Energy Management-as-a-Service Over Fog Computing Platform

Mohammad Abdullah Al Faruque, *Member, IEEE*, Korosh Vatanparvar, *Student Member, IEEE*

***Abstract*—By introduction of microgrids, energy management is required to control the power generation and consumption for residential, industrial, and commercial domains, e.g. in residential microgrids and homes. Energy management may also help us to reach zero net energy for the residential domain. Improvement in technology, cost, and feature size have enabled devices everywhere, to be connected and interactive, as it is called Internet of Things (IoT). The increasing complexity and data, due to the growing number of devices like sensors and actuators, requires powerful computing resources which may be provided by cloud computing. However, scalability has become the potential issue in cloud computing. In this paper, fog computing is introduced as a novel platform for energy management. The scalability, adaptability, and open source software/hardware featured in the proposed platform enable the user to implement the energy management with the customized control-as-services, while minimizing the implementation cost and time-to-market. To demonstrate the energy management-as-a-service over fog computing platform in different domains, two prototypes of home energy management and microgrid-level energy management have been implemented and experimented.**

***Index Terms*—Fog Computing, Microgrid, Home Energy Management, Control-as-a-Service, Software-as-a-Service, Networking Platform, Internet of Things**

I. Introduction and Related Work

POWER grids are getting more efficient and smarter as the new paradigm of microgrid has been introduced. A microgrid is comprised of distributed generators, energy storage, and loads which may connect to the power grid or operate autonomously [1]–[3]. Hence, an energy management system is essential to control the power generation and consumption. The microgrid has been shown to improve the reliability, efficiency, and profitability for residential and commercial installations [2]. However, in this paper, we focus on the residential domain for demand-side energy management.

In the year 2013, about 40.7% of the U.S. primary energy consumption is due to commercial buildings and residential homes existing in the power grid [4]. Reducing the energy consumption of the buildings and homes may help to decrease this number significantly. For instance, in the U.S. state of California, the California Energy Commission (CEC), planned to reach Zero Net Energy (ZNE) buildings by 2020 [5]. Also, the U.S. Department of Energy (DOE) seeks to develop technologies and techniques to improve the efficiency of the new and existing residential and commercial homes and buildings and thereby reduce the national energy consumption [6]. Hence, the need for energy efficient buildings and homes is growing rapidly. Moreover, the advancement in technology, the possibility of integrating low-power and high performance electronic devices have enabled us to build advanced embedded systems. Also, reduction in the cost and size of devices like: sensors, actuators, network adapters, switches, etc., have provided us with the opportunity to build sophisticated and low-cost energy management systems [7]–[12]. Typically, the reduction in energy consumption may be done using periodic energy cost feedback and remote monitoring the smart devices (demand response) [13], [14]. *In this paper, we focus on specific residential buildings - homes - for managing their energy consumption.*

In order to implement an energy management system, a platform is required which provides interactivity and interoperability between devices and flexibility of operation. In [15], a home energy management has been implemented over a networking platform in order to meet the mentioned requirements in lower cost, however, the operating domains of the system, scalability, heterogeneity, delay-sensitive devices, and controlling cost have not been considered properly.

One of the most important properties to be considered is the probability of penetrating into the consumer market and affordability of the platform for an ordinary consumer. Major requirements for the architecture which influence this penetration and affordability are: 1) interoperability; 2) scalability; 3) ease of deployment; 4) open architecture; 5) plug-n-play capability; and 6) local and remote monitoring [16], [17]. Moreover, meeting these requirements in a single package should also be cost-effective. Since consumers in residential buildings have limited budget and space for deploying the platform, we are considering them as the case study.

A. Motivational Case Study

Home Energy Management (HEM) as an example for energy management system has to meet the above-mentioned policies and rules. While different hardware, software, communication architectures have been proposed and compared by their power consumption, performance, etc. [14], [18]–[22], the cost of implementing the platform like: computing devices, software stack, communication devices, etc. are still high enough that hinder the process of deploying it for ordinary residential users. Moreover, the hardware and software architectures may not be able to handle the growing number of sensors and actuators alongside their heterogeneity. Control4, Honeywell, and various other companies [23]–[26] are providing the customers with HEM platforms which transform an existing home to a smart home. These products implement various functionalities such as: temperature control,

M. A. Al Faruque and K. Vatanparvar are with the Department of Electrical and Computer Engineering, University of California at Irvine, Irvine, CA 92697 USA e-mail: (alfaruqu@uci.edu).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2471260, IEEE Internet of Things Journal

efficient lighting, and management of smart devices. All these features are provided by a single device. Hence, the scalability, adaptability, and cost of the platforms may become an issue for customers. For instance, the products are strict about the number of devices connected to them for management and processing. Also, scaling the products for larger homes, may become significantly expensive. Furthermore, the users do not have the option of customizing the services they require and they have to purchase the whole package as it is.

