indeed can be affected by these means. To be specific, the likelihood of a malfunctioning control system can be reduced by a proper specification and implementation, an extensive evaluation and validation process, and observation of the control system's behaviour in the field. All these aspects are combined within the here presented process concept and presented step-wise within the following sections.

4.3 Prototyping Concept Phase I: Implementation

The first phase of the rapid control prototyping process is the implementation phase. Here the term implementation is meant in a broader sense, covering (a) the development of the control algorithm's specification, (b) the creation of a conceptual design on the basis of this specification, and (c) the concrete implementation. These three prototyping stages of Phase I are depicted in Figure 4.3 and are elaborated in detail in the following sections.

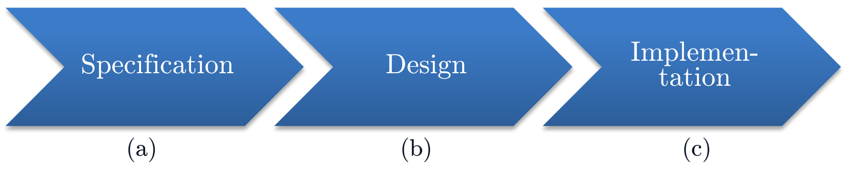


Figure 4.3: Stages of the agile prototyping concept – Phase I: Implementation.

4.3.1 Specification of the Control Algorithm

This first prototyping stage is directly related to process requirement `PR1 - Definition of specification`. The purpose of the specification is the description of functional behaviour of the control system in order that it fulfils an overarching goal (e.g. reduction of load imbalance, optimization of local consumption, maximization of DER hosting capacity). The functional specification is a written and constantly updated document describing the behaviour of the control system and its interaction with its environment in a very detailed but open way. It is one of the most relevant elements for a successful rapid prototyping process, as all other stages and phases of the process rely on this document.

External experts, especially from the grid operator side, are considered here and constantly integrated in the specification process (largely satisfying process requirement `PR10 - Grid operator integration...`). Furthermore, also requirements of external actors (i.e. customers and equipment vendors), described in Section 2.3.1, are considered in the specification (for the satisfaction of process requirement `PR8 - Consideration of customers`). Those can be given by law, individual contracts, or standards and norms.

To determine the detailed set of functional and non-functional requirements for the specification, scenarios are used to imaginary reproduce the control systems behaviour in all its intended and not intended situations. For the creation of these scenarios in cooperation with the diverse set of experts and stakeholders, a common jargon and common terms are used that could come from one of the already existing smart grid related frameworks (e.g. SGAM, NIST, etc.).

Even though the functional specification is written in an open way – meaning that it does not unnecessarily define technical details like the programming language in advance as long as there is no specific reason for that – some technical details, especially concerning those related to the environment and the adjunct interfaces, have to be clearly set. Those details could be the type of available and controllable active grid components and other grid related assets and their interfaces and communication protocols. This ensures that the right control system is specified for the respective distribution grid and its contained equipment. Thus, process requirement `PR12 - Consideration of power grid equipment` is satisfied.

The functional specification is no one-shot document but is constantly maintained and periodically reviewed and updated by one responsible development engineer. Especially due to the agile development type and the short cycle time, constant improvement of the specification and consistency to the current state of the control system under development is essential.

4.3.2 Conceptual Design of the Control Algorithm

While the functional specification is kept in an open form and comprises the close integration of (external) experts and stakeholders, the creation of the control system's conceptual design is done by experienced system development and software engineers (level 2 and level 3 after Cockburn; cf. Table 4.1). This development stage is directly related to process requirement `PR2 - Creation of the conceptual design`.

The (team of) development engineer(s) relies on the specification for the creation of the conceptual design. They apply and combine both established and novel methods of software engineering and electrical energy systems. By doing this, they create concrete technical concept consisting both of software and hardware design to fulfil the requirements and functions that are defined in the specification.

In order to ideally support the agile rapid prototyping process, the design of the control system has a modular or layered structure with increasing complexity and thereby supports the idea of early deployment and incremental improvement. Besides the fact that it allows for rapid deployment of the first implementations, such an incremental approach fosters the success of the overall development task and thus fulfils process requirement `PR9 - Rapidness and high success rate`.

This design phase closes the gap between the open specification and the concrete implementation by concretely (i.e. technologically) specifying the implementation details and defining an adequate software architecture. Such details are, for example, the used programming paradigms (e.g. procedural, object oriented, behavioural/descriptive), attributes, modularity, data types and structures, and complexity reduction by (functional) encapsulation. Already existing solutions, protocols, and interfaces are used to avoid reinventing the wheel.

The algorithmic approach defined in the specification phase has to be evaluated in order to select the appropriate programming language and in order to assess the computational requirements for the selection of the necessary hardware target system. The control system's hardware component, which later has to operate the developed control algorithm(s) in the field, has to be chosen in such a way that it fulfils the computational and interface requirements, as well as other (field) requirements like reliability, extended operating temperatures, vibration and humidity resistance.

In the conceptual design stage, many decisions have to be made in order to find ideally the best architecture for the control system specified in the first development stage. Thus, this task has to be performed by experienced system and software development engineers.

4.3.3 Implementation of the Control Algorithm and System

The implementation stage is the phase where the programmatical implementation of the previously specified and designed control algorithm is performed. This can be coding in the classical sense of literally writing program code, but can also be done by other means of algorithm implementation (e.g. visual programming languages, code generation, etc.). While the selection of the method for the implementation is subject of the developer, the behaviour of the created control algorithms is concretely determined by the specification and the control system architecture.

Typically, a suited integrated development environment (IDE) is used by the programmer(s) to not only program the algorithm but also to verify it, to perform unit tests and integration tests, and for debugging. All these tasks have to be performed in the implementation phase (ideally by multiple developers) in order for the rapid control prototyping process to fulfil process requirement `PR3 - Implementation of control algorithm`.

For the better discussion of the rapid control prototyping concept, at this point, the *control prototyping environment* is introduced. It can be seen as an imaginary workbench for control development, basically consisting of software but later also hardware elements and is depicted in Figure 4.4.

This workbench helps to get a comprehensive overview of the software and hardware actors involved in the rapid prototyping process, as well as of their interactions, relations, and data that has to be exchanged. The central element of this workbench is the control algorithm that is about to be developed, evaluated, revised, and improved. In this first phase of the concept for rapid control prototyping, the control prototyping environment could be substituted by a sophisticated IDE, but within the next evaluation phase this concept has to be extended.

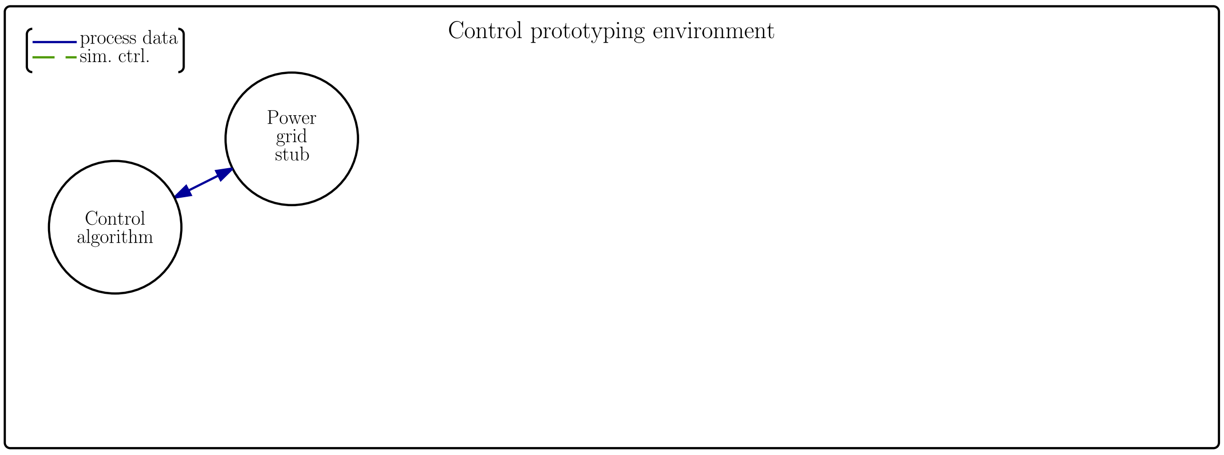


Figure 4.4: Control prototyping environment with actor set-up for the implementation of the control algorithm.

For the modular implementation according to the algorithms specification, and especially for the adjunct first integration tests, some means of interaction with the real world components would be very helpful for the developer. As the integration of real field components into the development process is very cost intensive or just not possible in such an early stage, stubs are used to imitate real world components. Those stubs are simple software replica of elements like e.g. voltage measurement nodes or electric vehicle supply equipment (EVSE). The stubs do not have to be detailed models of those elements and do not need to behave like the real world components but rather just need to mimic interaction with the control algorithm. In Figure 4.4 those stubs are represented by the Power grid stub. There might be also other stubs necessary for the developer (i.e. for each interacting set of components; e.g. to mimic the energy market or other components the control system later has to interact with).

As discussed in Section 4.6.2.1, different types of data have to be exchanged among the actors. Up to this development stage, only messages, e.g. in the sense of meter readings from the grid elements (stubs) or control commands from the control algorithm, are transferred.

Concerning the testing of the control algorithms, this development stage is well suited for extensive unit tests and especially integration tests, due to the fact that the stubs can be modified easily to mimic any conceivable behaviour and data. Besides testing the control algorithms behaviour on the basis of lost and implausible data, also stress tests can be conducted for performance evaluation. On the basis of such tests and performance evaluations, an early selection of varying control approaches can be done before proceeding with the finished control algorithm to the next phase – the evaluation phase.

Even if the development of the control algorithm is accomplished at this stage of development, its installation and operation on the target hardware still has to be performed in this development stage (in order to completely fulfil process requirement `PR4 - Implementation of control system`). This is not necessarily done directly in the first development iteration of the agile process but typically after the software algorithms have successfully been evaluated (by means of software evaluation).

Typically, the target hardware for the operation of the control algorithms is selected of available controller hardware, which could be industry grade PCs, programmable logic controllers (PLC), or simulator components. Furthermore, also an operation of the control algorithm in the data centre of the grid operator is feasible in some cases.

4.4 Prototyping Concept Phase II: Evaluation

Even if the mitigation of development risk begins with a proper specification and an effective system architecture, it is mainly carried out through the comprehensive evaluation of the control system. Therefore, this second phase is the main driver for the fulfilment of process requirement `PR11 - Mitigation of risk`. The evaluation of the control system can be split up into the pure software based evaluation of the control algorithms and the software and hardware based evaluation of the complete control system, which consists of the control algorithms operated on controller hardware.

For testing, as well as for the evaluation phase, the specification is the basic document to rely on. The most effective evaluation can be done by operation of the control system in the target environment, which is the distribution grid. Such an in-system-evaluation would pose an unpredictable threat to the distribution grid, which is not acceptable. In such cases, the evaluation is done by the use of simulations which realistically replicate the distribution grid. This kind of control concept evaluation with power grid simulation forms the first prototyping stage of the second phase, which is depicted in Figure 4.5 (a).

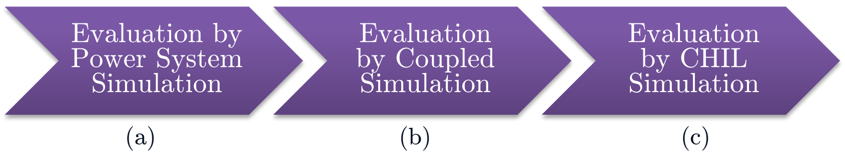

Evaluation by Power System Simulation
Evaluation by Coupled Simulation
Evaluation by CHIL Simulation
(a)
(b)
(c)

Figure 4.5: Development stages of the agile prototyping concept – Phase II: Evaluation.

In an extended approach for power grid control evaluation, also the impact of the used ICT system has to be considered. This is especially relevant if the controlled system is of distributed kind, meaning that sensors and actuators are not directly interlinked but via an ICT system, which typically is the case for networked smart grids. To consider the impact of the ICT system's non-ideal behaviour (e.g. packet delay and loss), it also has to be modelled in simulation. This is done via a separate communication simulator that has to be coupled to the power grid simulation. A thereby created coupled or co-simulation is used in the second development stage of the evaluation phase (cf. Figure 4.5 (b)).

Finally, also the behaviour of the controller hardware must be considered for the evaluation, as also this component has a significant impact on the performance of the control system. Therefore, at first the control algorithms are implemented on the target hardware to completely assemble the control system. This hardware control system is then evaluated in hardware-in-the-loop (HIL) simulation (cf. Figure 4.5 (c)) in the same way as the control algorithm has been evaluated in the previous stage.

The three mentioned prototyping stages (a-c), relating to the evaluation phase of the agile control prototyping process, are elaborated and conceptually designed in the following three subsections.

4.4.1 Evaluation by Power Grid Simulation

The first option to decide concerning the evaluation by power grid simulation is the type of simulation. This choice depends on the type of problem and the algorithmic approach, respectively. In case of a relatively long time scale and a discrete control approach, based on measurements from the grid and control commands to adapt set points of external actuators, quasi-stationary power flow simulation is sufficient. On the other hand, when dynamic effects with time scales in the range of the sinusoidal oscillation time or below have to be analysed, dynamic simulations based on dynamic systems theory have to be applied (e.g. for the analysis of harmonics). For this work the focus lies on quasi-stationary behaviour analysis due to the relatively long characteristic time of the control system in comparison to the sinusoidal oscillation.

The evaluation by power grid simulation is the very basic simulative evaluation approach. Therefore, the specific power grid to which the control system should be deployed later on, has to be modelled in the necessary level of detail. The basis of this model are the topology of the power grid, its power lines, transformers, loads, breakers, switches, and their respective parameters. The more accurate this model is in comparison to the real world, the more this type of simulation helps in the mitigation of risk.

Within this classical grid model, consumers and their applications are situated. All of these components that influence the power grid – either in the form of loads or (distributed) generation units – have to be considered in the power grid simulation in the form of models. Typically all loads within a household are summarized to one node in the simulation with a sum load profile. Also generation units can be considered in those nodes. The consumption profiles for the consumers are either real recorded profiles, statistically deduced (e.g. standard load profiles), or dynamically created by distinct simulators.

One key factor in modelling the power grid is that active and controllable components – such as loads and generation units that can communicate and interact with the control system – have to be modelled separately. They have to react on control commands in the same way as their real world representation would do and their models have to be integrated in the power grid simulation. Therefore, ideally, validated models provided by the equipment vendor should be used, if they exist. Also, possibly existing internal control loops and safety features (e.g. automatic overvoltage and frequency regulation in PV inverters) have to be modelled and considered in the power grid simulation for each and every individual instance that exists in the power grid. By considering the load and generation profiles and the active grid components, both consumers as well as power grid equipment are fully considered in the evaluation phase. (`PR8 - Consideration of customers` and `PR12 - Consideration of power grid equipment`.)

The evaluation by power grid simulation has some specific requirements, as the power grid simulation per se is a simulation of a continuous system, while the control approaches typically are operating discretely. This does not mean that power grid simulation needs to be real-time simulation, but it requires for some kind of coupling between the power grid simulator and the control algorithms in order for them to equally progress in (simulation) time.

In Figure 4.6, the fictive workbench – the control prototyping environment – is depicted, showing the control algorithm under evaluation as the leftmost actor. It is worth mentioning, that in this set-up the power grid stub is replaced by the power grid simulator, which forms the main evaluation set-up. The direct connection between these two elements is the message exchange path for measurements and meter readings in the upstream direction and for control commands and set points issued by the control algorithm in downstream direction.

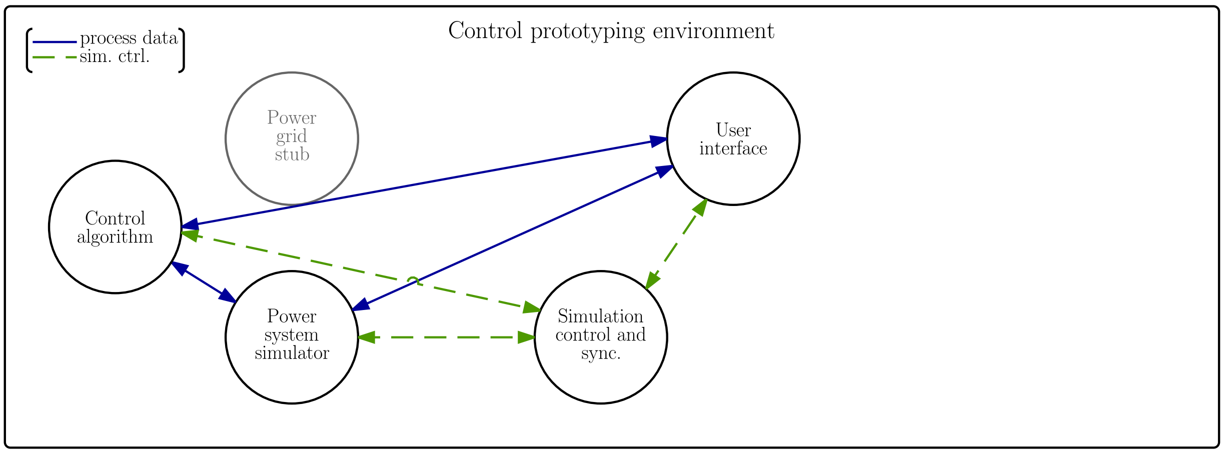


Figure 4.6: Control prototyping environment with actor set-up for the evaluation by power grid simulation.

As the control algorithm and the power grid simulator need to have an execution environment, which could be the developers PC, but also two distinct independently running machines, this communication path must support simulator distribution.

Another actor in the control prototyping environment is the simulation control and synchronization unit. This unit is responsible for maintaining simulation time of the control algorithm and the power grid simulator. Furthermore, it starts and stops the whole simulation and eventually distributes configuration parameters and scenario definitions. These simulation flow control data does not exist in the real world and thus is modelled separately. The external synchronization needs to be supported by the power grid simulator, which is the case for several products on the market (cf. Section 3.2.2).

Finally, one very convenient tool for the development and test engineers is the introduction of a user interface. Especially in the initial phase it is very useful for tracing both messages and simulation flow control data in order to assess and improve the evaluation set-up on the one hand and the control algorithm on the other.

The presented evaluation set-up is already a very effective method for the evaluation of the control algorithm's performance and behaviour and thus fulfils process requirement `PR5 - Simulative control algorithm testing` in a wide range. Unfortunately, it does not take the ICT system into account, which is a relevant issue for networked smart grid systems, due to the fact that the time constants of both systems are in the same range and thus are very likely to influence each other immensely.

4.4.2 Evaluation by Coupled Simulation

As stated at the end of the previous subsection, the simulative evaluation approach is not sufficient by only considering quasi-stationary power grid simulation. Also the ICT system and thus the used communication channel's behaviour has to be taken into account in the simulation by one or more distinct communication system simulators. This creates a classical coupled or co-simulation system where each domain is simulated in a dedicated (established) simulator and state variables are exchanged among these simulators at specific (synchronization) points in time (cf. Section 3.2.1).

The newly introduced communication simulator becomes an intermittent actor between all communicating actors. Its purpose is to realistically model the communication channel that is intended to be used later on in the field. Each message that is sent via a specific communication protocol is not directly sent to its target actor, but to the respective communication simulator. This simulator maintains a message queue and calculates the channel behaviour (e.g. message delay, packet loss, bit errors, congestion, etc.) on the basis of the channel's parameters.

To be able to properly model this channel, the used technology must be defined (in the specification) and the specific channel parameters have to be obtained from the targeted distribution grid. Several sophisticated communication simulators are available on the market (cf. Section 3.2.3), that could be used for that purpose. The challenge here is more the integration of the simulator into the co-simulation environment.

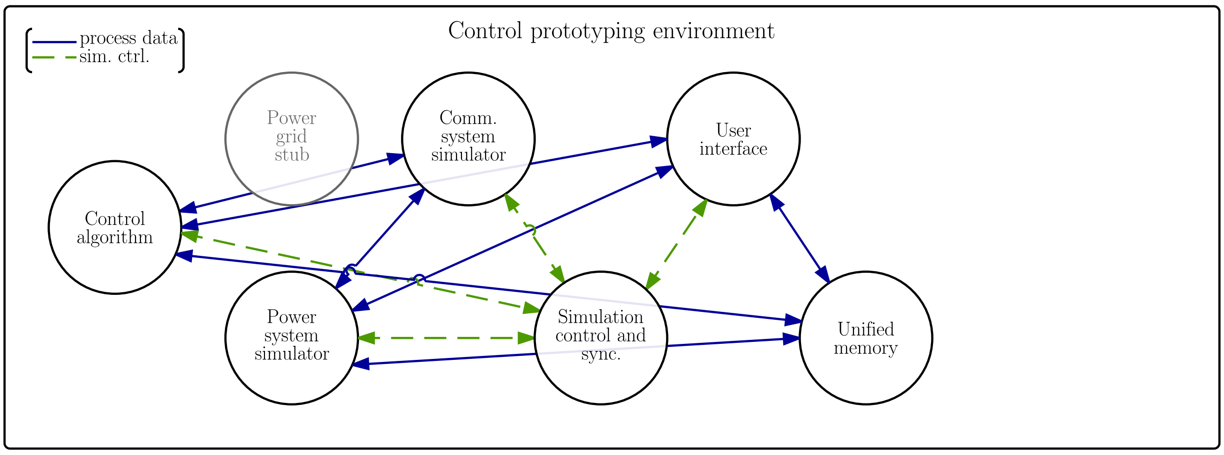


Figure 4.7: Control prototyping environment with actor set-up for the evaluation by coupled simulation.

As depicted in the control prototyping environment diagram in Figure 4.7, the communication simulator does not only interact with all communicating actors but also has to be integrated into the simulation flow control through interaction with the simulation control actor. This is necessary because also this simulator is very likely to be time-based and thus needs to be synchronized with the other actors.

Typically, this synchronization is performed on the basis of regular time slices with constant or variable width or by cyclic negotiation of the progression to the next point in simulation time. The former is performed as follows. The simulation control actor accepts a specific number of actors to participate in the co-simulation. After all actors registered and eventually necessary configuration parameters are distributed amongst them, the simulation time begins to progress. This is triggered by a signal from the simulation control actor allowing every simulator to progress their internal simulation for a specific time slice. In that short simulation period every simulator calculates and updates its internal state, probably sends or receives messages, and then reports the completion of the current simulation step. After all involved simulators finished their particular calculations, the system variables are exchanged amongst the physical simulators and the next time slice is triggered by the simulation controlling actor. This procedure is continuously repeated until a predefined end-time in simulation time is reached.

In this co-simulation set-up, the simulation control actor is the central coordinating element instructing every other simulator and thus handles the simulation control flow. All simulators must be able to interface and handle simulation control flow data issued by this actor.

Also in the co-simulation evaluation period, the user interface is a handy addition to the evaluation set-up. Not only it allows for proper observation of all transferred data; it can also be included as another simulation actor that allows the test engineer to slowly step through the first phases of the co-simulation and to precisely trace the interactions of all simulators with the simulation control actor and amongst each other.

For the later evaluation of the co-simulation results and thus of the control algorithm's performance, a unified memory is introduced into co-simulation. This persistence actor efficiently stores time series of variables and messages and provides them at any point in time for analysis purposes. Optionally, it provides data manipulation and conditioning features like re-sampling of data on a specific sampling interval or interpolation or extrapolation of missing data packages.

With the co-simulation approach it is possible to simulate the smart grid behaviour in an adequate way. Especially due to the lack of established monolithic methods for smart grid simulation, the combination of single system simulators to one co-simulation is a fitting solution to evaluate control algorithm behaviour and thus fulfils process requirement `PR5 - Simulative control algorithm testing`. Through the modelling and simulation of the power grid and the ICT system all necessary components are considered and thus also process requirement `PR12 - Consideration of power grid equipment` is satisfied.

4.4.3 Evaluation by CHIL Simulation

The controller-hardware-in-the-loop (CHIL) evaluation is the last stage of the prototyping concept's evaluation phase. It forms the bridge to the subsequent field operation phase due to the fact that in this CHIL stage, the previously implemented and in co-simulation evaluated control algorithm is already implemented on the target hardware.

Besides the previous co-simulation stage, this evaluation stage is essential for the intended risk mitigation. While with co-simulation the evaluation focus lay on the functional evaluation of the control algorithm, with this off-grid control system evaluation by CHIL analysis, the stable operation on the real target hardware and its interaction with the environment, as well as performance evaluation are anticipated.

The CHIL set-up proposed for this development stage is depicted in Figure 4.8 and is conceptually similar to the co-simulation set-up. The main difference is that the control algorithm in this case is directly operated on controller hardware. This so compiled control system is connected to the power grid and communication system co-simulation instead of the plain control algorithm.

The replacement of the so called "device-under-evaluation", which has been pure software until this point in evaluation and now has evolved to the completely integrated control system, leads to some necessary adaptations.

First of all, the control algorithm has to be deployed to the target hardware. In some cases this can be done just via deployment of the executable binary code and configuration parameters to the target machine. Depending on the architecture of the target system and the development system, cross compilation may be necessary. Some IDEs (e.g. Matlab) also offer code generation, which

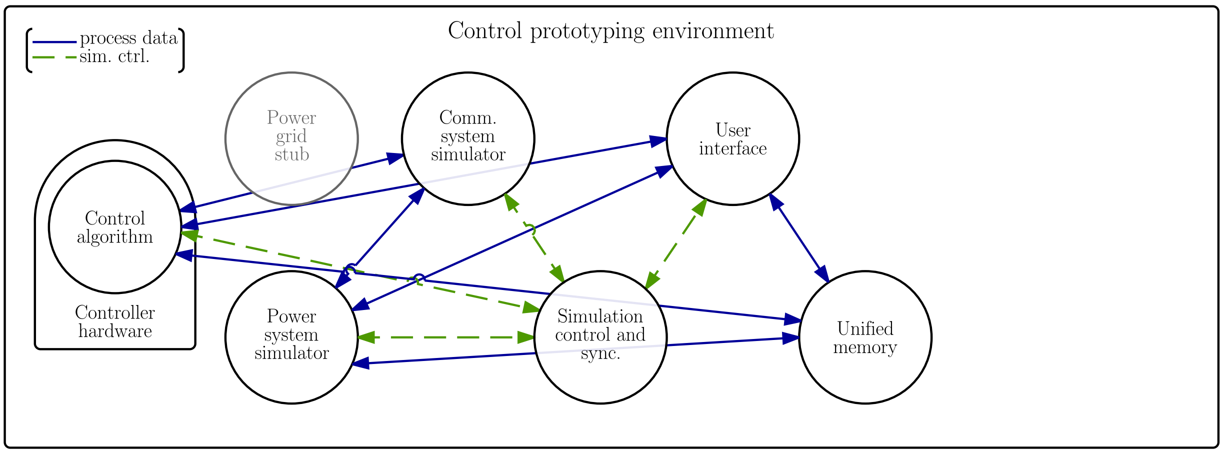


Figure 4.8: Control prototyping environment with actor set-up for the evaluation by CHIL simulation.

can be handy for this purpose. What should be avoided, is the necessity of re-implementation of the complete algorithm due to inconsistency in system architecture or programming languages, as this immensely increases the probability of errors and unwanted or unstable behaviour.

Secondly, interfacing the control system with the co-simulation set-up is performed. Unlike the control algorithm, control hardware only exchanges messages with the rest of the evaluation system. This corresponds to the real field situation. In case of Ethernet and TCP-based communication protocols, the integration into the simulated communication system can be done without hardware adaptation. Other (lower level) communication systems might require for a protocol adapter or driver unit.

The third and last fact to consider is that the control system is connected to the co-simulation as if it was in the field, which also changes the co-simulation methodology. This is caused by the fact that the control system cannot be synchronized by means of the simulation control actor, but rather is independently running on dedicated controller hardware. Thus, also the co-simulation has to run in real-time. This requires the simulation to perform at least as fast as real-time in order not to lack behind the real-time controller. Due to the coupling of different simulators and their inner computational complexity (e.g. convergence of Newton-Raphson power-flow calculation), the compliance with this hard requirement has to be checked individually for each set-up.

A preliminary summary of the evaluation phase shows that, by pure power grid analysis, the developed control system cannot sufficiently be evaluated due to the significant impact of the communication system on networked smart grid systems. Therefore, co-simulation of the power and communication system is employed which allows for a comprehensive simulation of the power grid system. The usage of established simulators is highly anticipated and their capability to be integrated into such a co-simulation system is essential. Furthermore, appropriate modelling of the respective power and communication grids together with all their contained elements is necessary to realistically reproduce real world situations and thus to effectively evaluate the prototyped control system. The CHIL evaluation stage adds another surplus to the evaluation phase by also considering the real controller hardware. Thus, many necessary tasks, like cross compilation, system compatibility checks, and performance evaluation can be performed prior to the deployment of the control system to the field. Process requirement `PR6 - Off-grid control system testing` can therefore be satisfied also within the evaluation phase.

4.5 Prototyping Concept Phase III: Field Operation

The presented prototyping concept does not only consist of the implementation and the evaluation of the control system. It covers the whole prototyping process also including field deployment and operation. In this section, the concept for the field operation phase, which is the third and last phase of the prototyping process, is presented.

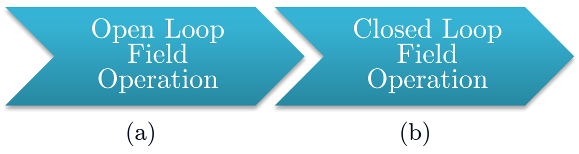


Figure 4.9: Steps of the Agile Prototyping Concept Phase III

This third phase consists of two stages and is schematically depicted in Figure 4.9. The two stages can be distinguished by their operation type: (a) open loop or parallel operation and (b) closed loop operation. These stages will be described in the following sections.

4.5.1 Open Loop Field Operation

The intermediary work that is not explicitly mentioned here but has to be done between Stage II and Stage III, is the deployment of the control system (i.e. the developed control algorithms on the target control hardware) to the field. Due to the fact that the control system has been completely assembled during the CHIL evaluation stage, the deployment process is relatively effortless because no re-implementation or excessive adaptations have to be made.

Figure 4.10 shows the fictive control prototyping environment, which is now represented by the field (i.e. the substation or control room, where the control system is operated). In this set-up, all simulation software actors are detached from the controller and replaced by their real world representations. To be specific, all field components – or rather the communication system to access these distributed field components – are accessed by the field automation gateway. Concerning the conceptual representation in Figure 4.10, the two actors *Power Grid Simulator* and *Communication System Simulator* are replaced by the *Field Automation Gateway*.

As depicted in Figure 4.10, in field operation mode, only messages are exchanged among the actors. The only computational element in this set-up is the developed control system, which is operated in real-time. There is no synchronization mechanism intended for any field component or other actor. Thus, the *Simulation Control* actor is also not included here.

The control system is operated in the field, where typically no direct access to the component is available, while it controls the critical power grid infrastructure. Due to this fact, it is highly anticipated to be able to supervise the controller at any time. Therefore, the *User Interface* actor is foreseen in the concept for field operation. This actor provides direct information about the grid and the control algorithm's status. Via this interface, also time series of measurements, which are persisted in a *Unified Memory* actor, can be obtained. Another relevant actor for the field operation mode is a *SCADA Interface* actor. This actor provides an uplink from the station level to be able to integrate data and operation modes of the control system into an overlay SCADA system.

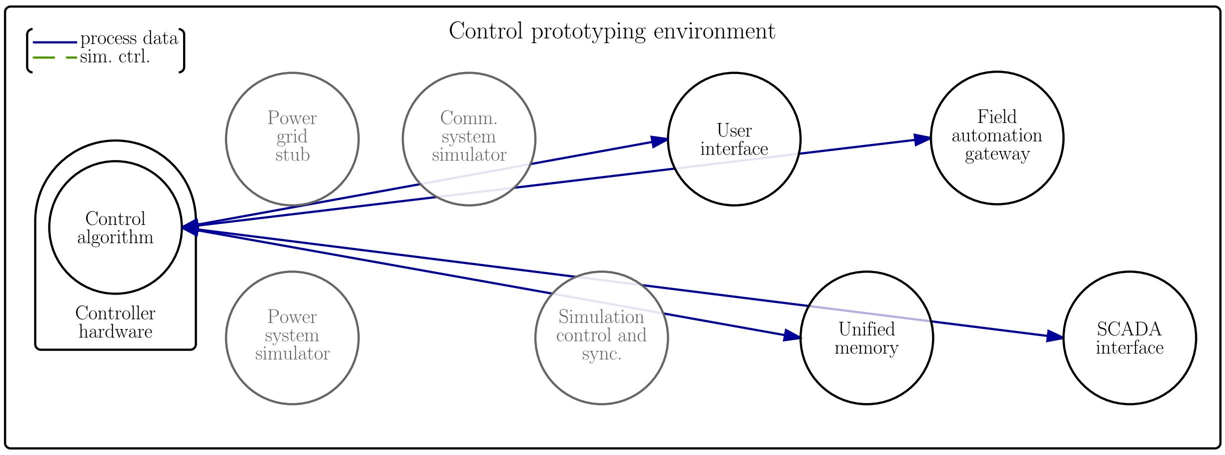


Figure 4.10: Control prototyping environment with actor set-up for the open and closed loop field operation phase.

Directly after the field deployment of the control system, this is the first operation mode which also takes the control expert into account as depicted in the sequence diagram shown in Figure 4.11.

The open loop operation is a parallel operation mode, where measurements are aggregated by the *Field Automation Gateway* and transferred to the *Controller*. The *Controller* processes this data and eventually issues a control command for the actors in the field. In open loop mode this control command is redirected to the *User Interface* where an expert has to assess the commands and acknowledge them in order to forward them to the *Field Automation Gateway* and the *Field Actuator*, consequently. Depending on the decision of the control expert, the respective command is forwarded or not. These optional actions are enclosed in the `opt`-section of the UML sequence diagram in Figure 4.11.

