An MQTT-Based Scalable Architecture for Remote Monitoring and Control of Large-Scale Solar Photovoltaic Systems

Salsabeel Shapsough(✉), Mohannad Takrouri, Rached Dhaouadi, and Imran Zualkernan

American University of Sharjah, Sharjah, UAE
salsabeelshapsough@gmail.com

Abstract. This paper presents a novel IoT-based architecture that utilizes IoT communication, software, and hardware technologies to enable real-time monitoring and management of solar photovoltaic systems at a large scale. The system enables stakeholders to remotely control and monitor the photovoltaic systems and evaluate the effect of various environmental factors such as humidity, temperature, and dust. The system was implemented and evaluated in terms of network delay and resource consumption. MQTT demonstrated an average network delay of less than 1 s, proving the architecture to be ideal for solar and smart grid monitoring systems. At the hardware, the evaluation showed the hardware to consume about 3% of the panel's capacity, while the application also utilized a very small percentage of the CPU. This lead to the conclusion that the proposed architecture is best deployed using low-cost constrained edge devices where a combination of efficient MQTT communication and low resources consumption makes the system cost-effective and scalable.

Keywords: IoT · Solar photovoltaic monitoring

1 Introduction

Studying the effect of dust and other atmospheric conditions on remote solar photovoltaic systems has been recognized as key to more efficient solar power generation. While soiling, especially the accumulation of dust particles on panels, is known to reduce solar panel efficiency [1], the cost of periodic cleaning can sometimes exceed the cost ensued by dust accumulation [2]. It is therefore important to monitor the PV panels soiling level to optimize the cost of cleaning, especially as the scale increases from few solar panels to solar farms. Similar challenges also exist with other environmental factors such as haze, moisture, and temperature [3, 4]. Furthermore, the tilt angle of a solar panel array can influence the solar radiance harnessed by the panel PV [5]. Controlling the angle, therefore, can not only maximize the power output, but also assist in building a model that predicts the optimal solar array installation in a specific region. Thus, the ability to remotely monitor and control multiple solar farms has since been recognized to hold the key to optimum solar power generation.

A.-S. K. Pathan et al. (Eds.): SGIoT 2018, LNICST 256, pp. 57–67, 2019.
https://doi.org/10.1007/978-3-030-05928-6_6

1.1 Background

Several designs for solar photovoltaic remote monitoring systems have been previously proposed [6]. A noticeable recent trend in the field has been the adoption of Internet of Things (IoT) concepts and technologies [7] for building such systems. An IoT system consists of numerous, geographically spread nodes that interact with their surroundings, and send information back to a server periodically, based on a certain event, or both. Information is either provided to stakeholders in real-time or stored to be available on demand. The scale of such systems enabled by emergence of new software and hardware technologies that cater for low-cost, scalability, reliability, and security.

1.2 The Internet of Things Paradigm

A generic Internet of Things system consists of five layers: perception, network, middleware, application, and business. In a solar PV system, the perception layer is represented by sensing and actuating elements at the panel level, while the network layer is represented by the wireless intranetworking within a solar farm, and the internetworking between solar farms, remote servers, and stakeholders. The middleware layer hosts technologies responsible for translating raw data into intelligible information, as well as storage and dissemination of information. The application and business layers represent interested stakeholders. While interfaces used by clients and researchers fall under the application layer, operations such as billing and integration with smart grid represent the business layer.

1.3 IoT-Aided Solar Monitoring

The value that IoT technologies can bring to remote monitoring of Solar PV systems has been recognized. Several works have suggested the incorporation of IoT elements in the field. Most such proposals cater to the edge of the IoT system, low-cost edge sensing and processing technologies such as Raspberry Pi, Arduino, BeagleBones, and others are used to acquire readings related to solar panel's power generation efficiency, as well as the surrounding environment [8–10]. Edge devices often act as nodes in mesh networks, connected by short-range wireless technologies such as ZigBee and Bluetooth. One shortcoming of the proposed systems is that beyond the edge, large-scale operations such as dissemination and management of data across the whole system are still performed either by classic internet technologies which are ill-suited for this purpose [11]. In this paper, we present a novel architecture for scalable, almost-real-time monitoring of Solar PV systems, based on a complete IoT paradigm that integrates IoT technologies across the system.