B. Research Challenges and Concept Overview

In summary, the problem of implementing energy management platform poses the following major challenges:

1) Performance, interoperability, and interactivity among heterogeneous devices in energy management platform.
2) Ability to customize the services, adaptability, and scalability of the energy management platform for various types of buildings, homes, and applications.
3) Cost of implementing the energy management platform, hardware, and software stack.

C. Our Novel Contributions

To address the above-mentioned challenges, a novel platform for energy management system is proposed that employs:

1) Interoperability, scalability, adaptability, and connectivity between smart devices over fog computing platform (see Section II) which consists of:
2) Low-power and low-cost devices for computation, storage, and communication.
3) Open source software architecture and hardware infrastructure (open architecture) built on a fog computing platform provides us with the ability to scale the platform (scalability).
4) The energy management software or control is implemented as a service using Devices Profile for Web Services (DPWS) [17], [19], [27] which also used for the discovery purpose to provide the plug-n-play feature. This Service-Oriented Architecture also abstracts the communication and hardware heterogeneity.

As shown in Fig. 1, fog computing brings connectivity and interoperability to the smart devices. Data transfer with the cloud and need for computation power from the cloud have been eliminated in this platform. Energy management has been implemented over this platform as a service and two prototypes have been implemented to demonstrate its features.

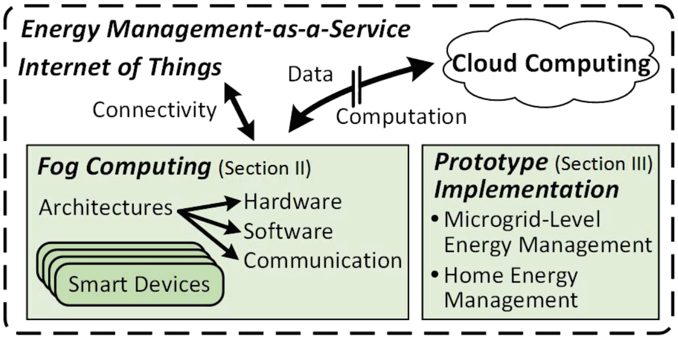


Fig. 1. Our Energy Management-as-a-Service Over Fog Computing Platform. The Dashed and Highlighted Parts are Our Novel Contributions.

II. Fog Computing Platform

Platform for an energy management system should have the features and meet the requirements mentioned in Section I in order to be economical and affordable for ordinary costumers. **Internet of Things (IoT)** *has become the new paradigm for connecting intelligent and self-configurable devices in a dynamic and global network platform. It overcomes the gap between the physical world and their representation in information systems [28]–[33].*

Hence, IoT has been seen as a possible platform for modern technologies, e.g. energy management system, smart home, and smart grid. The domain where this platform operates is scalable and dependent on the application, e.g. HEM monitors and controls the devices within a home [28].

Moreover, IoT may comprise of billions of devices that can sense, communicate, compute, and actuate. Although, the number of devices is highly dependent on the domain of the IoT operation, e.g. home or microgrid, it is growing rapidly over the years, even within a home. This has changed the traditional and manual way of controlling towards more cyber-integrated controlling. Also, this huge amount of data produced by the IoT will be possibly managed, processed, and analyzed, as part of the managing procedure [34]. Hence, **cloud computing** as a new way of centralized computing, integrates with IoT in order to provide the computing power [23]. Moreover, cloud computing provides the customers with infrastructure, platform, software, and sensor network, as services [12], [35]–[43]. These services are guaranteed to have the reliability and performance stated in the agreements, which are essential for the energy management platform.

However, increasing the number of devices in the IoT, i.e. energy management system, using cloud computing, may increase the response time and latency for some delay-sensitive devices such that it violates the timing requirements [44]–[46]. Hence, fog computing has been introduced to alleviate some of the above-mentioned problems [47].