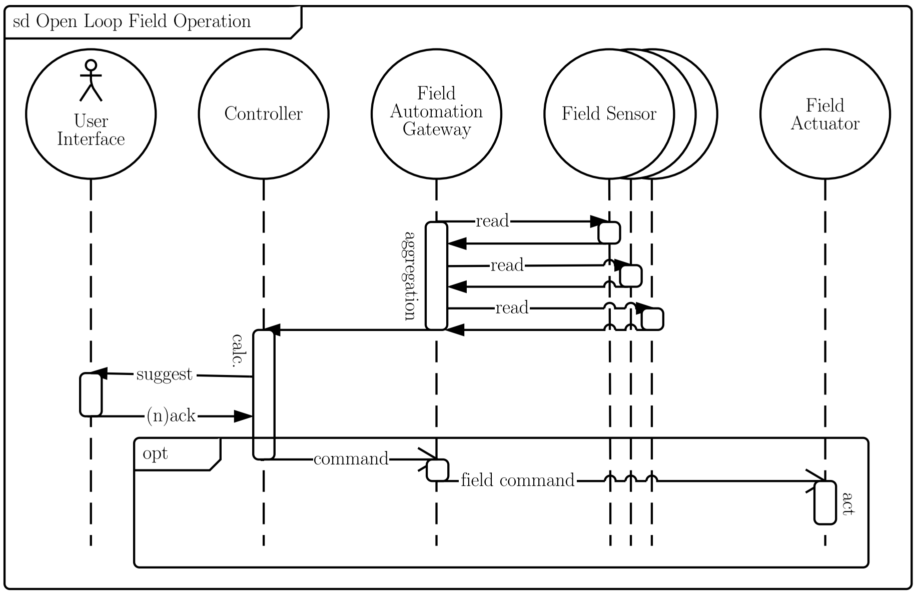


Figure 4.11: UML sequence diagram showing one open loop control iteration with the control expert optionally acknowledging the control command via the user interface.

The described open loop operation mode is a common method for the integration of new controllers in complex systems. It is another measure for the mitigation of risk along the agile development process and thus is included in the presented concept to further satisfy process requirement `PR11 - Mitigation of risk`.

4.5.2 Closed Loop Field Operation

The closed loop field operation is the final stage and the goal of the agile development process. In this operation mode, the developed control system operates in the field independently, without the need for expert interaction. The actor set-up is equivalent as for open loop operation mode, which is presented in Figure 4.10. The only difference to the open loop operation mode lies in the actor interaction, which is presented in the following sequence diagram in Figure 4.12.

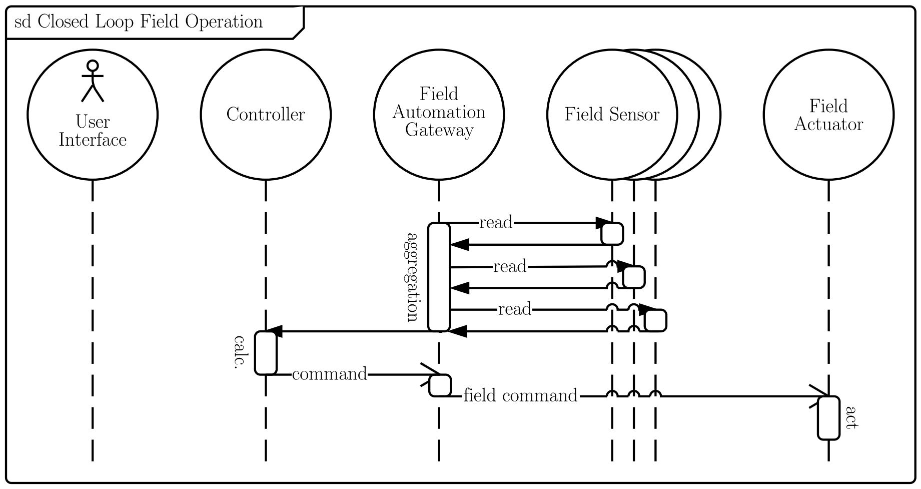


Figure 4.12: UML sequence diagram showing one closed loop control iteration without control expert interaction.

In closed loop operation mode, the control loop, which consists of measurements from *Field Sensors*, being processed by the *Controller*, which then issues a control command for influencing specific *Field Actors*, does not require user interaction any more.

Even though acknowledgements from a system expert are not necessary any longer, the *User Interface* is still a relevant actor. On the one hand, it is used to observe the control system's behaviour in the mid- and long-term and in special, unusual events. On the other hand it is used to extract time series data from the *Unified Memory* actor for long-term system behaviour assessment.

All presented methods are measures for the assessment of the control system's performance and suitability for fulfilling the overarching control goal. At this point in time, the prototyping process does not end, but rather remains in the final stage. Especially during the operation of the first instances of the newly developed control system, this final stage can be left again to go back to any of the previous control development stages in order to enhance the control system's performance or functionality or to fix issues identified during field operation.

4.6 Analysis of the Concept for the Agile Development Process

The presented concept for the development process is analysed hereafter according to its suitability to serve the three major objectives (rapidness, seamless field deployment, risk mitigation). Subsequently, the conceptual design is discussed according to its practicability, as the presented process should be a practicable approach for rapid control prototyping, which should also be applied by academia and industry. The results of this discussion directly flow into the realization of the process.

4.6.1 Elementary Objectives

The three elementary objectives of the process that should be reached by the satisfaction of the process requirements (which have been elaborated in Section 2.3) are

- its rapidness,
- the seamless field deployment, and
- its suitability to mitigate risk along the development process.

Rapidness is an objective that can directly be mapped to time that it takes for a development team to create a control system from the first specification until the deployment of the finally developed control system. As for a straight-forward task, this can easily be measured and compared, direct time comparison in the case of control prototyping for networked smart grid systems might not be as meaningful. This is caused by the facts that each prototyping task is individual for each specific distribution grid and that distribution grid control prototyping is a rather new discipline. The latter fact typically causes high learning effects due to the immense gain in experience of the development team.

One could argue, according to Section 4.1, that agile development processes are in general leading to results faster than plan-driven development methods. While this seems to be true at first sight, in an overall assessment this is just an apparent gain. In agile development, a first implementation can typically be deployed earlier, but does not provide the full functional range. The full set of features is created gradually by the iterative steps. Thus, the agility of the process might serve more for the overall prototyping success and the ability to consider changes in the specification than to the process' rapidness. Thus, the rapidness has to be evaluated by (development) experts, who are already experienced in the field of grid control development, in a subjective way.

The objective of seamless field deployment is aligned in certain particularities with the risk reduction objective, because the simpler the deployment step is, the less mistakes are being made. Since the deployment process per se is implicitly contained between development Stage II and Stage III of the process, making it simple influences the whole process architecture.

What does simplicity mean for this particular problem of field deployment? While one might argue that, when talking about deployment, there is no simple way, the focus here is on reduction of extra configuration and re-implementation efforts to make the deployment relatively simple. Thus the control algorithms, as well as the whole control system, has to be developed as field-near as possible; not in a regional sense but a technical one. The message syntax and semantic are chosen already in the first implementation phase in such a way that it will later be compatible with the field set-up. The interfaces, communication systems, and controller hardware are chosen similar or equal to the real field components. Software components (e.g. the user interface and the unified memory) are created in a generic way so that they can be ported directly to the field.

The mitigation of risk is a two-dimensional problem, as risk is commonly known as the (negative) consequences of an event in combination with the likelihood of its occurrence (e.g. stated by ISO 31000 [ISO09]). Due to the fact that it is hard to limit the consequences of a faulty control system by the definition of the adjunct prototyping process, the focus clearly lies on the reduction of its occurrence's likelihood. This likelihood reduction is maintained along all the three prototyping phases. The basis for risk mitigation thereby is laid by a proper specification and design. The lion share of the likelihood reduction is performed by the comprehensive evaluation phase (Phase II), where the three focus areas power grid, ICT, and control hardware are considered comprehensively and thus the whole spectrum of the control system and its interaction is covered. This is performed by the proposed co-simulation and CHIL evaluation set-up. While this is a very comprehensive approach, it cannot guarantee 100 % coverage. Thus, the evaluation is continued also in the field operation stage with an open loop operation mode and further observation during closed loop operation.

4.6.2 Discussion of the Conceptual Design

The conceptual design presented in this chapter evolves with the complexity of the development stages and incorporates several software and later also hardware components (referred to as actors). These actors are exchanging information amongst each other, which is distinguished as process data and simulation control data. Within this discussion the actor interaction, which is essential for the described process, is generalized. Actor-classes and message types are defined in a clearer way in order to prepare the concepts implementation. The concept is analysed concerning its scalability and an appropriate message exchange middleware is proposed.

4.6.2.1 Actor Classes and Message Types

All actors of the proposed concept that are interacting with the controller under development belong to one of the following three actor classes.

- Physical system simulators
- Discrete event simulators
- Hardware components

The physical system simulators are representing (continuous) real world processes like the power grid, thermal processes in houses, or electro-chemical processes in batteries. This simulators maintain physical state variables and their dynamics (e.g. by state-space representation with ordinary differential equations). Discrete event simulators are representing systems that are not directly related to time but based on events and their dependencies. (Packet based) communication system simulators are a good example for event based simulators. These simulators are using an internal event queue for maintenance of the process model. Hardware components are typically connected when the control system is deployed to the field or in the earlier hardware-in-the-loop development stages. This could be either real field components (i.e. in the lab or field) or real time digital simulators (RTS) as described in Section 3.3.

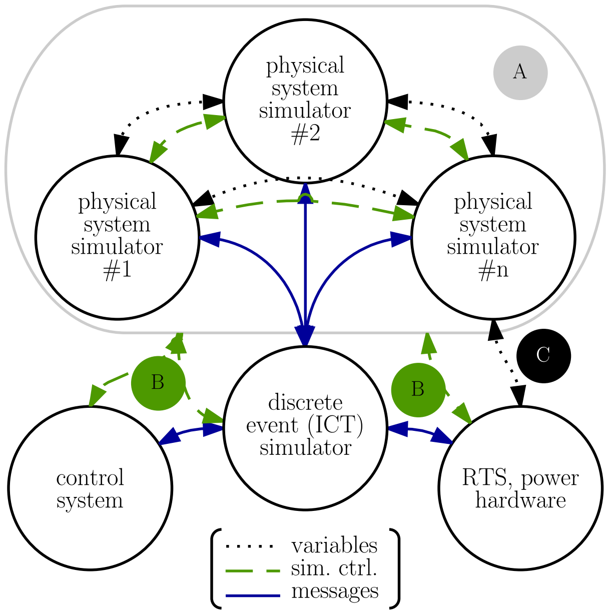

physical system simulator #2
A
physical system simulator #1
physical system simulator #n
C
B
B
discrete event (ICT) simulator
control system
RTS, power hardware
variables
sim. ctrl.
messages

Figure 4.13: Generalized actors interface diagram showing physical system simulators, discrete event simulator, and the control system – either as hardware or as a software component, exchanging three types of messages.

These three actor classes and their interactions are depicted in Figure 4.13. This figure also contains three different types of data and information that are being exchanged amongst the actors.

The black dashed lines in the figure represent exchange of physical state variables. These are internal states of the physical simulators that are influencing the simulation of the other physical system simulator(s) and thus have to be exchanged regularly. These physical variables only need to be exchanged in case of an interaction and mutual influence of multiple physical systems that are simulated in dedicated simulators. Such a compound of multiple physical system simulators represents the real world behaviour, which is indicated as (A) in Figure 4.13. Another special case where physical variables have to be exchanged, is hardware integration, where real physical inputs and outputs of a hardware component have to be synchronized with the component's simulated representation (e.g. integration of EVSE in power grid simulation). This case is marked as (C) in Figure 4.13.

Messages (represented by blue solid lines) exist in both, the real and the simulated environment, while physical variables are not explicitly exchanged in the real world representation of the simulated system but influence each other rather implicitly. Messages are explicitly issued or processed by components in the real world (e.g. controllers, sensors, actuators) or their simulated models. They typically are transmitted via a separate communication channel that has to be modelled and simulated in order to consider its influence on the simulated system.

Both, messages and physical variables, exist in the real system – either implicitly as physical states or as explicit messages. In simulation, this data have to be exchanged among the actors via appropriate links and channels in an efficient way. In the control prototyping environment figures of this chapter messages and physical variables as denoted as *process data*.

The third type of data, which is only necessary for simulation purpose, is simulation (flow) control data. This type of data (green dashed line in Figure 4.13) is essential for the co-simulation to keep the single simulators synchronized and to control and maintain their simulation progress. Dependent of the type of simulation, simulation control data is used for the configuration of the single simulators. It is used for the distribution of simulation scenarios among all simulators, to start and stop each simulation process, and to step through the single simulation processes.

Simulation control data is not only used for the synchronization of single (physical or discrete event) simulators, but also for the integration of hardware into these HIL simulations. The need for such a synchronization of hardware components – represented as (B) in Figure 4.13 – depends on the type of hardware to integrate and the type of simulation. While real-time simulation is necessary fir HIL simulation, simulation control data can be neglected for simple hardware components. On the contrary for complex hardware or control components or speed up real-time simulation this type of data is inevitable in order to properly integrate the hardware component(s) into the simulation process (e.g. initialization of a battery emulator).

4.6.2.2 Scalability

The concept that is presented in this chapter dynamically combines several actors in an imaginary control prototyping environment to fulfil the task necessary for the respective prototyping stage. Figures 4.4, 4.6, 4.7, 4.8, and Figure 4.10 show an apparently unmanageable amount of heterogeneous connections among the software and hardware actors of the process concept. Due to the need for data exchange among the actors, the complexity of the control prototyping environment rapidly increases with each and every added actor.

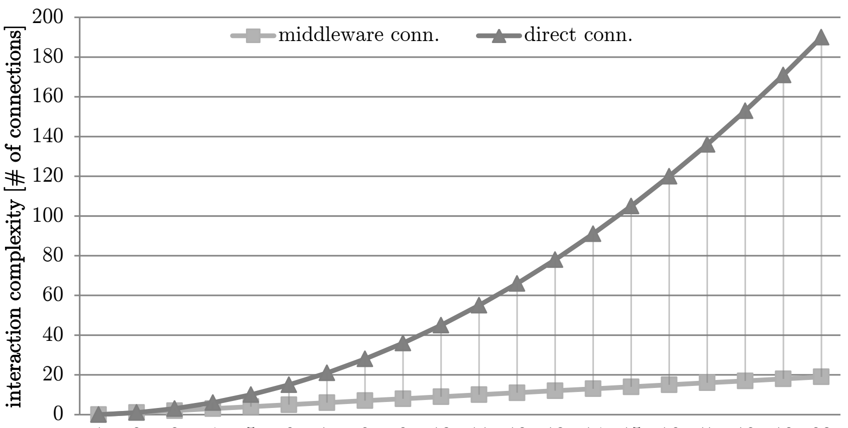


Figure 4.14: Scalability analysis comparing the direct connection complexity against a middleware using approach.

This increasing complexity can be expressed by the use of Landau notation and is depicted in Figure 4.14. An exact number of necessary connection cannot be determined, because this depends on the type of simulation and the used actors. Thus, in Figure 4.14 an upper and lower limit is presented. In case that each actor would have to interact directly with all other components, the upper limit of interactions (marked with triangles) would be reached. For a number of n_A actors this would lead to n_I necessary connections according to Equation (4.1). For the unpredictable exact number of necessary direct connections per actor, the rational number κ_d is introduced, which is defined by Equation (4.2).

$$n_I(n_A) = \kappa_d \cdot \frac{n_A \cdot (n_A - 1)}{2} \quad \in \quad \mathcal{O}(n_A^2) \quad \text{for} \tag{4.1}$$

$$\frac{2}{n_A - 1} \leq \kappa_d \leq 1 \quad \text{and} \quad n_A > 1 \tag{4.2}$$

As it can be seen at the upper limit presented in Figure 4.14, interaction complexity grows at $\mathcal{O}(n_A^2)$, which quickly becomes unmanageable, especially if actors require multiple connections for different data types (i.e. process and simulation control data).

To avoid such an unmanageable number of interfaces and connections, a common data exchange middleware is introduced (depicted in Figure 4.15). This middleware serves as a common data router that significantly reduces interaction complexity according to Equation (4.3), with $\kappa_{mw} \geq 1$. The result is represented by the lower limit depicted in Figure 4.14.

$$n_I(n_A) = \kappa_{mw} \cdot n_A \quad \in \quad \mathcal{O}(n_A) \tag{4.3}$$

With a universal and flexible interface it allows data exchange amongst all actors with only one connection per actor, resulting in $\kappa_{mw} = 1$.

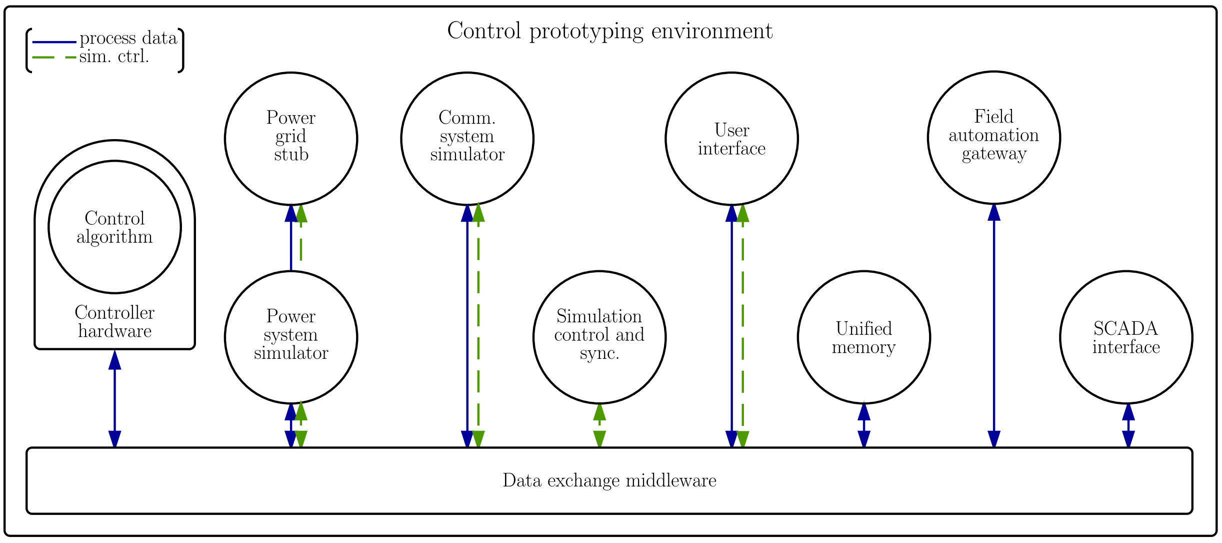


Figure 4.15: Control prototyping environment with actor set-up using a common data exchange middleware for the reduction of interaction complexity.

The introduction of the middleware reduces interaction complexity significantly from quadratic $\mathcal{O}(n_A^2)$ to linear $\mathcal{O}(n_A)$ growth. The use of the middleware is the only way to handle a control prototyping environment for rapid control prototyping. Due to this fact, it is an essential element of the prototyping process.

5 Design of a Prototyping Middleware

A comprehensive concept for the agile development process for rapid control prototyping has been elaborated in the course of this thesis and has been presented in the previous chapter. Its single development phases couple diverse tools with off-the-shelf simulators and require interaction between the developed control algorithms and the controller hardware. In the discussion at the end of the previous chapter, different actor classes have been identified and their interaction has been analysed. The information that has to be exchanged among the actors is assigned to three distinct data types: physical variables, messages, and simulation control data.

To enable this necessary information exchange, as well as to dynamically interlink all actors with the (prototypical) control algorithms, the use of a data exchange middleware is highly recommended to keep the system complexity in manageable bounds. There are solutions available to serve as such a middleware, which are described in Section 3.2. The analysis conducted in this section revealed that none of these solutions fulfil all the requirements resulting from the prototyping process. Especially the direct field deployment of the developed control algorithms is a necessary feature that is not supported by any of the analysed middleware solutions.

Thus, a lightweight solution for such a middleware, the Simulation Messages Bus (SMB), which has been created in the course of *Smart LV Grid* (cf. Chapter 2.2.3.2 and [FDL$^+$13, MF12]), is presented in this chapter. This middleware implementation does not only fulfil the requirements related to message exchange, but also allows for further functional extensions (i.e. co-simulation control, scenario management, simulator synchronisation, developer convenience, and direct field deployment).

5.1 Concept and Requirement Consideration

The prototyping middleware's primary purpose is to act as a common information exchanging entity to avoid unmanageable complexity of the control prototyping environment. Therefore, it should be designed in a generic, modular, and flexible way to be able to allow interaction of the heterogeneous set of actors, that were introduced with the prototyping concept in the previous chapter.

The interfaces of the middleware should be agnostic to the type of data transmitted in order to allow the exchange of all three data types defined in Section 4.6.2.1. While the topology of the data paths is inherently contained in the presented development stages, it has to be considered by the middleware. This issue could be overcome by the implementation of simple hub behaviour where all packets arriving at the middleware are forwarded to all connected actors. Due to performance and security reasons (especially in the field operation stages) this is not an appropriate solution. Rather some means of packet routing needs to be introduced.

In order to support the seamless field deployment of the developed control algorithm (process requirement `PR7`) – which is an elementary objective of the rapid prototyping process – the middleware has to be designed in such a way that it can be deployed together with the control algorithm(s). This would allow for a flexible set-up and operation also in the field and supersedes the porting or reimplementation of the algorithms. By this method, a simplification is reached that accelerates the deployment process (process requirement `PR9`) and greatly supports risk mitigation (process requirement `PR11`) by the avoidance of porting and reimplementation.

5.2 Exemplary Middleware Implementation

An implementation of the data exchange middleware, proposed in the discussion of the process concept (cf. Section 4.6.2), is designed within this section and presented according to related publications [FDL+13, MKFS13]. Furthermore, its exemplary implementation called Simulation Message Bus (SMB) is described subsequently and matches in its general structure the concept proposed in [KML15, Chap. 2]. At first however, the general architecture is presented and discussed followed by a detailed description of the entities required for the architecture.

5.2.1 General Architecture

In consequence of the heterogeneous set of actors (software simulators, hardware components, and the control system), the middleware architecture builds up on these actors' lowest common denominator, which is data transfer over TCP/IP socket connections. Figure 5.1 shows the basic architecture for that socket-based central data exchanging middleware – the Simulation Message Bus.

The middleware is built up on a core element that instantiates and combines a defined set of so called *proxies*. A proxy is an abstract object that provides two queues and two processing methods. Physical signals, messages, and simulation control data are transferred by means of those proxies' queues from one actor to the other(s) as so called *packets* contained in *envelopes*.

Figure 5.1 shows three arbitrary actors connected to the middleware. Concerning the type of connection, two different types of actors can be distinguished. The first type is an external (optionally distributed) actor, such as distinct physical systems simulators (e.g. power grid simulator). This actor type connects via TCP sockets (represented by the round connectors in the figure - used by Actor A and Actor B). The second type of actors are internal actors interfacing the middleware. They are implemented like a proxy – as a specialization of the proxy – but do not propagate the received packet to another proxy after processing it. Rather, this type of actors can be seen as middleware-internal end-node-proxies operating as independent processes of the middleware (cf. Actor C in the figure). This is very useful for the implementation of maintenance and convenience functions, e.g. logging or persistence issues, and for interfacing (e.g. web services).

Another elementary specialization of the proxy, depicted in Figure 5.1, is the connector. It provides capabilities for packet streaming via TCP socket connections to the external actors. This allows for the distribution of single actors (e.g. in co-simulation) amongst various (real or

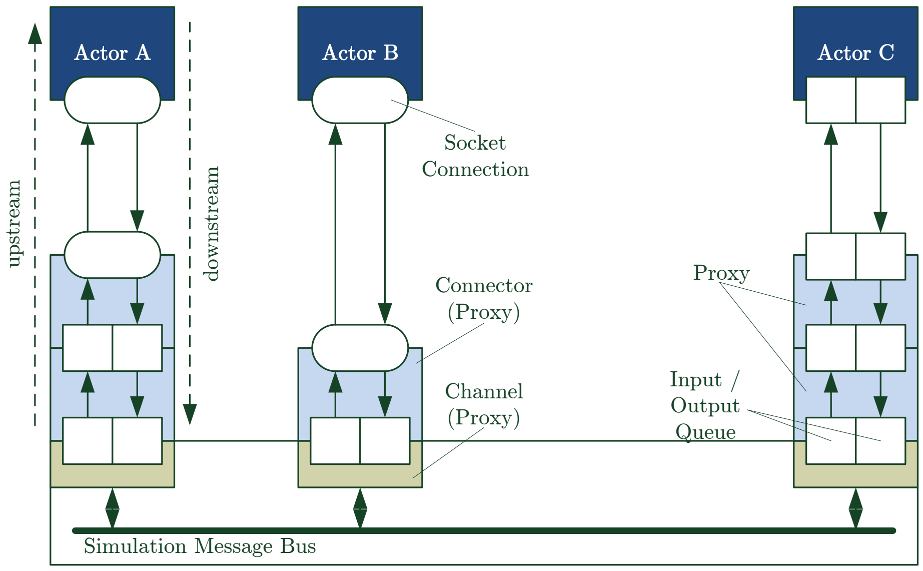


Figure 5.1: General architecture of the prototyping middleware connecting three actors in a flexible way [MKFS13].

virtual) machines and simulation cores. Furthermore, by means of this socket interface, external hardware components are integrated, e.g. for CHIL simulation.