2 Proposed Architecture

The proposed IoT-based architecture is shown in Fig. 1. The edge device at each panel represents the perception element. Each device is equipped with sensors and actuators and is capable of data processing and wireless communication. Data is transmitted wirelessly to reduce cabling costs. Wireless communication can be provided either by an access point that every group of edge devices can connect to over WiFi, Bluetooth, ZigBee, or Thread, or by connecting a 3G/4G module to each individual edge device.

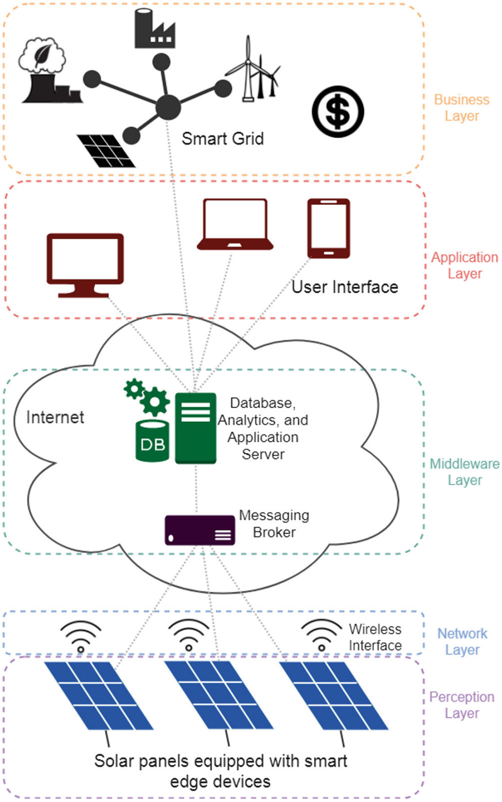

Fig. 1. System architecture

2.1 System Architecture

The first key middleware element is a messaging broker that supports a publish/subscribe communication protocol like Message Queuing Telemetry Transport (MQTT). As opposed to the HypterText Transfer Protocol (HTTP), -which operates in a one-to-one client-request/server-response architecture, MQTT allows event-based, one-to-many, many-to-one, and many-to-many messaging between remote nodes [12]. MQTT also consumes less computational and network resources, while also causing less latency [13]. This makes the protocol an ideal choice for deployment on low-resources edge devices in applications like solar monitoring. The broker keeps a register of subscribed nodes in order to route any new messages from publishers to their correct destinations. In the case of a node going down due to technical issues, the broker retains messages until the node is available again. Other middleware elements are: an application server to manage data and provide a web interface for users, a database to store data, and an analytics engine.

2.2 Communication Architecture

The key aspect of an MQTT-Based architecture is that the system supports the option of distributed servers, allowing load balancing for large-scale systems as more resources can easily be integrated into the system. Alternatively, different servers can be developed to host different types of applications, and connect to different analytics engines, making the system not only highly scalable, but also flexible. On the other end of the architecture, individual solar panels and entire solar farms can easily be added and removed with minimal disturbance to other part of the system. This is yet another advantage of publish/subscribe architectures, as the subscribers are separated from the publishers by the broker, which acts as a mediator between the two. Messages from publishers are delivered to subscribers based on "topics" maintained by the broker. Figure 2 demonstrates message routing at the broker.

