Grid Watch Dog: A Stream Reasoning Approach for Lightweight SCADA Functionality in Low-Voltage Grids

Konrad Diwold
Siemens AG
Vienna, Austria
konrad.diwold@siemens.com

Simon Mayer
Siemens AG, Pro2Future
GmbH, and TU Graz
Graz, Austria
simon.mayer@pro2future.at

Alfred Einfalt
Siemens AG
Vienna, Austria
alfred.einfalt@siemens.com

Josiane Xavier Parreira
Siemens AG
Vienna, Austria
josiane.parreira@siemens.com

Jack Hodges
Siemens AG
Berkeley, CA, USA
jack.hodges@siemens.com

Darko Anicic
Siemens AG
Munich, Germany
darko.anicic@siemens.com

Ralf Mosshammer
Siemens AG
Vienna, Austria
ralf.mosshammer@siemens.com

ABSTRACT

The advent of distributed energy resources in conjunction with recent developments in the context of smart embedded devices has fostered the increased integration of intelligent generators and prosumers into the energy system. As a consequence, previously passively operated grids have become highly dynamic environments, which require active monitoring and control. To allow for such monitoring and control, grid operators require lightweight SCADA-like functionality that can be flexibly adapted to dynamic grid contexts and is suitable for resource-constrained embedded systems. *Grid Watch Dog* enables system operators to create monitoring and control rules for low-voltage field devices on the fly, and to immediately deploy them in the field. The system combines a stream reasoner on field devices with an intuitive user interface for creating and managing rules. *Grid Watch Dog* has been implemented as a proof of concept in a smart grid testbed and we demonstrate its application in the context of monitoring a battery energy storage system.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

IOT '18, October 15–18, 2018, Santa Barbara, CA, USA

© 2018 ACM. ISBN 978-1-4503-6564-2/18/10...\$15.00

DOI: <https://doi.org/10.1145/3277593.3277601>

Author Keywords

Smart Grid; Stream Reasoning; Monitoring; SCADA; Grid Operations

INTRODUCTION

The number of active components (e.g., photovoltaics (PV), residential batteries, and home automation systems) in low-voltage distribution grids has been steadily increasing over the past years [23]. The Internet of Things (IoT) is a major driver for this increase, as novel computation and communication schemes are applied in the energy domain [12, 20] and facilitate new business models for the operation of battery storage systems, building automation systems and renewable generation.

In contrast to traditional loads, active components do not only passively consume energy within the grid but actively engage in consumption as well as generation. For distribution system operation, these developments have led to a paradigm shift: as their name suggests, distribution grids were originally built to *distribute* energy among *passive* consumers. Since distribution requires little to no control, distribution grids were and are usually operated passively, without any form of active monitoring, fault detection or control.

However, the majority of new distributed energy resources (DERs) such as rooftop PV-Systems and PV-battery systems are being deployed as part of the distribution grid, leading to increased volatility and complexity – consequently, passive operation can no longer guarantee safe operation. To give an example, strong local generation via DERs (e.g., high solar feed-in around midday) can reverse power flows in the grid and lead to current and voltage ranges beyond those specified in the system design. The low-voltage grid is thus in need

of new operation strategies in order to integrate, handle, and balance the various new actors within the grid and ensure its continued safe operation [24].

Over the past decade, a multitude of new approaches for the seamless integration of DERs in distribution grids as well as control strategies have been proposed and discussed [19, 24, 25]. These approaches to "Smart Grid Operation" range from the local deployment of technologies (e.g., PV inverters which regulate their power output according to local voltage / power measurements) that require no additional grid information to centralized monitoring and control strategies that depend on system information and thus require an underlying infrastructure that enables the various components to communicate with each other [13]. Again, the IoT constitutes an important driver for these developments by providing scalable solutions for the communication and interaction among devices in the context of smart grid operation [20].

Typically, the monitoring and control of an automation system is accomplished using *Supervisory Control and Data Acquisition* (SCADA) systems to do the bulk of the work [6]. In the energy domain SCADA systems are traditionally used in the context of medium-voltage (MV) and high-voltage (HV) grid operation – these grids have traditionally exhibited volatile and complex behavior (as traditional generation and industrial loads are integrated at these levels) and thus required observability and controllability. The application of traditional SCADA solutions in the low-voltage (LV) domain is currently not feasible due to the large number of LV grids that would need to be instrumented and the corresponding high upfront investments required. This is because the number of grids increases dramatically with decreasing voltage level: while a distribution system operator (DSO) might only need to operate a dozen MV grids, it usually operates hundreds of LV-grids that are connected to the MV-grids. A SCADA roll-out on the LV-grid level would thus constitute a significant investment.

