Noname manuscript No.
(will be inserted by the editor)

Advanced Configuration System for Cost-effective Integration of Distributed Energy Systems

How we changed complex integration processes into simplified out-of-the-box solutions, installed and maintained by electricians and operators.

Jan Müller · Christian Gitte · Martin Winter · Johan van der Geest

Received: date / Accepted: date

Abstract The transition from the classical energy grid to a smart grid implies the introduction of information and communication technologies. In recent years, research and industry are working on enabling smart grids by developing solutions to connect and integrate distributed energy systems. Even though these efforts have made significant progress, a main issue relates to the costs of additional technical solutions needed in a smart grid. In this work, we present a gateway which enables the connection and integration of distributed energy systems into a control system. The gateway hardware is based on a market ready industrial controller by Siemens. The main contribution of this work is a sophisticated software supporting the integration phase of the gateway. Since the integration is one of the main cost drivers of distributed energy systems, our solution reduces the integration costs significantly and enables new business models.

Keywords Smart Grid · Demand Side Management · Mass Customization · Inductive Programming

This work was funded by EIT Digital.

Jan Müller (✉) · Christian Gitte
Karlsruhe Institute of Technology (KIT),
Institute of Applied Informatics
and Formal Description Methods (AIFB),
Karlsruhe, Germany
E-mail: jan.mueller@kit.edu

Martin Winter
Siemens AG, CT Research and Technology Center
Munich, Germany

Johan van der Geest
TNO,
Groningen, The Netherlands

1 Introduction

In this paper we investigate how simplified configuration can enable decentralized energy resources efficiently in a mass customization scenario. We utilize controller hardware and field technology by Siemens to evaluate the concept. The core technology is a software called Configurations Wizard, which is able to handle models representing the capabilities of controllers and energy technology in the field. This paper is a result of the IMPaCT Project within EIT Digital, a pan-European cluster for innovation and entrepreneurship, funded by the European Commission in the Horizon 2020 research program.

In upcoming smart grid markets many new business opportunities for small and medium enterprises emerge. In Europe and more countries worldwide, due to the shift in energy legislation small, decentralized generation facilities gain much more importance. Furthermore, technologies to access load shifting potential at the residential and industrial level are in focus. Classic approaches to energy resources are dominated by technology created for industrial environments and individual software development for the field components. Such expensive approaches are typically cost-effective in environments with special requirements on reliability, like factories and power plants. It is not economically efficient to deploy such approaches to the consumer level, e. g., small photovoltaic systems, micro combined heat and, power solutions or charge stations for electric vehicles.

Nevertheless, we need to integrate new decentralized energy resources efficiently. The evolving mass market for energy resources at the consumer level is characterized by low margins on single installations. Therefore, we use a controller which is able to handle modular,

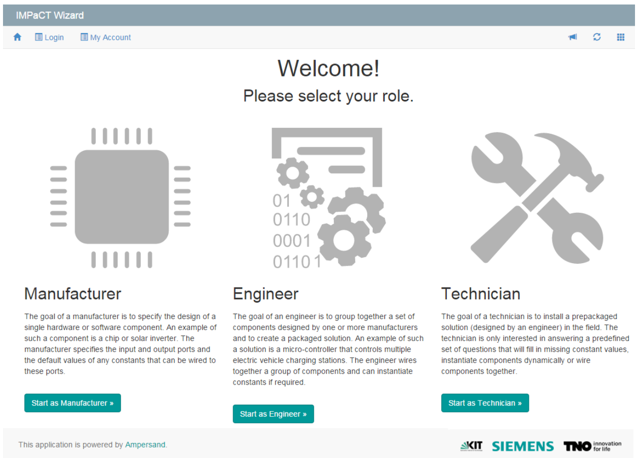

Fig. 1 Screenshot of the Configuration Wizard's log-in screen which provides different user interfaces based on the user role and know-how.

configurable services out of the box. This off-the-shelf controller is prepared for various smart grid applications scenarios. New drivers and control algorithms are available for download from the equipment manufacturer. Individual development of software extensions is still possible. The hardware itself is a specialized industrial computer with a bunch of connectors for network and measurement technology. In order to lower the costs, most of the components are adapted from consumer technology. The consumerization continues at the software level for operating system and platform technology. The complete controller is certified to fulfill industrial requirements like high temperature or high humidity environments. It is able to communicate and translate most common communication standards of the energy domain like IEC 61850 or Modbus.

