ICT Emulation Platform Setup Demonstration of Smart Grid Component Prototype Examples

Marcus Meisel*, Stefan Wilker*, Matthias Wess*, Alexander Wendt*, Thilo Sauter*† and Georg Kienesberger*
*TU Wien – Institute of Computer Technology
Gußhausstraße 27-29, 1040 Vienna, Austria
Email: {first.lastname}@tuwien.ac.at
†Danube University Krems – Center for Integrated Sensor Systems
Viktor Kaplan Str. 2 E, 2700 Wiener Neustadt, Austria
Email: {first.lastname}@donau-uni.ac.at

***Abstract*—The shift towards massively distributed energy generation demands more decentralized flexibility to meet strict power quality constraints of the electric grid. A cyber-physical system such as a smart grid can provide increased flexibility by utilizing additional information and communication technologies to better monitor the medium and low voltage distribution networks and to actively control grid-connected resources, ranging from loads to distributed generation, to electric mobility but at the cost of increased complexity. Essential future functionalities such as dynamic management of line use, fault detection and fast service restoration are only possible with appropriate sensors and actuators in place. These missing sensors and actuators on the distribution level are being developed today. This paper presents a standards based, low cost, open source, ICT emulation platform setup to test necessary networking concepts of these smart grid component prototypes already in various stages of development. Preliminary development results of the first example applications chosen: Customer Energy Management System, Smart Breaker, and Smart Meter, are shown in this work in progress paper.**

I. Introduction

Future smart grids are not just the current electrical system with some additional communication, but rather an ever growing, woven interconnection of devices, applications, in multiple domains, zones, and layers, incorporating new and existing protocols, data, functions, and stakeholders within a cyber-physical system that includes consumers, storage processes, distributed and centralized generation, new markets, and current and new power grid and IT-infrastructure. The present shoot up in volatile distributed generation, demands a similar distribution grid level increase of capacity management flexibility, to handle natural fluctuations of renewable energy sources while postponing high investments in grid reinforcements [1]. The challenges and complexity of the newly possible use cases and their combinations in different domains, zones, and layers (see Fig. 1) of the electric grid, demand interdisciplinary research teams and partners who also "by design" keep in mind the criticality of the infrastructures, new applications will operate on. After analyzing the advantages and disadvantages of two candidate realization frameworks, a system outline including new sensors and components in development right now [2] is described. This paper continues to show a low cost information- and communication technology (ICT) platform setup to emulate the communication infrastructure to develop, test, and validate innovative smart distribution grids sensor and actuator components already in various prototype stages. The paper concludes with first results of developed example applications and steps taken towards development of IT and communication network concepts and components, to provide new smart grid functionalities on the way towards enabling distributed flexibilities.

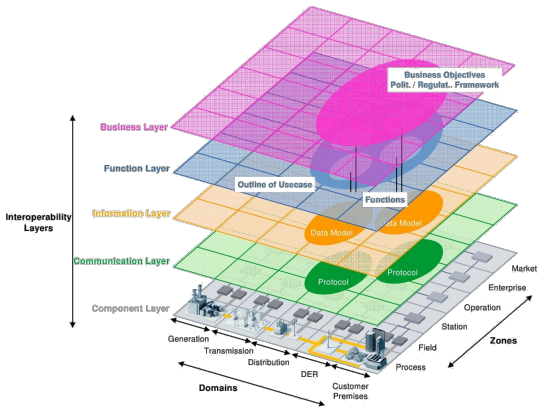

Fig. 1. Smart Grid Architecture Model (SGAM) by CEN-CENELEC-ETSI [3]

A. State of the Art

The following section presents two suggested frameworks for implementation of a Customer Energy Management System (CEMS) on exemplary control hardware.

1) Open Multi Utility Communication (OpenMUC): The OpenMUC framework [4], developed by Fraunhofer Institute for Solar Energy Systems, is licensed under General Public Licence (GPL). It is a JAVA based framework simplifying development of customized monitoring, logging, and controlling systems by the provision of already integrated communication protocol drivers, mostly from the energy domain. Since the Open Services Gateway initiative (OSGi)-plattform is the base of the framework, so called bundles can be easily installed and removed at runtime. Therefore developed systems are lightweight and customized. The OpenMUC architecture shown in [4] is built on a *Data Manager*, which is the central

element receiving and sending data to other independently running optional modules. There are four different categories of bundles.

1) Drivers describe the various communication protocols,
2) data loggers, available in ASCII and binary,
3) web user interfaces conveniently visualizing data, and
4) servers, for remote applications (e.g. smart phone/cloud apps) or local non-java applications.

OpenMUC bundles can be set up platform independent, since the framework is based on JAVA and OSGi [4].