Apart from the required investment, SCADA systems are complex and their parameterization requires significant time and effort and usually has to be carried out by experts. Also, adaptation of a monitoring and control process with SCADA is only possible to the extent foreseen by the developer who initially created the process. Several control systems provide the possibility to deploy scripts on on-system data-processing frameworks and thus to customize control strategies. However, these systems are not suitable for low-level embedded hardware due to their high computational requirements.

This study presents the *Grid Watch Dog* (GWD) smart grid application that applies *stream reasoning* in the context of grid monitoring in order to provide *lightweight rule-based SCADA functionality* for LV grids. The stream reasoner, which constitutes the core of the application, is specifically tailored for applications in constrained-device environments, thus reducing the potential required investment. Additionally, GWD facilitates the handling of monitoring rules by allowing users to create, deploy, update and delete rules via an intuitive user interface that guides the creation of rules based on *user intents*. The parameterization of rules is based on values that are pre-specified for common functionalities – in this way, the

user interface does not only simplify the creation of rules for non-experts but also ensures their correctness. Finally, rules are associated with actions that need to be carried out whenever a specific rule is triggered, such as informing the user or contacting other field applications.

The smart grid domain constitutes a challenging environment for stream processing and reasoning due to the sheer amount of data that needs to be processed and thus the required scalability of the approaches. A number of methods and approaches have been tested regarding their application in the smart grid domain [16, 22, 11, 21]. In contrast to GWD's embedded approach, these systems were designed for cloud environments (on stream processing platforms). Hosting the stream processing and reasoning on field devices with limited computational power is a design choice deliberately taken in order to counter the potential volatility of the communication infrastructure and allow the integration of the GWD in the grid operation process. In the context of smart grid operation, a number of semantic models have been and are currently being developed ranging from generalized thing descriptions that facilitate the interaction between different entities [14] to energy-specific data models that attempt to encapsulate all required aspects of a distribution system and prosumer operation [13, 15]. In contrast to these complex models, the data model used in our GWD proof of concept is limited as it only comprises sensor information and does not (yet) take machine interaction and other smart grid aspects into account. This minimal approach was chosen to quickly implement, test, and evaluate the application of embedded stream reasoning in a real-world smart grid operation deployment. To increase GWD's applicability in the future, more complex semantic models can be integrated.

The μ Reasoner, which is used within GWD for stream reasoning (see Section 2.2) is based on ETALIS [3, 4], a stream reasoning engine which follows the Complex Event Processing paradigm. ETALIS supports the EP-SPARQL language [2] to express stream reasoning tasks. ETALIS represents, for this application, a superior alternative to rule-based stream reasoning approaches such as LARS [5], as it enables complex event processing over multiple intervals that need not be disjoint [5].

To test and demonstrate GWD's capabilities, the problem of monitoring battery energy storage systems (BESS) was selected (see Section 3.2). The installation of such systems allows building operators to store energy produced by PV systems for later self-consumption or trading [17]. However the application of BESS in LV grids further decouples consumption and production, and thus impacts grid operation. In order to mitigate the added dynamics from BESSs in grid operation, the terms of BESS operation can be stipulated between the DSO and BESS operators. This however requires means of monitoring BESS behavior, which constitutes the GWD use case presented in this article.

GRID WATCH DOG

The primary aim of GWD is to provide lightweight rule-based SCADA functionality for LV grids. Figure 1 gives an overview of GWD's system architecture that features two main components: a front end for user-driven rule management that is

Figure 1: Left: GWD Overview – Users create rules by specifying *Intents* via the rule construction interface; the μ Reasoner applies them to LV grid data, and triggers associated actions if appropriate. Right: GWD Data Path – A classifier contextualizes raw data streams before passing them to the μ Reasoner.

hosted on the operator’s back end; and a field application that is hosted in the *intelligent secondary substation middleware Gridlink* [10]. Front end and field application communicate via REST APIs and are coupled via a common knowledge model and infrastructure [18].