Such a controller does still not fulfill the requirement of cost-efficient integration. Even if the runtime software of the controller is configurable and modular, one needs to know how to configure and set up the device. Our Configuration Wizard addresses this issue. It provides a simple and intuitive user interface for different kinds of users, based on their rights and knowledge level (see Figure 1).

Related work is available in different domains. Within the energy community several research projects and industry companies incorporate energy management systems, e. g. Organic Smart Home [1, 6], EF-Pi framework [13] or Qivicon [2]. Others developed communication standards or data models, like BEMI [10], EE-BUS [11] or the Common Information Model (IEC 61970 standards family). Our approach utilizes the aforementioned standards and can integrate the aforementioned energy management concepts. Basically, the resulting controller product can be integrated with every energy management system due to its open architecture and wide support of energy domain standards. Further related work is connected to service orchestration, graphical programming or Inductive Programming [3]. "If this than that"[1] (IFTTT) has launched a mesh up service for the connection of different web services. This relates to social media as well as technical services provided by smart phone or sensors in residential buildings. With our approach we share the mindset of simplicity as most important goal. Nevertheless, IFTTT focuses on private applications in non-critical environments. Our approach focuses on high reliability and application for industry located and residential energy applications. Microsoft introduced Visual Programming Language, a graphical notation for in the (educational) robotic domain [8]. Another representative from this domain is Open Roberta [4]. The graphical programming language Scratch is supposed to support beginners in programming [5]. Due to practical reasons, our approach does not utilize one of this languages. Instead we use a simple data model based on the controller's software architecture. More details are introduced in Section 3.

Parts of the following results have been compiled in internal research reports, restricted to EIT community [9, 14]. In this paper the work is published for the first time.

2 Scenario Description

The investigated scenario relates to typical applications of smart grid controllers. First, the controller is integrated in a charging spot for electric vehicles. Second, the controller is embedded in devices, e. g., in a combined heat and power engine (CHP). Third, the controller in used to access devices in a residential building, e. g., a heat pump. Fourth, the controller acts as a gateway for several devices, e. g., white goods. The first two scenarios are summarized as Embedded Mode, the other two scenarios are summarized as Gateway Mode. Figure 2 displays the four typical scenarios. Both modes have particular requirements regarding the configuration:

2.1 Embedded Mode

In Embedded Mode the controller typically controls system components like other controllers, sensors and actors. In general, it is embedded in the device case and invisible to the end user. The engineer responsible for

[1] https://ifttt.com/

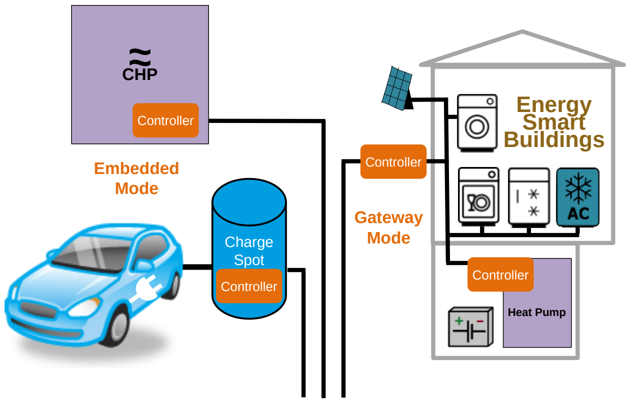

Fig. 2 Controller application scenarios in the embedded and the gateway mode.

the product design made the decision to use our controller because of the efficient integration possibilities. During the product development he can use the Configuration Wizard to model the product specific set of sensors and actors connected to the controller. He connects inputs and outputs of the installed components. Furthermore, he defines rules on how to transform the data or how to determine decision processes. He may also integrate own module, created by individual software development for the particular product set. The aforementioned engineer is the first user role to be considered for the Configuration Wizard.

Additionally, a role for the equipment manufacturer who created the controller has been defined. It should be able to define the component models supported by the controller and particular firmware revisions.

2.2 Gateway Mode

In this mode the controller is an external device connected to particular energy resources. It is responsible to integrate such resources into the energy system by supporting relevant grid control standards and provide access for operators of different kinds. Decentralized energy resources like small photovoltaic systems or heat pumps are typically installed by small local enterprises. Their technicians, e. g., electricians or installers, are responsible for the initial configuration and the proper connection of the controller and the energy device on hardware and software level. Due to the domain-specific skill level of technicians, they should not have access to all configuration functions. Full access may end up in misuse, mistakes and inefficiency. In particular, the efficiency is a strong reasons to provide role-based access to the Configuration Wizard for technicians. Furthermore, operators are a special kind of technicians. They may need to adjust particular limits like maximum power usage or the reactive power factor of the controlled energy resource.