2) Open Gateway Energy Management: Open Gateway Energy Management (OGEMA) 2.0 was developed as a result of the OGEMA and OpenMUC research by the Fraunhofer Institutes involved in the development of the predecessors. Now under General Public License (GPL) published software is designed as a gateway between user and smart grid [5]. Similar to OpenMUC, OGEMA uses the OSGi platform running on a JAVA virtual machine (JAVA VM) and therefore offers the same advantages concerning customization and adaptation to the final system. Other than in OpenMUC, OGEMA 2.0 categorizes installable applications in following layers:

1) Services by central services for administration via web,
2) applications for management and analysis systems,
3) resources defining data models of devices, and
4) communication drivers as lowest layer for device communication.

Compared to OpenMUC, the framework architecture shown on (http://www.ogema.org), has more available applications, a more capable architecture due to its layered structure, in which access to certain resources has to be permitted and approved by the administrator. This makes it more suitable for safety applications. OpenMUC on the other hand is a more lightweight software, securing access through a broad variety of communication protocols and simple data logging functions.

3) Integrating Components in a Secure Information/Automation Architecture: Any technical development in a smart grids context has to take into account the context of current research and development of communication architectures and system interfaces for smart grids, as well as the particular requirements on safety and security. Therefore, on the component and networking side, integration of sensors and actuators into distribution grids cannot be done by simply using off-the-shelf (OTS) equipment. This paper describes a platform setup of OTS devices with a bandwidth limiter and HTTP/SSL/HTTPS traffic monitor installed, that can be used during development as a low cost ICT emulation platform to ensure the quality of each device's function. Smart grid component functions span over multiple elements (existing and new sensors, existing controllers, existing and new actuators in distribution and customer domain) because of which testing communication in varying flavors is a critical aspect. New functionalities enabled by sensors and actuators discussed in this paper are:

1) protection functions for downstream system components (e.g., power consumers, installation facilities) or humans in case of a fault, comparable to circuit breakers,
2) power management by intentionally switching ON/OFF related power branches, and
3) system state information by additional sensor-based data to proactively indicate possible future faults on the grid, increasing the overall availability and reliability.

The components are designed alongside the Austrian reference architecture efforts [6], thus ensuring secure results with maximum compatibility.

B. System Concept Outline

An increasing number of networked smart grid applications are emerging. Together with the push for photovoltaics (PV) and other distributed, volatile generation into low voltage networks, pressure for innovative and cost-effective sensor and actuator technologies is created. Electric grid components that provide advanced functionality (eg. integrated communication capabilities), or components to retrofit existing devices with reasonable effort, are missing at distribution level. The market is beginning to request such devices on low and medium voltage level. A radically new, cost-efficient sensor and actuator component currently in development, along with necessary information/automation technology, are the scope of this paper. To support operation of involved components, an ICT infrastructure is required and conceptually shown in Fig. 2. These devices will fill gaps in the required observability for actively managed distribution grids. A key innovative approach is integration of power management and grid protection functions within one device, called the *Smart Breaker*. The component challenge is to integrate necessary components in a compact device with reasonable costs. Smart Breakers are located in the customer domain (see Fig. 1 low voltage, mostly residential/industrial customers) to provide protection functions, power management, measurement services, and communication for individual load/generation branches.

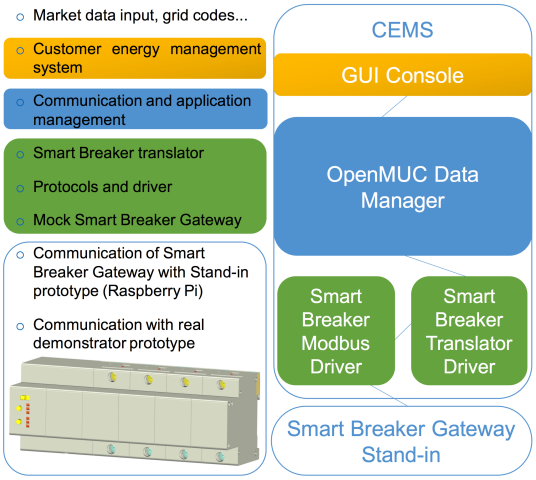

Fig. 2. System Concept integrating Smart Breaker and CEMS

Within a created ICT emulation platform, this paper describes the first development efforts of a CEMS, capable to

use the newly developed Smart Breakers in varying communication conditions. This paper shows the necessary setup to emulate different networking demands of interacting ICT components which include interfaces, gateway components, and service platforms.

II. Prototyping Emulation Platform

For the implementation of the CEMS on possibly different hardware components, OpenMUC was chosen. As described in section I-A1 the framework is a lightweight and simple architecture. The performance requirements are already met by off-the-shelf (OTS) hardware, which enables cheap prototyping of networked applications with multiple components. To perform ICT emulation of varying bandwidth connections and traffic inspection the commercial application *Charles Proxy* [7] was chosen as multi platform web debugging proxy.