Fog computing *moves the paradigm of cloud computing further to the edge of network. It is a platform which also may provide the IoT with the capability of pre-processing the data while meeting the low latency requirements [29].*

As in energy management system, the platform should be adaptive and scalable regardless of the system size. It also should provide the devices with the performance, interoperability, and interactivity features in order to help them transfer and process the data generated in an adequate response time. The energy management - as a control application - is implemented on fog computing platform. The energy management platform can be used for any type of buildings and various domains of operation, e.g. home or microgrid. The energy management may have various purposes like: 1) monitoring and metering the power consumption of each device, e.g. home power consumption; 2) managing the energy consumption by controlling the devices efficiently, e.g. intelligent lighting, Electric Vehicle (EV) charger [48], Heating, Ventilation, and Air Conditioning (HVAC) management, etc. The energy management platform is a system of systems. In order to design the platform, hardware (see Section II-A), software (see Section II-B), and communication (see Section II-C) architectures should be defined and integrated (see Section II-D) properly for these systems.

A. Hardware Architecture

The hardware of the energy management platform comprises of multiple devices, depending on its operating domain, e.g. home or microgrid. These devices may be categorized as following, based on their functionalities:

1) **Connecting:** these devices provide the connectivity among the existing and compatible devices. The wires, sockets, and antennas are considered as this category.
2) **Gateway:** various devices may have different connectivity standards and protocols, e.g. ZigBee, Bluetooth, and Ethernet [49]. The gateway devices establish a compatible connectivity between multiple devices if required.
3) **Sensor:** the energy management system needs to monitor the environment for timely changes, e.g. weather, light, energy price, etc. The sensors may digitize the analog signal generated by the environment.
4) **Actuator:** the energy management system may decide to configure multiple devices according to environment changes, in order to optimize a variable, e.g. energy consumption. These configurable devices are considered as actuators which may be locally or remotely controlled.
5) **Computing:** the devices that store, process, and analyze the data in the system. They may also implement sophisticated controllers to configure the actuating devices.

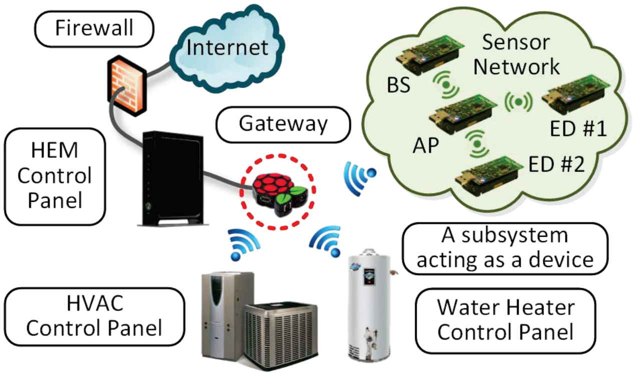


Fig. 2. The Hardware Architecture of the Fog Computing Platform for HEM.

All the devices existing in an energy management platform are considered as connected nodes in the network of the IoT. Moreover, multiple devices or nodes might be grouped as a subsystem in order to perform a single function. In a HEM platform, multiple smart devices, e.g. HVAC, are available to be monitored (sensed) and controlled (actuated) in order to reduce the total energy consumption. The essential computing device which handles all the monitoring and controlling tasks is called the **HEM control panel**. The main job of it is to discover and monitor different devices in the platform dynamically, handle different time-flexible load requests, and trigger and command the devices accordingly based on the algorithms implemented, in order to optimize a variable. To manage the energy consumption of a home, the HEM platform might need to retrieve information about the environment of the home. The physical world condition may be monitored by multiple sensors (e.g. temperature, humidity, light, etc.) in different places of the home. Moreover, different types of sensors may be implemented on a single device (e.g. TelosB mote module). Also, a smart device may have a computing node, implemented inside of it. This computing device, **device control panel**, enables the user or HEM control panel with the capability of configuring the device or a subsystem (see Fig. 2), in lower level and much more detail than the HEM control panel may be capable of, directly. Fig. 2 shows the hardware architecture used for communication and computation in the fog computing platform. As you see in the figure, the gateway may be eliminated, if the device or subsystem is capable of communicating with the platform directly.

B. Software Architecture

The computing nodes implement controllers to gather, store, process, and analyze data and manage devices. The open source and user-configurable routers run a distribution of Linux available in [50] on a MIPS processor. Using these routers as computing nodes may help the developer to easily program the controllers, compile, and run them on the router.