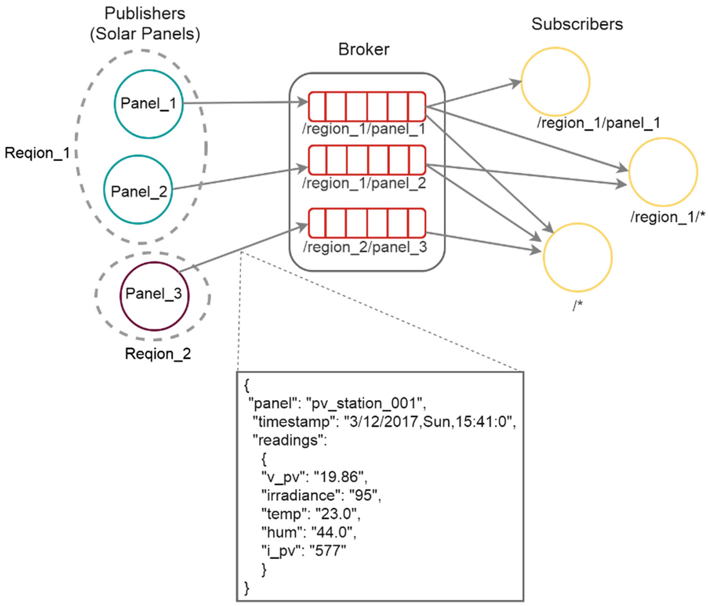

Fig. 2. Communication architecture

Every published message includes a topic following the format of a URL. For example, a PV panel located in "Region 1" of the solar farm and identified by the panel ID "Panel 1" tags its messages with "/region_1/topic_1". Subscribers can choose to subscribe to any topic by sending a one-time subscription request. Subscribers can subscribe to one panel (/region_1/panel_1), all panels in a region (/region_1/*), or all panels in the system (/*). The "*" character acts as a wildcard and allows subscribers to subscribe to numerous publishers with a single request. It is also possible for multiple panels to publish to one topic, e.g. "region_1/dust", or for subscribers to subscribe to sensor-related topics, e.g. "/*/dust". This enables subscribers interested in certain information, such as dust accumulation, for example, to receive updates any new panels added to the system, without the need to pre-register the panel at the subscribers.

2.3 Proof of Concept

Hardware Implementation. As proof of concept, a prototype of the system including hardware and software has been implemented and tested over the past few months. Two types of WiFi-connected Raspberry Pi-based edge stations were designed, implemented, and evaluated. The first type is "pv station", which is a station that is attached to and monitors an individual solar panel. The pv station is equipped with a combination of sensors and actuators to provide different types of monitoring and control. The second type, "c station", is equipped with a Raspberry Pi camera for surveillance. It is also possible to use the videos from the "c station" to perform image processing for research on weather and soiling effect on solar panels. The internal temperature of the Raspberry Pi in all stations is also monitored for troubleshooting. A description of each station's features is shown in Table 1, while the PV stations are shown in Fig. 3.

Table 1. Edge nodes

	C station	PV station 1	PV station 2	PV station 3
Purpose	CCTV	Solar irradiance and PV output monitoring	Solar irradiance and weather monitoring	PV output monitoring and IV characteristics measurement
Main components	• RPi Camera • RPi internal temperature sensor	• Pyranometer • 3 voltage sensors • 3 current sensors • RPi internal temperature sensor	• Pyranometer • Ambient temperature sensor • Humidity sensor • RPi internal temperature sensor	• 2 isolated voltage + current sensors • Temperature sensor • Relay modules • RPi internal temperature sensor
Report frequency	1/minute	1/second	1/minute	1/second
Data type	Video stream	text	text	text
Average packet size		508 Bytes	189 Bytes	346 Bytes

The two pv stations are self-sufficient in terms of power in that the batteries that powers the stations are charged using the solar panel each station monitors. The c station, on the other hand, is powered through a wall socket. Communication between edge

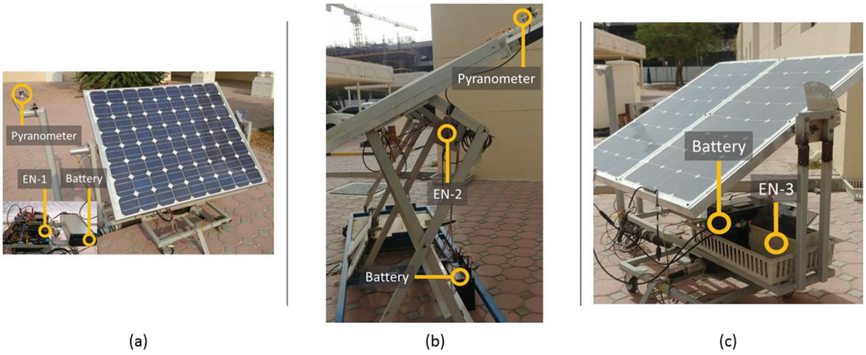

Fig. 3. Edge stations: (a) PV station 1, (b) PV station 2, (c) PV station 3

stations and the rest of the system is carried out over WiFi, while c station is connected via Ethernet to improve video streaming quality.