A. Prototyping Test System Setup

The described *Exemplary IT-Security-Integrated Control Device hardware* in [2], is a ARMv7 based system, meaning that not only the resources as memory and computing power but also dimensions and power consumption are compatible to OTS hardware. So the first step was to study and compare the test hardware to a RaspberryPI, to enable fast prototyping and testing of OpenMUC based applications, and dealing with enhanced security requirements of the control device hardware only during deployment in an established process.

1) Hardware: The aforementioned control device is based on the quad-core ARMv7 processor i.MX 6 and uses 512 megabytes of memory. In comparison to the RaspberryPI that uses a ARM Cortex-A7 processor, it is based on the ARM Cortex-A9. As physical interfaces to access the hardware, there are ethernet ports available. One of the ports represents a serial interface to access the hardware via appropriate software as for example PuTTY or Minicom. The ethernet port can be used to obtain access using the secure shell (SSH), offering the possibility of file transfer via Secure File Transfer Protocol (SFTP) – the only means necessary to migrate OpenMUC based applications.

2) Operating System Falcon: The operating system (OS), called *Falcon*, is based on a build-root linux system with several optimizations for real-time ability of the system. It is stored on a one gigabyte microSD drive, partitioned in the ext4 format. Despite the lack of several useful tools for security reasons, with current JAVA installed the OS meets the requirements of OpenMUC. Another crucial tool installed is the dropbear software package, a small SSH server and client implementing the version 2 of the secure shell protocol, and also enabling file transfer via SFTP. This setup allows to rapidly switch between any RaspberryPI/Eclipse development environment and deployment on the control device hardware.

B. Smart Grid Example Component Requirements

A main development requirement was the closed loop use-case sequence Fig. 3 and the necessary software components needed for the CEMS. It shows a series of messages from an in parallel developed medium voltage (MV) sensor, placed at a secondary substation, up to the SCADA of the distribution management system (DMS), containing voltage and power information. A yet to be determined way of communicating, that is not directly connected to the SCADA system itself, needs to react on the power-flow optimizations and communicate or forward needed data and commands (maybe) through the meter data management (MDM), smart meter (SM)/server, down to the CEMS where it implements a Modbus server to interface a smart breaker gateway (GW) switching 1-100 smart breakers, connecting 1-3 lines of loads.

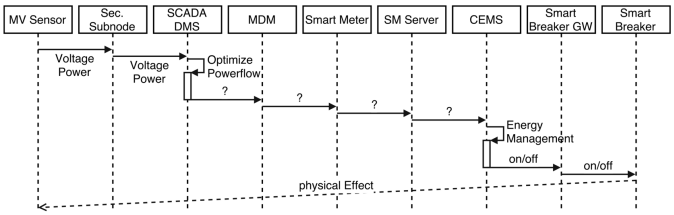

Fig. 3. Sequence diagram of a relevant closed loop use case

III. Preliminary Prototyping Results

A. Energy Management Component

Concluding form requirements in section II-B, the CEMS has three main functions:

1) It collects and receives commands and messages from hierarchically higher component through a (e.g. secure version of the IEC 61850) protocol.
2) Communication with Smart Breaker GW (including data collection and controlling via Modbus TCP/IP)
3) Making data accessible for users through reports (e.g. in a web interface)

These three requirements also define the three key components to be implemented. As OpenMUC already provides several data logging and visualization possibilities, the first prototyping step was to get OpenMUC running on all systems. Additionally the drivers for the protocols in question needed to be implemented. A further successful step was to show the desired RaspberryPI test environment was deployable to the special control device hardware and OS environment.

B. Customer Premises User Interface

A fourth generation prototype modified version of the OpenMUC web user interface is shown in Fig. 4. The frameworks web application is implemented with *AngularJS v1.3.0* and provides different HTML views based on the information coming directly from an application, mocking a smart meter. The left part of Fig. 4 shows a single graphical smart meter representation with regularly updated values, whereas the right part shows interactive smart breaker functionality.