The sensor devices, e.g. TelosB mote modules are compatible with TinyOS [51] which is an open source operating system (available in [52]) designed for low-power wireless sensors. Therefore, the algorithms implemented for monitoring, commanding the sensors and receiving data from them may be programmed easily and dynamically on the sensors. This flexibility may help the developers to program the sensors based on their requirements. In other words, different routing and discovery algorithms for different kinds of sensors (e.g. temperature, humidity, and light) may be implemented in different scenarios.

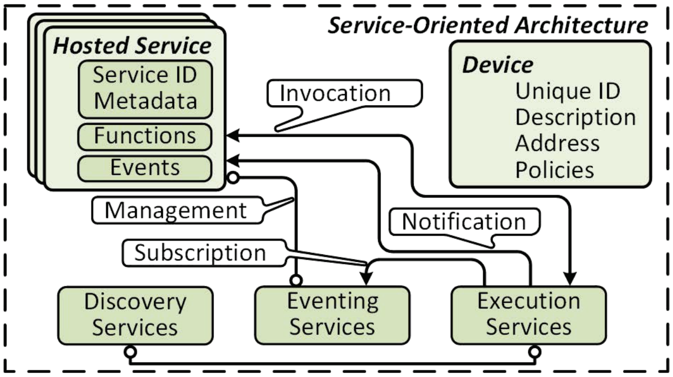


Fig. 3. Demonstration of the Software Architecture Implemented by SOA.

The control panels for each subsystem manage their devices in its own network through a predefined protocol. However, all the subsystems and devices connected in the main network need to follow a unique protocol defined by the HEM control panel (e.g. ZigBee). Therefore, the communications between devices are built around Devices Profile for Web Services (DPWS) from the Web Services for Devices (WS4D). Also, it relies on SOAP-over-UDP, SOAP, WSDL, and XML Schema. This protocol stack may be used for sending secure messages from device to device in a heterogeneous platform. The developers may utilize the WS4D-gSOAP toolkit available in [53] to implement the services needed for each device. Devices may host various DPWS-compliant services using:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2471260, IEEE Internet of Things Journal

1) **WS-Addressing:** provides an addressing mechanism for web services as well as messages in a transport-neutral matter.
2) **WS-Discovery:** provides a way to make the services discoverable by leveraging a protocol based on IP multicast.
3) **WS-MetadataExchange:** defines different data types and the operations to receive a metadata.
4) **WS-Transfer:** is used to transfer the metadata and is very similar to HTTP.
5) **WS-Eventing:** describes a protocol which allows web services to subscribe to or accept subscriptions for event notification messages.

Since the web services are platform-agnostic, using Service-Oriented Architecture (SOA), the platform maintains its flexibility of adding new devices, discovering devices and their respected services, synchronization between devices, and transferring structured data between them. Also, in HEM platform, the devices or the controllers required by a user may be added and managed as a service (Control-as-a-Service). Moreover, to establish synchronization capability and interoperability between devices in the network, e.g. HEM control panel, the device may host 'eventing' web services. In this way, the devices will be notified of any changes or events related to other devices if they have subscribed to those events. Fig. 3 shows how these services are implemented and configured in the platform in order to provide communication and certain functionalities required for the devices.

C. Communication Architecture

The devices in the platform communicate using the existing network adapters and interfaces. For instance, the routers in the HEM platform may have Wireless, Ethernet, Bluetooth, Universal Serial Bus (USB), etc., according to their specifications. Since a unique protocol (e.g. ZigBee) should be used for the main network, these interfaces need to be converted using gateway devices for the desired protocol, if necessary.

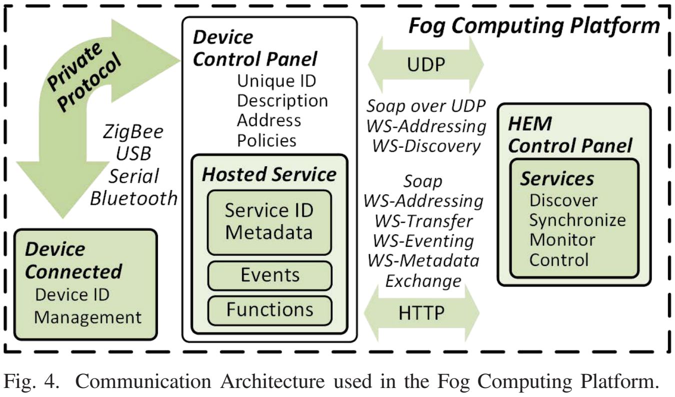


Fig. 4. Communication Architecture used in the Fog Computing Platform.