2.3 Requirements and Configuration Levels

The configuration wizard should fulfill the following requirements:

1. The wizard should be generic for technical environments.
2. The wizard should allow a distinct allocation of roles, i. e., the technician, the equipment manufacturer and the engineer, by role-specific access to configuration functions.
3. The wizard should provide configuration support based on expert knowledge.
4. The wizard should provide templates, easing common configuration tasks.
5. The wizard has to provide valid controller configuration files.
6. Simple validations of the configuration input have to be possible.
7. A common flexible configuration data model is needed, which can cover a large number of application cases.
8. A clear separation between the user interface and the back end systems is required.

In addition to these requirements, three configuration levels and their associated role have been defined:

1. *Application independent configuration provided by the manufacturer*: Configuration of basic controller functionality. This includes but is not limited to the integration of a third party applications. In this configuration step, operators can extend the functionality of the controller or device manufacturers can add drivers or preconfigured templates.
2. *Basic application configuration provided by the engineer:* Experts can preconfigure a controller in order to prepare it to a given application scenario. They can also configure what has to be configured by the technicians in the field. Commonly, the configuration performed is used in several controllers in the same application scenario but lacks some parameters for a specific controller instance installed in the field.
3. *Configuration of a controller instance provided by the technician:* This step is performed directly after the installation in the field. It is performed by the installer of the controller, e. g. a technician. The technician has to configure the parameters specific for the specific controller instance.

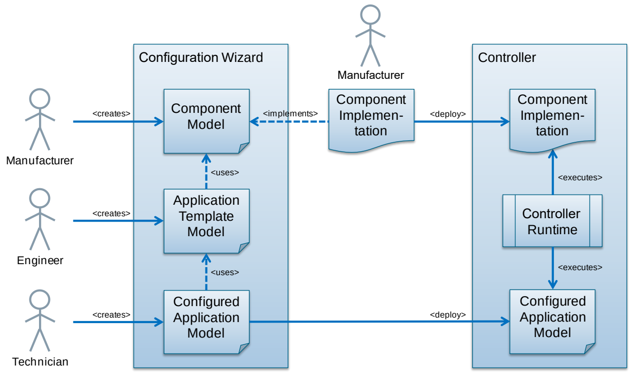

Fig. 3 Configuration of the controller using the Configuration Wizard.

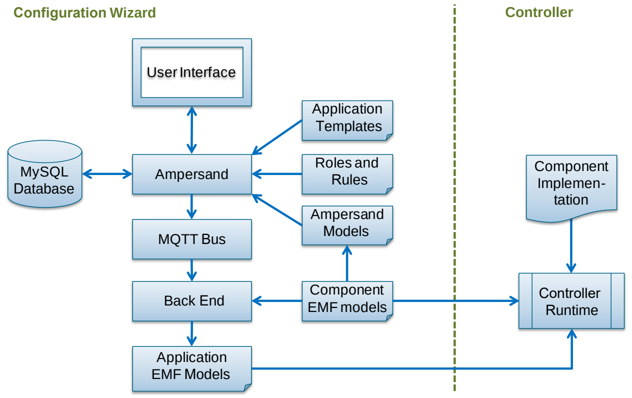

Fig. 4 Software Architecture of the Configuration Wizard.

3 Implementation

This section outlines the implementation of the concept described in Section 2. In particular, the configuration wizard and its interfaces to connected systems.

3.1 Configuration Process

In the IMPaCT project we have chosen to focus on the flexibility of the configuration wizard. This led to a configurable, rule-based wizard, which can be adapted to various scenarios and provides guidance for multiple user roles. The goal of every configuration process is to generate a so called Configured Application Model which contains all the information necessary for the operation of a controller. This model comprises the three configuration levels defined in Section 2.3. The three configuration levels result in the creation of a model describing the performed configuration.

The configuration process using the Configuration Wizard works as depicted in Figure 3. The roles of the manufacturer, the engineer and the technician create a specific configuration for a controller. Additionally, the implementations of the functions have to be provided by the manufacturer. These implementations together with the complete Configured Application Model result in an executable and configured application which is executed by the controller runtime.