The front end allows users to formulate new rules, which constitute an intent regarding a complex system state the user wishes to be informed about (e.g. "I want to be informed if the 15-minute average voltage values at any meter in the grid exceed 105% of that meter’s nominal voltage."). These rules are initially stored in a rule repository at the front end. Using the front end, a user can also manage existing rules which are either available in the rule repository (available actions: *delete rule*, *update rule*, *dispatch rule* to field application) or deployed on the field application (available actions: *pause rule*, *remove rule* from field application).

When a user-formulated rule is dispatched to the field application, it is translated into a set of reasoner rules [3] and deployed to our system’s reasoner. The reasoner operates on incoming data streams such as current grid data measured by smart meters and other metering devices by applying Datalog-like rules that are capable of capturing complex events and background knowledge. As available data streams can differ from grid to grid, a classifier is used to identify the available streams, which lowers the system’s configuration overhead. Whenever a rule is triggered, the reasoner executes the action associated with the rule (e.g., to inform the user or to trigger other applications that are deployed on the middleware). In the following subsections, we give more details about the underlying technologies used in the field application as well as the front end and the overall application composition.

Stream Classification

Through its front end, GWD permits users to formulate rules which apply to specific types of data streams that can differ from grid to grid (e.g., *European Line Voltage* vs. *American Line Voltage* streams). However, since the only information about a data stream that we can directly obtain from Gridlink is the deployment-specific *name of the data source*, streams need to be contextualized by labeling them with additional information before the system is able to select and apply the

associated rules. To automate this process, our system features a stream classification user interface that allows users to label data streams manually, but at the same time trains a classifier (k-nearest neighbor [7]) over the first three moments (Mean, Variance and Skewness) from data points in all labeled streams, and applies this classifier to all unlabeled streams. Specifically, our system’s classifier is able to learn two types of mappings: *Class Mappings* are the default mappings – when a user classifies a stream with statistical properties that can be generalized (e.g., *Voltage* data), our stream classifier requests a data sample (of configurable length `SAMPLE_LENGTH`) from the target stream and stores this sample in our system’s knowledge model together with the URI of the stream class it is associated with (using the URI of that class in our system’s knowledge model). The stored sample now serves as a statistical fingerprint of this stream class and the classifier uses it to classify additional streams. The other type of mappings in our system – *Individual Mappings* – directly map a specific Gridlink stream to a specific stream class in the GWD’s knowledge model. Here, the classifier does not attempt to generalize from an individual observation, which is a more suitable approach for streams with statistical properties that cannot be generalized in a straightforward way (e.g., power data).

When queried for a mapping of a new data stream, the GWD first checks whether there is already a stored individual mapping for that stream. If no such mapping is found, it obtains sample data from the data stream and compares the statistical fingerprint of that data with its stored sample data – the new data stream is then associated with the URI of the stored sample that has the smallest Euclidean distance to the data point’s sample. The classifier furthermore produces a confidence value, and the GWD considers all mappings above a given threshold (80% in our implementation) as valid.

μ Reasoner

Based on the contextualized data streams provided by the data classification component of the GWD, the μ Reasoner’s main purpose is to continuously monitor and control field devices, for example to detect in (near) real-time whether the voltage has exceeded a specified value. The μ Reasoner

aims to embed intelligence directly into devices – it is used in scenarios where the volume of data is too large to be sent directly to the cloud or a SCADA system. Instead, this data is processed locally, thereby allowing ultra-low latency reactions and limiting communication to the relevant data. Thus, the μ Reasoner is typically deployed on edge devices to monitor events and react to them locally.

The μ Reasoner provides real-time analytics such as anomaly detection and condition monitoring [8]. The formal language for expressing event patterns is based on the ETALIS language for events [3, 4]. Its features include: event pattern operators such as *sequence*, *conjunction*, *negation* and *disjunction*; sliding window operators based on time or counts; windowed aggregation functions such as *min*, *max*, *sum*, *average*, and *count*; temporal filters; projections; and splitting. The engine also allows for scripts for implementing custom event functions or actions. For example, smart meters produce values for *Active Power*, *Reactive Power* and *Voltage* for each phase. μ Reasoner then generates derived values using *Lua* scripts, such as *Aggregated Active and Reactive Power* over all phases and *Apparent Power*.