5.2.2 Configuration

The flexible architecture of the prototyping middleware, described above allows for an individual set-up for each development stage of the proposed concept. A channel is instantiated for each actor, either connected externally via a network connection or as a middleware-internal actor. The channels are equipped with an arbitrary number of stacked proxies to extend their functionality to best integrate the respective actor. Those individual middleware set-ups are stored as configuration files in the XML format following the syntax presented in the XML Schema Description (XSD) in Appendix A.1.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<Configuration>
  <Channel id="channel_1">
    <Proxy id="proxy1" type="proxyTypeA">
      <param name="param1" value="value1"/>
    </Proxy>
    <Proxy id="proxy2" type="proxyTypeB">
      <param name="param2" value="value2"/>
    </Proxy>
  </Channel>
</Configuration>
```

Listing 5.1: Minimalistic XML configuration describing the middleware architecture.

Listing 5.1 presents such a very basic XML architecture description. The presented architecture consists of only one channel that hosts two proxies. This channel is comparable to the channel of Actor A in Figure 5.1. The two proxies of the channel are stacked and thus forward incoming and outgoing packets to each other after processing them. The proxies are of different types, which refer to a specific implementation (e.g. routing proxy, connector proxy, etc.). Each channel can host an arbitrary number of proxies and each proxy is assigned a list of parameters relevant for its operation. The XML configuration is read during the start-up phase of the middleware. In this phase it is used to instantiate the defined number of channels and their related proxies and thus creates the destined architecture.

5.2.3 Data Format and Routing

Even though the middleware implementation aims to be a universal component, a minimum data format specification is necessary to ensure proper data routing. Thus, the envelope is defined as the basic data exchanging element and presented in this section. Also, routing functionality is described, which relies on the routing information contained in the envelope's header. An optional data model for information exchange in the prototyping process is proposed subsequently. Finally, the proxy concept is refined and some proxy implementations are presented.

5.2.3.1 Envelope Definition

The basic structure of the envelope, which is the basic information exchange element in the middleware, is illustrated in Figure 5.2. It consists of two basic elements, which are the routing path and the payload. While the former contains information about the routing targets, the latter is a generic element that can be of any object type. As there is no data model inherently implied in the envelope definition, arbitrary transport layer protocols can be encapsulated in the envelope structure. This allows for a very generic usage of the prototyping middleware.

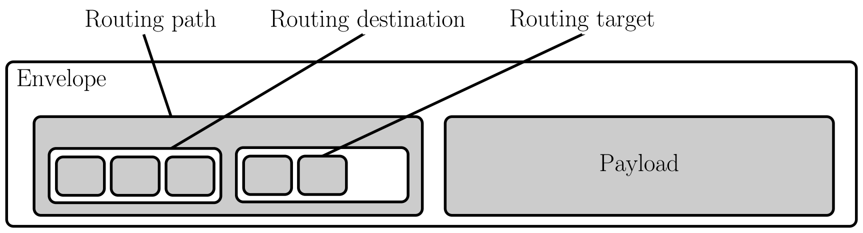


Figure 5.2: The data format of the envelope that is used for information exchange by the middleware [MF12].

The routing path hosts routing targets, which are the IDs of the distinct channel, in a specific manner in order to allow for flexible routing, described in the next subsection.

5.2.3.2 Routing

The topological information, that was implicitly included in the direct actor connections of the presented concept, is translated to the explicit routing information of the middleware implementation. Routing of the envelopes ensures that the information – in the form of envelopes carrying the payload – is delivered to the correct actor(s).

Due to the fact that each actor directly relates to a channel, the unique channel IDs (referred to as routing targets) are compiled to a routing table, which is the basis for the middleware's routing functionality. The routing path, depicted in Figure 5.2 consists of a set of routing destinations.

Those routing destinations contain one or more routing targets. This unordered grouping of routing targets to an ordered set of routing destinations allows for a multicast as well as for a multipoint routing of the envelopes.

In case of an envelope reaching the middleware, its routing header is analysed. The first routing destination is used as the set of target channels where the envelope is simultaneously forwarded to as a multicast. This routing destination is removed from the routing header before forwarding the envelope. By this ordered set of routing destinations, a multipoint routing path through the set of actors becomes possible.

5.2.3.3 Packet Data Model

The different types of information that are exchanged during the complete prototyping process have a very similar structure. They primarily consist of sensor readings and actuator set points, physical signals, or simulation flow control commands. An exemplary use case is as follows: sensor readings are requested by the control system, which receives a reply message either from the simulated measuring component or the real field device. For the purpose of such information exchange, a lean data model for *packets* is defined and illustrated in Figure 5.3. These packets are used for information exchange in the prototyping process and are transferred as envelope payload via the middleware.

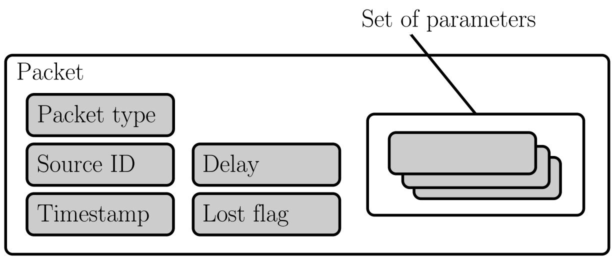


Figure 5.3: The packet data model for information exchange along the whole prototyping process [MF12].

The packet data model contains the following parameters, that are also listed in the packet grammar description in Listing A.2 of Appendix A.1.

- The `packet type` attribute is an enum object that is intended to be set to one of the four values: `create`, `request`, `update`, and `delete`.
- The `source id` attribute represents the channel ID of the sending source. This makes it possible, e.g. for the receiver of a request, to send its answer back to the correct channel and thus to the correct actor.
- The `timestamp` attribute is used to tag the timestamp of a measurement. Furthermore, it is of high interest for (co-)simulation development stages in order to maintain simulation time and events within the virtual simulation time.

These attributes are common attributes which are usable for any of the development stages. Especially for the simulation incorporating stages, two further attributes are available, which are:

- The `delay` attribute is used for the communication channel simulation to indicate the duration of packet delay caused by the impact of the communication channel.
- The `lost flag` attribute is also used for the consideration of ICT influence during a simulation. It indicates whether the packet got lost during the simulated transmission via the communication channel.

Finally, an arbitrary set of parameters is provided in the packet data model. This is the actual information storage for user/actor data. In this list an arbitrary number of parameters can be stored. Each parameter is a triplet of the fields `id`, `var`, and `value` represented as `[id].[var]=[value]` (e.g. `trafo08.mluActivePower=1337`). For request packets, the value field is likely to remain empty.

5.2.4 Proxies

Proxies are the building blocks of the middleware and provide a modular and extensible architecture. They are realized as abstract objects that can be specialized in their implementations to fulfil specific tasks. The most relevant implementations of the proxies for the prototyping process are presented subsequently.

5.2.4.1 Proxy Concept

The proxies are stackable and extensible objects that are used for information handling and processing by means of transmission and processing of objects like envelopes and packets. The proxy's internal structure is depicted in Figure 5.4.

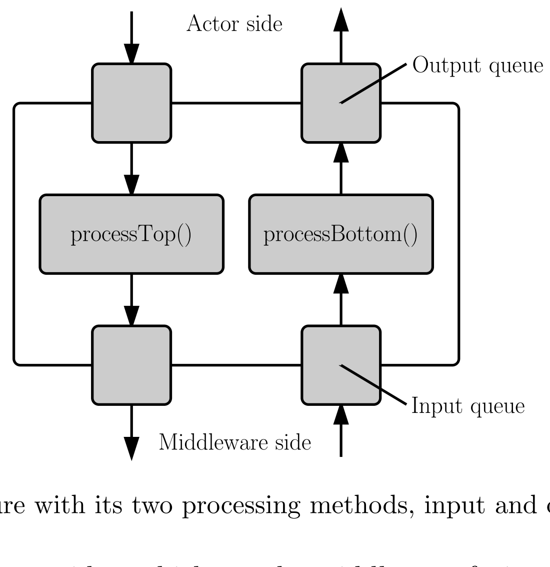


Figure 5.4: The proxy structure with its two processing methods, input and output queues [MKFS13].

Each proxy is accessible from two sides, which are the middleware facing side (the bottom side in the figure) and the actor facing side (the top side in the figure). Incoming and outgoing objects are stored in receive and transmission buffers (blocking queues); two on each side. While the two transmission buffers are members of the proxy, the two receive buffers are referenced objects of the next stacked proxy.

Besides these buffers, two processing methods (`processTop()` and `processBottom()`) are contained in the proxy. In the basic proxy implementation, these methods simply forward the objects from the receive buffers to the transmission buffers. To create advanced functionality in advanced proxy implementations derived from the basic proxy, these two methods must be overridden.

To operate the proxy's functionality, two processes are instantiated in each proxy. These two processes independently take objects from the receive buffers, process them by the described processing methods, and put the resulting object to the transmission buffers. One process is responsible for the top-processing side, the other one for the bottom-processing side.

For the case that, due to any reason, one of the queues should not be worked off and thus runs into risk of overflowing, a high watermark level is defined. If this limit of objects is reached, either the whole buffer content will be discarded or a predefined share of the oldest contained objects. This behaviour can be configured by the proxy's parameters in the configuration file (cf. Section 5.2.2).

5.2.4.2 Proxy Implementations

In order to establish a proper connection between the actors and the middleware, the following proxy implementations were realized in the course of the *Smart LV Grid* project (cf. Section 2.2.3.2), besides others. They are used in different prototyping middleware configurations to fulfil the implementation, evaluation, and field operation tasks of the process' distinct development stages.

Routing Proxies

The middleware is made to handle objects of type envelope, presented in Section 5.2.3.1. Those envelopes contain a routing header, besides the actual payload, that is used for proper delivery of the information. As illustrated in the figures (4.4, 4.6, 4.7, 4.8, and 4.10) of the control development environments in Section 4.2, the routing of information is constant for each development stage and therefore related to the actual middleware configuration. Thus, it makes sense to implement the routing proxy that overtakes the task of wrapping the actors' data packets into envelopes and setting the routing header.

This proxy typically is the one closest to the middleware and receives a list of routing destinations consisting of routing targets (i.e. the destined channels' IDs – cf. Section 5.2.3.1). As a consequence of the wrapping and unwrapping feature of this proxy, all other proxies stacked behind this one, as well as the respective actor, have to deal with packets for all further operations.

The described proxy realisation as a routing proxy provides static routing capability. By the use of the middleware with this proxy together with the proposed socket connection, the minimum configuration for a prototyping middleware is realized. Especially for dynamic actors or actors with diverse functionalities, the static routing is not very convenient. Those actors (e.g. the communication system simulator) require dynamic routing on the basis of packet information (e.g. routing measurement requests to the power grid simulator, while routing the replies to the control system).

For that purpose another proxy implementation was created which is a dynamic routing proxy. This is a more complex proxy as it performs packet inspection in its processing methods. The packet content is checked to match routing patterns (using regular expressions) and the routing destinations are set accordingly. This increased flexibility comes at the cost of higher computational efforts caused by the necessary individual packet inspections.

Streaming Proxy

Serialisation of the information objects (i.e. packets and envelopes) is necessary in order to allow for a flexible interfacing of external actors that are unable to interpret the binary stream of these objects. Thus, the streaming proxy is a specific translation proxy that translates between binary information objects and serialized character string representations of those elements. As a consequence, while this proxy's middleware facing interfaces handle envelopes and packets, its actor facing interfaces only handle character strings.

Up to now, two formats are supported by the streaming proxy, which are XML and JSON character strings. The related XML schema description is presented in Listing A.2 of Appendix A.1. Due to its popularity, the XML streaming capability ensures the compatibility with a wide range of tools and simulators that also support XML data exchange. Furthermore, JSON proved its usefulness due to its limited overhead (in contrast to the XML format) and consequently high transmission performance.

Socket Connector Proxy

In order to connect external actors like power grid or communication system simulators, or the control system under development or evaluation, a networking interface is necessary. This is realized by the socket connector proxy implementation providing a TCP socket connection. This proxy can be seen as an endpoint-proxy because no further proxy can be stacked atop this one.

The proxy usage is illustrated in Figure 5.1 on page 75. Its configuration parameters are the unique port number for the specific channel's socket connection and a binary transmission flag. The latter is used in the internal implementation to distinguish between either text based information exchange – which is, for example, used in case of JSON or XML stream transmission provided by a streaming proxy stacked underneath – or binary object (packet, envelope) streaming. Depending on these proxy parameters, which are individually set for each socket connector proxy instance in the configuration file, TCP server socket connections are initialized and accepting client connections on the predefined ports.

Sync Proxy

The sync proxy is the first proxy implementation presented in this section that is an end-node-proxy. This means that it is an individual actor that is not operated independently but as a process of the middleware. Also its configuration is contained in the middleware configuration. This is done due to the fact that the operation and configuration of this proxy is directly related to the particular development stage configuration and thus to the related middleware configuration. The following parameters are used to configure the sync proxy.

`clients` This integer value defines the number of actors, which are sync clients, to be coordinated and synchronized by the proxy. This parameter has to be greater or equal to two.

`scenarios` This string value references a CSV file containing a detailed list of scenarios, which should be processed by the sync proxy.

`trigger` This boolean value defines whether the sync proxy should listen to external control signals, e.g. given by a user through a graphical user interface (GUI).

The sync proxy is a complex component which is the implementation of the *Simulation Control* actor of the prototyping concept presented in Section 4. It is necessary in nearly every development stage to fulfil the following task:

Maintaining all actors involved in the control development environment, either in

- co-simulation mode with virtual simulation time in equidistant time-slices or in
- real-time simulation mode with hardware integration.

These tasks involve simulation flow control by means of starting and stopping the simulation process, initializing each simulator with necessary parameters, distributing simulation scenarios among the simulators and finally stepping through simulation time (as described in Section 3.2.1). To allow for this functionality, the sync proxy is realized with a state machine described subsequently with the aid of Figure 5.5.

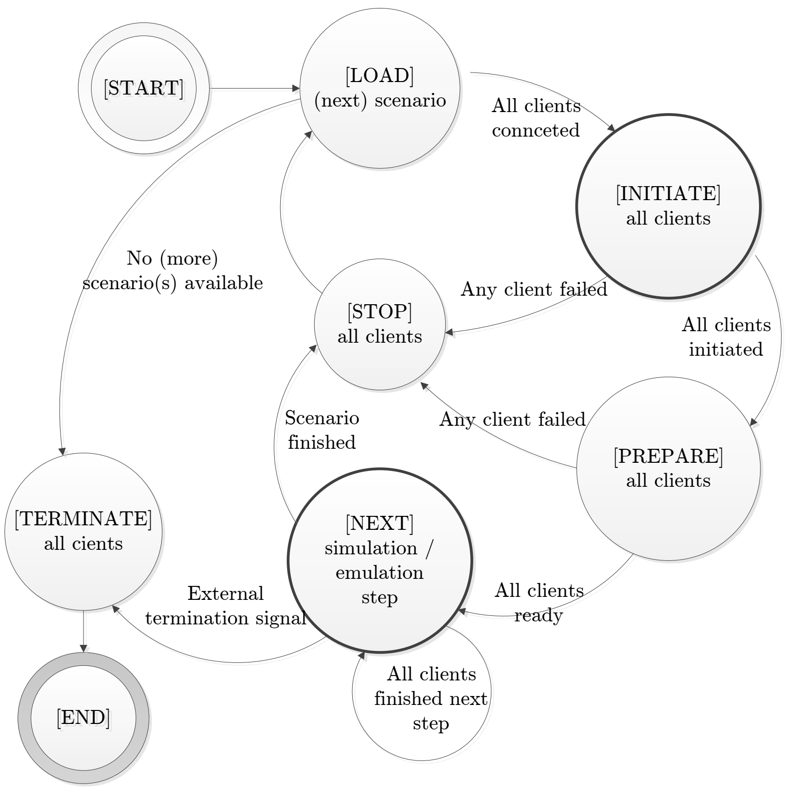


Figure 5.5: Sync proxy state transition diagram for co-simulation and real-time simulation.

This state diagram describes the consecutive execution of several complete simulation runs on the basis of a scenario file (cf. Appendix A.2). Each line in the scenario file represents one co-simulation run simulating a defined time range with specific parameters, such as a reference to the power grid file, generation and consumption profiles (e.g. depending on the weekday or the season), control system configurations, etc.

The commands issued in distinct states are packed into the predefined packet data model (cf. Section 5.2.3.3) in the form `sync.[command|param]=[Null|0|1|String]`. For sync commands, null values are typically used in requests, zero and one values are used as (N)ACK indicators in the corresponding replies. In case that external signals are accepted through the configuration parameter `trigger`, the state machine also reacts on `sync.ext_go`, `sync.ext_stop`, and `sync.ext_terminate` for GUI-controlled starting, stopping or termination of the simulation run.

Subsequently, the single states of the state diagram are listed, together with a detailed description of their functions and transition conditions to the other states.

[**START**] In the START state the sync proxy loads the scenario file and initializes internal data structures. No packets are sent in this state. The next consecutive state is the LOAD state.

[**LOAD**] This state is the first state of a scenario run. The scenario is loaded as a line of the scenario file. The sync command 'hello' is sent to all actors and their replies are awaited. In case that the predefined number of sync actors positively reply to the 'hello' request, the next state is the INITIATE state. Otherwise the STOP state is anticipated.

[**INITIATE**] The INITIATE state allows all clients to initiate their internal states and configuration according to the scenario parameters (e.g. the loading of the power grid file). These parameters are sent within an 'init' sync command to all involved actors. After a successful initiation, each actor sends an acknowledge message to the sync proxy or otherwise a NACK in case of a failed initialisation. In the latter case the sync proxy sets the next state to the STOP state. If all actors have successfully acknowledged their initialisation, the state machine proceeds with the PREPARE state.

[**PREPARE**] This state is foreseen as preparation state for all actors to the start of the co-simulation. This is especially useful for the calculation of initial parameters and state variables that will be exchanged during the first simulation run. Furthermore, configuration parameters for the controllers can be loaded in this state. A 'start' sync command is sent to all actors. If all actors acknowledge this command the next state will be the NEXT state, which is the begin of the co-simulation or the real-time simulation, depending on the simulation mode.

[**NEXT**] The NEXT state is the state where the simulation process takes place. Depending on whether the simulation mode is regular co-simulation or real-time simulation, either many NEXT-iterations are performed until the simulation end time is reached, or only one (long lasting) NEXT-iteration is performed where each simulator independently operates in real-time mode, optionally with a speed up factor, until the end time is reached. While the former operation is strictly synchronized with a signal and message exchange between the simulation time slices, the latter real-time operation is loosely coupled only by simulation flow control. The message and signal exchange thereby can take place at any time in order to allow for a real-time integration of hardware components. After exceeding the simulation time either in co-simulation or real-time simulation, the next state is set to be the STOP state or the TERMINATE state in case of an external termination signal.

[**STOP**] In this state a 'stop' sync command is sent to all involved actors for confirmation. The proxy internal buffers and signals are cleared and thus prepared for the next scenario simulation run. The only consecutive state is the LOAD state that is set after each actor has acknowledged its stoppage.

[TERMINATE] The TERMINATE state represents the final completion of all scenario simulations or an advanced termination caused by an external signal. It also includes sending of a 'termination' sync command to all actors to indicate simulation shut-down. Finally, the END state is set as a final state.

[END] In this state also the sync proxy terminates and thus stops operation.

Web Service Proxy

The web service proxy is similar to the sync proxy a standalone actor, which is implemented as an end-node-proxy. It provides a rich set of functionalities for the complete prototyping process, by embedding a lightweight Jetty [5] web-server with RESTful web service (by embedding Jersey [13]) functions into a proxy. From the control prototyping environment introduced in Section 4.2, the *User Interface* actor and the *SCADA Interface* actor are realized by this proxy.

Its implementation of a lightweight web server allows for the hosting of a web page, which is the user interface. Furthermore, by the RESTful web services, a generic interface to the middleware is established. This can be accessed by the grid operator's SCADA system. It also serves as data source for the dynamic web interface's graphical representations of selected values (e.g. by voltage and power graphs).

All data packets which are sent to the web service proxy are cached internally and provided for external access by a specific cache access web service answering HTTP GET request in the following form (cf. Listing 5.2).

```
> GET http://localhost:8080/rest/getValue/channel_grid/trafo8/activePwr HTTP/1.1

HTTP/1.1 200 OK
Date: Mon, 23 Feb 2015 15:18:20 GMT
Server: Jetty(6.1.22)
Content-Length: 8
Content-Type: text/plain; charset=utf-8
Cache-Control: max-age=0
Expires: Mon, 23 Feb 2015 15:18:20 GMT