3.2 Software Architecture

The overall structure of the software architecture is depicted in Figure 4. All three roles defined in Section 2 access the Configuration Wizard through a common user interface. The user interface is based on Ampersand [7, 12], which is a powerful rule-based platform. It builds the overall configuration, exposing to each role exactly those actions which are required to fulfill the tasks of that role. Ampersand has a backing database storing all model data necessary in the user interface, i. e, the application templates, information about the roles and the Ampersand models. The Ampersand model is a simplified version of the component model, containing only the information necessary in the user interface. The configuration model generated in Ampersand, either as a complete model or as a modification of an already existing model, is then pushed to the back end using a MQTT message bus. The back end converts the configuration model to application configuration files that are used by the controller. The controller runtime provided on the controller finally uses the component models, the component implementations and the application configuration files to execute the complete configured application.

3.3 Data Model

In order to enable a large number of application cases for the Configuration Wizard, a common flexible configuration data model has been developed. It is based on the Eclipse Modeling Framework[2] (EMF) which provides the following main features used in this context:

1. EMF is first used to create a meta-model, which defined the possible model elements, their attributes and relations.
2. The meta-model can directly be used to generate Java code with classes representing the model elements.
3. Model instances can be fragmented with the parts loaded from multiple locations, while still providing a consistent overall view.

The model is graph based and contains nodes which can be connected using edges. It defines the following

[2] https://eclipse.org/modeling/emf/

main objects: A Component is the description of an executable node. It mainly has an interface with input and output ports. Component descriptions are bundled with the executable code of the component. An Application defines a collection of component instances and edges, which interconnect output ports and input ports of component instances. The Applications are built using the Configuration Wizard and are deployed to the controller. The model provides mechanisms for distributed configuration and validation, also used to establish template support of the configuration wizard. The controller runtime, however, just sees a single, coherent and consistent overall model, on which it is able to operate.

In addition to the EMF model, a simplified model for the Ampersand framework has been created. While the EMF model is optimized for execution by the controller runtime, the Ampersand model is the more optimal data model for rules validation. However, the two models can be mapped onto each other.

3.4 User Interfaces

Based on the special requirements for the different roles, we decided to create two different user interfaces founding on the same back end architecture. For most application scenarios the form-based Configuration Wizard (see Figure 5) is the best choice. It provides a simple, predefined form based layout to the user based on its role. Forms and masks are generated based on Ampersand rules. New templates support new roles and application scenarios id necessary.

The form-based approach is simple and powerful, but limited to forms. Therefore, a second approach for the special needs of technical product development has been investigated. This Graph-Based Interface (see Figure 6) is based on the concepts of graphical programming. Components, inputs and outputs of the (embedded) controller are represented in a visual model. The engineer selects necessary parts based on the real hardware setup within the controlled device. Furthermore, he connects interfaces by wires and defines constants as well as configuration parameters accordingly.

4 Evaluation

In order to evaluate the Configuration Wizard and the Graph-Based Interface presented in the is paper, two different application scenarios have been investigated. To evaluate the embedded mode, the application of the controller in a charge station scenario has been chosen, while the gateway mode is evaluated in a scenario

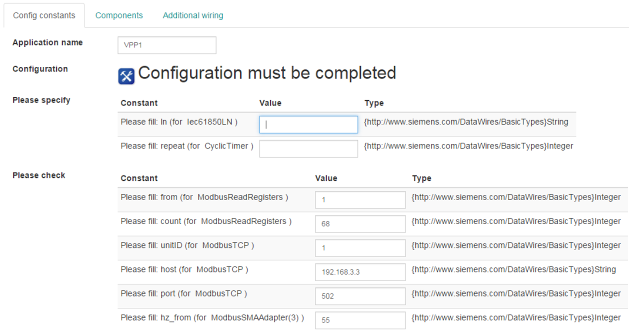

Fig. 5 Form based user interface of the Configuration Wizard.

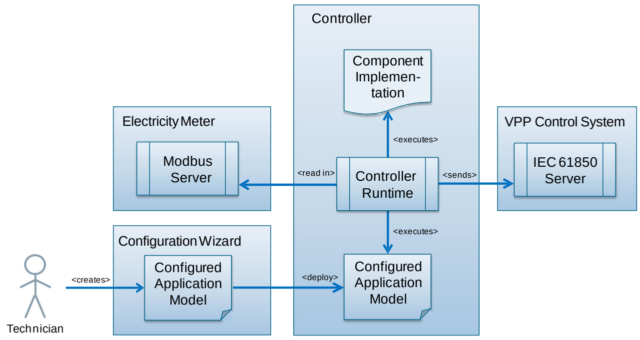

Fig. 7 Evaluation scenario in the evaluation of the gateway mode.

comprising the connection of an electricity meter to a virtual power plant (VPP). In this section, both evaluation scenarios are described more closely.