The μ Reasoner’s event processor is integrated with a Datalog engine which provides inference capabilities. Each edge node running the μ Reasoner is capable of reporting which resources (data points) it has access to, as each node has local knowledge mapping low-level data (e.g., from Modbus registers¹) to application-level knowledge. Thus, by interpreting local knowledge, the μ Reasoner can derive which data points are accessible (e.g., smart meter voltages) and use this knowledge for detecting temporal patterns.

By featuring event processing and deductive capabilities, the μ Reasoner is capable of performing *stream reasoning* tasks, i.e., to conjunctively reason over streaming events in combination with static or slowly evolving knowledge, and to continuously derive a streaming output under time constraints [1]. μ Reasoner runs on an embedded device such as for example a device with ARM Cortex-M3 Processor and 64KB RAM or more. Unfortunately at the present no performance results are available. Regarding the expressive power of μ Reasoner language, please refer to [3] where the language has been defined and published. The difference between ETALIS [3] and μ Reasoner is that ETALIS uses the full expressiveness of Prolog in the where clause of rules, and μ Reasoner is limited to Datalog.

Front end and application composition

GWD’s front end was developed using *vert.x*, a polyglot event-driven application framework², and the field application was developed as an application for the *Gridlink* middleware, with the classification and reasoning capabilities bundled in a single application. Communication between the GWD components is realized via REST APIs – the API of the field application is depicted in the following:

`/gwd/rules:`

`[get]: get all currently deployed rules`
`[post]: deploy new rule (json body) on GWD`

`/gwd/rule/{ruleID}:`

`[get]: get status of rule (active / inactive)`
`[delete]: delete rule from GWD`
`[put]: update rule on GWD (json body)`

`/gwd/streams:`

`[get]: get IDs of available data streams`

When the front end is started it connects to the GWD field application and requests the available streams. These received streams are consequently available for rule composition. Figure 2 displays the composition front end: A rule is defined by a *rule name*, a *description*, at least one *rule element*, a *logical combinator* of these rule elements, and a description of the *action* that should be triggered whenever the rule is satisfied (see 2 (a)). Each rule element (see 2 (b)) reflects a particular monitoring condition (e.g., "15-minute average of meter values *a* or *b* or *c* exceeds a certain threshold") and is directly linked to the available data streams. Complex rules can be defined by combining these elements using logic operators.

Rules are stored as JSON objects in a database in the front end, which allows their retrieval at a later point as well as their annotation and semantic enrichment for ease of use on the operator side. The rule management user interface is depicted in Figure 3. When a rule is deployed on the field device, the corresponding JSON object is transmitted to the GWD field application, where it is validated and translated into a set of reasoner rules, which can consequently be instantiated on the reasoner. An example for the transformation of a rule object into reasoner rules is given in Figure 4.

Whenever a rule is satisfied, the GWD field application takes the action specified in the rule object. These actions include the notification of a specified endpoint via a POST request and/or the notification of other applications hosted on the field system by means of publishing the rule satisfaction event in a topic on the Gridlink message bus, which interested applications can subscribe to.

While an intelligent substation is constrained in terms of memory and CPU when compared to standard computational systems, it is still able to host several μ Reasoner instances in one GWD application at runtime, given sufficient resources. This allows the instantiation of an exclusive μ Reasoner instance for each deployed rule, with each reasoner hosting the rules required to monitor the rule object. The benefit of this setup is that data streams can be mapped to the reasoner instances which require them and each reasoner instance only needs to evaluate input that is relevant for its rules.

EXPERIMENTS AND RESULTS

Stream Classification Tests

We performed a number of tests to verify that the stream classifier can successfully classify data streams that are relevant in

¹modbus.org/

²<https://vertx.io/>

(a)

(b)

Figure 2: GWD’s rule composition interface.

the context of a smart grid operations infrastructure, and that its performance is high enough for this purpose. To do this, we obtained a total of 1’433’596 individual data point values from several data streams from field measurements of a *Siemens SICAM P855* device: *Root Mean Square (RMS) Voltage* (3 phases); *RMS Current* (3 phases); *System Frequency*; *Total Active Power*; *Active Power* (3 phases); *Total Reactive Power*; *Reactive Power* (3 phases); and *Active Power Factor* (3 phases) over a time frame of 16.5h from midnight until 16:25pm. In addition, we obtained publicly available data on *Voltage Total Harmonic Distortion* (THD) as well as *Instantaneous Flicker* over a single day from a deployment at Lawrence Berkeley National Lab³ in the USA.