60439.75
```

Listing 5.2: HTTP GET request and response to obtain a cached (simulated or real-world) power value of a transformer in plaintext format.

For more complex requests (e.g. time series requests), also XML and JSON formats are implemented in the web service. Furthermore, the implementation supports data encryption via transport layer security (TLS) and HTTP basic authentication.

Mapping Proxy

The mapping proxy is an intermediate proxy that can be used to map packet syntax on the basis of a mapping file defining a translation table for `[id].[var]` combinations to other combinations. This proxy has proven its usefulness in one specific situation, which is the field deployment of the developed control system. In case that the naming convention of the simulation does not correspond to the one in the field, by this mapping proxy this inconsistency can be fixed easily.

6 Implementation and Application of the Integrated Prototyping Process

The subsequent sections of this chapter describe the implementation of the integrated control development process. This exemplary implementation bases on the abstract concept, which has been defined in Chapter 4. The coupling middleware Simulation Message Bus (SMB) proposed in Chapter 5 is used to dynamically interlink the single actors of the process in each phase of the control prototyping process. This also implies the connection of the control hardware for controller-hardware-in-the-loop (CHIL) evaluation, as well as the seamless deployment of the control algorithm to the field. The description of the process implementation is enhanced by showing excerpts from its successful application for control development, done in the course of the research project *DG DemoNet – Smart LV Grid* (cf. Section 2.2.3).

6.1 Process and Application Context

The process for rapid control prototyping is implemented as proposed in Chapter 4 in order to fulfil the stakeholder requirements that have been extracted in the requirements engineering process in Section 2.3.1 according to the IntelliGrid approach. Its agile structure, that is composed of three phases with subordinated prototyping stages, is illustrated once again in Figure 6.1. This figure represents the concrete implementation of the process rather than its abstract concept. It emphasises the flexibility of the process where in each stage a short-cut to previous stages is provided in order to be able to react quickly to changing requirements.

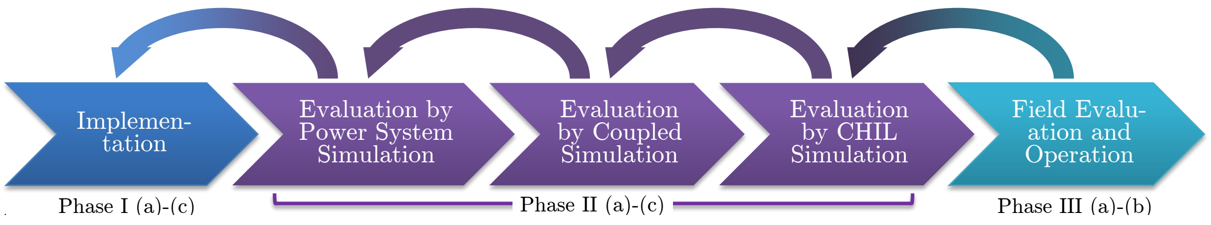

Implemen-tation
Evaluation by Power System Simulation
Evaluation by Coupled Simulation
Evaluation by CHIL Simulation
Field Evalu-ation and Operation
Phase I (a)-(c)
Phase II (a)-(c)
Phase III (a)-(b)

Figure 6.1: Process for rapid control prototyping with a focus on risk reduction along the process.

The joint national research project *DG DemoNet – Smart LV Grid*[1] provides the context for the presented process implementation. The process has been developed on the basis of this project

[1]funded within the programme Neue Energien 2020 by the Austrian Klima- und Energiefonds under project number 829867

and applied by domain experts and engineers related to the project. There were four Austrian distribution grid operators involved, which were Salzburg AG, Netz OÖ GmbH, Linz AG, and Netz Burgenland Strom GmbH. Technology partners were SIEMENS AG Austria (providing the communication system, the transformers with OLTCs, controller hardware, etc.) and Fronius International GmbH (providing PV inverters). Beside these industrial partners, research was conducted by Vienna University of Technology (Institute of Computer Technology and Energy Economics Group) as well as AIT – Austrian Institute of Technology.

One of the goals of the project was the development of a control system for selected LV distribution grids in Austria in order to maximize DG hosting capacity. The target grid that has been selected in Upper Austria is operated by Netz OÖ GmbH and located in the municipal of Eberstalzell, a typical rural region with approx 2,300 inhabitants and 170 grid nodes.

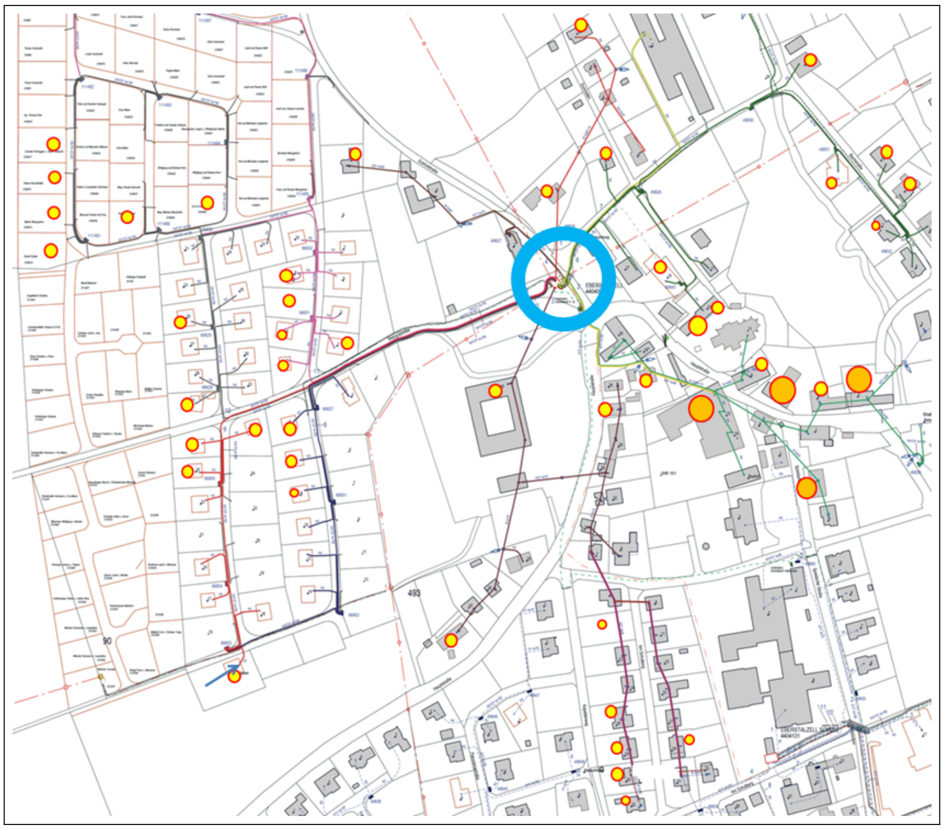


Figure 6.2: LV distribution grid for control system operation in Eberstalzell, Upper Austria (with friendly permission of A. Abart/Netz OÖ GmbH). The 630 kVA transformer (marked with a circle) with its three-step OLTC feeds around 170 nodes via four cable and three landline branches. The small circles on the depicted nodes represent the PV installations with their rated maximum power correlated to the circles' diameter (kindly provided by Netz OÖ GmbH).

The distribution grid's structure is illustrated in Figure 6.2. This figure shows a relatively large distribution grid with a 630 kVA transformer with a three-step ($\widehat{=}$ 4.8 %) OLTC in the secondary substation, which is marked by a circle. This transformer supplies the local grid via four cable and three landline branches. In this LV grid, around 60 rooftop mounted, grid-connected PV installations are operated with an accumulated peaking power of around 300 kWp. Furthermore, approximately every second household operates remotely controllable EVSE. These controllable components, together with the PV inverters and the OLTC, are the actuators that intended to be remotely controlled by the control system.

The rest of this chapter describes the process implementation in the course of *DG DemoNet – Smart LV Grid* followed by an excerpt about its application within the project. The implementation description of each process stage contains the set-up of the SMB-based control prototyping environment. To keep an overview, extensive chunks of information, like the related SMB configuration files, are moved to the appendices and referenced accordingly.

6.2 Prototyping Process Phase I: Implementation

Process phase I consists of three prototyping stages, which are (a) the creation of the control system specification, (b) the design of the control algorithm(s), and (c) the concrete implementation. These are the first three stages of the complete prototyping process, whose implementation and application is described hereafter.

Implementation of Phase I, Stages (a-c)

For the creation of the control system specification, expert workshops and discussions are performed. The overarching goal of maximizing DG hosting capacity in the specific power grid is communicated amongst all experts and partitioned into sub-problems and tasks. The basis for the creation of the control approach and its specification are standard methods for voltage control derived from literature and the experience of experts gained in related research projects and other application domains. Prior to the concrete specifications, the prevailing boundary conditions are presented in meetings by domain experts, i.e. the grid operators. For LV grid control prototyping the boundary conditions primarily concern the assets that are available in the field, their capabilities (concerning communication and remote control), and the available temporal and financial resources. Within the workshops, the experts create scenarios that cover different states and cases of the power grid and its elements and thereby derive the intended behaviour of the control system.

On the basis of the specification, the conceptual design for the control system is created by AIT's and SIEMENS' control system engineers on the basis of established software and system engineering methods. Proven object-oriented software design methods for a modular implementation or the control system are used. To match the principal idea of agile development to very quickly have the first controllers in operation, a multi-stage control structure with fall-back options is planned. In the design phase, also the given ICT infrastructure and assets are considered. On the basis of the specified functions and given boundary conditions, concrete hardware and software components are selected for the implementation.

For the concrete implementation stage, Microsoft Visual Studio is chosen as integrated development environment (IDE). The control algorithms are implemented in the VC++ programming language. Thus, in this early prototyping stage, the control prototyping environment consists of three components depicted in Figure 6.3. The major component is the control algorithm under development, which is already designed to be connectible to the SMB, which is the second component. The connection of the SMB and the control algorithm is established by a routing proxy (R), a streaming proxy (S), and a socket connector proxy (SC) (cf. Section 5.2.4.2). For all socket connector proxies, non-well known ports (according to RFC 6335 [WCT$^+$11]) are chosen in order not to conflict with other software operated on the developers PC.

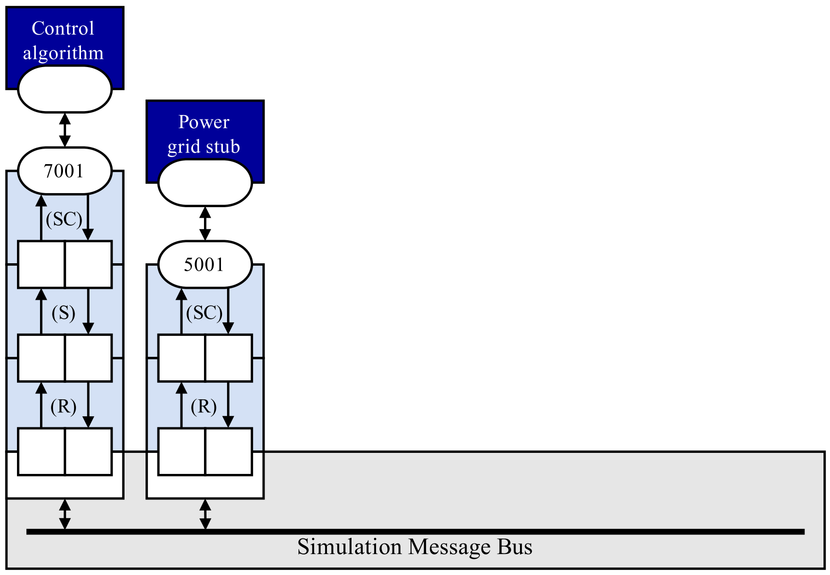


Figure 6.3: SMB configuration for Phase I, Stage (c): implementation of the control algorithm. The implementation of the first algorithm drafts are created by the use of a power grid stub, which allows for basic interaction tests.

The first socket connector proxy instance exposes the port 7001 for the control algorithm. The streaming proxy that is stacked underneath is necessary in order to serialize the Java packet objects (cf. Section 5.2.3.3) as JSON strings to the VC++ implementation of the control algorithm. The routing proxy has only one routing target, which is the channel to which the power grid stub is connected. This stub element imitates the power grid, or its sensor and actuator components, respectively. It is used as the counterpart to the control algorithm in very early interaction and behaviour tests of the algorithm. The power grid stub provides fake measurement readings and receives OLTC step-commands from the controller, routed via the SMB. As a Java component the stub is capable of handling the binary streams of packet objects and thus does not require an intermittent streaming proxy. The provision of a socket connection on port 5001 by the SMB's second socket connector proxy instance is sufficient. The concrete architectural configuration of SMB set-up depicted in Figure 6.3 can be obtained from Listing A.4 in Appendix A.3.

Application of the Prototyping Stages

The application of the first three prototyping stages in the joint research project *DG DemoNet – Smart LV Grid* results in the implementation of several control algorithms and a control supervisor unit. In several expert workshops, representatives of the involved distribution grid operators, experts from SIEMENS AG Austria, AIT – Austrian Institute of Technology and other stakeholder representatives specified a hierarchical control structure for the project's LV grids. The created control structure is depicted in Figure 6.4 and consists of five stages.

The basic concept of the control system structure is that several control algorithms are operated in parallel in the field managed by a supervisor. All algorithms are calculating set points and control commands according to the received measurements and their control strategy. They send their internal status and the control commands to the supervisor unit, which is responsible of selecting one control stage and forwarding its control commands. The selection is performed by the priority of the respective stage and its status, i.e. its readiness level.

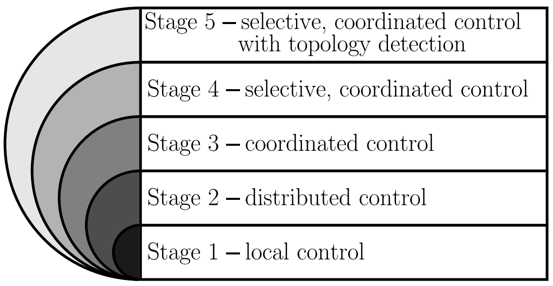

Stage 5 – selective, coordinated control with topology detection
Stage 4 – selective, coordinated control
Stage 3 – coordinated control
Stage 2 – distributed control
Stage 1 – local control

Figure 6.4: Five control stages with increasing complexity that have been specified through the prototyping process' Phase I in the course of the related research project *Smart LV Grid* [EZS+13].

While Stage 1 of the control system (in Figure 6.4) is designed to only control the transformer's OLTC on the basis of voltage measurements from the transformer's bus bars – and thus does not require remote communication facilities –, other more complex stages use remote meter readings. Stage 2 reads predefined critical smart meters to get the voltage levels of the complete distribution grid supplied by the respective substation, and sets the transformer's OLTC according to this information. Stage 3 operates similarly as Stage 2, but also sends set points for the PV inverters' P(U) and Q(U) characteristics as broadcast messages. The even more complex Stage 4 and Stage 5 are specified to use single-cast messages for the set points and incorporate topological information. These stages have not been implemented so far. More details concerning the developed control strategies are available in [EKBL12, EZS+13].

6.3 Prototyping Process Phase II: Evaluation

The evaluation phase is the most relevant phase of the rapid control prototyping process. It is designed to overtake the lion share of risk mitigation within the prototyping process. This is done by different types of simulation in three evaluation stages focusing on (a) the power grid, (b) the communication system, and (c) the control system's hardware component. The implementation of these three stages with its concrete soft- and hardware components is described in detail in the subsequent sections.

6.3.1 Evaluation by Power Grid Simulation

For the evaluation of the implemented control algorithms in power system simulation, DIgSILENT PowerFactory is used. PowerFactory is an established power system simulator that provides a wide range of simulation modes ranging from electro-magnetic transient, over root means square (RMS) simulation to steady state power flow simulation. The latter is chosen for the evaluation of the control algorithms, as they are intended to solve voltage limit issues which typically have a very long time scale. Thus, the evaluation of related measures does not require consideration of fast transients or harmonic effects.

Implementation of Phase II, Stage (a)

The power system simulation related evaluation of Stage II (a) requires a connection of the implemented control algorithms with the power system simulator. PowerFactory provides some useful interfaces for this interaction, like an OPC client and proprietary DLL interfaces with an API [SSAS13]. Within the work, conducted in related research project, these interfaces have been used in order to fulfil three tasks that are necessary for the fully automated interaction with PowerFactory.

- Remote update of physical variables via the TCP/IP socket interface
- Remote reading of physical variables via the TCP/IP socket interface
- Remote control of the steady-state power flow simulation either in off-line or on-line mode (real-time simulation or emulation; cf. Section 3.2.1)

The integration of PowerFactory via its API forced the use of the C++ programming language because the API is only available via the dynamic link library (DLL) `digAPI.dll`. The developed power system simulator depicted in Figure 6.5 acts as a wrapper for PowerFactory and adds additional features to ensure compatibility with the SMB's data format and the used simulation process flow and synchronisation mechanisms (cf. description of the sync proxy in Section 5.2.4.2).

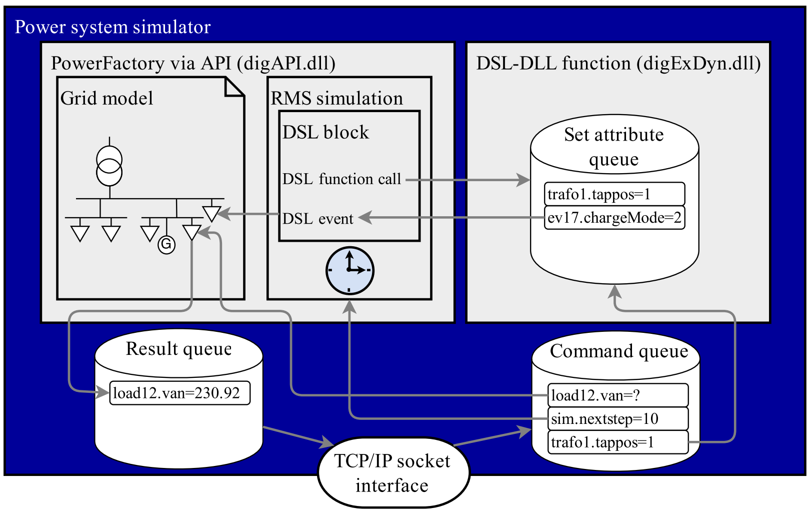


Figure 6.5: Power system simulation: integration of DIgSILENT PowerFactory for steady-state power flow calculation [SSAS13].

After its start, the power system simulator directly instantiates PowerFactory as power system simulator. The API is then used for loading of the power grid model and control of the RMS simulation core. Load profiles and generation profiles (in the form of negative loads) are loaded from predefined CSV files, which are assigned respectively in the grid model.

For remote data and command processing and in order to fulfil the three previously mentioned tasks, two queues are incorporated in the power system simulator implementation. The command queue, enqueues all incoming packets, which can be either requests of physical variables, updates for physical variables, or simulation process flow and synchronization commands.

Variable requests can be handled directly by the PowerFactory API (`digAPI.dll`), even during RMS simulation runtime. The physical variable is read from the grid model's state variables and packed into a reply packet that is forwarded to the result queue. These packets are sent via the TCP/IP socket connection back to the SMB. To avoid unpredictable behaviour, reading of physical variables is only performed between simulation steps where all physical states are in an equilibrium.

In contrast to the reading commands, the updates of physical variables cannot be performed directly via PowerFactory's `digAPI.dll` during simulation. Therefore, a predefined block in DSL (DIgSILENT Simulation Language) has to be used. In this DSL block, events are generated that influence the simulated objects' states in the running simulation. As the DSL block is not accessible via the already established connection, another DSL block-related connection is created by the use of the DSL-DLL `digExDyn.dll`. An update function is invoked through the DSL block, which gets attributes from the attribute queue that are then updated in the running grid simulation through the DSL event [FDL+13].

The third task, beside requesting and updating physical model variables, is the remote simulation process flow control and synchronization. Basic simulation flow control is supported by PowerFactory's `digAPI.dll` API. Compatible functions for initiation, starting, and stopping of the simulation in accordance to the sync proxy's state machine (cf. Section 5.2.4.2) are realized by these API calls.

Figure 6.6 shows the configuration of the control prototyping environment for the evaluation of the control algorithms by power system simulation. The corresponding detailed configuration is presented in Listing A.5 of Appendix A.3. Equal to the previous set-up for the implementation phase, the SMB forms the common basis for data exchange among all actors. This involves the exchange of all three data types that have been identified in the discussion of the concept in Section 4.6.2: messages, physical variables, and simulation flow control data.

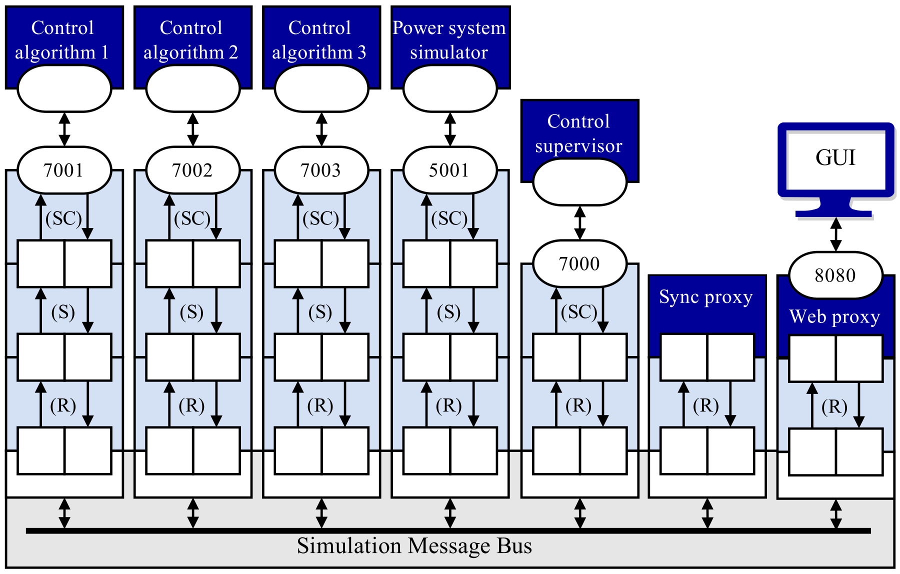


Figure 6.6: SMB configuration for Phase II, Stage (a): evaluation by power system simulation. The three implemented control algorithms are connected to the power system simulator DIgSILENT PowerFactory via the SMB. Synchronization of the components is taken care of by the sync proxy. An HTML GUI provides control, manipulation, and observation features.

Three out of five designed control algorithms have been implemented in the previous prototyping stage. They are connected similar as in the previous prototyping stage via the 7000er TCP/IP port range to the SMB in the evaluation set-up (illustrated in Figure 6.6). They interact with the control supervisor, which constantly monitors the control stages' state and forwards the control commands of the control stage with the highest priority and an active state to the controlled system, which is the power system simulator.

The power system simulator, which is connected to the TCP/IP port 5001, instantiates and uses DIgSILENT PowerFactory for power flow simulation as described above. This simulator replaces the power grid stub of the previous prototyping stage. By an appropriate model it represents the real power grid in which the control system is intended to operate. The wrapper that governs the PowerFactory instance by its two APIs receives and answers (voltage) requests and handles set points (e.g. for the OLTC, the PV inverters, or other actively controllable grid components). Due to the fact that the wrapper is also written in the C++ programming language in order to be able to use PowerFactory's dynamic link libraries (DLL), its connection to the SMB is supported by a streaming proxy (S). Like with the control algorithms, this proxy serializes the binary packet objects to JSON-strings, which are then transferred via the socket connector proxy to the power system simulator.

This prototyping stage is the first one that uses the sync proxy in order to coordinate the simulation flow among the distributed actors, which in this case are the power system simulation and the control algorithms. The sync proxy acts as the synchronisation master. It guides all actors in a stepped manner through the quasi-static power flow simulation with constant time slices, by following the state machine presented in the proxy's description in Section 5.2.4.2. While other, more dynamic synchronization mechanisms might provide better performance (e.g. by the use of negotiation techniques to find the next synchronisation point instead of a constant synchronisation interval), this one has been chosen, because the physical system simulator anyway updates its states in a cyclic manner and the approach is simple and sufficient for the problem at hand. Nevertheless, a variation of the synchronisation interval, even during simulation, is principally supported by the sync proxy's implementation.

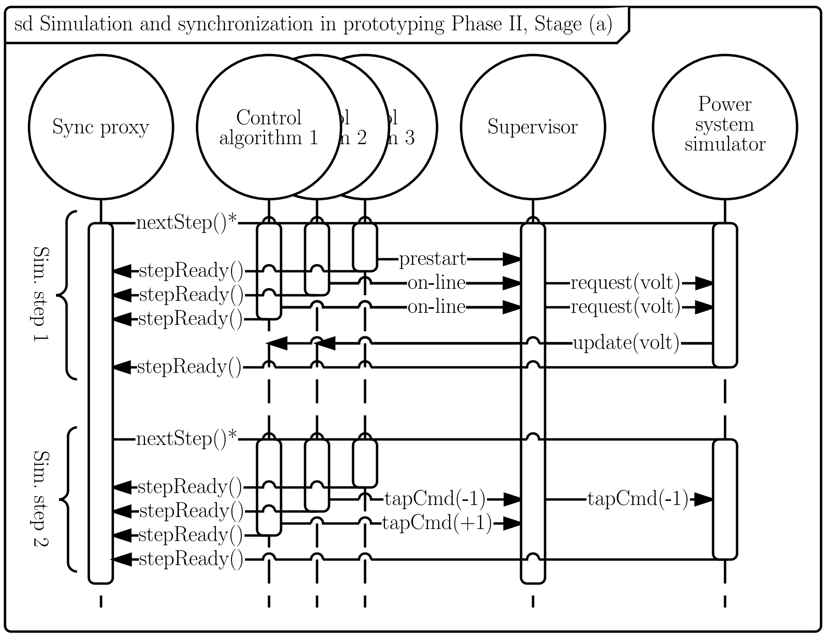


Figure 6.7: Sequence diagram for the simulation and synchronisation process in Phase II, Stage (a) of the prototyping process.

The sequence diagram in Figure 6.7 illustrates the simulation and synchronisation process for this prototyping stage on an intuitive example. The whole process is coordinated by the sync proxy. After the not depicted initiation phase, the off-line simulation is started with a broadcast (`nextStep()*`; the asterisk indicates the broadcast) to all involved actors. This broadcast informs the actors to progress the simulation time by a predefined time step (e.g. one minute), to perform any action necessary within this virtually elapsed time, and to finally update their internal states.

In this example, the control algorithms send their states to the supervisor, which is not a simulation actor in the inner sense because it does not have any temporal behaviour. The supervisor only acts on the basis of causal subjection similar to a multiplexer or gateway. Besides the status updates, the algorithms also request (voltage) measurements form the (simulated) power system. The power system simulator directly answers these requests with the cached values of the previous power flow calculation. Both the power system simulator and the control algorithms report their finalization of the current time slice calculations by an adequate command (`stepReady()` in Figure 6.7). In the next simulation step – which is again initiated by the sync proxy's broadcast – the control algorithms calculate (i.e. integrate) the voltage deviations and trigger appropriate control commands. These commands are sent to the control supervisor. In this case, only the command of the active control algorithm with the highest priority is forwarded to the respective actor in power system simulation. In the presented example this is Control algorithm 2. In order to be capable of this coordinated simulation operation mode and to be conform with the simulation time progression by time slices, the control algorithms under design need to have a time-wrapper that allows them to abstract real-time to the simulated time.

This kind of off-line simulation runs as fast as possible. Directly after each of the simulation-involved actors confirmed its completion of the current time slice, the next one is triggered. This rapidness makes it very difficult for the developer to check the control algorithms behaviour in detail. Therefore the GUI is connected to the control prototyping environment as the rightmost element in the schematic illustration in Figure 6.6.

Furthermore, the GUI provides features for the visualisation of the control algorithms' states and details of every element of the power system simulation. Additionally, it provides control of the simulation flow. For detailed stepwise analysis, the developer can use the GUI to take part as a distinct actor in the coordinated simulation flow and step through it or even manipulate or inject messages or physical values. All these observation and developer convenience functions, provided by the GUI, are presented with illustrations in greater detail in the following Section 6.5.

Application of the Prototyping Stage

The control prototyping environment has been applied in *DG DemoNet – Smart LV Grid* according to the configuration depicted in Figure 6.6 and described above. The three control algorithms that have been designed and implemented in the implementation phase of the process have been tested with this set-up. The operation of first algorithm drafts with power system simulation quickly lead to new insights. Several agile iterations between the design, implementation, and this evaluation phase were performed in order to create proper working control algorithms.

Figure 6.8 depicts the plot of a typical control algorithm action in this evaluation stage. This plot results from the coordinated simulation of one of the rural LV grids that have been selected for the research project. It has been created with DIgSILENT PowerFactory by the previously described simulation approach. The detailed power system simulation is depicted in Figure A.1 of Appendix A.4.

The upper chart of Figure 6.8 shows the bus bar voltages and the three phase voltages of selected critical nodes of the LV grid printed over time. Furthermore, the voltage limits of the operating control algorithms are shown. Due to high solar irradiation voltages are at a high level during

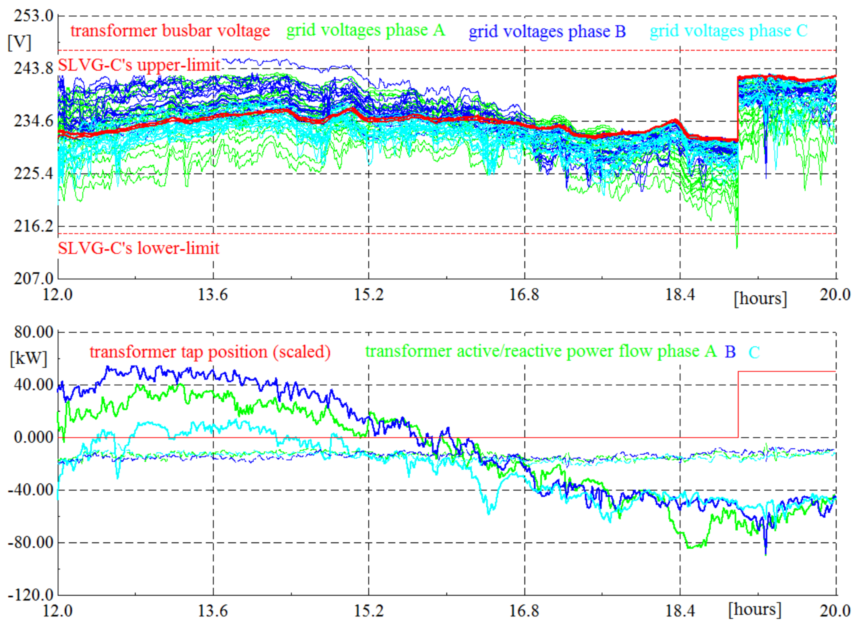


Figure 6.8: Application of the control algorithm and power system simulation in the *DG DemoNet – Smart LV Grid*'s distribution grid of Eberstalzell. The upper diagram shows the transformer's busbar voltages and the three phase voltages of selected critical grid nodes together with predefined voltage limits for the controller. The lower diagram shows the OLTC's tap position together with the transformer's active and reactive power transmission over the three phases.

noon and early afternoon. Towards the evening hours, the voltage levels drop, which leads to a violation of the lower voltage limit. The control algorithms counteract the violation by issuing a tap up command to the OLTC, which is performed around 7:00 p.m. in this simulation.

The lower chart of Figure 6.8 shows the active and reactive power that flow over the transformer. Also the tap position of the transformer's OLTC is shown in this chart. As one can see in the chart, the investigated LV grid feeds active power back to the overlay MV grid during the noon hours due to the high share of installed PV generation units. During the evening hours – when the previously described control action is issued – the LV grid drains active power from the overlay power grid.

6.3.2 Evaluation by Coupled Simulation

The evaluation by coupled simulation stage of the prototyping process considers both domains that are relevant for networked smart grid systems: the electric distribution grid and the conjunct ICT network. These two domains are simulated in dedicated simulation tools that are coupled and synchronized to form a co-simulation in order to allow for comprehensive control algorithm evaluation.

The following section describes the introduction and implementation of a product specific ICT network simulation into the previously presented control prototyping environment. Subsequently, the application of the so created co-simulation, for the evaluation of the developed control algorithms in the *DG DemoNet – Smart LV Grid* project, is presented.

Implementation of Phase II, Stage (b)

The implementation of this prototyping stage demands for the integration of an ICT network simulator that is capable of realistically modelling and simulating the communication system, which is used in the concrete distribution grid. This means that by the use of the communication simulator, the influence of the communication channel on message exchange should be reproduced similarly to the real world. This can be done after identification and analysis of the ICT network and a selection and configuration or proprietary implementation of an appropriate communication simulator.

In the *DG DemoNet – Smart LV Grid* project, a product-specific power line communication (PLC) system was used (SIEMENS AMIS CX-1 profile [SIE11]), which was not supported by any of the established communication system simulators. Thus, a proprietary communication simulator based on a statistical channel model was implemented. Its implementation is schematically depicted in Figure 6.9 and described subsequently.

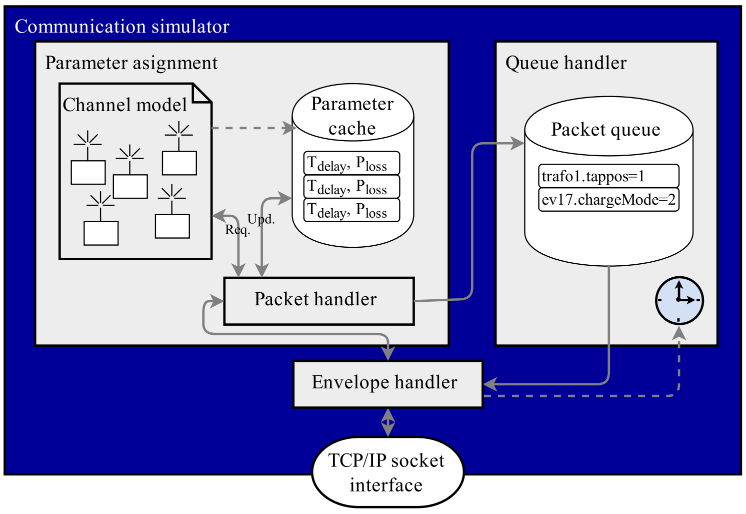


Figure 6.9: Internal structure of the communication simulator implementation. The PLC channel is modelled in a statistical way on the basis of field measurements. Resulting communication parameters are assigned to the packets, which pass the queue handler before further propagation to other actors of the co-simulation.

The implementation of the communication simulator consists of two main modules, which are a parameter assignment entity and a queue handler. Besides these two, another module is used for the handling of envelopes. This last module's purpose is the unpacking of received envelopes and the packing of packets and the assignment of the routing header to the created envelope. This is necessary due to the intermittent nature of the communication simulator and the related fact that messages might have different routing targets depending on their packet type.

After an incoming envelope is unpacked, it is sent to the communication parameter assignment module. This module distinguishes between request packets and update packets. In case that a request packet arrives, two sets of communication parameters are obtained from the statistical channel model. Each set contains a delay time and a loss probability value for the addressed node in the power system (simulation). While one of the parameter sets is directly assigned to the request packet, the other is temporarily stored in the parameter cache. This is necessary because the statistical channel model bases on field measurements which describe the transmission delay and loss probability for each two-way request-response handshake. Later, when the related update packet passes the communication simulator, the packet handler obtains the communication parameters from the parameter cache and thus assigns the correct values for the request-response packet pair.

When an incoming packet – independent of whether it is of request or update type – has received the set of communication parameters, it is directly forwarded to the queue handler, which is the second major element of the communication simulator. There, all packets are stored in an ordered output queue depending on their previously assigned delay time. Also packets which are marked as lost are processed equally because in reality they still influence (delay) other packet's transmission (by a time-out limited waiting of the receiver), which is thereby considered also in simulation.

The packet queue of the queue handler has two purposes. It ensures a proper order of all messages that are transmitted in the simulated ICT network and thus is comparable to event queues of other communication system simulators. The second purpose is the delay of packets according to their assigned delay time. This has to be done in two different ways according to the mode of simulation. If regular (off-line) simulation is performed, where each simulator calculates its internal states for the next time slice as fast as possible, the packets are worked off the packet queue just as their resulting sending time fits into the current time slice of simulation time. In case of real-time simulation (e.g. for CHIL evaluation), the packets are really delayed by the queue handler until the delay time is exceeded.

The previously presented implementation of the communication simulator is integrated in the control prototyping environment as illustrated in Figure 6.10. The SMB allows for a seamless iteration of the communication simulator as a new synchronized co-simulation component by the creation of a new channel that provides a connection on port 6001. Furthermore, the routing configuration is adapted in order to re-route messages between the control algorithms and the power system simulator via the communication simulator. The resulting XML-configuration for this set-up is enclosed in Listing A.6 of Appendix A.4.

Interestingly, the communication simulator is directly connected to the SMB via a socket connector proxy (S) but without a routing proxy (R). This is because the communication simulator takes care of the dynamic routing of the packets by its own envelope handler (cf. Figure 6.9). All other actors of the environment are operated as in the previous prototyping stage. The co-simulation process with all involved actors is depicted in the sequence diagram in Figure 6.11.

The communication simulator is a new intermittent actor that also takes part in the synchronization handshake and relays messages between the power system simulator and the control algorithms or supervisor, respectively. The illustration shows three synchronization intervals, each synchronizing a specific amount of simulation time. After communicating their internal states to the supervisor, each ready control algorithm requests measurements from the power grid. These measurements do not directly reach the power system simulator, but end in the communication simulator where they are handled as described earlier in this section (cf. Figure 6.9).

In the presented example in Figure 6.11, the update packet is assigned a shorter delay than the simulation cycle duration. Thus, it is directly forwarded by the communication simulator to the power system simulator. On the contrary, the related update packet is delayed longer and thus temporarily remains in the communication simulator's queue. The same delay behaviour applies to the tap-change command, which is issued by Control algorithm 2. This one is delayed equally to the update of *voltA* and thus forwarded in the following simulation interval. Depending on the

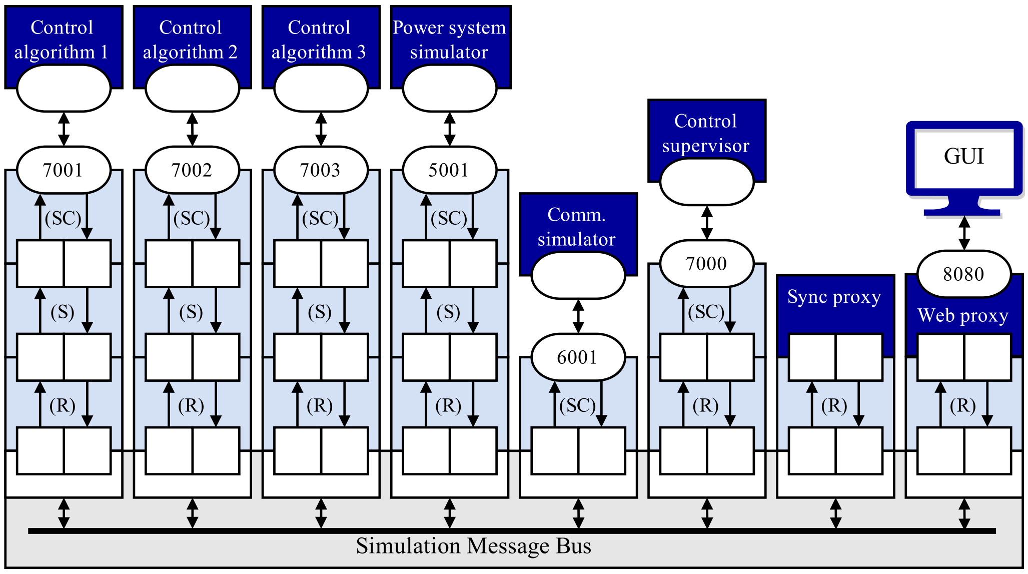


Figure 6.10: SMB configuration for Phase II, Stage (b): evaluation by coupled simulation. The three implemented control algorithms are connected to a power and communication system co-simulation. This allows for the consideration of the influence caused by the communication system in the field.

duration of the simulation intervals and the delay of the packets derived from the communication channel model, the forwarding of the packet can be delayed even over several simulation cycles in both update and request directions.

Application of the Prototyping Stage

Subsequently, the application of the power and communication system co-simulation is described. The presented example is one of the analysis that were conducted in the course of the project *DG DemoNet – Smart LV Grid*, where the impact of communication delay on the OLTC control is evaluated. The full analysis is presented in [FSK15].

The goal of the conducted analysis is the evaluation and assessment of OLTC control behaviour (which is contained in each of the developed control algorithms) on the basis of varying the communication channel's quality. Therefore, the channel model of the communication simulator is extended to allow for parameter variation. The total set of field communication measurements – which are the basis for the statistical channel model – is divided into four quartiles as depicted in Figure 6.12.

Out of these four quartiles three test cases are generated by the selection of either the two highest performing quartiles (*Optimistic* test case), the two middle quartiles (*Neutral* test case), or the two lowest performing quartiles (*Pessimistic* test case). To have some more extreme test cases, two artificial test cases were generated and added to the set of test cases. All five resulting cases are listed in Table 6.1 together with the resulting test set's average packet delay time and average loss probability. The *Reference* case is defined without any deteriorative properties, like delay and packet loss.

[2]A test set consists of 10,000 samples to analyse the resulting average test case parameters

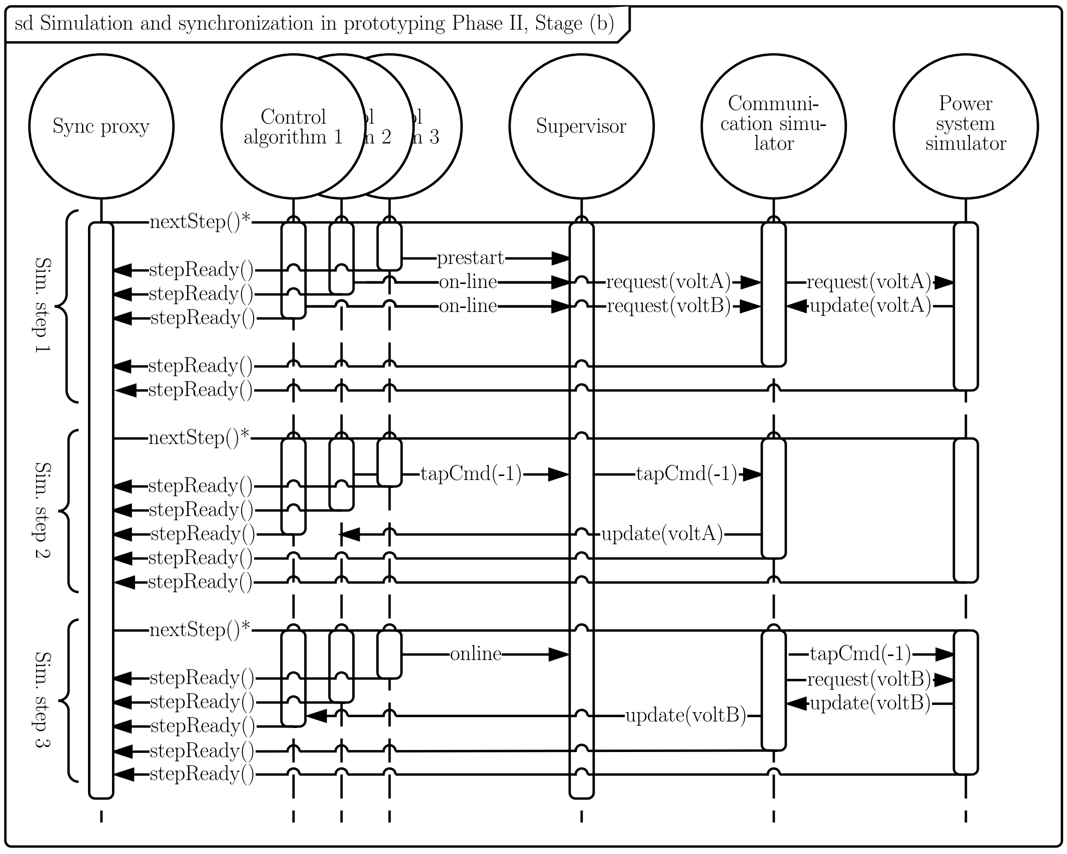


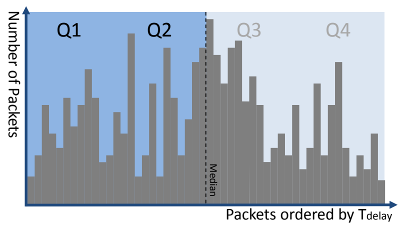
Figure 6.11: Sequence diagram for the simulation and synchronisation process in Phase II, Stage (b) of the prototyping process.

The co-simulation for the evaluation of the controller behaviour uses the LV distribution grid of Eberstalzell and recorded communication logs of the respective AMIS PLC system [SIE11]. The simulation has been conducted with one-second simulation cycles and power system RMS simulation resolution in order to be able to obtain accurate simulation results. For this analysis, PV generation profiles of an average sunny summer day with mixed clear sky and cloudy conditions were chosen. Load profiles were chosen for an average working day in summer.

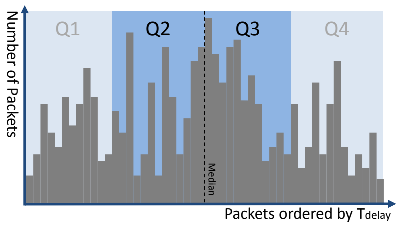
The following Figure 6.13 shows two simulation result charts. Its Subfigure 6.13a represents the *Reference* case while Subfigure 6.13b illustrates voltage values and the OLTC's tap position for the test case *Artificial 2*. In each graph, the three topmost voltage lines represent the maximum and minimum voltage values of all grid nodes, as well as the transformer's bus bar voltage. The dashed lines are the voltage limits for the controller, which are set to $U_n \pm 14V$. The stepped curves at the bottoms of the charts represent the current tap position, which is directly affected by the control algorithms. For better illustration it is depicted with an offset of 210.

The chart for the *Reference* test case in Subfigure 6.13a shows the ideal behaviour of the control system. It immediately reacts to any recognized voltage band violation by issuing tap change commands to the OLTC. Its fast reactivity allows to keep all voltages within the predefined limits throughout the whole simulated day. Even quick voltage deviations, like the one at around 11:50 a.m., can be compensated in an adequate way.

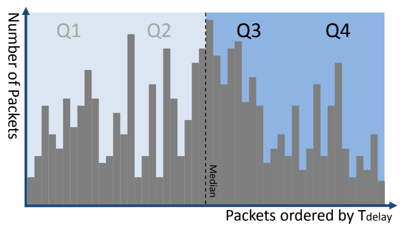
The other, most extreme test case is shown in Subfigure 6.13b and depicts power system be-



(a) Quartile Q1 and Q2 selected for test case *Optimistic*



(b) Quartile Q2 and Q3 selected for test case *Neutral*



(c) Quartile Q3 and Q4 selected for test case *Pessimistic*

Figure 6.12: Schematic figures of the quartile selection for the communication simulation test cases on the basis of field-measured communication data.

Table 6.1: Five test cases with varying communication channel parameters and reference scenario without deteriorative properties.

Test case	**selected quartiles**	**test set's**[2] **AVG(t_{delay})**	**test set's**[2] **AVG(P_{loss})**
Reference	n/a	0 ms	0 %
Optimistic	Q1–Q2	360 ms	15 %
Neutral	Q2–Q3	435 ms	33 %
Pessimistic	Q3–Q4	500 ms	50 %
Artificial 1	n/a (gaussian)	2,500 ms	50 %
Artificial 2	n/a (gaussian)	5,000 ms	50 %

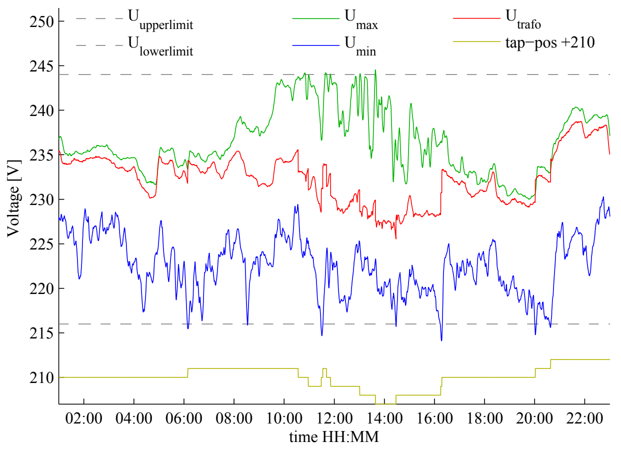
haviour with an average packet delay of five seconds and 50 % packet loss probability. Such low communication channel quality results in several voltage limit deviations throughout the whole day. An enlarged view of the voltage deviation is depicted in Figure 6.14.

This figure shows a zoomed comparison of the *Reference* and *Artificial 2* cases around 8:00 p.m. of the simulated day. It is interesting that the communication delay of some seconds result in a delayed control action of up to 15 minutes. This is caused by the cyclic request behaviour of the used ICT system in combination with the channel's high loss probability.

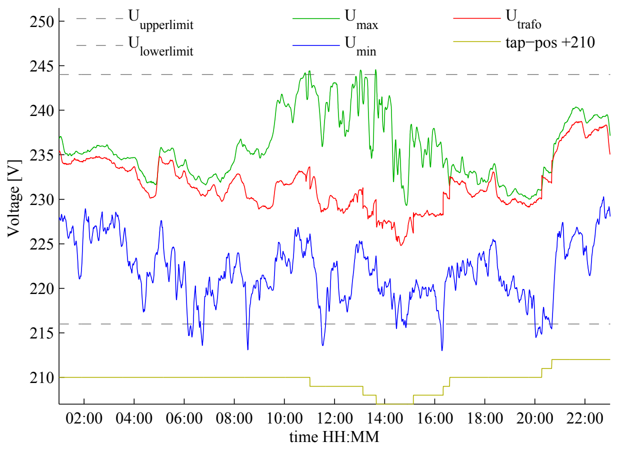
In order to be able to compare the five test cases quantitatively, a metric is introduced taking both the duration and the amplitude of the voltage violation into consideration. Thus, the limit violation of the comparison presented in Table 6.2 is measured as a scalar product in the form of $\Delta U^2 \cdot \Delta t$. These values are averaged, in order to be comparable to EN 50160 limitations, on one minute, five minutes, and ten minutes intervals. Also other limits given by EN 50160 are analysed and presented in Table 6.2.

This co-simulation results clearly show the impact of varying communication channel quality on the control algorithms that were implemented in the course of the *DG DemoNet – Smart LV Grid* research project. It can be seen that both test cases *Optimistic* and *Neutral* lead to almost no voltage deviation in the ten-minute averaging interval. The voltage band does not exceed the 28 V limit, which is a critical value for the control system assessment.

Another interesting learning from the conducted analysis is the reduction of tap changes caused by bad communication channel parameters. While reduction of tap changes is desired by distribution



(a) *Reference* test case without communication delay.



(b) *Artificial 2* test case with communication delay and packet loss.

Figure 6.13: Simulation results for co-simulation without and with consideration of communication delay.

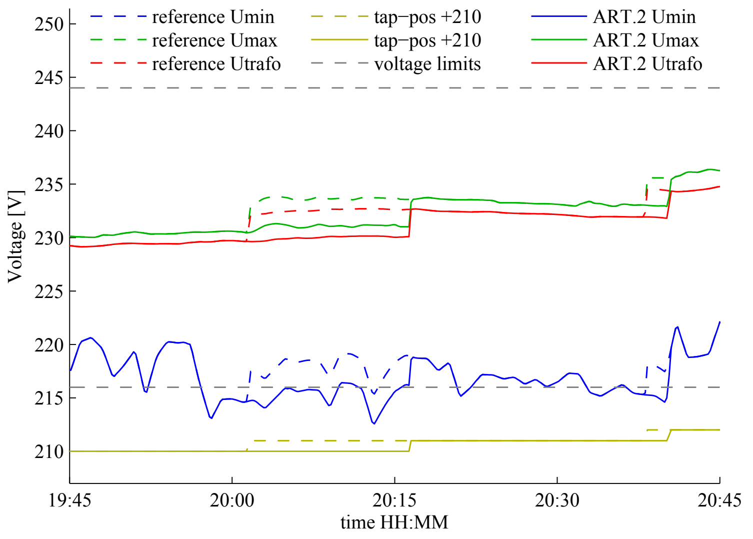


Figure 6.14: Comparison of the voltage limit violation caused by the communication delay on the non-ideal PLC communication channel (*Artificial 2*) with the ideal behaviour (*Reference*).

grid operators in order to reduce material wear and related expense, this shall not be fostered by a bad communication channel and on the costs of power quality. Rather a limitation of tap changes is done by the definition of an integration interval in the control algorithms, which is set to five minutes in the analysed controllers. This averaging interval is also the reason for the relatively high level of voltage limit violations in the one- and five-minutes averaging interval of the deviation analysis for the *Reference* test case (cf. 2nd and 3rd column of Table 6.2).

The three test cases *Optimistic*, *Neutral*, and *Pessimistic* represent typical PLC communication channel behaviour in the analysed rural distribution grids. They show that the high delay and loss rate adversely affect power quality by the constantly increasing numbers for the averaged limit violation factor. The delayed or even lost notifications of voltage band violation in the grid lead to an unwanted increase or decrease of voltage levels and thus lead to a spreading of the voltage band (cf. 6th column of Table 6.2). Thus, the voltage band is violated in the *Pessimistic*

Table 6.2: Results of experimentally evaluated test cases.

Test case	**avg. limit violation [V^{2}s]**			**10' violation time [%]**	**10' voltage band [V]**	**tap changes [1]**
avg. interval [min.]	1'	5'	10'			
Referece	4,516	677	0	0.00	27.6	15
Optimistic	5,044	1,044	0	0.00	27.7	14
Neutral	5,975	1,224	7	0.69	28.0	14
Pessimistic	7,223	2,020	217	1.91	28.5	13
Artificial 1	10,522	3,669	1,409	5.56	29.3	10
Artificial 2	19,566	9,278	5,288	7.99	30.6	8

test case by 0.5 V.

When having a look at the two artificial test cases, one can see that with such communication channel parameters, reasonable control of the low voltage grid is not possible. The controller lacks of representative information from the power grid and thus is not able to react adequately. In case that control action is issued by the controller, the high loss rate and missing transmission control of the used CX-1 protocol [SIE11] do not provide proper control conditions.

6.3.3 Evaluation by CHIL Simulation

The previous evaluation stage allowed for consideration of the two major smart grid domains, the electrical power grid and the ICT network, by coupled simulation. As a next step of the prototyping process and in order to continuously mitigate risk, also the controller hardware has to be considered in the evaluation phase. By the deployment of the control algorithms to the target hardware and testing of the thereby created control system in the co-simulation set-up, a controller-hardware-in-the-loop (CHIL) evaluation is performed. This comprehensive CHIL evaluation method is the last evaluation stage in the presented rapid prototyping process. Its implementation as well as its application in the context of the adjunct research project is presented as follows.

Implementation of Phase II, Stage (c)

The implementation of the CHIL prototyping stage on the basis of the SMB middleware is depicted in Figure 6.15. Its structure is very similar to the previously presented co-simulation set-up. While the actor set-up, and thus the SMB configuration, remains the same, the control algorithms are deployed to the controller target hardware and are not operated any longer on the developer's machine.

Due to the flexible structure of the SMB that offers socket connections for control algorithm interfacing, the transition of the algorithms from their development status to the operational status is simply performed by copying of the executable binaries to the target hardware. This

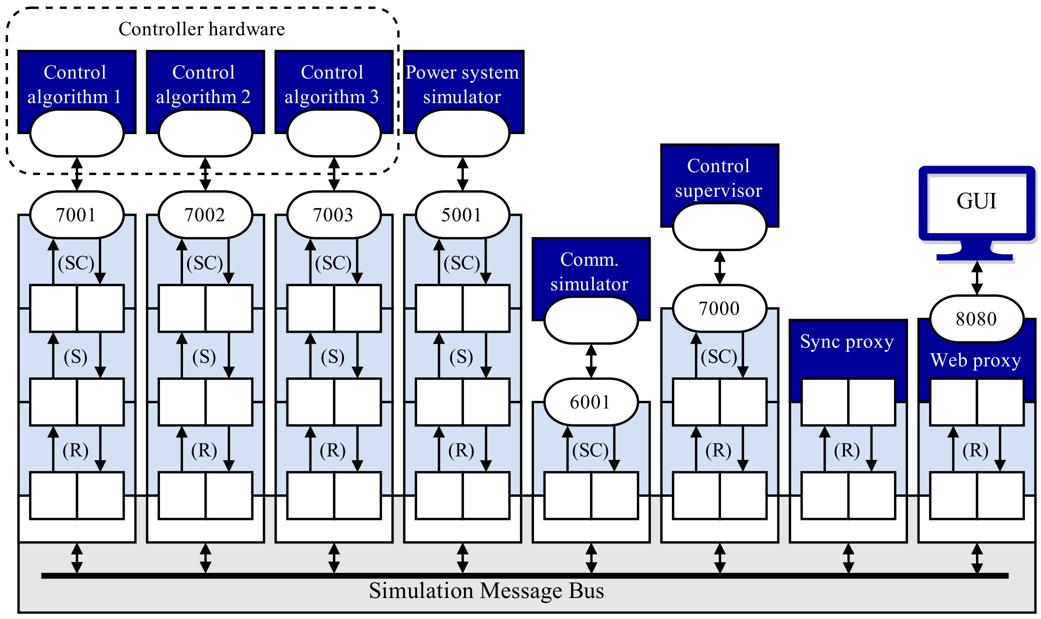


Figure 6.15: SMB configuration for Phase II, Stage (c): evaluation by CHIL simulation. The three control algorithms are operated on dedicated controller hardware and remotely connect to the SMB.

step represents the seamless field deployment, which is one of the main contributions of the presented process.

Because in this CHIL evaluation phase, the real controller hardware is interacting with the coupled simulation, the type of simulation has to be changed from off-line to on-line (real-time) simulation. This is necessary in order to mimic real-time power grid behaviour for the control system, which also operates in real-time. As a consequence of the field-like operation of the control system, also no simulation flow control messages (from the sync proxy) are exchanged with the control algorithms. While during off-line simulation the sync proxy prompts all actors to simulate distinct equidistant time slices in a continuous cyclic manner (cf. Figure 6.11), in this real-time simulation all actors run independently on the basis of their own clock.

This independent real-time operation of all involved actors is illustrated in the sequence diagram in Figure 6.16. After an initiation and configuration phase, which is not contained in the figure but can be retracted from the sync proxy's state diagram in Figure 5.5, a start signal is broadcasted among all actors. This start signal (`doRTS(t)`), that is sent from the sync proxy to all other actors, tells them to start an independent real-time simulation run for a specific time *t*.

While all event-based actors simply react to incoming messages, the physical system simulator (i.e. the power system simulator) needs to perform steady state load flow simulation on a specific granularity level. In order to avoid system inconsistency, it must be guaranteed, that the calculation time of the simulator does not exceed the duration of the simulated time step. Consequently, the physical system simulation must not be slower than real-time. This can be reached by a proper selection of the simulation time granularity, accuracy, and computational performance. In the presented set-up, the communication simulator has a check mechanism implemented that reports lacking packets on the basis of the system time and the packet's time stamp. Thus, it is possible to automatically check the integrity of each simulation run.

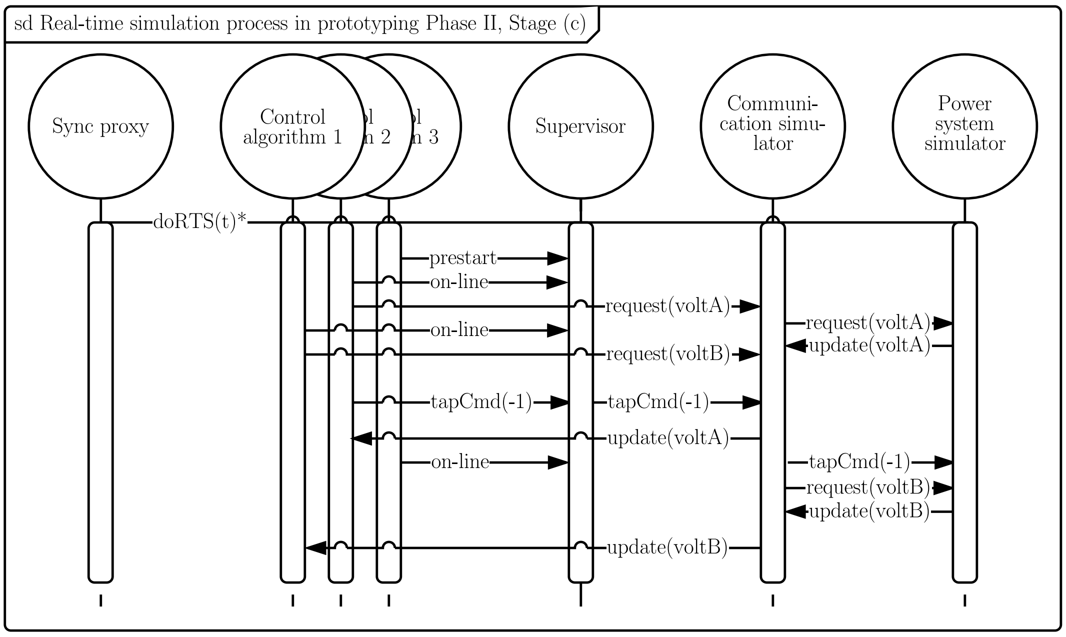


Figure 6.16: Sequence diagram for the real-time simulation process in Phase II, Stage (c) of the prototyping process.

Application of the Prototyping Stage

The CHIL evaluation stage is applied in the course of the *DG DemoNet – Smart LV Grid* project in several integration stages published in [FESM14]. This integration stages differ in their control system integration level and consequently approach the field set-up of the control system with every stage.

The first CHIL implementation stage is equivalent to the implementation described in the previous section. Thereby, the three control algorithms are deployed to the target hardware depicted in Figure 6.17. This hardware control component is an industry-grade box PC that can be mounted within the secondary substation and configured remotely via an Ethernet uplink to the grid operator's ICT network. However, for this first CHIL implementation stage, the controller hardware is located in the lab and only operates the control algorithms. These algorithms connect via TCP/IP socket connections to the SMB middleware on the developer's PC.

By this set-up, the operation of the algorithms in a different environment is tested. In the second CHIL integration stage also all other necessary control system actors (e.g. the control supervisor and the web proxy for providing the user interface and the SCADA uplink) are deployed to the controller hardware. The resulting complete control system is equivalent to the field control system. For the final evaluation stage in CHIL, it is operated together with the co-simulation consisting of the PowerFactory power system simulator and the AMIS PLC communication simulator (cf. application details in Section 6.3.1 and 6.3.2).

6.4 Prototyping Process Phase III: Field Operation

The field operation phase is the final phase of the prototyping process and consists of two stages. The stages are the open loop evaluation and the closed loop operation stage. Their principal

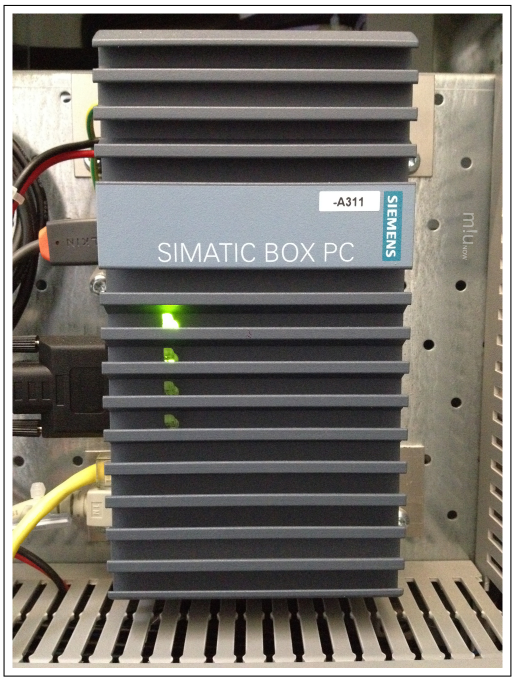


Figure 6.17: Target hardware for the control system: SIEMENS SIMATIC POX PC.

concept and the control command transmission, either checked by a control expert or directly forwarded to the field components, are presented in Section 4.5. Hereafter, the implementation of the third phase and its application in the course of the related project are presented.

Implementation of Phase III, Stages (a-b)

In the CHIL evaluation stage, the control system is completely compiled by the deployment of the control algorithm binaries and the auxiliary components like the control supervisor and the web proxy together with the Simulation Message Bus (SMB) middleware to the target hardware.

Figure 6.18 shows the final set-up of the control system with the SMB connecting all relevant control system components. This software set-up is completely operated on the field controller hardware depicted in Figure 6.17 of the previous section. Compared to the CHIL set-up, all simulation components are detached from the SMB to have only the control system components remaining, connected to the SMB as they are. Instead of the simulated power and communication system, the real field components are attached to the control system via a field automation gateway component, the IEC-60870 gateway.