The sensor devices, e.g. TelosB mote modules, may use the standards-based, low-power wireless technology ZigBee to communicate with each other. Moreover, the standard used for the communication between different nodes may be Smart Energy Profile (SEP 2.0), which is a standard for IP-based control for home energy management and it is supported by these devices, but this profile is not ready yet (standard, protocol stack and hardware). Furthermore, the flexibility of the ZigBee networks can handle about 65535 devices in a network. All the devices are tagged using a unique Identification (ID) number. These IDs are hard coded into the sensor devices while programming and maybe used for routing and transferring data in the same network. Also, the services help the HEM control panel to discover new devices added via plug-n-play. The HEM control panel will verify and authorize the new devices connected to the HEM platform.

To further extend the range that sensor network supports, we may utilize a two-level hierarchical network. In other words, some of the sensor devices are programmed to only sense, which are called End Devices (ED) and may be placed in different corners of a room. Then, a sensor node may be programmed to act as an Access Point (AP) and connects to all the end devices in that room. Also, in a higher level, another sensor device may act as a Base Station (BS) which connects to multiple access points in all the rooms. The end devices transfer the data through their access points and then to the base station. The number of access points connected to each base station or the number of end devices connected to the access points are the variables which are adjusted based on the structure of the building and the requirements. The base station sends all the data gathered from all end devices to the HEM control panel directly or indirectly (using gateway), at the final stage. The gateway used here is a Raspberry Pi. The sensor nodes may communicate with a Raspberry Pi using serial connection over ZigBee standard, and then the Raspberry Pi connects to the HEM control panel through Ethernet. However, if the router has the capability of communicating via ZigBee directly, or the router had the compatible driver to control the sensor module through USB, the Raspberry Pi would be eliminated. Fig. 4 demonstrates the protocols and web services used for the communications between devices.

Furthermore, the user may interface with the platform using the web pages designed for the control panel of each device. The HEM control panel interface is setup on the main router. However, the interfaces for other devices may be provided by the vendors of the devices. Moreover, Asynchronous JavaScript and XML (AJAX) technique may be utilized to retrieve the information from the devices and view it on the web page and trigger different functions and services implemented on the control panel in order to process data. Furthermore, connecting the HEM control panel to the Internet enables the user to monitor and control the home locally or remotely through internet.

D. Integration of Architectures

As we have explained in the last three subsections, the flexibility and low cost infrastructure helps us to add any device, sensor, actuator, and their respected services. The functions and controllers are implemented as services. The services of the devices have to be defined using WS4D to be able to communicate with the routers and each other. On the other hand, if a subsystem designed for an application-specific purpose, needs to be added to the platform, regardless of its own protocol and architecture, it needs to be compatible with the protocol stack used in the HEM platform, or we can make it compatible by using gateway devices.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2015.2471260, IEEE Internet of Things Journal

III. Prototype Implementation

To test the energy management platform, a HEM and a microgrid-level prototypes have been implemented to demonstrate different domains of operation for energy management.

HEM Prototype: the HEM platform is implemented for one home as shown in Fig. 5. The detailed specifications for the hardware used are listed in Table I. In the home, multiple smart devices such as: HVAC, water heater, and EV charger are implemented. Each device is monitored and controlled by the HEM control panel. Also, the devices have their own control panels to monitor their status and set their configurations. In this prototype, all the devices are software modeled, however, in the real-world implementation, the control panels of the devices will be provided by their vendors.

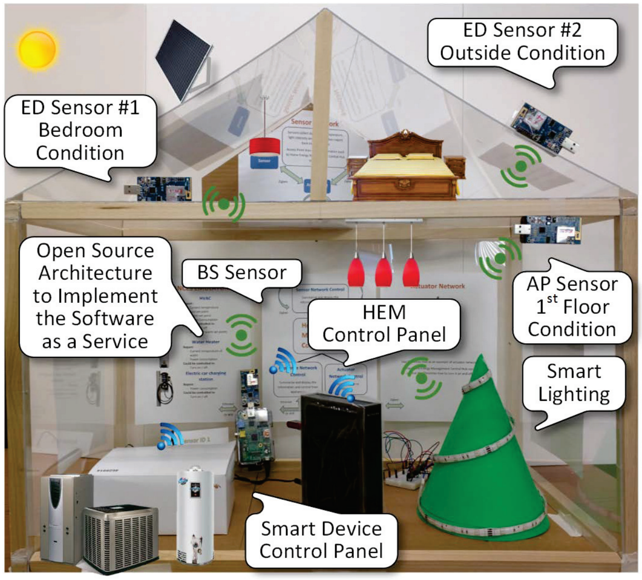


Fig. 5. HEM Prototype Demonstrating the Fog Computing Platform.