4.1 Gateway Mode: Electricity Metering

In this evaluation scenario, we examine a scenario in which the controller is used as a communication gateway together with the Configuration Wizard. It reads out the data of an electricity meter and sends it to a VPP control system (see Figure 7). In the evaluation we want to simulate the role of a technician who is going to configure the Modbus connection between the electricity meter and the controller as well as the connection between the controller and the VPP control system using the IEC 61850 protocol through the configuration wizard. The technician has to configure some of the Modbus parameters, such as the polling interval and register indices, as well as the IEC 61850 client parameters, such as the host name, port number and user credentials.

After the technician role has been selected, the user can decide to (re)configure an existing application, or to configure a new application. As we have no applications configured yet, we will create a new one. An application

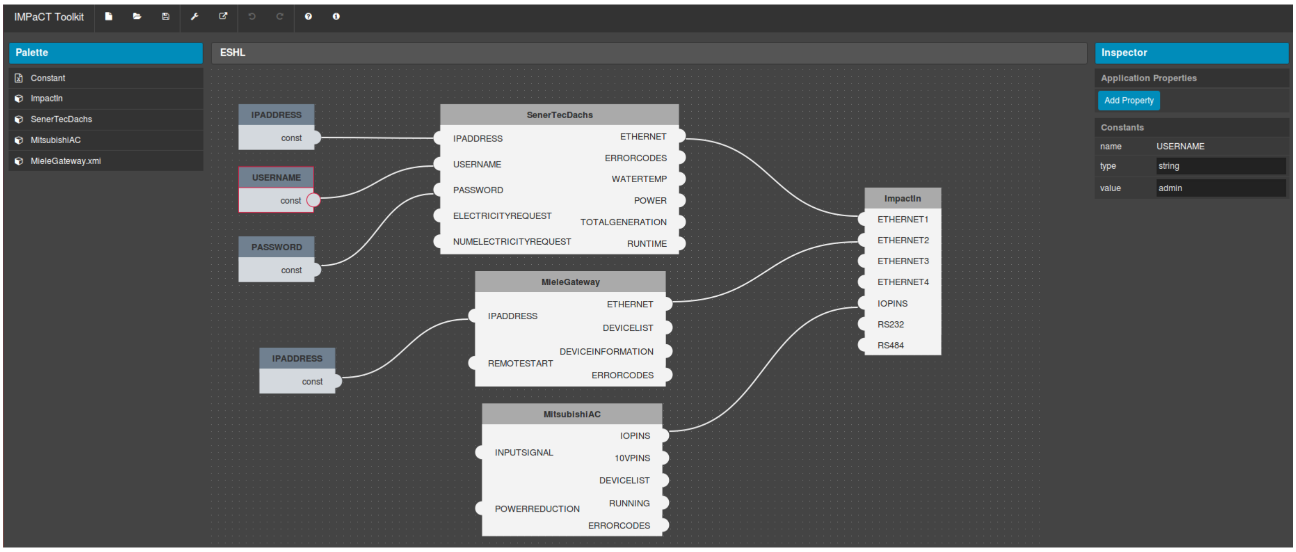

Fig. 6 Alternative user interface of the Configuration Wizard based on graphical programming.

can be empty, or can be instantiated from a template. For this project we have created a template that can be used for configuring the connection to a Modbus device. This template is the components, ports, pre-configured wiring and constants and can easily be modified. Figure 5 shows a configuration that is almost completed, only two more constants need to be filled in. When the technician presses the Sync button, a message is constructed containing the Configured Application Model and deployed to the controller. We have verified this model with the expected input used in earlier tests, and concluded that a valid model file was generated. In addition, the scenario has been set up using real hardware, and the data transfer from the electricity meter to the VPP control system has been tested.

4.2 Embedded Mode: Charge Station

This evaluation setup relates to the engineer role, using the controller and the alternative configuration interface to configure the electrical sensors and actors forming a charge station for electric vehicles.

During evaluation, exemplary parameters of the charge station settings could be configured. One test relates to the maximum voltage of the charger as well as the blink frequency of a control light indicator. The evaluation itself was performed using an power train prototype which behaves similar to an electric vehicle and which will be used to perform a charging process. We measured the power consumption of the charger to monitor the effect of limiting the charging voltage. This is a typical test setup for charge station manufacturers during the product development and a good evaluation for the engineer-focused graphical programming interface.