Messages (e.g. for meter readings and set point transmission) that have previously been forwarded to the simulated field components via the communication system simulator, are directly sent to the IEC-60870 gateway in this field set-up. The gateway acts as a relay station and translation component to the field components and protocols (i.e. IEC 60870-5-104). The counterpart in the field is a AMIS data concentration unit, which is responsible for data aggregation and forwarding via the AMIS specific CX-1 PLC protocol.

The implementation of the two prototyping stages also contains the graphical user interface (GUI) that is provided by the web proxy as a web-based GUI. The open loop control stage implies a

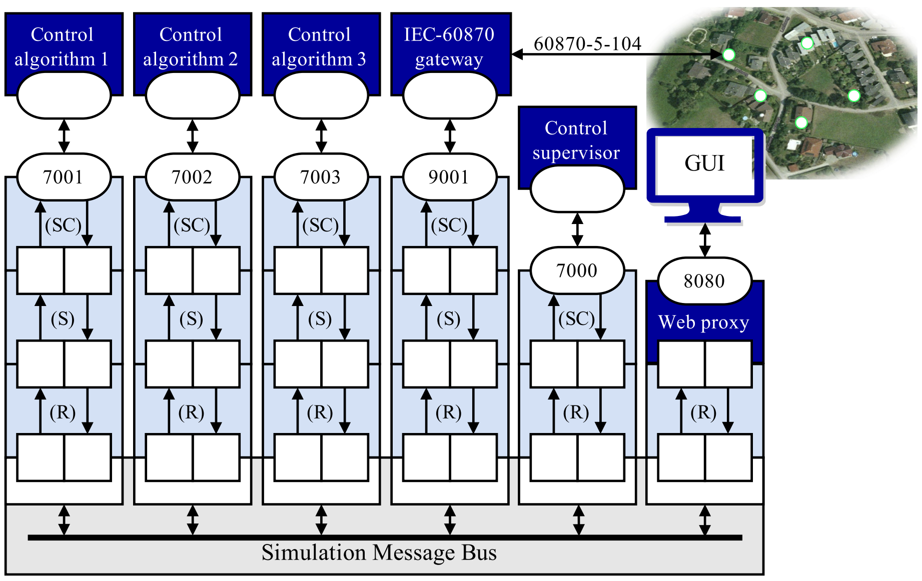


Figure 6.18: SMB configuration for Phase III, Stages (a) and (b). All simulative actors are replaced by an IEC-60870 field gateway component connecting the control algorithms to the real power grid.

control system expert to check the commands that are issued by the control system before their transmission to the actual actuator(s). Therefore, an extra widget for the manual approval of control commands is available in the GUI, which is depicted in Figure 6.21 of the next section.

Application of the Prototyping Stages

The application of the two field-related prototyping stages is performed in *DG DemoNet – Smart LV Grid* directly after the CHIL evaluation. Therefore, the final control system is deployed to the field (i.e. the secondary substation) and connected via the IEC-60870 gateway to the AMIS data concentrator. Through remote network access over the grid operator's network infrastructure (which in this case is protected by a VPN), control system experts can access the GUI to perform several actions.

At first after field deployment, open loop operation of the control system is performed. In this parallel operation mode, the control algorithms, which can be monitored via the algorithm widget (depicted in Figure 6.20), collect field measurements and use them in their internal control strategy to create control commands for specific field actuators (i.e. OLTC tap change commands or reactive power variation commands for distributed PV inverters). These commands are assessed by the control supervisor and displayed in the open loop widget of the GUI (cf. Figure 6.21).

In this widget, all control commands are queued and wait for approval by the control system expert. This expert has the option to either accept the command and thereby admit its forwarding to the field actuator, or decline the command in case that it does not seem to be suited for the current situation in the power grid. All accepted and declined commands are listed in a command log, which can also be seen at the bottom of the figure.

After the control expert evaluated each control algorithm's performance and suitability for the distribution grid under control, he is able to change the control system's operation mode from open to closed loop mode by the button on top of the open loop widget. By changing to closed loop mode, the commands from the control algorithms are directly forwarded from the control supervisor to the IEC 60870 gateway and thus to the grid components. This mode is the final operation mode, in which the control system remains as long as no problems arise during operation. Nevertheless, it is possible to switch back to open loop operation at any time in order to acknowledge control commands manually again.

In closed loop mode, several widgets of the GUI are used for the observation of the control system. The region map widget depicted in Figure 6.19 provides an overview of all relevant nodes in the controlled distribution grid. For each node, as well as for the transformer, voltage levels and power flow can be monitored. Both live and historical data can be depicted as charts or exported as time series (cf. left widgets of Figure 6.19).

The algorithm widget depicted in Figure 6.20 is used to monitor the state of all control algorithms (maximal five control stages). The supervisor at this state provides five slots for control algorithms. The slots can be manually disabled in case a control algorithm should not control the distribution grid. Furthermore, for validation purpose, a cyclic daily rotation of the active slot is implemented, which can also be enabled via the GUI's algorithm widget (see bottom part of Figure 6.20).

6.5 Graphical User Interface

The graphical user interface (GUI) is introduced in the abstract prototyping concept in the first evaluation stage (cf. Section 4.4). In the evaluation phase of the concept, the GUI fulfils the following tasks. It is used for the observation of physical state variables, like voltages, active and reactive power, the tap position and others. The development and test engineers employ the GUI to manually inject measurements or control commands to modify the system states in order to evaluate the control system's behaviour and performance. Additionally, it is used to control and monitor the (coupled) simulation in all three simulation-based evaluation stages.

The GUI is not only for the evaluation phase of high value. Also in the field operation phase, the GUI is designated to serve the engineer with the following features. It provides a comprehensive overview of the control system and its control algorithms as well as of the controlled distribution grid and its components' states. The GUI allows for the manual control of the transformer's tap changer and supports the control system experts in the open loop operation mode by providing an acknowledgement feature for all issued control commands.

The implementation of the GUI that has been created in the course of the *DG DemoNet – Smart LV Grid* project is depicted in the three figures of this section. It is written in HTML5 and uses JavaScript in the form of jQuery and AJAX elements for dynamic element creation and content updates. Furthermore, the access to the GUI is protected by HTTP Basic Auth routines.

At first, Figure 6.19 shows the complete GUI design with its menu bar on top providing links to several widgets, which can be aligned on the dashboard below in an arbitrary way. The most relevant widgets are described subsequently.

On the right hand side of Figure 6.19 a region map widget is depicted. It shows the rural area where the control distribution grid is located. The coloured circles represent the grid nodes. The circles' appearance represents the current power quality. The higher the voltage is, the warmer the colour of the circle becomes and the diameter of the circles represents the voltage magnitude

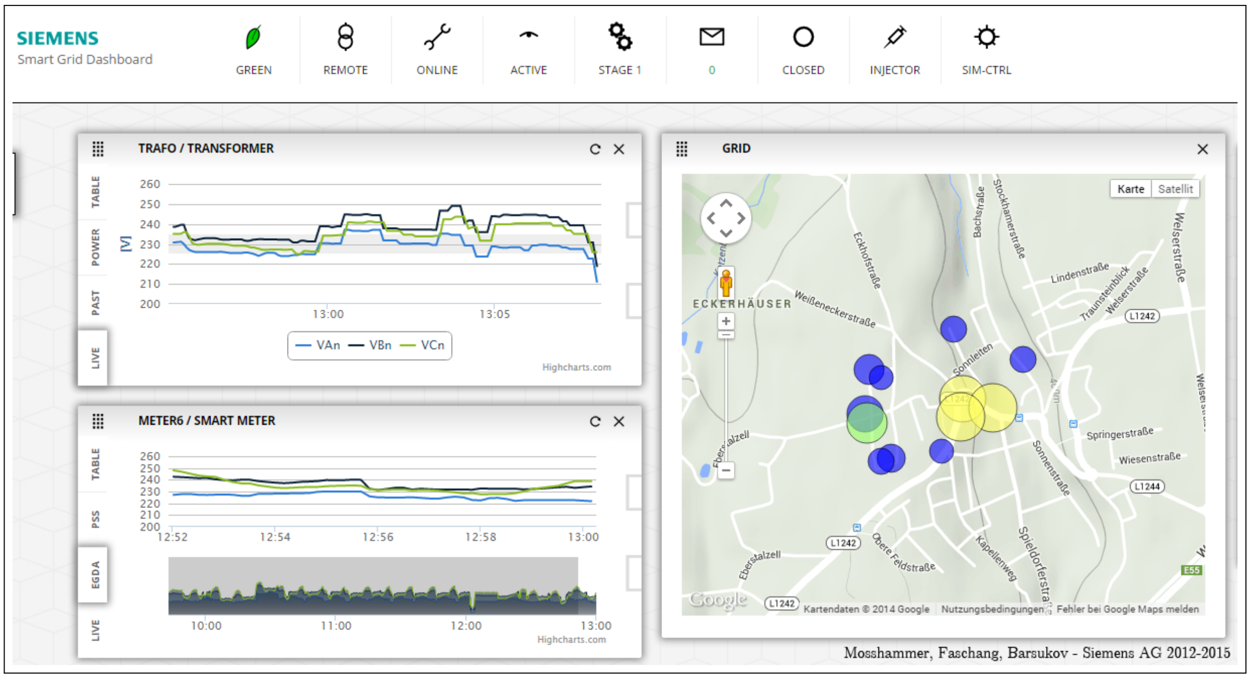
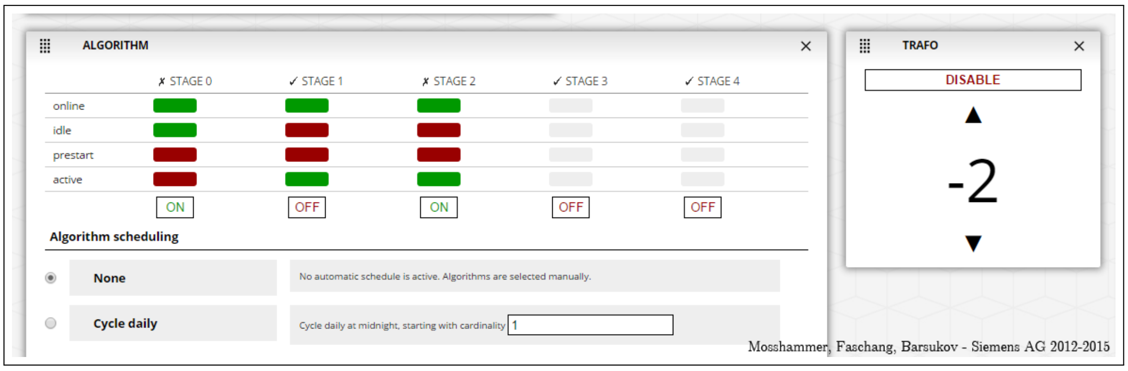


Figure 6.19: Graphical user interface of the prototyping process showing the region map widget with relevant nodes (right hand side) and two instances of the chart widget for the transformer and for the node *METER6* (left hand side).

deviation of the three phase voltages. In the example shown in the figure, voltage levels are rather low. The low voltages can also be seen in the voltage charts on the left hand side of the figure. Such charts are available for each and every monitored node of the distribution grid and for the transformer. They provide live measurements from the field, as well as historical data and an export feature for this data as (optionally re-sampled) time series. For the transformer, also active and reactive power can viewed over time.


ALGORITHM

	✗ STAGE 0	✓ STAGE 1	✗ STAGE 2	✓ STAGE 3	✓ STAGE 4
online	🟩	🟩	🟩	⬜	⬜
idle	🟩	🟥	🟥	⬜	⬜
prestart	🟥	🟥	🟥	⬜	⬜
active	🟥	🟩	🟩	⬜	⬜
	ON	OFF	ON	OFF	OFF

Algorithm scheduling

[✓ None] No automatic schedule is active. Algorithms are selected manually.

Cycle daily — Cycle daily at midnight, starting with cardinality 1

TRAFO

DISABLE

▲

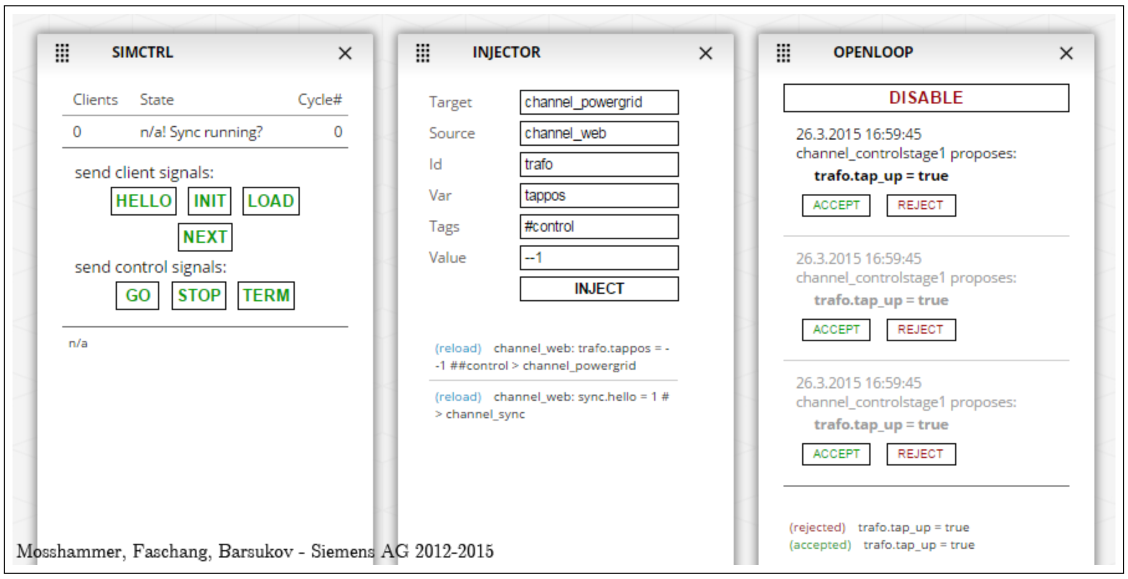
-2

▼

Mosshammer, Faschang, Barsukov - Siemens AG 2012-2015

Figure 6.20: Graphical user interface of the prototyping process showing the control algorithm widget and the transformer control widget.

Figure 6.20 shows the algorithm overview widget. In this widget the operational states of all active control algorithms are shown and the control system can be manipulated by enabling and disabling up to five control algorithm slots. This can be done either manually by the ON/OFF buttons or by an automated algorithm scheduling mechanism. The mechanism cyclically enables only one control algorithm. This feature is very useful in order to compare the algorithms' behaviour and performance with each other's. On the right hand side of Figure 6.20, the trafo widget is depicted. It shows the current position of the transformer's on-load tap-changer (OLTC) and allows for manual modification of this position. Because this manual changing of the OLTC position might cause harm to the distribution grid and power quality, manual tapping can be disabled. Furthermore, a safety mechanism limits the tapping-actions per minute.



Target	channel_powergrid
Source	channel_web
Id	trafo
Var	tappos
Tags	#control
Value	--1

(reload) channel_web: trafo.tappos = --1 ##control > channel_powergrid

(reload) channel_web: sync.hello = 1 # > channel_sync

26.3.2015 16:59:45
channel_controlstage1 proposes:
trafo.tap_up = true

26.3.2015 16:59:45
channel_controlstage1 proposes:
trafo.tap_up = true

26.3.2015 16:59:45
channel_controlstage1 proposes:
trafo.tap_up = true

(rejected) trafo.tap_up = true
(accepted) trafo.tap_up = true

Figure 6.21: Graphical user interface of the prototyping process showing the simulation control widget, the injector widget, and the open loop control widget.

The three widgets depicted in Figure 6.21 are especially useful for the evaluation phase. The leftmost widget allows for supervision and control of the stepped simulation process for control system evaluation by power grid simulation, coupled simulation, and controller-hardware-in-the-loop simulation. This widget directly communicates with the simulation control actor (which is implemented as sync proxy). It shows the total number of connected simulation clients, the state of the sync proxy's state machine and the current simulation cycle. Besides basic simulation flow control (by sending GO, STOP, and TERM control signals to the sync proxy's state machine) it also allows for detailed simulation analysis. This can be done by the participation of the control expert in the simulation as distinct simulation client. He can step through each single simulation step by sending simulation client signals (i.e. HELLO, INIT, LOAD, NEXT) via this widget.

The widget depicted in the middle of Figure 6.21 allows for packet injection for several purposes. It can be used for data manipulation or control command injection for testing purposes. This injection widget is especially useful for stressing the control algorithms with extreme or syntactically incorrect data. The rightmost widget of the last figure is the open loop widget. It is only used in open loop field operation mode to manually acknowledge or decline control commands that are issued by the control system under evaluation.

7 Results, Evaluation, and Conclusion

After showing the successful implementation of the proposed concept in Chapter 6, in this section the results are presented in the form of a summary of the developed process together with an evaluation and a critical discussion. The evaluation of the rapid prototyping process is performed in a threefold way. At first, a process verification is performed on the basis of the requirements defined in Section 2.3. Secondly, the process is validated based on the presented application through external experts and by field operation results. In a final evaluation phase, the main contribution of this work is discussed focussing on the fulfilment of the research question, which is the core issue of this work.

Finally, the complete work that has been conducted in this theses is concluded. A summary from an overlooking point of view is given and conclusions are drawn in retrospective. In the end, an outlook on future work and research activities is presented.

7.1 Results of the Rapid Control Prototyping Process

The concept for the prototyping process has been created on the basis of the stakeholders for rapid control prototyping for networked smart grids and their adjunct requirements that have been identified by the IntelliGrid methodology. This concept defines an agile implementation, evaluation, and deployment process that focuses on three elementary objectives. These objectives are the process' rapidness, the seamless field deployment, and the process' suitability to mitigate risk along the development process.

Within the *DG DemoNet – Smart LV Grid* project and in cooperation with experts from the industry (e.g. SIEMENS AG Austria, Fronius International GmbH), distribution system operators (e.g. Netz OÖ GmbH, Salzburg AG, Linz AG) and research institutions (AIT – Austrian Institute of Technology, Vienna University of Technology), the process was implemented and applied by these experts with the goal to maximize the hosting capacity for renewable energy sources (RES) in distribution grids.

As a result of this work and the process application by the consortium, three control systems for low voltage distribution grids in the Austrian provinces Upper Austria and Salzburg were developed, extensively evaluated, and deployed. The following distribution grids – depicted in Figure 7.1 – are concerned.

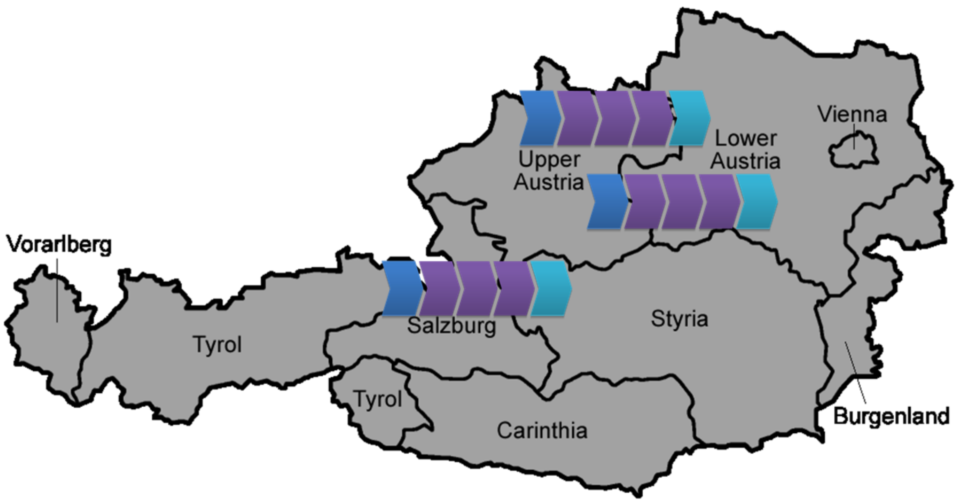


Figure 7.1: Map of Austria showing its provinces and the indicators for the three low voltage grid control systems, which have been developed and deployed by the application of the rapid control prototyping process. They are located in Eberstalzell and Littring (Upper Austria) as well as in Köstendorf (Salzburg).

- Eberstalzell (Upper Austria – Netz Oberösterreich GmbH)
- Littring (Upper Austria – Netz Oberösterreich GmbH)
- Köstendorf (Salzburg – Salzburg AG)

As a result of the process application, all these distribution grids are equipped with a control system described in Section 6. For each control system in the three distribution grids, the control prototyping process is currently in the final prototyping process stage (Phase III, Stage (b) – cf. Section 6.4). They are actively monitored and supervised in their autonomous operation state. In the case that the need for improvements or extensions of the control approach emerges, the process can easily be resumed by reviving the agile development process in any of its stages and working it through until ending up again with the improved and operating control system.

At the end of the adjunct research project, the control systems fulfil their primary objective, which is the stable operation of rural distribution grids with a high share of distributed generation (DG) from renewable resources (RES). Furthermore, also electric vehicle supply equipment is considered in the distribution grid of Salzburg AG. Additional details and other project-related information can be obtained from the final report [FFG15].

7.2 Process Evaluation

The process evaluation consists of three evaluation steps. The first one is the detailed verification of the process on the basis of the requirements it is intended to fulfil. The second evaluation step is the validation of the process, which is done in a twofold way according to the scientific research method presented in Section 1.5 of the introduction. Finally, as the third and last evaluation step, this thesis' research question is examined along with the description of the main contribution.

7.2.1 Process Verification by Requirements Satisfaction

The verification of the process is performed by decision criteria that are directly related to the fulfilment of each single requirement for the process. Table 7.1 lists these high level requirements that were identified within the IntelliGrid stakeholder process in Section 2.3.1. The following section verifies how these requirements are considered and satisfied by the control prototyping process.

Table 7.1: Process Requirements (PR) for the control prototyping process segmented into associated groups.

#	Process requirement `name`
PR1	`Definition of specification`
PR2	`Creation of the conceptual design`
PR3	`Implementation of control algorithm`
PR4	`Implementation of control system`
PR5	`Simulative control algorithm testing`
PR6	`Off-grid control system testing`
PR7	`Mitigation of risk`
PR8	`Seamless field deployment`
PR9	`Rapidness and high success rate`
PR10	`Grid operator integration and ability to handle changing requirements`
PR11	`Consideration of customers`
PR12	`Consideration of power grid equipment`

The first four process requirements (PR) are directly related to the implementation phase of the process. The specification stage, which is the first prototyping stage and typically performed by the development engineers, directly fulfils `PR1`. The creation of the conceptual design of the control concept (`PR2`) is covered by the second stage of the process. The third stage of the process covers both, the pure software implementation of the control algorithms (`PR3`) and their implementation in control hardware components (`PR4`) to form the complete control system. These first four requirements are straightforward and directly covered by Phase I of the prototyping process. The three stages of this phase were successfully applied in Section 4.3.

Process requirement `PR5` concerns the simulative control algorithm testing and is covered by the first two stages of the process' evaluation phase. These two stages are formed by the evaluation through power system simulation and the evaluation through coupled communication system and power system simulation. The putative over-achievement of `PR5` is caused by the fact that the two simulative evaluation stages are the core tools of the process for the achievement of `PR7`, which is the risk mitigation. Not only the simulative evaluation stages but also the controller-hardware-in-the-loop (CHIL) evaluation stage are conducive for `PR7`. CHIL evaluation is used for off-grid control system testing and evaluation and thus satisfies `PR6`.

The power grid operator integration and the ability to handle their changing requirements (`PR10`) directly increase the success rate of the development project (`PR9`). The integration of the power grid operator is inherently satisfied by the proper compilation of the project team. A system expert of the grid operator is considered in each project team in order to fulfil the first part of `PR10`. The ability to handle changing requirements – which is considered as the second part of `PR10` – is addressed by the general structure of the process and its agile structure. Agility pervades the entire process and allows the developer to jump back to a previous prototyping stage at any time.

For the satisfaction of the seamless field deployment requirement (`PR8`), which also contributes to the risk mitigation-requirement (`PR7`), the proposed middleware is created in a specific way. In Chapter 5, the middleware is designed in such a way that the control algorithms remain connected to the middleware throughout the whole prototyping process. The resulting seamless field deployment can be seen clearly, when the control system is deployed to the field by simply removing the simulators and by connecting the field automation gateway entity instead.

Commercial, private, and institutional customers (`PR11`) are taken into account by the integration of their consumption and generation characteristics in the power system simulation. This information can either be considered as simple profiles or time series in the power flow simulation, or introduced as dedicated models in the co-simulation. For the consideration of power grid equipment (`PR12`) also the power system simulation is foreseen. Dedicated models in the power system simulator are used to mimic the behaviour of the grid components.

7.2.2 Process Validation by Field Operation Results

The validation of the presented concept and its implementation as a rapid control prototyping process is done in accordance to the classification by Shaw [Sha02]. As presented in Figure 1.8, the chosen validation method is of empirical kind. To be concrete, validation is performed with the experience that the experts from the related research project gained during the application of the prototyping process.

Figure 7.2 shows evaluation results of the control system that was created by the application of the presented rapid control prototyping process by the experts of the *DG DemoNet – Smart LV Grid* project. In [SEH^{+}15, SBB^{+}15], Schwalbe et al. show the functionality of the developed control system containing three control stages. Furthermore, they evaluate the performance of the control system in three field tests. The results of the field tests, which were conducted in Eberstalzell, are depicted below in Figure 7.2. The field test results of the other two controlled distribution grids are contained in Appendix A.6.

The performance of the control system is assessed by the occupation of the available EN 50160 voltage band. This data is related to the hosting capacity of the respective distribution grid. The less voltage band is used, the more margin remains for the additional installation of distributed generation from renewable energy sources.

The first bar-chart in Figure 7.2 is the reference scenario and depicts the projected partitioning of the ±10 % voltage band given by EN 50160. It contains voltage band variation sectors for the

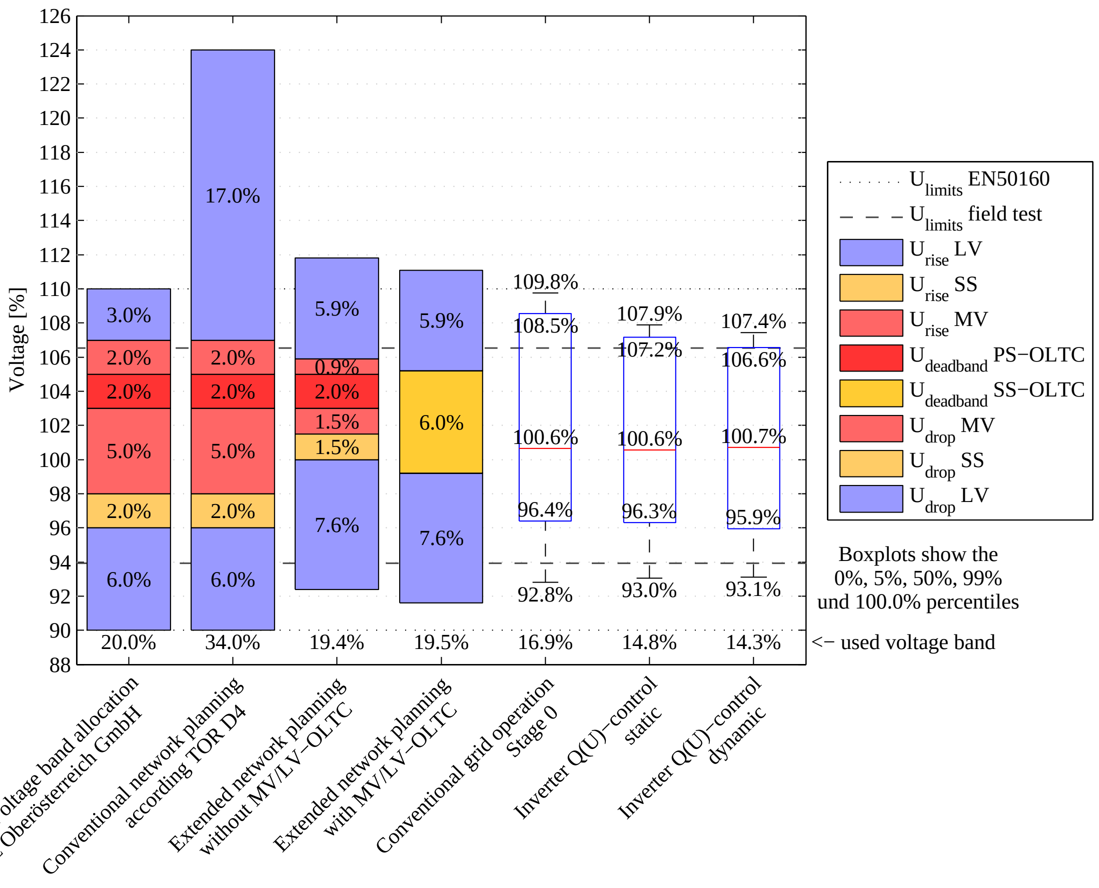


Figure 7.2: Evaluation results of the control system operated in the LV distribution grid in Eberstalzell/Upper Austria [SBB+15] (with friendly permission of Roman Schwalbe).

dead-band of the primary substation's (PS) and secondary substation's (SS) on-load tap-changer (OLTC) and for the voltage variations caused in the medium voltage (MV) and low voltage (LV) band. The limits of these sectors are individually set by the grid operator Netz Oberösterreich GmbH for the respective LV grids. Additionally, the EN 50160 voltage limits are marked as well as the voltage limits for the control system, which have been chosen narrower than defined by EN 50160 in order to challenge the control system.

The second bar-chart represents the hypothetical voltage situation in Eberstalzell's distribution grid according to a TOR D4 [Ene13] assessment, which is also used for conventional worst-case network planning. The high voltage magnitude is caused by the assumption that each of the single-phase PV installations is connected to the same phase, which is not realistic but must be considered for worst case network planning.

More realistic network planning methods can be applied when monitoring options are available in the grid to be able to assess the real voltage deviation in the overlay MV grid. This leads to the extended network planning voltage band estimations, which are represented by the third and fourth bar-chart. In the fourth bar-chart the influence of the MV grid and its components is untended due to the decoupling of the MV and LV grid by an OLTC.

The three rightmost bar-charts show five percentiles of the three control strategies' field operation measurements. The percentiles are the 0 %, 5 %, 50 % (median), 99 %, and the 100 % percentiles of all measurements that were collected during the three months lasting evaluation period. The numbers below the bar-charts show the voltage band occupation. It can clearly be seen, that the voltage band occupation is reduced by the use of the control system. The occupied voltage band is constantly decreasing even further by the use of the more complex control stage. Interestingly, the lower limit and the 5 %-percentile limit hardly change. This results from the fact that the created control system primarily tends to reduce the occupied voltage band and secondly aims for the centring of the occupied voltage band within the permitted range.

These results prove the general suitability of the process for rapid control prototyping. For the validation by experience, each prototyping stage of the process has been applied multiple times in order to develop the presented control systems. The implementation phase lies the basis for the control system. In this phase the experts' experience has been combined with the distribution system operator's requirements to specify the control systems. The three stages of the evaluation phase allowed for the comprehensive assessment of the implemented control system. Each of the three evaluation stages helped to identify issues of the control system that have been improved by an agile iteration back to the implementation phase. Besides the assessment of the control hardware, the CHIL evaluation also contained the deployment of the control algorithms to the controller hardware which has greatly contributed to the rapid and seamless field deployment of the control system. The final field operation phase has been gratefully accepted by the distribution system operator especially for the GUI-based supervision of the actively operation control system.

Besides this validation, also the process' three main contributions, which are the rapidness, the seamless field deployment, and the mitigation of risk along the prototyping process, are verified hereafter.

The rapidness of the process is assessed by the comparison of control system development conducted during the *Smart LV Grid* project with control development conducted in projects of the project chain *DG DemoNet*. Prior to the *Smart LV Grid* project, a set of projects from the same project chain (namely *DG DemoNet Concept* [Bun08], *DG DemoNet Validation* [FFG13a], and *BAVIS* [FFG12a]; discussed in Section 2.2.3) were conducted with the goal to create a control system for the maximisation of DG hosting capacity in rural Austrian MV distribution grids. Due to the fact that this goal is similar to the goal pursued in *Smart LV Grid* and the rural characteristic of the affected distribution grids, these projects are well comparable. Due to the applied characteristic of the research projects and the involved industry partners, the control development problem is closely related and similar to industrial control development tasks.

For the assessment of the technology maturity of the control system for the maximization of DG hosting capacity, the Technology Readiness Levels (TRL) are used. This metric has first been introduced by NASA [SPR88] and is also used by the EU Horizon 2020 Work Programme [Eur14]. Without applying a structured process, the control development in the MV-related projects took the team of researchers, engineers, and domain experts approximately seven years and three projects in order to come from the stage where the technology concept was formulated (TRL 2) to the stage where the technology was validated in industrially relevant environment (TRL 5) (cf. Appendix A.7). In comparison to that, by applying the structured process presented in this work, rapid control prototyping lead to a development time of only three years for a comparable task in the LV domain with a similarly composed team of developers and experts.

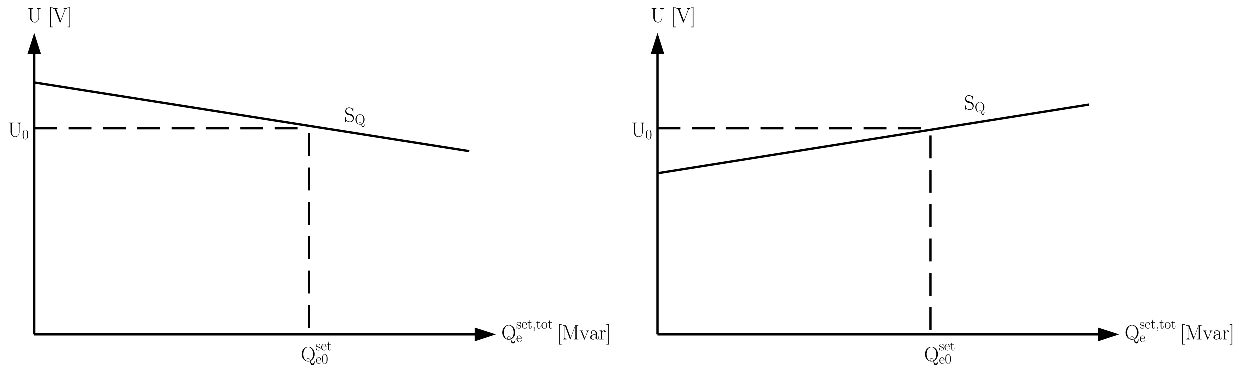
The seamless field deployment, which is the second main contribution of the presented rapid control prototyping process, also greatly contributes to the rapidness of the process. This deployment method is a fixed feature of the prototyping process and is characterized by the fact that the control algorithms neither need to be re-implemented, nor re-compiled, or connected in a different way for the field deployment. These facts are obvious and can directly be seen in the application of the process (cf. Chapter 6).

The risk mitigation is one of the most relevant properties of the developed process. This relevance is reflected by the extensive evaluation phase of the process. The validation of the process' risk mitigation capability is done by the application of the process within the project *DG DemoNet – Smart LV Grid*. During the application of the evaluation phase, several potential threats were identified in a relatively early stage and thus high costs caused by a malfunctioning control system were mitigated. Three examples of such threats that were identified are given below.

During the first long-term evaluation of the control system by power system simulation, the control algorithms suddenly stopped operation and crashed due to a memory leak in the controller's source code. This fault is mentioned as (A) in Figure 7.4. Not discovering this programming error would have led to a non-functioning control system in the field.

In the power system and communication system co-simulation stage of the process' evaluation phase, a misconfiguration of the control system pointed out (mentioned as (B) in Figure 7.4). This forced the control system to withdraw field measurements which were delayed for a certain amount of time due to the power-line-carrier (PLC) based communication system. Not discovering this problem would have caused the control system to act on the basis of deprecated field measurements.

The most serious malfunction that has been discovered during the application of the rapid control prototyping process was a misinterpretation of the control system's specification by the developer (mentioned as (C) in Figure 7.4). This lead to an inverse characteristic of the PV inverters' voltage droop control (illustrated in Figure 7.3).



(a) Correct voltage droop **(b)** Inverse voltage droop

Figure 7.3: Comparison of the PV inverters' correct and inverse voltage droop characteristics

Instead of a negative voltage droop (cf. Subfigure 7.3a), which is typical for a set of cooperating generators (discussed in Section 2.1), the controller issued an inverse voltage droop (cf. Subfigure 7.3b). Not discovering this issue would have caused an increase of reactive power generation in case of high voltage levels which would have further worsen the power quality and have finally lead to a critical over-voltage or the shut-down of components.

The three presented threats that have been identified within the application of the rapid prototyping process are depicted in Figure 7.4. In this figure the evaluation phase is depicted with its three stages. Furthermore, three fault classes are contained to which the identified threats

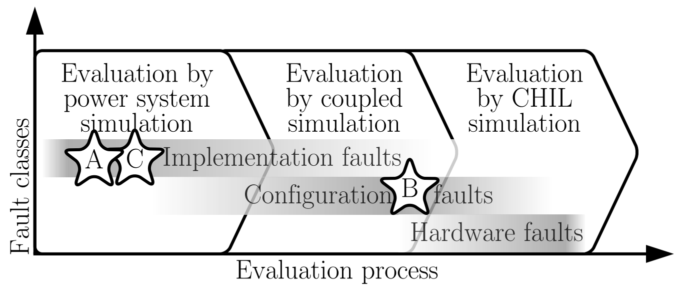

Evaluation by power system simulation
Evaluation by coupled simulation
Evaluation by CHIL simulation
A
C
Implementation faults
Configuration faults
B
Hardware faults
Fault classes
Evaluation process

Figure 7.4: Assignment of the identified potential faults/threats to fault classes and assignment of those classes to the three evaluation stages of the rapid control prototyping process.

are assigned. The final conclusion that can be drawn is the assignment of fault classes to the evaluation stages of the process. As depicted, implementation faults turned out to be identified in the early evaluation by power system simulation stage. Configuration faults are rather identified in the more sophisticated co-simulation or even CHIL evaluation stage, while hardware-related fault identification is clearly located in the CHIL evaluation stage.

7.2.3 Main Contribution

The research question of this thesis formulated in Section 1.4 is

Can control systems for low voltage distribution grids efficiently be designed, implemented, and tested in such a way that there is as little risk as possible and that the controlled power grid is not adversely affected?

and can now be positively answered.

The application of the presented process has clearly shown the feasibility to efficiently design, implement, and test control systems for low voltage distribution grids in such a way that there is as little risk as possible and that the controlled power grid is not adversely affected. The process' suitability has successfully been validated by the application of the process in an extensive Austrian research project. For the fulfilment of the research question, four problem statements (PS 1 – PS 4) were defined, which have been accomplished as follows. The work conducted for the fulfilment of these four problem statements is the main contribution of this work and forms the presented process for rapid control prototyping.

- *PS 1 – Investigate the method of control system development, identify related stakeholders, and define relevant requirements.*
  Suitable methods for control system development have been identified in the area of software and systems engineering (cf. Section 3.1). They have been modified and extended by a hardware deployment and field operation stage in order to serve for rapid control system prototyping (cf. Section 4.2). Relevant stakeholders have been identified in the course of related research projects (cf. Section 2.3.1). On the basis of these stakeholders, use cases (i.e. user stories) were formulated, which describe the control development process from the viewpoint of each stakeholder (cf. Section 2.3.2). With the IntelliGrid approach [IEC08] an established methodology has been used to extract the requirements from these stakeholders

and user stories, which have to be fulfilled by the prototyping process. Furthermore, methods and (software) tools have been researched and analysed from other similar application domains like the automotive or power-hardware industry that are useful for the control development task in the energy domain (cf. Section 3.2, Section 3.3, and Section 3.4).

- *PS 2 – Elaborate and define a concept for rapid control prototyping for smart low voltage grids, which fulfils the requirements defined in PS 1.*
  After an abstraction and assessment of the control development task (cf. Section 4.1), an agile concept has been elaborated and defined (cf. Section 4.2) on the basis of the analysed available methods and tools. This concept describes a process with three main phases (Implementation – cf. Section 4.3, Evaluation – cf. Section 4.4, and Field Operation – cf. Section 4.5), which are segmented into underlaid prototyping stages. For each of the phases' prototyping stages, all relevant actors have been identified and their interactions have been analysed. A subsequent discussion of the suggested process' structure (cf. Section 4.6.2) revealed an unmanageable increase of complexity. This problem has been solved by the introduction of a prototyping middleware (cf. Chapter 5) for the effective connection of all necessary software- and hardware-actors.

- *PS 3 – Realize a fully integrated implementation of the concept, which has been elaborated in PS 2.*
  The fully integrated implementation of the process concept, which has been created within this thesis, is presented in Chapter 6. The implementation is directly aligned to the process structure, which has been defined in the elaborated concept mentioned in PS 2. It thus consists of the same prototyping phases and stages (Implementation – cf. Section 6.2, Evaluation – cf. Section 6.3, and Field Operation – cf. Section 6.4). The implementation of these phases uses both established simulators (e.g. DIgSILENT PowerFactory) and proprietary tools (e.g. a model of the power-line-carrier communication channel, a co-simulation process flow and synchronization entity, etc.). The core element of the process implementation is the prototyping middleware, which is presented in Chapter 5, that allows for the seamless field deployment and the flexibility of the prototyping process.

- *PS 4 – Finally, discuss and evaluate the suitability of the created rapid control prototyping process.*
  The evaluation of the proposed process for rapid control prototyping has been done on the basis of its application in the course of the *DG DemoNet – Smart LV Grid* research project. Its suitability is discussed and critically reflected in Chapter 7.3. One of the key results of this assessment is that the process is well suited for the often happening changes of requirements and for rapid development and improvements of centralized control systems. Finally, an outlook on further use and potential improvements is given.

7.3 Conclusion

The fast progression in the power grids' paradigm shift and especially the associated integration of distributed renewable energy sources (RES) has led to a vast amount of research questions and tasks. To ensure system stability and power quality, it is necessary to keep pace with this progress by investigating new techniques and solutions, as done in this thesis by the development of an integrated process for rapid control prototyping. In this chapter, a reflection on the problem statement and the presented solution for control development for networked smart grid systems is given. The solution is critically examined and discussed, followed by a prospective outlook.

7.3.1 Summary

To counteract the increase of greenhouse gas (GHG) emissions and associated negative environmental influences, as well as to increase independence from limited fossil energy imports, changes in power generation and distribution systems are necessary. In Central Europe, these changes are primarily made by the introduction of distributed RES and by the increase in generation and distribution efficiency. To be able to integrate a high share of distributed renewable generation units into the power grid without running into risk of regional grid bottlenecks or balancing bottlenecks, an extension of the power grid infrastructure by the introduction of information and communication technology (ICT) is required. This combination of the power and the communication network together with all its newly possible applications, sensors, and actuators forms the *Smart Grid*.

In the near future, the introduction of ICT does not only allow for remote metering, but also for many other new ICT-driven applications, like generation or demand side management (DSM). Due to the volatile and highly fluctuating characteristic of renewable generation and its rapidly increasing share of the generation mix, it is absolutely necessary to be able to control both, the generation and the demand side in order to keep the power supply system stable and voltages within the predefined limits. Low voltage generation side units, like photovoltaic (PV) inverters or combined heat and power (CHP) plants, and high power consumption units, like electric vehicle (EV) charging equipment or heat pumps in households, are not the only new remotely controllable entities. Also transmission and distribution grid entities like switches and transformers with on-load tap-changers will be remotely controllable in the *Smart Low Voltage Grid*.

For the control of all these newly controllable grid units, powerful and well-adapted control algorithms are necessary. These control algorithms are a mission-critical entity in the *Smart Grid*, as they are designated to maintain grid stability and power quality. They are used to avoid regional grid bottlenecks, as well as supra-regional balancing issues. As the power grid is critical infrastructure that has to work at any time and cannot be taken off-line in order to test control concepts or algorithms, development, testing, and evaluation of new control systems has to be done off-line – for example by (co-)simulation with the integration of hardware components.

The investigation of the current situation of control algorithm development for low voltage grids showed that tools are available, which are dedicated to specific sub-tasks of control algorithm development (cf. Chapter 3) in similar or other application domains. Nevertheless, there is neither a defined smooth process, nor an integrated tool chain available for the tasks that are necessary for rapid and seamless smart grid control development. This lack of a coordinated process is directly addressed within this thesis and leads to the definition of the related research question and the problem statements, which are presented in Section 1.4.

Following an inductive-hypothetical research strategy, the problem has been abstracted by the identification of the relevant stakeholders and their requirements for a seamless rapid prototyping process. The theory for this work resulted from experience that has been gained in related research projects together with the research question. The concrete hypothesis is that it is possible to efficiently design, implement, and test control systems for low voltage distribution grids by a well-structured development process (presented in Chapter 4) in such a way that there is as little risk as possible and that the controlled power grid is not adversely affected.

The concept for such a control development process that has been elaborated within this work relies on an agile development approach resulting from a multi-dimensional process analysis and method selection (cf. Section 4.1). The concept is intended to fulfil three major objectives, also referred to as the three main contributions of the process, which are the process' rapidness, the seamless field deployment of the developed control system, and the process suitability to mitigate risk along the development process.

The single stages of the resulting rapid control prototyping process have been grouped into three consecutive phases. The first phase (cf. Section 4.3) is the implementation phase and consists of the specification, the conceptual controller design, and its concrete implementation. The second phase (cf. Section 4.4) is the evaluation phase, which is the most comprehensive one. It consists of three evaluation stages, which consequently approach the situation of the control system in the real distribution grid and thus allow for extensive tests and risk mitigation. The third and final phase (cf. Section 4.5) is the field operation phase that contains the open and closed loop field operation stages of the developed control system.

In alignment to the research strategy, the proposed strategy has been implemented (cf. Chapter 6) for control development in the course of the joint national research project *DG DemoNet – Smart LV Grid* and validated by its application through external experts and their application results. These results showed the suitability of the proposed process and its implementation for rapid control prototyping for networked smart grid systems. It has been validated by a successfully developed control system that consists of three control algorithms and is operated in three Austrian low voltage distribution grids. Furthermore, the three main contributions have been validated in an empirical way by the results of the process' application (cf. Section 7.2.2).

7.3.2 Critical Reflection

Resulting from the strong integration of RES into the power generation mix, two major problems emerge. The first one is the regional grid bottleneck, meaning that single lines or transformers may be overloaded or voltage limits may be violated. The second one concerns supra-regional balancing problems caused by a high regional concentration of RES (e.g. PV power in Bavaria/Germany or offshore wind power in the North Sea), where a high amount of electric energy would have to be transmitted among long distances. The first problem is directly addressed by the solution presented in this work and the control systems developed by the presented prototyping process. The latter one cannot directly be solved due to the regional perception of the developed control systems. To be able to solve also this second problem in a (semi)automatic manner, a universal overlay control system has to be designed, in order to control and coordinate all regional cells or distribution systems. In order to solve also the control development task for this overlay control system with the presented rapid control prototyping process, the following adaptations of the process would be necessary. The simulation model would have to be changed completely from a distribution grid simulation to a transmission grid simulation. In such a simulation, the single distribution grids would have to be abstracted and modelled as single simulation components with sensor and actuator capabilities. Also the presented deployment process would not be feasible for control of supra-regional balancing problems.

Further critical issues concern the Simulation Message Bus (SMB) middleware and the co-simulation and are discussed subsequently. The SMB has been designed proprietarily in order to combine all necessary tools and functions for the implementation and successful application of the prototyping environment. This was indispensable due to the fact that none of the already existing frameworks and middleware solutions provided full functionality and all necessary features, even though they would potentially have been of higher quality or performance than the prototypically self-designed SMB.

Furthermore, the packet propagation delay caused by the middleware itself is not considered in the current implementation. This is only relevant for real-time simulation and does not concern off-line simulation. In real-time simulation the unconsidered middleware delay only adds up some milliseconds, which can be neglected for real packet delays of up to seconds. For other, potentially faster applications this deviation should be considered.

Co-simulation, as it is done in the presented solution, is only capable of quasi-stationary simulation with variable step size. This is not suited for simulating dynamic behaviour of the electric energy system. More research efforts have to be invested to integrate also transient simulation into the co-simulation framework, which is another challenging task. Especially the dynamic simulation of multiple physical systems (cyber-physical energy systems) under the consideration of real power hardware components would be the next development stage for the presented solution.

The development of control algorithms for low voltage power grids requires an individual model of each of these power grids. Not only the power grids' structure has to be modelled, but also every actuator within these grids. The digitalization and realistic modelling of every single power grid is a huge amount of work. Therefore, ways should be investigated to automate this process and to be able to use standardized models and interfaces for sensors and actuators.

Even though the presented process is well structured and breaks the development task down into smaller tasks, which are easier to handle by the control development engineers, its application showed that support from an expert is necessary in each prototyping stage. Especially the proprietary connection of the simulators had to be tuned several times before it was conveniently usable. Despite the prototyping process' clear structure, in the end the responsibility of following it always remains with the developer that uses the process and the related tools in order to deliver high quality work.

7.3.3 Outlook

This work presents a first step towards seamless and integrated control development by a concrete process. As the presented approach is tailored to the development of a centralized control systems for low voltage distribution grids, the plan for the near future is its expansion to serve smart grid control system developers in a more generic way. Potential further applications are the already mentioned adaptation and application of the rapid prototyping process for control systems in higher voltage and control levels of the power grid infrastructure. The dynamic development of high level control systems in combination with the presented solution for low voltage systems would allow for control of the complete power grid infrastructure in an integrated way and thus to optimize local energy consumption, reduce trans-regional power transfer, and maximize generation from renewable energy sources.

Furthermore, an adaptation of the process for control development based on a distributed (multi) agent-based structure is envisioned in further research activities. Due to the distributed nature of this control approach and the agents' mutual influence, it is difficult to develop the control system and even more challenging to assess it in lab infrastructure. Therefore, the integration of control hardware components into coupled simulation as it is presented in this work, will be a highly valuable feature. The ability to test real control components' interaction with a potentially high number of simulated components is highly interesting and relevant for the development of such agent-based power grid control systems.

Beside these short-term plans, the long-term vision is the creation of an integrated control development environment that is capable of realistically simulating a cyber-physical energy system among all voltage levels and among all relevant physical domains in a dynamic manner. This environment should be capable of dynamically exchanging models of power grid components and directly importing topological information from the grid operator's asset management system. The developed control algorithms should be able to detect topology changes automatically and to react dynamically on the basis of such events. The deployment process should be as convenient as downloading an application to a smart phone, should also take security issues into consideration, and should support the dynamic exchange of control algorithms amongst agents.

The presented work is the first step into this envisioned future of rapid control prototyping for networked smart grid systems. Even though it does not solve all the envision tasks yet, it lies the basis and is, due to its modular structure, well-suited for an extension and further development into the right direction.

A Appendices

A.1 Syntax for the Middleware Configuration and Packets

Listing A.1 shows the XML schema description for the middleware configuration.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:jaxb="http://java.sun.com/xml/ns/jaxb"
  xmlns:xjc="http://java.sun.com/xml/ns/jaxb/xjc"
  jaxb:extensionBindingPrefixes="xjc"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified"
  version="2.0"
  jaxb:version="1.0">

  <xs:element name="Configuration" type="tConfiguration" />

  <xs:complexType name="tConfiguration">
    <xs:sequence>
      <xs:element name="Channel" type="tChannel" minOccurs="2" maxOccurs="
          unbounded" />
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="tChannel">
    <xs:sequence>
      <xs:element name="Proxy" type="tProxy" minOccurs="0" maxOccurs="unbounded"
          />
    </xs:sequence>
    <xs:attribute name="id" type="xs:string" use="required" />
  </xs:complexType>

  <xs:complexType name="tProxy">
    <xs:sequence>
      <xs:element name="param" type="tParam" minOccurs="0" maxOccurs="unbounded"
          />
    </xs:sequence>
    <xs:attribute name="id" type="xs:string" use="required" />
    <xs:attribute name="type" type="xs:string" use="required" />
    <xs:attribute name="channel" type="xs:string" use="optional" />
  </xs:complexType>
  <xs:complexType name="tParam">
    <xs:attribute name="name" type="xs:string" use="required" />
    <xs:attribute name="value" type="xs:string" use="required" />
  </xs:complexType>

</xs:schema>
```