The home is being monitored by a network of four sensor devices (TelosB mote modules). The sensor network is defined as a subsystem inside the platform (system of systems). They sample the temperature, humidity, and lighting inside and outside of the home. Also, different types of commands to enable/disable the sensors or set the configurations, e.g. sampling rate for the sensors, may be sent by the sensor control panel to each device by specifying the ID number of that device.

In the HEM platform, a HVAC controller has been implemented as a service. Algorithm 1 illustrates the pseudocode for managing the HVAC in order to reduce its power consumption. The controller receives the temperature set points configured by the user (T_c, T_h), the Demand Response (DR) signal (DR) sent by the transformer, the current room temperature (T_r), and HVAC operation mode ($mode$) which toggles between $cooler$ and $heater$, as inputs. The outputs of this controller are the adjusted temperature set points and the status of the HVAC ($status$) which toggles between on and off. Initially, the threshold for turning on/off the HVAC is defined. The controller turns off/on the HVAC system based on the operation mode and the temperature difference between the room and the user preference. The adjusted HVAC status and the temperature set points are sent to the HVAC.

TABLE I
The Hardware Specifications of the HEM Prototype.

Router	Product Name	NETGEAR
	Product Number	WNR3500L
	WiFi Performance	300 Mbps
	WiFi Band	2.4 GHz
	Security	WPA/WPA2-PSK
		SPI NAT Firewall
		DoS Attack Prevention
	Processor	480 MHz MIPS 74K
	Memory	128 MB NAND Flash
	Ports	4x Ethernet
		1x USB 2.0
Sensor	Product Name	TelosB Mote
	Product Number	TPR2420CA
	Processor	16-bit
	Program Memory	Flash 48 KB
	RAM	10 KB
	Communication	Serial UART
		RF 2.4 to 2.4835 GHz
		USB
	Sensors	Light 320 to 730 nm
		IR 320 to 1100 nm
		Humidity 0 to 100% RH
		Temperature -40 to 123.8˚C
Gateway	Product Name	Raspberry Pie
	Model	B
	Processor	700 MHz
	RAM	512 MB
	Communication	1x Ethernet
		2x USB 2.0
		HDMI

Algorithm 1: Smart HVAC Control-as-a-Service

```
Input: Set Points $T_c, T_h$
Input: DR Signal $DR$
Input: Room Temperature $T_r$
Input: HVAC Mode $mode$
Output: Set Points $T_c, T_h$
Output: HVAC Status $status$
// define threshold for turning on/off the HVAC
$Threshold = 1$
// limit set points when DR signal is triggered
if $DR$ == $true$ then
    $T_c = 79$
    $T_h = 65$
/* turn on/off the HVAC based on temperature, set
   points, and operation mode                     */
if $mode$ == $heater$ then
    if $T_h - T_r > Threshold$ then
        $status = on$
    else
        if $T_r > T_h + Threshold$ then
            $status = off$
if $mode$ == $cooler$ then
    if $T_r - T_c > Threshold$ then
        $status = on$
    else
        if $T_r < T_c - Threshold$ then
            $status = off$
return $status, T_c, T_h$
```

EV charger controller has been also implemented as a service in the HEM platform. Algorithm 2 illustrates a controller which schedules the charging time of the battery in order to reduce the electricity cost while meeting the departure time of the user. The controller receives user-specified departure time ($0 \leq t_d \leq 23$), the current time of the day ($0 \leq t_c \leq 23$), and the current battery status ($0 \leq SoC \leq 100$). The output of the controller is the adjusted current to charge the EV battery (I). Initially, total battery capacity ($Capacity$), the start time (t_{os}) and end time (t_{oe}) for the off-peak hours, and the maximum charge rate which the charger is able to provide (max_I), are defined. The controller schedules the charging process over the off-peak hours first. Then, if more energy is required, more charging will be allocated during the on-peak hours. The charge rate is assigned based on the current time.