Backend and Web Application Server. The backend server is the main subscriber to real-time readings. The server was built using NodeJS and Express and is hosted on Amazon Web Services (AWS) which provides a mix of a Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). Readings are stored into DynamoDB, which is a document-based NoSQL database. The main server's Express interface exposes four main functionalities: a standalone solar irradiance interface (Fig. 4), a real-time monitoring, control, and surveillance interface for individual panels (Figs. 5 and 7), a history interface that allows access to archived data logs from specific dates and times (Fig. 6), and an analytics interface. Access privilege levels can range from casual users who can only view the real-time solar irradiance, to high level administrators who are authorized to send control commands to the panels, view analytics, and access the surveillance camera.

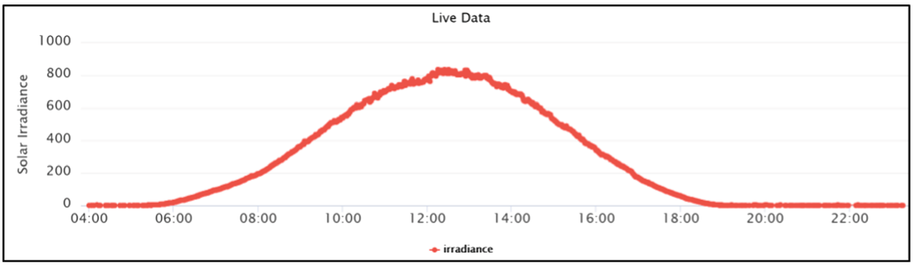

Fig. 4. Solar irradiance over a full day

The importance of the surveillance camera comes from enabling administrators to compare measured values to actual real-time footage. For example, an irregularity was previously noticed in solar irradiance data at around 08:45 am where the irradiance value experiences a sudden jump. Reviewing the surveillance video (Fig. 7) revealed

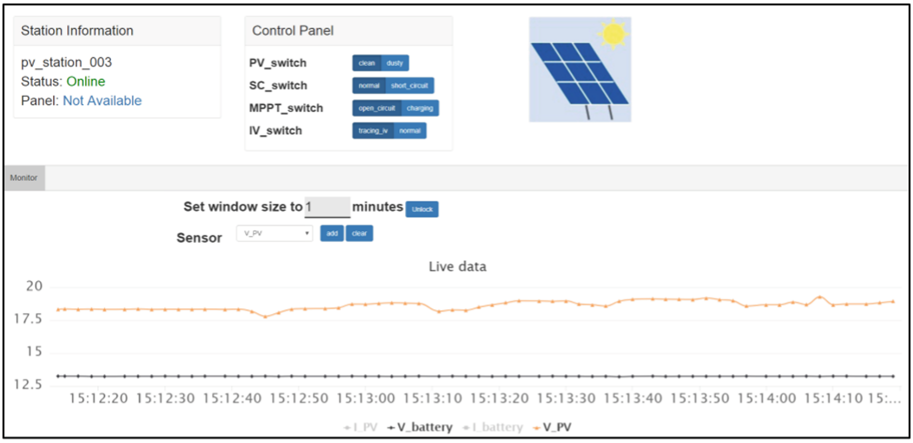

Fig. 5. Real-time interface for PV station 2 showing humidity and temperature

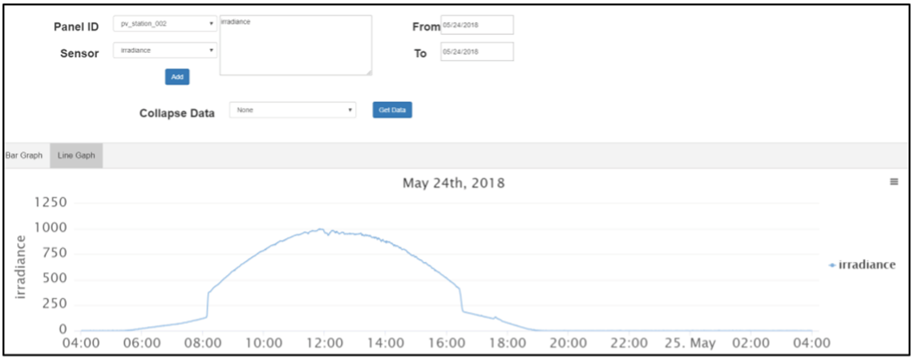

Fig. 6. History interface showing solar irradiance readings log from PV station 2 on May 24th, 2018

that the shading was due to the sun rising from behind the neighboring buildings. Furthermore, the stored footage provides visual data that can be used along with image processing and machine learning algorithms to perform various studies to predict soiling levels and study its effect on the solar power output.