To verify that the stream classifier can classify these data streams with sufficient accuracy, we performed 1000 iterations of the following test setup and with the parameter `SAMPLE_LENGTH` set to values between 3 and 103 (in increments of 5):

- *Training*: For each stream, select, at random, an index in the time series and train the classifier using the data point values at that index and the following `SAMPLE_LENGTH` indices.
- *Classification*: For each data stream, select, at random, an index in the time series and classify the data point using the values at that index and the following `SAMPLE_LENGTH` indices. Record an error every time the category given by the classifier is different from the true category.

When applied to the complete data set with all data streams, our classifier delivers very good results: the classification error stays below 2% for all `SAMPLE_LENGTH` settings at and above 30. In addition, when we exclude the data streams that should be mapped using an *Individual Mapping* (according to our experience), the error rates drop to zero for all sample lengths – as expected, our statistical classifier has no trouble distinguishing between e.g., *Voltages* (that hover around a nominal value such as 230V or 110V) and *Frequencies* (that do the same, for different values). Also the *time required* for training and classification of the data stream samples (measured at a `SAMPLE_LENGTH` parameter of 30) is acceptable, at 8ms

per sample (averaged over 1000 iterations on all tested data streams).

GWD-based battery energy storage system monitoring in distribution grids

Recent years have seen increased installation of PV-systems in combination with battery energy storage systems (BESS) in buildings. This combination allows to store energy produced by PV systems for later consumption (and thus help to increase self-consumption) or trading [17]. Given that these systems are usually governed by intelligent control mechanisms, which differ among systems and operation modes, they add yet more dynamics to distribution grids as their use decouples weather, consumption and production. A number of studies have investigated the impact of BESS on the grid as well as the potential for actively utilizing these systems to stabilize the grid [9, 26].

The potential impact BESS can have on distribution systems is depicted in Figure 5: Figure 5 (a) shows a situation where the power obtained via the PV system is directly fed into the distribution grid (e.g., because the system’s battery is already fully charged). Figure 5 (b) depicts a scenario where the battery is charged via the distribution grid thus creating load peaks which can put stress on the distribution grid if they occur in times of peak load. As can be seen in the figure, BESSs can exhibit very dynamic load and consumption spikes. Besides the example in Figure 5, there are many more ways a BESS impact the distribution system depending on sizing, controller and other factors.

If the overall intended operation mode of a BESS is known in advance, it is possible to monitor the behavior of the system on the basis of meter values. An exemplary overall operation mode could be to maximize the self-consumption of generated power and avoid load peaks beyond the limit specified in the grid connection contract in order to avoid a penalty. If a violation of the operation mode is detected, the operator of the battery can be notified immediately in order to reduce financial losses. However, given that the specific characteristics to be monitored depend on the sizing of the battery as well as its potential changing operation modes, this requires a flexible

³maps.pqube.com

Figure 3: GWD's rule deployment/management interface.

Figure 4: Sketch of mapping a user-defined rule object into a set of reasoner rules

monitoring solution, which is the reason why this problem was selected to test GWD.

Experimental Setup and Scenario

The application of GWD was tested in the smart grid testbed of the *Aspern Smart City Research* (ASCR) campus. This testbed comprises several distribution grids equipped with metering devices and intelligent secondary substations. Within the testbed, a number of BESS located in buildings as well as secondary substations operate under different operational strategies in order to test the impact of storage systems on distribution system operation.

The GWD field application was implemented and deployed on an intelligent secondary substation in the Aspern testbed, while the GWD front end was hosted in the testbed's back end. The investigated distribution grid features a single BESS with a capacity of 150 kWh and a peak load / consumption

of 150 kW. During normal operation this BESS tries to maximize self-consumption of generated energy while limiting the occurrence of consumption peaks.

Based on historical sensor data, the building's active (P) and reactive power (Q) tuples under normal operation could be established. The tuple set constitutes the fluctuation of the active and reactive power and can thus be seen as a sort of fingerprint regarding the battery system's intended and thus normal behavior. As can be seen in Figure 6, the P/Q activity under normal operation clusters well and allows P/Q boundary conditions to be established which can be used to monitor the behavior of the battery system.