Algorithm 2: Smart EV Charger Control-as-a-Service

```
Input: Departure Time $t_d$
Input: Current Time $t_c$
Input: Battery Status $SoC$
Output: Charge Rate $I$
// define the battery total capacity
$Capacity = 60KWh$
// define start and end time for off-peak hours
$t_{os} = 22$
$t_{oe} = 11$
// define the maximum charge rate possible
$max_I = 4KW$

// evaluate capacity remaining to charge
$Capacity_{rem} = (100 - SoC)/100 * Capacity$
// the time interval when there is off-peak hours
$Time_{off} = (min(t_{oe}, t_d) - max(t_{os}, t_c))\%24$
// the charge rate during off-peak hours
$I_{off} = min(max_I, Capacity_{rem}/Time_{off})$
// the charged capacity during off-peak hours
$Charged_{off} = I_{off} * Duration_{off}$
/* evaluate capacity remaining after charging
   during off-peak hours                                */
$Capacity_{rem} = Capacity_{rem} - Charged_{off}$
/* the time interval remaining to charge
   subtracting the off-peak hours                       */
$Time_{rem} = (t_d - t_c - Time_{off})\%24$
// the charge rate during on-peak hours
$I_{on} = min(max_I, Capacity_{Remaining}/Time_{Remaining})$

// deciding the current charge rate based on time
if $t_c \in Duration_{off-peak}$ then
    $I = I_{off}$
else
    $I = I_{on}$
return $I$
```

Microgrid-Level Prototype: the energy management platform in the microgrid-level comprises three homes connected to a transformer. A control panel is implemented in the transformer-level to monitor and manage the power consumption of each home. The transformer-level control panel monitors the load of each home and the current condition of the transformer. Based on this information and the controller, the control panel may decide to send commands to the HEM for each home, in order to reduce their power consumption by a specific value, which is called DR. The reduction in the power consumption may prevent over-loading and over-heating the transformer which may improve the efficiency and longevity. The transformer control panel is implemented in one router and other three homes with their emulated smart devices are implemented in other six routers (see Fig. 6).

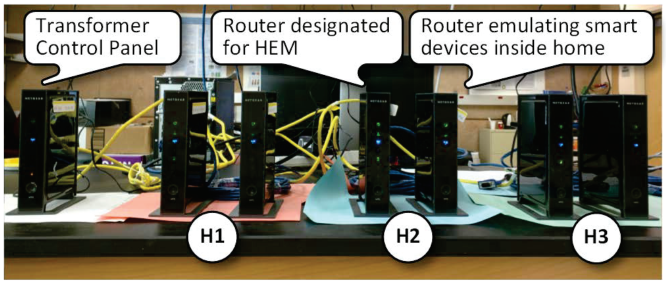

Transformer Control Panel
Router designated for HEM
Router emulating smart devices inside home
H1
H2
H3

Fig. 6. Microgrid-Level Energy Management Prototype.

In the microgrid-level energy management platform, a transformer management has been implemented as a service. Algorithm 3 illustrates the pseudocode for monitoring the homes connected to the transformer in order to prevent overloading the transformer. The controller receives an array of the homes connected (H) and the current load on the transformer ($load_c$), as inputs. The output of the controller is the overloading status of the transformer ($status$). Initially, the maximum load in which the transformer can operate efficiently ($load_{max}$) and the maximum load a home should have ($home_{max}$) are defined. The controller checks whether the transformer is overloaded. In case of overload, a DR signal is sent to each home which consumes more than the threshold.