Fig. 7. Surveillance cam screenshots between 08:45 and 09:15am, showing the shading effect

3 Evaluation

The edge stations are the most important elements in the system, yet they have the highest constraints in terms of resources. While the backend server can easily be scaled with backup and load balancing software, the edge stations must function as reliable, standalone units. Therefore, the system evaluation focuses on the RPi's ability to ensure reliability at a low resources cost. The metrics taken account in the evaluation are power consumption, CPU utilization, I/O operations, and end-to-end delay. The RPi's internal temperature is also measured as the system is expected to be exposed the region's harsh, high-temperature weather.

Experimental Setup. All three edge stations were allowed to run for a full day while resource consumption, internal temperature, and network traffic were monitored. Power consumption was measured using YoctoAmp [14], an isolated USB ammeter which sampled the current consumed by the edge station at 1 sample/second. Internal temperature was measured using a built-in function in the RPis, while the CPU utilization and network write rate was measured using the Linux-based performance monitoring tool NMON [15]. The internal temperature was taken at a rate of 1 sample/minute, while the CPU utilization was measured at a rate of 1 sample/second. This is primarily due to the fact that while CPU utilization can change significantly within seconds, the internal temperature remains relatively stable and changes more slowly. Finally, the network end-to-end delay was defined as the time needed for an MQTT packet to travel from the publisher (edge station) to a subscriber and was measured by assigning timestamps at the times of sending and receiving. In order to ensure the accuracy of the end-to-end delay measurement, the clocks of the publisher and the subscriber had to be perfectly synchronized. One way to achieve this is to use the same edge station as the subscriber to its own packets. However, in order to completely avoid the extra subscriber process affecting other metrics such as the CPU utilization and power, the network delay experiments were run separately. The stations were connected to 100-Watt solar panels.

Results. The main findings of the experiment are shown in Table 2. As expected, the station that consumed the highest amount of resources was the camera station.

Compared to the other station, continuous video streaming over WiFi utilizes the CPU by about 60% more and consumes around 50% more power. As for the PV stations, which are the main components of the system, the two stations demonstrate similar consumption of resources. The first conclusion that can be drawn is that the report frequency has insignificant effect on the resource consumption: as the performance is similar despite a difference of a factor of 60 between the sampling frequencies of the two PV stations. Secondly, the low percentage of CPU consumption and the relatively-high power consumption (which amounts to a little under 3% of the solar panel's output) suggest that the stations may not need a device as powerful as the RPi at all. Consequently, the suggested architecture can be improved based on WiFi-enabled microcontrollers with a fraction of the computational power such as an ESP or a smaller Arduino-based microcontroller, thus reducing the cost of hardware and power consumption, while maintaining the same level of reliability. Finally, since the overall goal is to eventually integrate the solar park into a smart grid system, the system latency was compared to general delay requirements in smart grid applications reviewed in [16], which was approximated to be 1 s. The low end-to-end delay proves MQTTS to be well-suited for secure, real-time communication for such a system.

Table 2. Experimental findings

Metric		C station	PV station 2	PV station 3
Power consumption	# of samples	3600	3600	3600
	Mean	3.142 W	2.237 W	2.148 W
	Stdv.	0.165	0.0730	0.0518
Raspberry Pi CPU utilization	# of samples	3600	3600	3600
	Mean	4.824%	2.931%	2.708%
	Min	2.1%	0%	0.3%
	Max	16.2%	15.8%	28%
Raspberry Pi internal temp	# of samples	1440	1440	1440
	Mean	58.475 °C	56.464 °C	51.120 °C
	Stdv.	6.277	6.509	2.156
Network write rate	# of samples	3600	3600	3600
	Mean	245.9 KB/s	2.6 KB/s	2.6 KB/s
	Min	209.1 KB/s	0.1 KB/s	0.1 KB/s
	Max	622.9 KB/s	29.2 KB/s	26 KB/s
MQTT round-trip latency	# of samples	1440	1440	1440
	Mean	0.802 s	0.800 s	0.801 s
	Stdv.	0.787	0.8188	0.806

4 Conclusion

The proposed system architecture employs a network of distributed edge nodes that collect sensor readings from the environment as well as the solar photovoltaic panels and transmits them back to the main station over WiFi. In addition to maintaining a large-scale database, the main station hosts an application server that offers monitoring of readings in real time, an overview of historic data, geolocation-based visualization of connected solar farms, and big data analytics. The system is used to enable and support the study of the effects of environmental factors, on solar farm, such as dust accumulation and its effect on power generation efficiency. The system also enables stakeholders to monitor and control critical operations. This is made possible via Internet of Things communication, software, and hardware technologies which are inherently designed for large scale distributed ecosystems.