The web interface extends according to the connected and selected devices that the user is interested in. The user can interact with the *On/Off* buttons and send a Modbus signal towards that specific mocked smart breaker (in this example

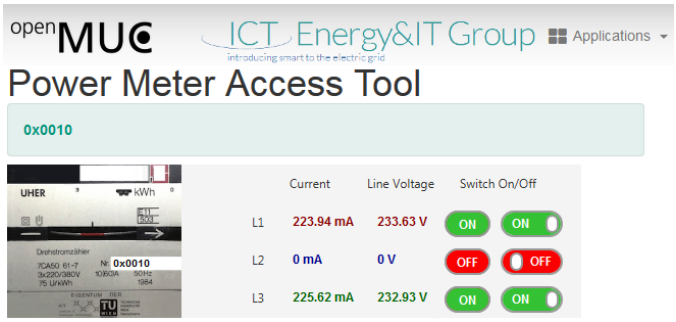

Fig. 4. OpenMUC integrated active view of Smart Breaker

ID 0x0010) in order to shut down or turn on the physically connected line. The displayed devices can be activated and deactivated during runtime, due to the *Apache Felix* framework that is used by OpenMUC. The used OSGi implementation of Apache Felix provides the possibility to add and remove so called bundles on the fly, which enables a highly modular approach on the different parts of the architecture.

Fig. 5. Smart Breaker (left), CEMS (right) deployed on networked hardware

C. Networking Emulation

Following successful *RaspberryPi 3* deployment (OS: Raspbian Jessie), each instance is executing a special task. One interacts as the CEMS, that allows the interaction to the rest of the system via web. Other RaspberryPi devices act as smart breakers or smart meters and provide data on several channels for the CEMS, which is able to translate and visualize them (see Fig. 5). The setup allows to have several communicating devices with (differently) defined networking properties, using the debian version of charles proxy, to emulate varying (slower) network connections including latency. This allows, in the example of the CEMS and smart breaker as smart grid prototypes, to develop robust networking concepts at all development stages, from different components implementing the mocked functionalities up to replacing them by their physical devices.

IV. Conclusion and Future Work

The (dis)advantages of the two frameworks OpenMUC and OGEMA in the specific use case for a CEMS, and smart breaker system were discussed. OpenMUC was chosen, as it implements the needed drivers for communication, and is more lightweight. Therefore the compatibility of OpenMUC and exemplary control hardware was studied, as well as deployment methods to OTS hardware. As a result, development was conducted on RaspberryPi's on which the needed IEC 61850 standard or Modbus drivers for communication with the smart breaker GW or SM server were implemented. For testing and verification of the communication, different specialized RaspberryPi instances were used for simulation and mocking of different devices, on all of which Charles proxy was configured to emulate eg.: different connection bandwidths of the networked devices. This system setup allows each component to be replaced by its successor, implementing the same protocols or interfaces, already at these different prototype stages, to test behavior under defined communication infrastructure conditions.

Future work with the provided ICT emulation platform is put equally as much in the development to demonstrate a way to integrate existing components, and step by step replacing them with new components, as it is put in the evaluation of functionalities arising from communication latencies, bandwidth restrictions, packet loss, etc., enabled by this platform setup, next to eg. cyber security validation of chosen protocol implementations.

Acknowledgment

This paper is based on findings of the project *Integration of Innovative Distributed Sensors and Actuators in Smart Grids – Project iniGrid*, commissioned as flagship project by *Österreichische Forschungsförderungsgesellschaft mbH (FFG)* as part of *e!MISSION.at 4th call for proposals*.

References

[1] dena, *Dena Netzstudie II*. Berlin: Deutsche Energie-Agentur GmbH, 11 2010.

[2] S. Hutterer, W. Hauer, J. Meindl, and F. Kupzog, "Secure integration and rollout of iec 61850-based smart components within the inigrid project," *23rd International Conference on Electricity Distribution*, p. 4, 15.-18. June 2015. [Online]. Available: http://www.cired.at/pdf/CIRED2015_0713_final.pdf

[3] Smart Grid Coordination Group, "Smart grid reference architecture, v3.0 final," CEN-CENELEC-ETSI, Technical Report, November 2012. [Online]. Available: ftp://ftp.cencenelec.eu/EN/EuropeanStandardization/HotTopics/SmartGrids/Reference_Architecture_final.pdf

[4] Fraunhofer ISE. (2014, 09) Openmuc framework overview. [Online]. Available: https://www.openmuc.org/index.php?id=11

[5] Fraunhofer ISE, ISE and IWES. (2014, 12) Ogema - introduction of concepts, terminology and framework services. [Online]. Available: http://www.ogema.org//wp-content/uploads/2014/12/OGEMA_2.0_introduction_v2.0.2.pdf

[6] M. Meisel, A. Berger, L. Langer, M. Litzlbauer, and G. Kienesberger, "The rassa initiative – defining a reference architecture for secure smart grids in austria," in *Energy Informatics*, ser. Lecture Notes in Computer Science, S. Gottwalt, L. König, and H. Schmeck, Eds. Springer International Publishing, 2015, vol. 9424, pp. 51–58. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-25876-8_5

[7] XK72 Karl von Randow. (2016, 05) Charles web debugging proxy. [Online]. Available: http://www.charlesproxy.com