For user interface and the back end systems were deployed to dedicated servers within the one subnet. Utilizing the alternative configuration interface, we changed configuration value like maximum voltage in several test series. Changed values were updated in the back end and sent to the charging controller. Referring to the voltage example, the measurement of the power consumption of the charger successfully reflected the change in the maximum voltage of the charger. In addition, the blink frequency of the indicator control light was also changed using the Configuration Wizard. It successfully changed according to the desired configuration.

5 Conclusion

Within the IMPaCT-project we demonstrated successful how integration of decentralized energy resources can look like in the future. Our controller is able to support a broad range of applications scenarios by its generic software and powerful hardware interfaces. But this capability needs to be accessible and simple to use. Our research addressed this issue – we created a state-of-the-art software architecture and two potent user interfaces. The combinations of controller and Configurations Wizard is perfect to fulfill the need for reliable and cost-effective smart grid integration technology.

Acknowledgements We gratefully acknowledge the financial support from the European Commission in Horizon 2020 and EIT Digital which provided the environment for this paper.

References

1. Allerding, F.: Organic Smart Home: Energiemanagement fr Intelligente Gebude. Ph.D. thesis, Karlsruhe (2014). Accepted.: Karlsruhe, KIT, PhD-Thesis, 2013
2. Deutsche Telekom AG: Qivicon fr Unternehmen – Smart Home-Potentiale erkennen und gewinnbringend nutzen (2016). URL https://www.qivicon.com/de/fuer-unternehmen. Accessed on 03 March 2016
3. Gulwani, S., Hernández-Orallo, J., Kitzelmann, E., Muggleton, S.H., Schmid, U., Zorn, B.: Inductive Programming Meets the Real World. Commun. ACM **58**(11), 90–99 (2015). DOI 10.1145/2736282
4. Jost, B., Ketterl, M., Budde, R., Leimbach, T.: Graphical programming environments for educational robots: Open roberta - yet another one? In: Multimedia (ISM), 2014 IEEE International Symposium on, pp. 381–386 (2014). DOI 10.1109/ISM.2014.24
5. Maloney, J., Resnick, M., Rusk, N., Silverman, B., Eastmond, E.: The scratch programming language and environment. Trans. Comput. Educ. **10**(4), 16:1–16:15 (2010). DOI 10.1145/1868358.1868363. URL http://doi.acm.org/10.1145/1868358.1868363
6. Mauser, I., Müller, J., Allerding, F., Schmeck, H.: Adaptive Building Energy Management with Multiple Commodities and Flexible Evolutionary Optimization. Renewable Energy **87, Part 2**, 911 – 921 (2016)
7. Michels, G., Joosten, S., Woude, J., Joosten, S.: Relational and Algebraic Methods in Computer Science: 12th International Conference, RAMICS 2011, Rotterdam, The Netherlands, May 30 – June 3, 2011. Proceedings, chap. Ampersand, pp. 280–293. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
8. Morgan, S.: Programming Microsoft®Robotics Studio. Microsoft Press, Redmond, WA, USA (2008)
9. Müller, J., Winter, M., van der Geest, J.: IMPaCT Configuration Wizard Concept Deliverable 14002-D02. EIT Digital (2015). Unpublished.
10. Nestle, D., Ringelstein, J., Waldschmidt, H.: Open Energy Gateway Architecture for Customers in the Distribution GridArchitektur eines offenen Gateways für Energiemanagement und Energieeffizienz für Verteilnetzkunden. it-Information Technology Methoden und innovative Anwendungen der Informatik und Informationstechnik **52**(2), 83–88 (2010)
11. Otter, R.: Mit Bits und Bytes gegen babylonisches Sprachengewirr. VDI Nachrichten **16**, 12–13 (2015). 17 April 2015
12. Tarenskeen, D., Bakker, R., Joosten, S.: Using ampersand in it architecture. CONFENIS 2013 - 7th International Conference on Research and Practical Issues of Enterprise Information Systems pp. 206–214 (2013)
13. Toersche, H.A., Hurink, J.L., Konsman, M.J.: Energy management with TRIANA on FPAI. In: PowerTech, 2015 IEEE Eindhoven, pp. 1–6. IEEE (2015)
14. Winter, M., van der Geest, J., Müller, J.: IMPaCT Software Description Deliverable 14002-D03. EIT Digital (2015). Unpublished.