Listing A.1: Middleware configuration syntax in XSD format

Listing A.2 shows the XML schema description for the middleware packets.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:jaxb="http://java.sun.com/xml/ns/jaxb"
  xmlns:xjc="http://java.sun.com/xml/ns/jaxb/xjc"
  jaxb:extensionBindingPrefixes="xjc"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified"
  version="2.0"
  jaxb:version="1.0">

  <xs:element name="Packet" type="tPacket" />

  <xs:complexType name="tPacket">
    <xs:sequence>
      <xs:element name="Param" type="tParam" minOccurs="0" maxOccurs="unbounded"/
          >
    </xs:sequence>
    <xs:attribute name="PacketType" type="tPacketType" use="optional" />
    <xs:attribute name="SourceID" type="xs:string" use="optional" />
    <xs:attribute name="Timestamp" type="xs:string" use="optional" />
    <xs:attribute name="Delay" type="xs:string" use="optional" />
    <xs:attribute name="Lost" type="xs:boolean" use="optional" />
  </xs:complexType>

  <xs:simpleType name="tPacketType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="create" />
      <xs:enumeration value="request" />
      <xs:enumeration value="update" />
      <xs:enumeration value="delete" />
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="tParam">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="id" type="xs:string" use="required" />
        <xs:attribute name="var" type="xs:string" use="optional" />
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>

</xs:schema>
```

Listing A.2: Middleware packet grammar in XSD format

A.2 Simulation Scenario Description

The scenario file, that has been used in the prototyping process of the *DG DemoNet – Smart LV Grid* project is presented in the following Listing A.3.

```
SCENARIONAME;GRIDNAME;MODE;SIMSTEPSIZE_MS;EMUSCALEFACTOR;BEGINTIME;ENDTIME;
    FILE_AMISLOG;FILE_DATAPOINTS_EGDA;FILE_DATAPOINTS_MODBUS;FILE_CTRL1_CONFIG;
    FILE_CTRL2_CONFIG;FILE_CTRL3_CONFIG;FILE_CTRL4_CONFIG
Sommer/Montag/Schoen;SmartLVGrid\EBERSTALZELL Strang 1 4 5 API;Simulation
    ;1000;1;01.01.2012 10:32;01.01.2012 16:32;./resources/amis.csv;./resources/
    datapints_egda.csv;./resources/datapints_modbus.csv;./resources/ctrl_config1.
    csv;./resources/ctrl_config2.csv;./resources/ctrl_config3.csv;./resources/
    ctrl_config4.csv
Winter/Dienstag/Schlecht;SmartLVGrid\EBERSTALZELL Strang 1 4 5 API;Simulation
    ;1000;2;02.02.2012 11:34;01.01.2012 17:55;./resources/amis.csv;./resources/
    datapints_egda.csv;./resources/datapints_modbus.csv;./resources/ctrl_config1.
    csv;./resources/ctrl_config2.csv;./resources/ctrl_config3.csv;./resources/
    ctrl_config4.csv
Herbst/Sonntag/Gut;SmartLVGrid\EBERSTALZELL Strang 1 4 5 API;Emulation
    ;1000;3;01.01.2012 10:52;01.01.2012 17:55;./resources/amis.csv;./resources/
    datapints_egda.csv;./resources/datapints_modbus.csv;./resources/ctrl_config1.
    csv;./resources/ctrl_config2.csv;./resources/ctrl_config3.csv;./resources/
    ctrl_config4.csv
Fruehling/Mittwoch/Schoen;SmartLVGrid\EBERSTALZELL Strang 1 4 5 API;Emulation
    ;1000;4;01.01.2012 06:32;01.01.2012 17:55;./resources/amis.csv;./resources/
    datapints_egda.csv;./resources/datapints_modbus.csv;./resources/ctrl_config1.
    csv;./resources/ctrl_config2.csv;./resources/ctrl_config3.csv;./resources/
    ctrl_config4.csv
```

Listing A.3: Simulation scenarios consisting of several parameters stored in CSV format.

The scenario file is used by the sync proxy, which is the implementation of the *Simulation Control* actor and an element of the middleware proposed in Chapter 5. The scenario file stores the following relevant simulation parameters in CSV format.

- `SCENARIONAME` - This is a unique name for the simulation scenario.
- `GRIDNAME` - The gridname identifies the power grid (file) that has to be used in the power system simulator for this specific simulation scenario.
- `MODE` - This sets the simulation mode to either off-line or on-line simulation in order to support hardware integration for CHIL evaluation.
- `SIMSTEPSIZE_MS` - For the off-line simulation this parameter defines the length of one simulation step size.
- `EMUSCALEFACTOR` - For on-line simulation this factor defines the speed-up of the simulation. In case of real-time simulation this factor is set to one.
- `BEGINTIME` - This is the begin date and time of the simulation period.
- `ENDTIME` - This is the end date and time of the simulation period.
- `FILE_AMISLOG` - This is the source file for the communication simulator that models the AMIS PLC channel. This file contains statistical data about packet delay and loss.

- FILE_DATAPOINTS_EGDA - For the modelling of the communication system, this file contains a list of all data points that are accessed via EGDA communication.
- FILE_DATAPOINTS_MODBUS - For the modelling of the communication system, this file contains a list of all data points that are accessed via Modbus communication.
- FILE_CTRL*_CONFIG - This is a set of configuration file names that are forwarded to the respective control algorithms.

A.3 SMB Middleware Configurations

The hereafter presented listings contain the XML configurations for the Simulation Message Bus (SMB) middleware, which have been written for the application of the rapid control prototyping process (described in Chapter 4).

SMB Middleware Configuration for Prototyping Phase I, Stage (c)

Listing A.4 shows the SMB middleware configuration for the prototyping Phase I, Stage (c).

```xml
<?xml version="1.0" encoding="UTF-8" ?>

<Configuration
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="C:\Eclipse\resources\smbconfiguration.xsd">

  <!-- Power Grid Stub -->
  <Channel id="channel_stub">
    <Proxy id="stub_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy"
        >
      <param name="destination" value=".* -> channel_cvcu,channel_logging" />
    </Proxy>
    <Proxy id="stub_connector" type="at.siemens.smb.proxy.impl.
        SocketConnectorProxy">
      <param name="port" value="5001" />
      <param name="binary" value="true" />
    </Proxy>
  </Channel>

  <!-- Control Algorithm -->
  <Channel id="channel_ctrl">
    <Proxy id="ctrl_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy"
        >
      <param name="destination" value=".* -> channel_stub,channel_logging" />
    </Proxy>
    <Proxy id="ctrl_streaming" type="at.siemens.smb.proxy.impl.
        ObjectStreamingProxy" >
      <param name="context" value="at.siemens.smb.data.packet.Packet" />
      <param name="format" value="JSON" />
    </Proxy>
    <Proxy id="ctrl_connector" type="at.siemens.smb.proxy.impl.
        SocketConnectorProxy">
      <param name="port" value="7001" />
      <param name="binary" value="false" />
    </Proxy>
  </Channel>

  <!-- File Logging -->
  <Channel id="channel_logging">
    <Proxy id="logging_filelog" type="at.siemens.smb.proxy.impl.FileLogProxy">
      <param name="file" value="./log/measurements.log" />
      <param name="max_records" value="10000" />
      <param name="max_history" value="10" />
    </Proxy>
  </Channel>

</Configuration>
```

Listing A.4: SMB middleware configuration for prototyping Phase I, Stage (c)

SMB Middleware Configuration for Prototyping Phase II, Stage (a)

Listing A.5 shows the SMB middleware configuration for the prototyping Phase II, Stage (a).