References

1. Burton, P.D., King, B.H.: Application and characterization of an artificial grime for photovoltaic soiling studies. IEEE J. Photovoltaics **4**, 299–303 (2014). https://doi.org/10.1109/JPHOTOV.2013.2270343
2. Zapata, J.W., Perez, M.A., Kouro, S., et al.: Design of a cleaning program for a pv plant based on analysis of energy losses. IEEE J. Photovoltaics **5**, 1748–1756 (2015). https://doi.org/10.1109/JPHOTOV.2015.2478069
3. Liu, H., Nobre, A.M., Yang, D., et al.: The impact of haze on performance ratio and short-circuit current of PV systems in Singapore. IEEE J. Photovoltaics **4**, 1585–1592 (2014). https://doi.org/10.1109/JPHOTOV.2014.2346429
4. Huang, P., Zhao, W., Li, A.: The preliminary investigation on the uncertainties associated with surface solar radiation estimation in Mountainous Areas. IEEE Geosci. Remote Sens. Lett. **14**, 1071–1075 (2017). https://doi.org/10.1109/LGRS.2017.2696973
5. Khoo, Y.S., Nobre, A., Malhotra, R., et al.: Optimal orientation and tilt angle for maximizing in-plane solar irradiation for PV applications in Singapore. IEEE J. Photovoltaics **4**, 647–653 (2014). https://doi.org/10.1109/JPHOTOV.2013.2292743
6. Rahman, M.M., Selvaraj, J., Rahim, N.A., Hasanuzzaman, M.: Global modern monitoring systems for PV based power generation: a review. Renew. Sustain. Energy Rev. (2017). https://doi.org/10.1016/j.rser.2017.10.111
7. Degener, S.: The internet of things: enabling intelligent solar assets. Renew. Energy Focus **17**, 136–137 (2016). https://doi.org/10.1016/j.ref.2016.06.004
8. Adhya, S., Saha, D., Das, A., et al.: An IoT based smart solar photovoltaic remote monitoring and control unit. In: 2016 2nd International Conference on Control, Instrumentation, Energy Communication (CIEC), pp. 432–436 (2016)
9. Jihua, Y., Wang, W.: Research and design of solar photovoltaic power generation monitoring system based on TinyOS. In: 2014 9th International Conference on Computer Science Education, pp. 1020–1023 (2014)
10. Liu, G., Qiu, H., Zhu, L., Chen, Y.: Architecture and experiment of remote monitoring and operation management for multiple scales of solar power plants. In: 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), pp. 2489–2495 (2017)

11. Mashal, I., Alsaryrah, O., Chung, T.-Y., et al.: Choices for interaction with things on Internet and underlying issues. Ad Hoc Netw. **28**, 68–90 (2015). https://doi.org/10.1016/j.adhoc.2014.12.006
12. Yokotani, T., Sasaki, Y.: Comparison with HTTP and MQTT on required network resources for IoT. In: 2016 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC), pp. 1–6 (2016)
13. Naik, N.: Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In: 2017 IEEE International Systems Engineering Symposium (ISSE), pp. 1–7 (2017)
14. Yocto-Amp - Tiny isolated USB ammeter (AC/DC). http://www.yoctopuce.com/EN/products/usb-electrical-sensors/yocto-amp. Accessed 27 Mar 2017
15. nmon for Linux | Main / HomePage. http://nmon.sourceforge.net/pmwiki.php. Accessed 27 Mar 2017
16. Kansal, P., Bose, A.: Bandwidth and latency requirements for smart transmission grid applications. IEEE Trans. Smart Grid **3**, 1344–1352 (2012). https://doi.org/10.1109/TSG.2012.2197229