Results

A GWD rule was generated and deployed in the field, which alerts the operator whenever the active power values per phase (averaged over 5 minutes) leave the range of 1.5kW

Figure 5: Impact of PV-battery system on a distribution grid: (a) feed-in of surplus power into the distribution system, (b) battery charging

Figure 6: Active (P) / Reactive (Q) power activity of the battery system during normal operation.

Figure 7: Active and Reactive power over a day measured at the point of common coupling of a building with an integrated PV-battery system.

to 27.85kW or whenever the reactive power values per phase (averaged over 5 minutes) leave the range of 0kVar to 5kVar. The rule was applied to active and reactive power values measured at the point of common coupling of the building. The 5-minute averaging factor was incorporated in the rule to prevent the system from reacting to false-positive transient events.

Figure 7 shows time series of active and reactive power measured at the point of common coupling during a day on which the battery changed its operation mode due to a controller update. The time spans marked in red constitute the times that the GWD was able to detect a violation according to the rule, and then notified the front end. As can be seen not all

violations were detected, which is due to the time factor used. By reducing the averaging time, the system can be tuned to detect events of shorter duration.

CONCLUSION

In this article, we introduced the Grid Watch Dog (GWD), a lightweight, stream-reasoning-based SCADA system for LV grids. We discussed GWD's ability to consume and immediately deploy rules for monitoring and controlling LV field devices that are defined by system operators on the fly, and gave an overview of its system architecture, in particular the interactions between the user interface, the streams classification endpoint, and the embedded μ Reasoner. Finally, we presented first results from a proof-of-concept deployment of GWD that demonstrates its application in real-world grid monitoring scenarios: GWD is currently deployed in the Aspern Smart City testbed and is used for rule-based monitoring of battery systems that operate in different modes like peak shaving, self-consumption optimization, reduction of asymmetrical phase loading, and combinations of these modes. Future work will look into the application of GWD in other domains of smart grid operation. In addition the application of GWD on other embedded systems (such as smart meters) and the incorporation of distributed reasoning among a set of GWDs is subject of future research.

Acknowledgements

The presented work was funded and supported by the *Austrian Research Promotion Agency (FFG)* (#846141 SCDA & #853555 FACDS & #854184 COMET), the *Austrian Climate and Energy Fund (KLIEN)* and the *California Energy Commission (EPC-15-046, Smart Management of Smart Grids)*. The presented application was implemented and tested in the Smart Grid testbed of *Aspern Smart City Research (ASCR)*.

REFERENCES

1. Darko Anicic. 2012. *Event Processing and Stream Reasoning with ETALIS: From Concept to Implementation*. Omniscriptum GmbH & Co KG.
2. Darko Anicic, Paul Fodor, Sebastian Rudolph, and Nenad Stojanovic. 2011. EP-SPARQL: a unified language for event processing and stream reasoning. In *Proc. 20th International Conference on World Wide Web (WWW '11)*. ACM, 635–644.
3. Darko Anicic, Paul Fodor, Sebastian Rudolph, Roland Stühmer, Nenad Stojanovic, and Rudi Studer. 2010. A rule-based language for complex event processing and reasoning. In *International Conference on Web Reasoning and Rule Systems*. Springer, 42–57.
4. Darko Anicic, Sebastian Rudolph, Paul Fodor, and Nenad Stojanovic. 2012. Stream Reasoning and Complex Event Processing in ETALIS. *Semantic web* 3, 4 (Oct. 2012), 397–407.
5. Harald Beck, Minh Dao-Tran, Thomas Eiter, and Michael Fink. 2015. LARS: A Logic-based Framework for Analyzing Reasoning over Streams. In *Proc. Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 1431–1438.