```
<?xml version="1.0" encoding="UTF-8" ?>

<Configuration
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="C:\Eclipse\resources\smbconfiguration.xsd">

  <!-- Power System Simulator PowerFactory -->
  <Channel id="channel_pf">
    <Proxy id="pf_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy">
      <param name="destination" value=".*sync.* -> channel_sync,channel_logging"
         />
      <param name="destination" value=".* -> channel_ctrl1,channel_ctrl2,
         channel_ctrl3,channel_logging" />
    </Proxy>
    <Proxy id="pf_streaming" type="at.siemens.smb.proxy.impl.ObjectStreamingProxy
       ">
      <param name="context" value="at.siemens.smb.data.packet" />
    </Proxy>
    <Proxy id="pf_connector" type="at.siemens.smb.proxy.impl.SocketConnectorProxy
       ">
      <param name="port" value="5001" />
      <param name="binary" value="false" />
    </Proxy>
  </Channel>

  <!-- Supervisor -->
  <Channel id="channel_sv">
    <Proxy id="sv_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy">
      <param name="destination" value="* -> channel_pf,channel_web,
         channel_logging" />
    </Proxy>
    <Proxy id="sv_connector" type="at.siemens.smb.proxy.impl.SocketConnectorProxy
       ">
      <param name="port" value="7000" />
      <param name="binary" value="true" />
    </Proxy>
  </Channel>

  <!-- Control Algorithm Stage 1 -->
```

```
<Channel id="channel_ctrl1">
  <Proxy id="ctrl1_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
      ">
    <param name="destination" value=".*sync.* -> channel_sync,channel_logging"
        />
    <param name="destination" value=".* -> channel_sv,channel_logging" />
  </Proxy>
  <Proxy id="ctrl1_streaming" type="at.siemens.smb.proxy.impl.
      ObjectStreamingProxy">
    <param name="context" value="at.siemens.smb.data.packet.Packet" />
    <param name="format" value="JSON" />
  </Proxy>
  <Proxy id="ctrl1_connector" type="at.siemens.smb.proxy.impl.
      SocketConnectorProxy">
    <param name="port" value="7001" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>

<!-- Control Algorithm Stage 2 -->
<Channel id="channel_ctrl2">
  <Proxy id="ctrl2_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
      ">
    <param name="destination" value=".*sync.* -> channel_sync,channel_logging"
        />
    <param name="destination" value=".* -> channel_sv,channel_logging" />
  </Proxy>
  <Proxy id="ctrl2_streaming" type="at.siemens.smb.proxy.impl.
      ObjectStreamingProxy">
    <param name="context" value="at.siemens.smb.data.packet.Packet" />
    <param name="format" value="JSON" />
  </Proxy>
  <Proxy id="ctrl2_connector" type="at.siemens.smb.proxy.impl.
      SocketConnectorProxy">
    <param name="port" value="7002" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>

<!-- Control Algorithm Stage 3 -->
<Channel id="channel_ctrl3">
  <Proxy id="ctrl3_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
      ">
    <param name="destination" value=".*sync.* -> channel_sync,channel_logging"
        />
    <param name="destination" value=".* -> channel_sv,channel_logging" />
  </Proxy>
  <Proxy id="ctrl3_streaming" type="at.siemens.smb.proxy.impl.
      ObjectStreamingProxy">
    <param name="context" value="at.siemens.smb.data.packet.Packet" />
    <param name="format" value="JSON" />
  </Proxy>
  <Proxy id="ctrl3_connector" type="at.siemens.smb.proxy.impl.
      SocketConnectorProxy">
    <param name="port" value="7003" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>
```

```
  <!-- Sync Proxy -->
  <Channel id="channel_sync">
    <Proxy id="sync_routing" type="at.siemens.smb.proxy.impl.RoutingProxy">
      <param name="destination" value="*" />
    </Proxy>
    <Proxy id="sync_proxy" type="at.siemens.smb.proxy.impl.SyncProxy">
      <param name="clients" value="4" />
      <param name="trigger" value="false" />
      <param name="scenariofile" value="./resources/sync_scenarios.csv" />
    </Proxy>
  </Channel>

  <!-- Web Proxy -->
  <Channel id="channel_web">
    <Proxy id="web_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy">
      <param name="destination" value="* #command -> channel_pf,channel_logging"
          />
      <param name="destination" value="* #sync -> channel_sync,channel_logging" /
          >
      <param name="destination" value="* -> channel_logging" />
    </Proxy>
    <Proxy id="web_proxy" type="at.siemens.smb.proxy.impl.WebServiceProxy">
      <param name="maxpacketlogsize" value="10000" />
      <param name="webserverport" value="8080" />
      <param name="www_index" value="login.html" />
      <param name="www_root" value="./resources/login/" />
      <param name="ssl_https_enabled" value="false" />
      <param name="auth_enabled" value="true" />
      <param name="auth_realm" value="./resources/jetty/jetty-auth-realm.
          properties" />
      <param name="webapp_war" value="./resources/jetty/exploded_war" />
      <param name="webapp_context" value="/auth/dashboard" />
    </Proxy>
  </Channel>

  <!-- File Logging -->
  <Channel id="channel_logging">
    <Proxy id="logging_filelog" type="at.siemens.smb.proxy.impl.FileLogProxy">
      <param name="file" value="./log/measurements.log" />
      <param name="max_records" value="10000" />
      <param name="max_history" value="10" />
    </Proxy>
  </Channel>

</Configuration>
```

Listing A.5: SMB middleware configuration for prototyping Phase II, Stage (a)

SMB Middleware Configuration for Prototyping Phase II, Stages (b) and (c)

Listing A.6 shows the SMB middleware configuration for the prototyping Phase II, Stages (b) and (c).

```
<?xml version="1.0" encoding="UTF-8" ?>

<Configuration
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:noNamespaceSchemaLocation="C:\Eclipse\resources\smbconfiguration.xsd">

<!-- Power System Simulator PowerFactory -->
<Channel id="channel_pf">
  <Proxy id="pf_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy">
    <param name="destination" value=".*sync.* -> channel_sync,channel_logging"
       />
    <param name="destination" value=".* -> channel_comsim,channel_logging;
       channel_ctrl1,channel_ctrl2,channel_ctrl3,channel_logging" />
  </Proxy>
  <Proxy id="pf_streaming" type="at.siemens.smb.proxy.impl.ObjectStreamingProxy
     ">
    <param name="context" value="at.siemens.smb.data.packet" />
  </Proxy>
  <Proxy id="pf_connector" type="at.siemens.smb.proxy.impl.SocketConnectorProxy
     ">
    <param name="port" value="5001" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>

<!-- Communication Simulator -->
<Channel id="channel_comsim">
  <Proxy id="comsim_connector" type="at.siemens.smb.proxy.impl.
     SocketConnectorProxy">
    <param name="port" value="6001" />
    <param name="binary" value="true" />
  </Proxy>
</Channel>

<!-- Supervisor -->
<Channel id="channel_sv">
  <Proxy id="sv_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy">
    <param name="destination" value="* -> channel_comsim,channel_logging;
       channel_pf,channel_web,channel_logging" />
  </Proxy>
  <Proxy id="sv_connector" type="at.siemens.smb.proxy.impl.SocketConnectorProxy
     ">
    <param name="port" value="7000" />
    <param name="binary" value="true" />
  </Proxy>
</Channel>

<!-- Control Algorithm Stage 1 -->
<Channel id="channel_ctrl1">
  <Proxy id="ctrl1_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
     ">
    <param name="destination" value=".*sync.* -> channel_sync,channel_logging"
       />
    <param name="destination" value=".* -> channel_sv,channel_logging" />
  </Proxy>
  <Proxy id="ctrl1_streaming" type="at.siemens.smb.proxy.impl.
     ObjectStreamingProxy">
    <param name="context" value="at.siemens.smb.data.packet.Packet" />
    <param name="format" value="JSON" />
  </Proxy>
  <Proxy id="ctrl1_connector" type="at.siemens.smb.proxy.impl.
     SocketConnectorProxy">
    <param name="port" value="7001" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>

<!-- Control Algorithm Stage 2 -->
<Channel id="channel_ctrl2">
  <Proxy id="ctrl2_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
      ">
    <param name="destination" value=".*sync.* -> channel_sync,channel_logging"
        />
    <param name="destination" value=".* -> channel_sv,channel_logging" />
  </Proxy>
  <Proxy id="ctrl2_streaming" type="at.siemens.smb.proxy.impl.
      ObjectStreamingProxy">
    <param name="context" value="at.siemens.smb.data.packet.Packet" />
    <param name="format" value="JSON" />
  </Proxy>
  <Proxy id="ctrl2_connector" type="at.siemens.smb.proxy.impl.
      SocketConnectorProxy">
    <param name="port" value="7002" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>

<!-- Control Algorithm Stage 3 -->
<Channel id="channel_ctrl3">
  <Proxy id="ctrl3_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
      ">
    <param name="destination" value=".*sync.* -> channel_sync,channel_logging"
        />
    <param name="destination" value=".* -> channel_sv,channel_logging" />
  </Proxy>
  <Proxy id="ctrl3_streaming" type="at.siemens.smb.proxy.impl.
      ObjectStreamingProxy">
    <param name="context" value="at.siemens.smb.data.packet.Packet" />
    <param name="format" value="JSON" />
  </Proxy>
  <Proxy id="ctrl3_connector" type="at.siemens.smb.proxy.impl.
      SocketConnectorProxy">
    <param name="port" value="7003" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>

<!-- Sync Proxy -->
<Channel id="channel_sync">
  <Proxy id="sync_routing" type="at.siemens.smb.proxy.impl.RoutingProxy">
    <param name="destination" value="*" />
  </Proxy>
  <Proxy id="sync_proxy" type="at.siemens.smb.proxy.impl.SyncProxy">
    <param name="clients" value="5" />
    <param name="trigger" value="false" />
    <param name="scenariofile" value="./resources/sync_scenarios.csv" />
  </Proxy>
</Channel>

<!-- Web Proxy -->
<Channel id="channel_web">
  <Proxy id="web_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy">
```

```
      <param name="destination" value="* #command -> channel_pf,channel_logging"
         />
      <param name="destination" value="* #sync -> channel_sync,channel_logging" /
         >
      <param name="destination" value="* -> channel_logging" />
    </Proxy>
    <Proxy id="web_proxy" type="at.siemens.smb.proxy.impl.WebServiceProxy">
      <param name="maxpacketlogsize" value="10000" />
      <param name="webserverport" value="8080" />
      <param name="www_index" value="login.html" />
      <param name="www_root" value="./resources/login/" />
      <param name="ssl_https_enabled" value="false" />
      <param name="auth_enabled" value="true" />
      <param name="auth_realm" value="./resources/jetty/jetty-auth-realm.
         properties" />
      <param name="webapp_war" value="./resources/jetty/exploded_war" />
      <param name="webapp_context" value="/auth/dashboard" />
    </Proxy>
  </Channel>

  <!-- File Logging -->
  <Channel id="channel_logging">
    <Proxy id="logging_filelog" type="at.siemens.smb.proxy.impl.FileLogProxy">
      <param name="file" value="./log/measurements.log" />
      <param name="max_records" value="10000" />
      <param name="max_history" value="10" />
    </Proxy>
  </Channel>

</Configuration>
```

Listing A.6: SMB middleware configuration for prototyping Phase II, Stages (b) and (c)

SMB Middleware Configuration for Prototyping Phase III

Listing A.7 shows the SMB middleware configuration for the prototyping Phase III, Stages (a) and (b).

```
<?xml version="1.0" encoding="UTF-8" ?>

<Configuration
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="C:\Eclipse\resources\smbconfiguration.xsd">

  <!-- Automation system (IEC60870) -->
  <Channel id="channel_ssn">
    <Proxy id="proxy_ssn_routing" type="at.siemens.smb.proxy.impl.
        DynamicRoutingProxy">
      <param name="destination" value="*:heartbeat_dc.*=* -> channel_sv" />
      <param name="destination" value="* -> channel_ctrl1,channel_ctrl2,
         channel_ctrl3,channel_web,channel_logging" />
    </Proxy>
    <Proxy id="proxy_ssn_connector" type="at.siemens.smb.proxy.impl.
        SocketConnectorProxy">
      <param name="port" value="9001" />
      <param name="binary" value="true" />
    </Proxy>
```

```
</Channel>

<!-- Supervisor -->
<Channel id="channel_sv">
  <Proxy id="sv_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy">
    <param name="destination" value="* #openloop -> channel_web,channel_logging
        " />

    <param name="destination" value="* -> channel_ssn,channel_web,
        channel_logging" />
  </Proxy>
  <Proxy id="sv_connector" type="at.siemens.smb.proxy.impl.SocketConnectorProxy
      ">
    <param name="port" value="7000" />
    <param name="binary" value="true" />
  </Proxy>
</Channel>

<!-- Control Algorithm Stage 1 -->
<Channel id="channel_ctrl1">
  <Proxy id="ctrl1_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
      ">
    <param name="destination" value=".* -> channel_sv,channel_logging" />
  </Proxy>
  <Proxy id="ctrl1_streaming" type="at.siemens.smb.proxy.impl.
      ObjectStreamingProxy">
    <param name="context" value="at.siemens.smb.data.packet.Packet" />
    <param name="format" value="JSON" />
  </Proxy>
  <Proxy id="ctrl1_connector" type="at.siemens.smb.proxy.impl.
      SocketConnectorProxy">
    <param name="port" value="7001" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>

<!-- Control Algorithm Stage 2 -->
<Channel id="channel_ctrl2">
  <Proxy id="ctrl2_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
      ">
    <param name="destination" value=".* -> channel_sv,channel_logging" />
  </Proxy>
  <Proxy id="ctrl2_streaming" type="at.siemens.smb.proxy.impl.
      ObjectStreamingProxy">
    <param name="context" value="at.siemens.smb.data.packet.Packet" />
    <param name="format" value="JSON" />
  </Proxy>
  <Proxy id="ctrl2_connector" type="at.siemens.smb.proxy.impl.
      SocketConnectorProxy">
    <param name="port" value="7002" />
    <param name="binary" value="false" />
  </Proxy>
</Channel>

<!-- Control Algorithm Stage 3 -->
<Channel id="channel_ctrl3">
  <Proxy id="ctrl3_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy
      ">
    <param name="destination" value=".* -> channel_sv,channel_logging" />
    </Proxy>
    <Proxy id="ctrl3_streaming" type="at.siemens.smb.proxy.impl.
        ObjectStreamingProxy">
      <param name="context" value="at.siemens.smb.data.packet.Packet" />
      <param name="format" value="JSON" />
    </Proxy>
    <Proxy id="ctrl3_connector" type="at.siemens.smb.proxy.impl.
        SocketConnectorProxy">
      <param name="port" value="7003" />
      <param name="binary" value="false" />
    </Proxy>
  </Channel>

  <!-- Web Proxy -->
  <Channel id="channel_web">
    <Proxy id="web_routing" type="at.siemens.smb.proxy.impl.DynamicRoutingProxy">
      <param name="destination" value="* #command -> channel_ssn,channel_logging"
          />
      <param name="destination" value="* -> channel_logging" />
    </Proxy>
    <Proxy id="web_proxy" type="at.siemens.smb.proxy.impl.WebServiceProxy">
      <param name="maxpacketlogsize" value="10000" />
      <param name="webserverport" value="8080" />
      <param name="www_index" value="login.html" />
      <param name="www_root" value="./resources/login/" />
      <param name="ssl_https_enabled" value="false" />
      <param name="auth_enabled" value="true" />
      <param name="auth_realm" value="./resources/jetty/jetty-auth-realm-ETZ.
        properties" />
      <param name="webapp_war" value="./resources/jetty/exploded_war" />
      <param name="webapp_context" value="/auth/dashboard" />
    </Proxy>
  </Channel>

  <!-- File Logging -->
  <Channel id="channel_logging">
    <Proxy id="logging_filelog" type="at.siemens.smb.proxy.impl.FileLogProxy">
      <param name="file" value="./log/measurements.log" />
      <param name="max_records" value="10000" />
      <param name="max_history" value="10" />
    </Proxy>
  </Channel>

</Configuration>
```

Listing A.7: SMB middleware configuration for prototyping Phase III, Stages (a) and (b)

A.4 PowerFactory Simulation

The following Figure A.1 shows a screen-shot of the Eberstalzell LV distribution grid, that has been simulated in coupled simulation in order to evaluate the performance and suitability of designed control algorithms.

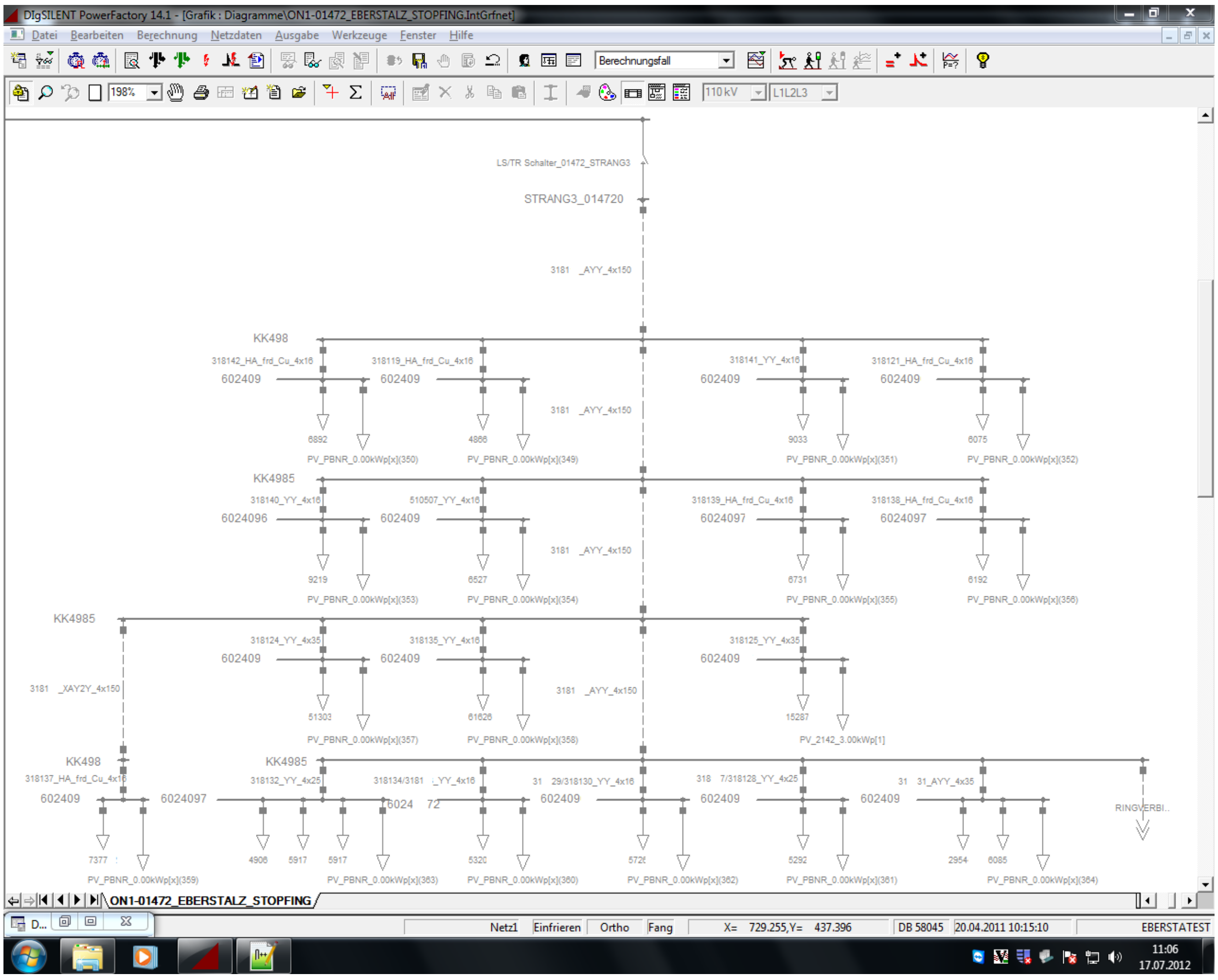
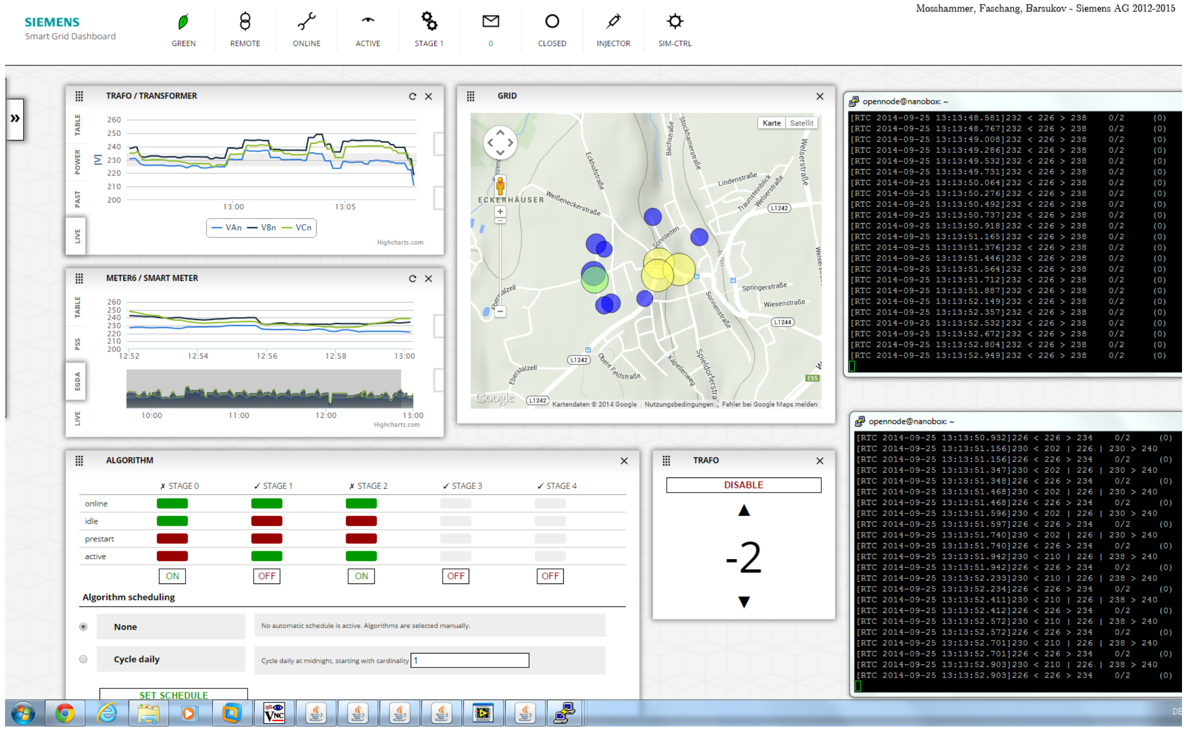


Figure A.1: The power system simulation of the Eberstalzell LV distribution grid. Depicted is the PowerFactory user interface, which shows the graphical representation of one selected branch with bus bars and lines to the consumers.

A.5 User Interface Screenshot



opennode@nanobox: ~

```
[RTC 2014-09-25 13:13:48.581]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:48.767]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:49.008]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:49.286]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:49.532]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:49.731]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:50.064]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:50.276]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:50.492]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:50.737]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:50.918]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:51.165]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:51.376]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:51.446]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:51.564]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:51.712]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:51.887]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:52.149]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:52.357]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:52.532]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:52.672]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:52.804]232 < 226 > 238    0/2      (0)
[RTC 2014-09-25 13:13:52.949]232 < 226 > 238    0/2      (0)
```

opennode@nanobox: ~

```
[RTC 2014-09-25 13:13:50.932]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:51.156]230 < 202 | 226 | 230 > 240
[RTC 2014-09-25 13:13:51.156]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:51.347]230 < 202 | 226 | 230 > 240
[RTC 2014-09-25 13:13:51.348]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:51.468]230 < 202 | 226 | 230 > 240
[RTC 2014-09-25 13:13:51.468]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:51.596]230 < 202 | 226 | 230 > 240
[RTC 2014-09-25 13:13:51.597]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:51.740]230 < 202 | 226 | 230 > 240
[RTC 2014-09-25 13:13:51.740]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:51.942]230 < 210 | 226 | 238 > 240
[RTC 2014-09-25 13:13:51.942]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:52.233]230 < 210 | 226 | 238 > 240
[RTC 2014-09-25 13:13:52.234]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:52.411]230 < 210 | 226 | 238 > 240
[RTC 2014-09-25 13:13:52.412]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:52.572]230 < 210 | 226 | 238 > 240
[RTC 2014-09-25 13:13:52.572]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:52.701]230 < 210 | 226 | 238 > 240
[RTC 2014-09-25 13:13:52.701]226 < 226 > 234    0/2      (0)
[RTC 2014-09-25 13:13:52.903]230 < 210 | 226 | 238 > 240
[RTC 2014-09-25 13:13:52.903]226 < 226 > 234    0/2      (0)
```

Figure A.2: A screenshot of the complete user interface developed for the control prototyping process together with SSH command line listings of the remotely operating control algorithms output.

A.6 Control System Evaluation Results

Figure A.3 and Figure A.4 show the results of the control system that have been created by the rapid control prototyping process in the course of *DG DemoNet – Smart LV Grid* project.

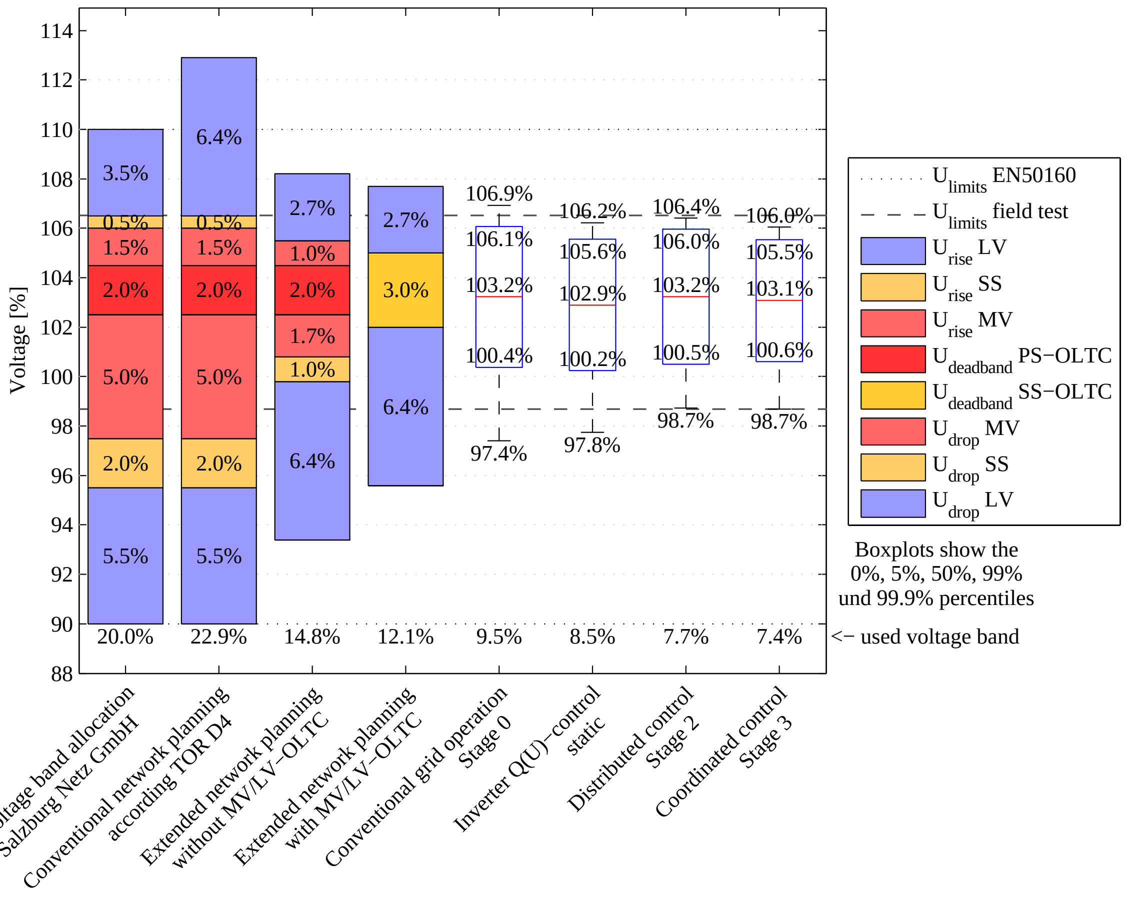


Figure A.3: Evaluation results of the control system operated in the LV distribution grid in Köstendorf/Salzburg [SBB+15]. The first four bar-charts represent the grid operators network planning scenarios and the last three represent voltage band occupation resulting from the control system (Stage 1, 2, and 3) operation in the field (with friendly permission of Roman Schwalbe).

The first bar-chart is the reference scenario and depicts the projected partitioning of the ±10 % voltage band given by EN 50160. It contains voltage band variation sectors for the dead-band of the primary substation's (PS) and secondary substation's (SS) on-load tap-changer (OLTC) and for the voltage variations caused in the medium voltage (MV) and low voltage (LV) band. The partitioning is individual for the respective grid operators (Salzburg Netz GmbH in Figure A.3 and Netz Oberösterreich GmbH in Figure A.4). Furthermore, the EN 50160 voltage limits are marked as well as the voltage limits for the control system, which have been chosen narrower in order to challenge the control system.

The second bar-chart represents the voltage situation in Littring's and Köstendorf's distribution grids according to a TOR D4 [Ene13] assessment, which is also used for conventional worst-case

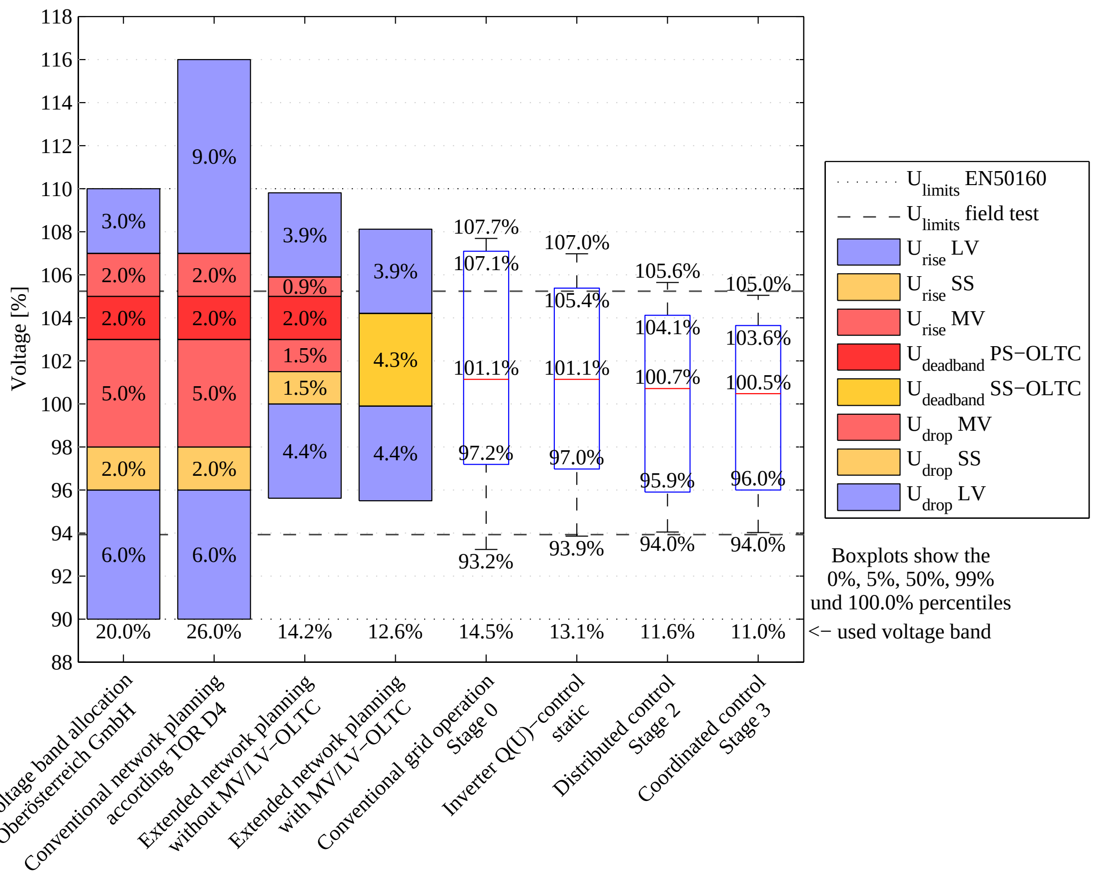


Figure A.4: Evaluation results of the control system operated in the LV distribution grid in Littring/Upper Austria [SBB+15]. The first four bar-charts represent the grid operators network planning scenarios and the last three represent voltage band occupation resulting from the control system (Stage 1, 2, and 3) operation in the field (with friendly permission of Roman Schwalbe).

network planning. The high over-voltage is caused by the assumption that each of the single-phase PV installations is connected to the same phase.

More realistic network planning methods can be applied when monitoring options are available in the grid to be able to assess the real voltage deviation in the overlay MV grid. This leads to the extended network planning voltage band estimations, which are represented by the third and fourth bar-chart. In the fourth bar-chart the influence of MV grid and its components is untended due to the decoupling of MV and LV grid by an OLTC.

The three rightmost bar-charts show five percentiles of the three control strategies' field operation measurements. The percentiles are the 0 %, 5 %, 50 % (median), 99 %, and the 100 % percentiles of all measurements. The numbers below the bar-charts show the voltage band occupation. It can clearly be seen, that in both distribution grids the voltage band occupation is reduced by the use of the control system. The occupied voltage band constantly decreases even further by the use of the more complex control stage.

A.7 Definition of the Technology Readiness Levels

According to the European Commissions Horizon – 2020 Work Programme 2014-2015 [Eur14], the Technology Readiness Levels (TRL) are defined as follows.

TRL 1 basic principles observed

TRL 2 technology concept formulated

TRL 3 experimental proof of concept

TRL 4 technology validated in lab

TRL 5 technology validated in relevant environment
(industrially relevant environment in the case of key enabling technologies)

TRL 6 technology demonstrated in relevant environment
(industrially relevant environment in the case of key enabling technologies)

TRL 7 system prototype demonstration in operational environment

TRL 8 system complete and qualified

TRL 9 actual system proven in operational environment
(competitive manufacturing in the case of key enabling technologies; or in space)

References of Scientific Publications

[AB10] Tansu Alpcan and Tamer Başar. *Network Security: A Decision and Game-Theoretic Approach.* Cambridge University Press, October 2010.

[Aic12] Christian Aichele. *Smart Energy.* Vieweg Verlag, Friedr, & Sohn Verlagsgesellschaft mbH, 2012.

[And12] Göran Andersson. Dynamics and Control of Electric Power System. Technical report, ETH Zürich, February 2012.

[Arm04] Phillip G. Armour. *The Laws of Software Process: A New Model for the Production and Management of Software.* CRC Press, June 2004.

[ASG12] S. Adhikari, F. Schupfer, and C. Grimm. Co-simulation framework for variation analysis of radio frequency transceivers. In *System, Software, SoC and Silicon Debug Conference (S4D), 2012*, pages 1–6, September 2012.

[ASHL13] F. Aalamifar, A. Schlögl, D. Harris, and L. Lampe. Modelling power line communication using network simulator-3. In *2013 IEEE Global Communications Conference (GLOBECOM)*, pages 2969–2974, December 2013.

[ASRU13] F. Andren, T. Strasser, S. Rohjans, and M. Uslar. Analyzing the need for a common modeling language for Smart Grid applications. In *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, pages 440–446, July 2013.

[BA04] Kent Beck and Cynthia Andres. *Extreme Programming Explained: Embrace Change.* Addison-Wesley Professional, November 2004.

[Bec03] Kent Beck. *Extreme Programming: die revolutionäre Methode für Softwareentwicklung in kleinen Teams ; [das Manifest].* Pearson Deutschland GmbH, 2003.

[Bev14] Hassan Bevrani. *Robust Power System Frequency Control.* Springer, 2014.

[BO04] Björn Möller and Lennart Olsson. Practical Experiences from HLA 1.3 to HLA IEEE 1516 Interoperability. In *Proceedings of 2004 Fall Simulation Interoperability Workshop, 04F-SIW-045, Simulation Interoperability Standards Organization*, September 2004.

[Bon14] H. E. G. Bonin. *Systems Engineering in Public Administration: Proceedings of the IFIP TC8/WG8.5 Working Conference on Systems Engineering in Public Administration, Luneburg, Germany, 3-5 March 1993.* Elsevier, May 2014.

[Bru12] Helfried Brunner. DG DemoNet Smart LV Grid - Increasing the DER Hosting Capacity of Distribution Networks - Voltage Control from Simulation to Field Test, April 2012.

[BT04] Barry W. Boehm and Richard Turner. *Balancing Agility and Discipline: A Guide for the Perplexed.* Addison-Wesley Professional, 2004.

[Bun08] Bundesministerium für Verkehr, Innovation und Technologie. Aktiver Betrieb von elektrischen Verteilnetzen mit hohem Anteil dezentraler Stromerzeugung – Konzeption von Demonstrationsnetzen. Projektbericht 13a/2008, Bundesministerium für Verkehr, Innovation und Technologie, Wien, January 2008.

[Bun09] Bundesministeriums der Justiz (Deutschland). Gesetz für den Vorrang Erneuerbarer Energien (Erneuerbare-Energien-Gesetz - EEG), January 2009.

[Bun12] Bundeskanzleramt Österreich. Bundesgesetz über die Förderung der Elektrizitätserzeugung aus erneuerbaren Energieträgern (Ökostromgesetz 2012 - ÖSG 2012), July 2012.

[CEN11] CENELEC. EN50160:2010 - Voltage Characteristics in Public Distribution Systems. Technical report, March 2011.

[CEN12] CEN-CENELEC-ETSI. Smart Grid Reference Architecture. Technical Report, CEN-CENELEC-ETSI Smart Grid Coordination Group, November 2012.

[CK06] François E. Cellier and Ernesto Kofman. *Continuous System Simulation.* Springer, New York, 2006 edition, April 2006.

[CL13] Pascal Cantot and Dominique Luzeaux. *Simulation and Modeling of Systems of Systems.* John Wiley & Sons, March 2013.

[CLL10] Chee Kai Chua, Kah Fai Leong, and Chu Sing Lim. *Rapid Prototyping: Principles and Applications.* World Scientific, 2010.

[Cob11] Charles G. Cobb. *Making Sense of Agile Project Management: Balancing Control and Agility.* John Wiley & Sons, February 2011.

[Cra07] Valentin Crastan. *Elektrische Energieversorgung 1.* Springer, Berlin; New York, 2007.

[Cra12] Valentin Crastan. *Elektrische Energieversorgung 2.* Springer-Verlag Berlin Heidelberg, 2012.

[CW12] Valentin Crastan and Dirk Westermann. *Elektrische Energieversorgung 3.* Springer-Verlag Berlin Heidelberg, 2012.

[CWN$^+$12] Seung Tae Cha, Qiuwei Wu, A.H. Nielsen, J. Ostergaard, and In Kwon Park. Real-Time Hardware-In-The-Loop (HIL) Testing for Power Electronics Controllers. In *Power and Energy Engineering Conference (APPEEC), 2012 Asia-Pacific*, pages 1–6, March 2012.

[DBBVdK$^+$07] K.. De Brabandere, Bruno Bolsens, J.. Van den Keybus, A. Woyte, J. Driesen, and R. Belmans. A Voltage and Frequency Droop Control Method for Parallel Inverters. *IEEE Transactions on Power Electronics*, 22(4):1107–1115, July 2007.

[DFW97] Judith S. Dahmann, Richard M. Fujimoto, and Richard M. Weatherly. The Department of Defense High Level Architecture. In *Proceedings of the 29th conference on Winter simulation*, WSC '97, pages 142–149, Washington, DC, USA, 1997. IEEE Computer Society.

[DNR$^+$14] Christian Dänekas, Christian Neureiter, Sebastian Rohjans, Mathias Uslar, and Dominik Engel. Towards a Model-Driven-Architecture Process for Smart Grid Projects. In Pierre-Jean Benghozi, Daniel Krob, Antoine Lonjon, and Hervé Panetto, editors, *Digital Enterprise Design & Management*, number 261 in Advances in Intelligent Systems and Computing, pages 47–58. Springer International Publishing, 2014.

[Dro04] Sven Dronka. *Die Simulation gekoppelter Mehrkörper- und Hydraulik-Modelle mit Erweiterung für Echtzeitsimulation.* Shaker, 2004.

[EKBL12] A. Einfalt, F. Kupzog, H. Brunner, and A. Lugmaier. Control strategies for smart low voltage grids - The project DG DemoNet - Smart LV Grid. In *Integration of*

Renewables into the Distribution Grid, CIRED 2012 Workshop, pages 1–4, May 2012.

[Ene13] Energie-Control GmbH. Technische und organisatorische Regeln für Betreiber und Benutzer von Netzen - Hauptabschnitt D4: Parallelbetrieb von Erzeugungsanlagen mit Verteilernetzen. Technical report, E-Control, Wien, 2013.

[EP09] Rat Europäisches Parlament. Richtlinie 2009/28/EG des Europäischen Parlaments und des Rates vom 23. April 2009 zur Förderung der Nutzung von Energie aus erneuerbaren Quellen und zur Änderung und anschließenden Aufhebung der Richtlinien 2001/77/EG und 2003/30/EG, June 2009.

[EPR12] EPRI - Electric Power Research Institute. IntelliGrid Smart Grid Roadmap Methodology and Lessons Learned. Technical Update 1026747, EPRI - Electric Power Research Institute - Power Delivery & Utilization, December 2012.

[EUD12] Directive 2012/27/EU of the European Parliament and of the Council of 25 October 2012 on energy efficiency, amending Directives 2009/125/EC and 2010/30/EU and repealing Directives 2004/8/EC and 2006/32/EC. Official Journal of the European Union, 2012.

[eur11] Standardization mandate to support European Smart Grid deployment (M/490), March 2011.

[Eur13] European Environment Agency, editor. *Annual European Union greenhouse gas inventory 1990-2011 and inventory report 2013*. 2013.

[Eur14] European Commission. Horizon 2020 – Work Programme 2014 - 2015 (European Commission Decision C (2014)4995 of 22 July 2014). Technical report, European Commission, June 2014.

[EZS+13] A. Einfalt, F. Zeilinger, R. Schwalbe, B. Bletterie, and S. Kadam. Controlling active low voltage distribution grids with minimum efforts on costs and engineering. In *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*, pages 7456–7461, November 2013.

[FDL+13] Mario Faschang, Pavlos Dimitriou, Thomas Leber, Friederich Kupzog, Ralf Mosshammer, and Roman Schwalbe. DG DemoNet Smart LV Grid - Deliverable 2.1: Description of the final Co-simulation environment. Technical report, Vienna, July 2013. unpublished.

[FESM14] M. Faschang, A. Einfalt, R. Schwalbe, and R. Mosshammer. Controller hardware in the loop approaches supporting rapid prototyping of smart low voltage grid control. In *Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), 2014 IEEE PES*, pages 1–5, October 2014.

[FFG12a] FFG. Beitrag zum aktiven Verteilernetzbetrieb durch Innovative Spannungsregelung – BAVIS. Publizierbarer Endbericht, Klima- und Energiefonds - Österreichische Forschungsförderungsgesellschaft mbH (FFG), February 2012.

[FFG12b] FFG. Smart Grids Modellregion Salzburg: Zentrale Spannungs- (U) und Blindleistungsregelung (Q) mit dezentralen Einspeisungen in der Demoregion Salzburg. Publizierbarer Endbericht 825468, Klima- und Energiefonds - Österreichische Forschungsförderungsgesellschaft mbH (FFG), 2012.

[FFG13a] FFG. Aktiver Betrieb von elektrischen Verteilnetzen mit hohem Anteil dezentraler Stromerzeugung – Validierung von Spannungsregelungskonzepten. Endbericht, Klima- und Energiefonds - Österreichische Forschungsförderungsgesellschaft mbH (FFG), 2013.

[FFG13b] FFG. DG DemoNetz Validierung - Deliverable D6: Funktionale Beschreibung einer Kommunikationsplattform für Spannungsregelungskonzepte. Deliverable, Klima- und Energiefonds - Österreichische Forschungsförderungsgesellschaft mbH (FFG), 2013.

[FFG15] FFG. DG DemoNet Smart LV Grid - Final Report / Endbericht. Endbericht, Klima- und Energiefonds - Österreichische Forschungsförderungsgesellschaft mbH (FFG), Vienna, 2015. (to be published).

[FSK15] Mario Faschang, Roman Schwalbe, and Friederich Kupzog. Experimental Sensitivity Analysis of Low Voltage Control Strategies on Communication Properties. In *PowerTech (POWERTECH), 2015 IEEE Eindhoven*, Eindhoven, July 2015.

[GH14] Rune Gustavsson and Shahid Hussain. The Proper Role of Agents in Future Resilient Smart Grids. In Juan M. Corchado, Javier Bajo, Jaroslaw Kozlak, Pawel Pawlewski, Jose M. Molina, Benoit Gaudou, Vicente Julian, Rainer Unland, Fernando Lopes, Kasper Hallenborg, and Pedro García Teodoro, editors, *Highlights of Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection*, number 430 in Communications in Computer and Information Science, pages 226–237. Springer International Publishing, 2014.

[GKL06] M. Geimer, T. Krüger, and P. Linsel. Co-Simulation, gekoppelte Simulation oder Simulatorkopplung? Ein Versuch der Begriffsvereinheitlichung. 2006.

[GRSW12] Reinhard Grünwald, Mario Ragwitz, Frank Sensfuß, and Jenny Winkler. *Regenerative Energieträger zur Sicherung der Grundlast in der Stromversorgung.* Büro für Technikfolgen-Abschätzung beim Deutschen Bundestag (TAB), Berlin, 2012.

[GSO07] J. Duncan Glover, Mulukutla Sarma, and Thomas Overbye. *Power Systems Analysis and Design.* Cengage Learning, May 2007.

[GWP$^+$14] Christopher Greer, David A. Wollman, Dean E. Prochaska, Paul A. Boynton, Jeffrey A. Mazer, Cuong T. Nguyen, Gerald J. FitzPatrick, Thomas L. Nelson, Galen H. Koepke, Allen R. Hefner Jr, Victoria Y. Pillitteri, Tanya L. Brewer, Nada T. Golmie, David H. Su, Allan C. Eustis, David G. Holmberg, and Steven T. Bushby. NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 3.0. Technical Report NIST SP 1108r3, National Institute of Standards and Technology, October 2014.

[HH11] Roy M. Harrison and Ronald E. Hester. *Nuclear Power and the Environment.* Royal Society of Chemistry, 2011.

[HJD10] Elizabeth Hull, Ken Jackson, and Jeremy Dick. *Requirements Engineering.* Springer Science & Business Media, October 2010.

[HM14] Jahangir Hossain and Apel Mahmud. *Large Scale Renewable Power Generation: Advances in Technologies for Generation, Transmission and Storage.* Springer Science & Business Media, January 2014.

[IEC86] IEC - International Electrotechnical Commission. IEC 868 - Flickermeter. Functional and design specifications. Technical report, 1986.

[IEC08] IEC - International Electrotechnical Commission. IEC/PAS 62559 - IntelliGrid Methodology for Developing Requirements for Energy Systems. Publicly available specification IEC/PAS 62559, IEC / EPRI, January 2008.

[IEC10] IEC - International Electrotechnical Commission. IEC 61000-4-15 Electromagnetic compatibility (EMC) - Part 4-15: Testing and measurement techniques - Flickermeter - Functional and design specifications. Technical report, 2010.

[IEE97] IEEE. Proposed terms and definitions for flexible AC transmission system (FACTS). *IEEE Transactions on Power Delivery*, 12(4):1848–1853, October 1997.

[IEE10] IEEE. IEEE 1516-2010 - IEEE Standard for Modeling and Simulation (M&S)

High Level Architecture (HLA)– Framework and Rules. Technical Report 1516, IEEE, 2010.

[ISO09] ISO - International Organization for Standardization. ISO 31000:2009 Risk management – Principles and guidelines. Norm, International Organization for Standardization - ISO/TC 262, 2009.

[JRMRM12] Javier Juárez, Carlos Rodríguez-Morcillo, and José Antonio Rodríguez-Mondéjar. Simulation of IEC 61850-based Substations Under OMNeT++. In *Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques*, SIMUTOOLS '12, pages 319–326, ICST, Brussels, Belgium, Belgium, 2012. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[JSA$^+$14] P. Jonke, B. Sumanta, F. Andren, J. Stockl, and T. Strasser. Rapid prototyping of distributed energy resources integrating ICT and power electronics design. In *IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*, pages 3591–3597, October 2014.

[KHV$^+$11] M. Korki, N. Hosseinzadeh, H.L. Vu, T. Moazzeni, and Chuan Heng Foh. A channel model for power line communication in the smart grid. In *Power Systems Conference and Exposition (PSCE), 2011 IEEE/PES*, pages 1–7, March 2011.

[KLS$^+$14] Markus Kammerstetter, Lucie Langer, Florian Skopik, Friederich Kupzog, and Wolfgang Kastner. Practical Risk Assessment Using a Cumulative Smart Grid Model. pages 31–42, April 2014.

[KML15] Siddhartha Kumar Khaitan, James D. McCalley, and Chen Ching Liu. *Cyber Physical Systems Approach to Smart Electric Power Grid.* Springer, March 2015.

[KP11] Friederich Kupzog and Peter Palensky. Smart energy distribution. In *The Industrial Electronics Handbook*, pages 36–1–36–8. CRC Press, 2011. invited.

[KPA$^+$04] P. Kundur, J. Paserba, V. Ajjarapu, G. Andersson, A. Bose, C. Canizares, N. Hatziargyriou, D. Hill, A. Stankovic, C. Taylor, T. Van Cutsem, and V. Vittal. Definition and classification of power system stability IEEE/CIGRE joint task force on stability terms and definitions. *IEEE Transactions on Power Systems*, 19(3):1387–1401, August 2004.

[Kun94] Prabha Kundur. *Power System Stability and Control.* McGraw-Hill Education, January 1994.

[LAH11] V. Liberatore and A. Al-Hammouri. Smart grid communication and co-simulation. In *2011 IEEE Energytech*, pages 1–5, May 2011.

[Lun04] Jan Lunze. *Regelungstechnik 2 - Mehrgrößensysteme, Digitale Regelung.* Springer, 3. auflage edition, September 2004.

[MAL09] RAJIB MALL. *Fundamentals of Software Engineering.* PHI Learning Pvt. Ltd., May 2009.

[MF12] Ralf Mosshammer and Mario Faschang. DG DemoNet Smart LV Grid - WP2 Simulation Message Bus User Manual. Technical report, Siemens AG Österreich, Vienna, 2012. unpublished.

[MKFS13] R. Mosshammer, F. Kupzog, M. Faschang, and M. Stifter. Loose coupling architecture for co-simulation of heterogeneous components. In *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*, pages 7570–7575, November 2013.

[MOD14] K. Mets, J.A. Ojea, and C. Develder. Combining Power and Communication Network Simulation for Cost-Effective Smart Grid Analysis. *IEEE Communications Surveys Tutorials*, 16(3):1771–1796, 2014.

[MW03] Frank Maurer and Don Wells. *Extreme Programming and Agile Methods - XP/Agile Universe 2003: Third XP and Second Agile Universe Conference, New Orleans, LA, USA, August 10-13, 2003, Proceedings.* Springer Science & Business Media, July 2003.

[Nö14] Martin Nöhrer. *Flexible Co-Simulationsumgebung zum Testen von Ladeinfrastrukturen für Elektromobilität.* Diploma Thesis, Vienna University of Technology, Vienna, 2014.

[Nat13] National Instruments. Whitepaper: Hardwareintegration in NI LabVIEW. Whitepaper, July 2013.

[NKM+07] J. Nutaro, P.T. Kuruganti, L. Miller, S. Mullen, and M. Shankar. Integrated Hybrid-Simulation of Electric Power and Communications Systems. In *IEEE Power Engineering Society General Meeting, 2007*, pages 1–8, June 2007.

[NLS14] M. Nohrer, F. Lehfuss, and J. Stockl. Flexible test system architecture for electric vehicle charging infrastructure. In *Electric Vehicle Conference (IEVC), 2014 IEEE International*, pages 1–7, December 2014.

[OO11] Dietrich Oeding and Bernd R. Oswald. *Elektrische Kraftwerke und Netze.* Springer DE, January 2011.

[Pac12] Pacific Northwest National Laboratory. GridLAB-D – A Unique Tool to Design the Smart Grid. Brochure, Pacific Northwest National Laboratory, November 2012.

[PD11] P. Palensky and D. Dietrich. Demand Side Management: Demand Response, Intelligent Energy Systems, and Smart Loads. *IEEE Transactions on Industrial Informatics*, 7(3):381–388, August 2011.

[PM12] Katalin Popovici and Pieter J. Mosterman. *Real-Time Simulation Technologies: Principles, Methodologies, and Applications.* CRC Press, August 2012.

[Reb02] Eckhard Rebhan. *Energiehandbuch: Gewinnung, Wandlung und Nutzung von Energie.* Springer DE, May 2002.

[Rei99] Gunther Reinhart. *Rapid prototyping: Methoden für die reaktionsfähige Produktentwicklung ; Augsburg, 13. Oktober 1999.* Herbert Utz Verlag, 1999.

[RLS+14] S. Rohjans, S. Lehnhoff, S. Schuette, Filip Andren, and T. Strasser. Requirements for Smart Grid simulation tools. In *2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)*, pages 1730–1736, June 2014.

[Rub12] Kenneth S. Rubin. *Essential Scrum: A Practical Guide to the Most Popular Agile Process.* Addison-Wesley, July 2012.

[SB09] Bernhard Schweizer and Martin Busch. Numerische Ansätze zur gekoppelten Simulation, November 2009. FLUIDON-Konferenz - Simulationen im mechatronischen Umfeld - Aachen.

[SBB+12] Matthias Stifter, Benoit Bletterie, Helfried Brunner, Daniel Burnier de Castro, Sawsan Henein, Filip Andren, Roman Schwalbe, Werner Tremmel, A. Abart, Reinhard Nenning, Frank Herb, and R. Pointner. Dg demonetz validierung: Innovative spannungsregelung von der simulation zum feldtest. In *12. Symposium Energieinnovation*, pages 1–8, Graz, 2012. TU Graz.

[SBB+15] Schwalbe, Brunner, Bletterie, Abart, Traxler, Radauer, and Niederhuemer. DG-DemoNet Smart LV Grid – Extending hosting capacity of LV grids by advanced planning and voltage control. In *Proceedings of 2015 International Symposium on Smart Electric Distribution Systems and Technologies (EDST)*, Vienna, August 2015. (to be published).

[SBKK12] Florian Skopik, Thomas Bleier, Markus Kammerstetter, and Georg Kienesberger.

Smart grid security guidance: Eine sicherheitsinitiative für intelligente stromnetze. In *Proceedings of the 42. annual conference of the German computer society*. Springer Lecture Notes in Infomratics, 2012. Vortrag: 42. Jahrestagung der Gesellschaft für Informatik e.V. (GI) (INFORMATIK 2012), Braunschweig, D; 2012-09-16 – 2012-09-21.

[SBN10] L. Snider, J. Bélanger, and G. Nanjundaiah. Today's power system simulation challenge: High-performance, scalable, upgradable and affordable COTS-based real-time digital simulators. In *2010 Joint International Conference on Power Electronics, Drives and Energy Systems (PEDES) 2010 Power India*, pages 1–10, December 2010.

[SEH+15] Schwalbe, Einfalt, Heidl, Abart, Radauer, and Brunner. DG-DemoNet Smart LV Grid – Robust Control Architecture to Increase DER Hosting Capacity. In *Proceedings of CIRED - 23rd International Conference on Electricity Distribution*, Lyon, June 2015.

[Sei14] Christian Seitl. *Entwicklung eines echtzeitfähigen Batteriemodells für Power Hardware-in-the-Loop Simulationen.* 2014.

[Sha02] Mary Shaw. What makes good research in software engineering? *International Journal on Software Tools for Technology Transfer*, 4(1):1–7, October 2002.

[SIE11] SIEMENS AG. SIEMENS AMIS CX1-Profil (Compatibly/Consistently Extendable Transport Profile V.1) Layer 1-4. Technical Report D20-002-1.00, 2011.

[SMB10] SMB Smart Grid Strategic Group (SG3). IEC Smart Grid Standardization Roadmap. Technical Report Ed. 1.0 - 2009-12, International Electrotechnical Commission, June 2010.

[Sol82] Henk G. Sol. *Simulation in Information Systems Development.* Krips Repro, 1982.

[SPR88] Stanley R. Sadin, Frederick P. Povinelli, and Robert Rosen. The NASA technology push towards future space mission systems. October 1988.

[SS12a] Stefan Scherfke and Steffen Schütte. mosaik – Architecture Whitepaper. Whitepaper, OFFIS, Oldenburg, 2012.

[SS12b] S. Schutte and M. Sonnenschein. Mosaik - Scalable Smart Grid scenario specification. In *Simulation Conference (WSC), Proceedings of the 2012 Winter*, pages 1–12, December 2012.

[SSAS13] M. Stifter, R. Schwalbe, F. Andren, and T. Strasser. Steady-state co-simulation with PowerFactory. In *2013 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES)*, pages 1–6, May 2013.

[SSKD11] T. Sauter, S. Soucek, W. Kastner, and D. Dietrich. The Evolution of Factory and Building Automation. *IEEE Industrial Electronics Magazine*, 5(3):35–48, September 2011.

[SSS12] Steffen Schütte, Stefan Scherfke, and Michael Sonnenschein. Mosaik - Smart Grid Simulation API - Toward a Semantic based Standard for Interchanging Smart Grid Simulations. In Brian Donnellan, João A. Peças Lopes, João Martins, and Joaquim Filipe, editors, *SMARTGREENS*, pages 14–24. SciTePress, 2012.

[SST11] S. Schutte, S. Scherfke, and M. Troschel. Mosaik: A framework for modular simulation of active components in Smart Grids. In *2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS)*, pages 55–60, October 2011.

[SWA+13] Matthias Stifter, Edmund Widl, Filip Anren, Atiyah Elsheikh, Thomas Strasser, and Peter Palensky. Co-Simulation of Components, Controls and Power Systems based on Open Source Software. In *General Meeting, Annual Conference on IEEE Power and Energy Society*, 2013.

[VH08] András Varga and Rudolf Hornig. An Overview of the OMNeT++ Simulation Environment. In *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, pages 60:1–60:10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[Way13] Adrian Waygood. *An Introduction to Electrical Science*. Routledge, June 2013.

[WCT+11] Magnus Westerlund, Stuart Cheshire, Joseph Touch, Michelle Cotton, and Lars Eggert. RFC6335 - Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Transport Protocol Port Number and Service Name Registry. Request for Comments RFC6335, IETF, August 2011.

[WdWP+12] Dan Wang, B. de Wit, S. Parkinson, J. Fuller, D. Chassin, C. Crawford, and N. Djilali. A test bed for self-regulating distribution systems: Modeling integrated renewable energy and demand response in the -D/MATLAB environment. In *Innovative Smart Grid Technologies (ISGT), 2012 IEEE PES*, pages 1–7, 2012.

[Wea13] E. Widl and et al. The FMI++ Library: A High-level Utility Package for FMI for Model Exchange. In *MSCPES: Modeling and Simulation of Cyber-Physical Energy Systems*, Berkely, USA, 2013.

[WHOB06] W. Weichselberger, M. Herdin, H. Ozcelik, and E. Bonek. A stochastic MIMO channel model with joint correlation of both link ends. *IEEE Transactions on Wireless Communications*, 5(1):90–100, January 2006.

Internet References

[1] Agile Alliance. *Manifesto for Agile Software Development*, visited 2015-01-26. http://www.agilemanifesto.org/.

[2] Bundesnetzagentur für Elektrizität, Gas, Telekommunikation, Post und Eisenbahnen. *Netzausbau - Netzentwicklungsplaene und Umweltbericht*, visited 2014-02-17. http://www.netzausbau.de/cln_1932/DE/Verfahren/NEP-UB/NEP-UB-node.html.

[3] DAI-Labor as part of Fakultät IV Elektrotechnik und Informatik at the Technische Universität Berlin. *NeSSi2*, visited 2015-03-05. http://www.nessi2.de/.

[4] Deutsches Bundesministerium für Wirtschaft und Energie. *Arbeitsgruppe Erneuerbare Energien-Statistik (AGEE-Stat)*, visited 2015-05-27. http://www.bmwi.de/DE/Themen/Energie/Energiedaten-und-analysen/arbeitsgruppe-erneuerbare-energien-statistik.html.

[5] The Eclipse Foundation. *Jetty - Servlet Engine and Http Server*, visited 2015-01-26. http://eclipse.org/jetty/.

[6] Energie-Control Austria. *E-Control Austria – Ökostrom-Einspeisemengen und Vergütungen*, visited 2015-05-27. http://www.e-control.at/de/statistik/oeko-energie/oekostrommengen.

[7] FMI development group. *FMI — Functional Mock-up Interface*, visited 2015-03-07. http://www.fmi-standard.org.

[8] IEC - International Electrotechnical Commission. *IEC 61850: Power Utility Automation*, visited 2014-02-13. http://www.iec.ch/smartgrid/standards/.

[9] NS-3 Consortium. *ns-3 project*, visited 2015-03-05. http://www.nsnam.org/.

[10] OFFIS e.V. Oldenburg - Germany. *mosaik — A flexible Smart Grid co-simulation framework*, visited 2015-03-07. http://mosaik.offis.de/.

[11] OPAL-RT TECHNOLOGIES Inc. *RTS HIL PHIL RCP — OPAL-RT, World simulation technology leader*, visited 2015-03-09. http://www.opal-rt.com/.

[12] OpenADR Alliance. *OpenADR*, visited 2014-02-11. http://www.openadr.org/.

[13] Oracle Corporation. *Jersey*, visited 2015-01-26. https://jersey.java.net/.

[14] Regatron AG Switzerland. *Regatron.com - Home*, visited 2015-03-08. http://www.regatron.com/.

[15] RTDS Technologies Inc. *RTDS Technologies Inc. — Real Time Digital Power System Simulation*, visited 2015-03-09. http://www.rtds.com/.

[16] SmartGrids - European Technology Platform. *European technology platform for the electricity networks of the future*, visited 2015-01-24. http://www.smartgrids.eu/.

[17] Team SimPy. *SimPy 3.0.7 documentation*, visited 2015-03-07. http://simpy.readthedocs.org.

[18] United Nations - Framework Convention on Climate Change. *Kyoto Protocol - United Nations Framework Convention on Climate Change*, visited 2013-12-26. `http://unfccc.int/resource/docs/convkp/kpeng.pdf`.

[19] U.S. Department of Energy - GridLAB-D Simulation Software. *GridLAB-D Simulation Software*, May visited 2013-11-10. `http://www.gridlabd.org/`.

Curriculum Vitae

Personal Information

Name	Mario Faschang
Date of birth	April 23, 1985 in Braunau am Inn, Austria
Nationality	Austria
Email	`mario.faschang@gmx.at` `mario.faschang.at@ieee.org`

Education

since 2011/07	PhD student at Vienna University of Technology
2009–2011	Master programme Computer Technology at Vienna University of Technology
2009	Aeronautics and Telecomunication at Universitat Politècnica de Catalunya
2005–2009	Bachelor programme Electrical Engineering and Information Technology at Vienna University of Technology
2000–2004	Federal Higher Technical Institute, Braunau am Inn, Austria

Professional Experience

since 2014/12	R&D engineer at AIT–Austrian Institute of Technology, Energy Department
2015/01-03	Visiting scientist at Carl von Ossietzky University of Oldenburg, OFFIS Institute
2012–2014	R&D engineer at SIEMENS AG Austria, Corporate Technology
2011–2014	Research assistant at Vienna University of Technology, Institute of Computer Technology

Selected Research Projects

since 2014/12	SC Demo Aspern – ICT-integration for smart buildings and smart grids involving social and municipality aspects in Aspern
2014	ICT4RobustGrid – ICT requirements for operation of advanced and robust smart grids
2012–2014	SIEMENS CT Test Facility "Intelligent Low Voltage Grid"
2011–2014	DG DemoNet-Smart LV Grid – Control concepts for active low voltage network operation with a high share of distributed energy resources
2011	SGMS - V2G-Interfaces – Smart Grids Modellregion Salzburg: Erstellung eines Umsetzungsplans zur Vehicle-to-Grid Interfaceentwicklung

Selected Publications

- M. Faschang, R. Schwalbe, F. Kupzog: *Sensitivity Analysis of Low Voltage Control Strategies on Communication Properties*; for 2015 IEEE PowerTech Eindhoven (accepted – publication in progress), 29 June-2 July 2015 Eindhoven, NL
- M. Faschang, S. Rohjans, F. Kupzog, E. Widl, S. Lehnhoff: *Requirements for Real-Time Hardware Integration into Cyber-Physical Energy System Simulation*; for 2015 Workshop on Modeling and Simulation of Cyber-Physical Energy Systems (MSCPES), 2015 IEEE IES, pp.47-52, 13-16 Apr. 2015, Seattle, WA, USA
- F. Faschang, A. Einfalt, R. Schwalbe, R. Mosshammer: *Controller Hardware in the Loop Approaches Supporting Rapid Prototyping of Smart Low Voltage Grid Control*; for Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), 2014 IEEE PES, pp.1,5, 12-15 Oct. 2014
- M. Faschang, M. Nöhrer, J. Stöckl, F. Kupzog: *Extensible Co-Simulation Framework for Electric Vehicle Charging Infrastructure Testing*; in Proceedings of IEEE Smart Grid Communications (SmartGridComm) 2014, pp.182,187, 3-6 Nov. 2014
- M. Faschang, F. Kupzog, R. Mosshammer, A. Einfalt: *Rapid Control Prototyping Platform for Networked Smart Grid Systems*; in Proceedings of IECON 2013 – 39th Annual Conference of the IEEE Industrial Electronics Society, (2013)
- R. Mosshammer, F. Kupzog, M. Faschang, M. Stifter: *Loose Coupling Architecture for Co-Simulation of Heterogeneous Components - Support of Controller Prototyping for Smart Grid Applications*; in: IECON 2013 – 39th Annual Conference of the IEEE Industrial Electronics Society, (2013)
- M. Faschang, F. Kupzog: *Interfacing Vehicle Charging Systems with User and Grid Requirements*; in Journal Informatik-Spektrum, 36 (2013), 1; pp.27-34
- M. Faschang, F. Kupzog, P. Dimitriou, et al.: *Co-Simulation of Power- and Communication- Networks for Low Voltage Smart Grid Control*; in Proceedings of the Smart Grids Week - Bregenz 2012", (2012)