6. Stuart A Boyer. 2016. *SCADA supervisory control and data acquisition, Fourth Edition*. The Instrumentation, Systems and Automation Society.
7. Feng Chen, Pan Deng, Jiafu Wan, Daqiang Zhang, Athanasios V Vasilakos, and Xiaohui Rong. 2015. Data mining for the internet of things: literature review and challenges. *International Journal of Distributed Sensor Networks* 11, 8 (2015), 431047.
8. Opher Etzion and Peter Niblett. 2010. *Event Processing in Action*. Manning Publications Co., Greenwich, CT, USA.
9. Robert L Fares and Michael E Webber. 2017. The impacts of storing solar energy in the home to reduce reliance on the utility. *Nature Energy* 2, 2 (2017), 17001.
10. Mario Faschang, Stephan Cejka, Mark Stefan, Albin Frischenschlager, Alfred Einfalt, Konrad Diwold, Filip Pröbstl Andrén, Thomas Strasser, and Friederich Kupzog. 2017. Provisioning, deployment, and operation of smart grid applications on substation level. *Computer Science-Research and Development* 32, 1-2 (2017), 117–130.
11. Raul Castro Fernandez, Matthias Weidlich, Peter Pietzuch, and Avigdor Gal. 2014. Scalable stateful stream processing for smart grids. In *Proc. 8th ACM International Conference on Distributed Event-Based Systems*. ACM, 276–281.
12. Giancarlo Fortino and Paolo Trunfio. 2014. *Internet of things based on smart objects: Technology, middleware and applications*. Springer.
13. Vehbi C Gungor, Dilan Sahin, Taskin Kocak, Salih Ergut, Concettina Buccella, Carlo Cecati, and Gerhard P Hancke. 2011. Smart grid technologies: Communication technologies and standards. *IEEE Transactions on Industrial Informatics* 7, 4 (2011), 529–539.
14. Sebastian Kaebisch and Takuki Kamiya. 2017. Web of Things (WoT) Thing Description. *First Public Working Draft, W3C* (2017).
15. Jason Koh, Steven Ray, and Jack Hodges. 2017. Information Mediator for Demand Response in Electrical Grids and Buildings. In *Proc. IEEE 11th International Conference on Semantic Computing (ICSC)*. IEEE, 73–76.
16. Björn Lohrmann and Odej Kao. 2011. Processing smart meter data streams in the cloud. In *Innovative Smart Grid Technologies (ISGT Europe), 2011 2nd IEEE PES International Conference and Exhibition on*. IEEE, 1–8.
17. Rasmus Luthander, Joakim Widén, Daniel Nilsson, and Jenny Palm. 2015. Photovoltaic self-consumption in buildings: A review. *Applied Energy* 142 (2015), 80–94.
18. Simon Mayer, Jack Hodges, Dan Yu, Mareike Kritzler, and Florian Michahelles. 2017. An Open Semantic Framework for the Industrial Internet of Things. *IEEE Intelligent Systems* 32, 1 (2017), 96–101.
19. KS Reddy, Madhusudan Kumar, TK Mallick, H Sharon, and S Lokeswaran. 2014. A review of Integration, Control, Communication and Metering (ICCM) of renewable energy based smart grid. *Renewable and Sustainable Energy Reviews* 38 (2014), 180–192.
20. S Sofana Reka and Tomislav Dragicevic. 2018. Future effectual role of energy delivery: A comprehensive review of Internet of Things and smart grid. *Renewable and Sustainable Energy Reviews* 91 (2018), 90–108.
21. R Shyam, Sachin Kumar, Prabakaran Poornachandran, and KP Soman. 2015. Apache spark a big data analytics platform for smart grid. *Procedia Technology* 21 (2015), 171–178.
22. Yogesh Simmhan, Qunzhi Zhou, and Viktor Prasanna. 2011. Semantic information integration for smart grid applications. In *Green IT: Technologies and Applications*. Springer, 361–380.
23. Thomas Stetz, Jan von Appen, Fabian Niedermeyer, Gunter Scheibner, Roman Sikora, and Martin Braun. 2015. Twilight of the Grids: The Impact of Distributed Solar on Germany's Energy Transition. *IEEE Power and Energy Magazine* 13, 2 (2015), 50–61.
24. Thomas Strasser, Filip Andrén, Johannes Kathan, Carlo Cecati, Concettina Buccella, Pierluigi Siano, Paulo Leitao, Gulnara Zhabelova, Valeriy Vyatkin, Pavel Vrba, and others. 2015. A review of architectures and concepts for intelligence in future electric energy systems. *IEEE Transactions on Industrial Electronics* 62, 4 (2015), 2424–2438.
25. Maria Lorena Tuballa and Michael Lochinvar Abundo. 2016. A review of the development of Smart Grid technologies. *Renewable and Sustainable Energy Reviews* 59 (2016), 710–725.
26. Jan von Appen, Julio H Braslavsky, John K Ward, and Martin Braun. 2015. Sizing and grid impact of PV battery systems-a comparative analysis for Australia and Germany. In *Proc. 2015 International Symposium on Smart Electric Distribution Systems and Technologies (EDST)*. IEEE, 612–619.