Algorithm 3: Smart Transformer Control-as-a-Service

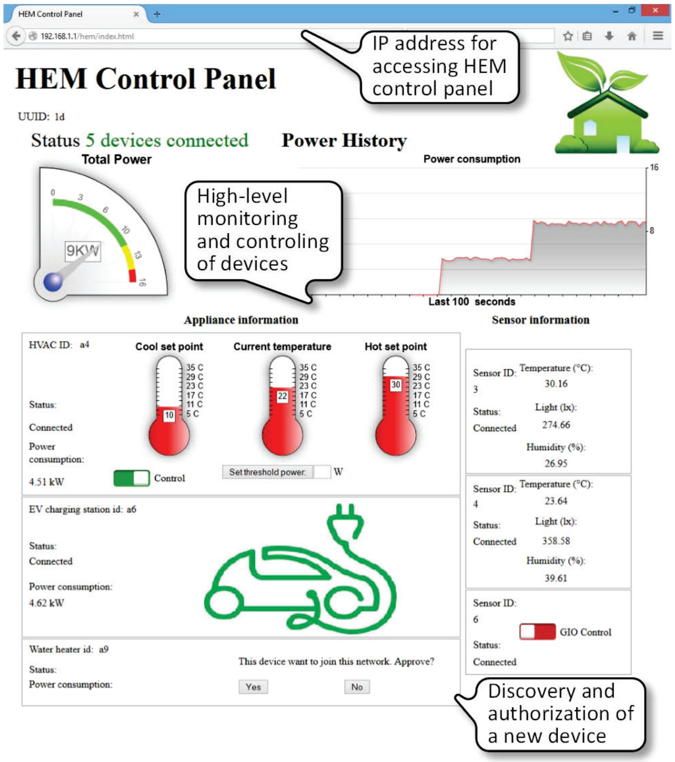
```
Input: Homes Connected to the Transformer $H$
Input: Current Transformer Load $load_c$
Output: Overload Status $status$
// define transformer rating load
$load_{max} = 20KW$
// define threshold for each home
$home_{max} = 4KW$
// assume no overloading in transformer
$status = false$

// check whether the transformer is overloading
if $load_c > load_{max}$ then
    for $h \in H$ do
        if $h_{load} > home_{max}$ then
            /* trigger DR signal for the home with
               higher consumption than threshold  */
            $Trigger\_DR(h)$
    $status = true$
return $status$
```

IV. Experimental Results on Case Studies

To demonstrate and experiment the features of the energy management over fog computing platform as a service, multiple services are implemented and evaluated on the platform (see Section III). The complete working prototypes of our case study demonstrations are presented in [54] (for detailed specifications see Section II).

A. Home Energy Management

In the HEM-level, the services like managing sensor network, efficient lighting, smart EV charging, and smart HVAC



IP address for accessing HEM control panel

HEM Control Panel

UUID: 1d

Status 5 devices connected

Power History

High-level monitoring and controling of devices

Appliance information

HVAC ID: a4

Status: Connected

Power consumption: 4.51 kW

Cool set point: 10

Current temperature: 22

Hot set point: 30

EV charging station id: a6

Status: Connected

Power consumption: 4.62 kW

Water heater id: a9

Status:

Power consumption:

This device want to join this network. Approve?

Sensor information

Sensor ID: 3 Status: Connected Temperature (°C): 30.16 Light (lx): 274.66 Humidity (%): 26.95

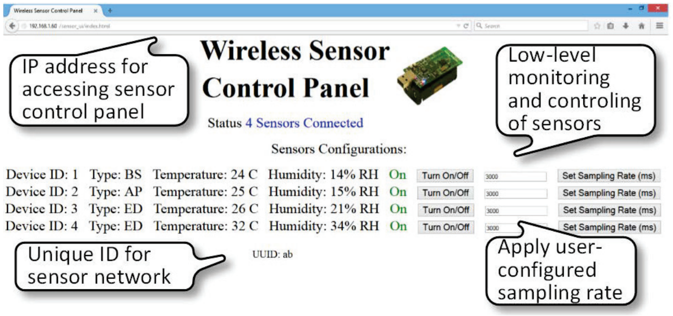
Sensor ID: 4 Status: Connected Temperature (°C): 23.64 Light (lx): 358.58 Humidity (%): 39.61

Sensor ID: 6 Status: Connected GIO Control

Discovery and authorization of a new device

Fig. 7. HEM Control Panel for Monitoring and Controlling Smart Devices.

controlling, are added to the energy management system. The algorithms for each of the services are implemented in the HEM fog computing platform (Algorithms 1 and 2). The HEM control panel views the current devices connected to the platform. Through HEM platform, the user may turn on/off each device (see Fig. 7). Fig. 8 shows the user interface for managing the sensor network in the home. The user may check the temperature and humidity in different parts of the home. Using smart EV charging, the EV charger will efficiently decide when and how to charge the EV such that it does not violate power consumption restrictions while meeting the departure time specified by the user (see Fig. 10). The HVAC consumption is mainly dependent on the temperature set points adjusted by the user. By monitoring the temperature in different places of the home and knowing the energy price, the HVAC controller may efficiently decide on the temperature set points (see Fig. 9). This process may reduce the power consumption of the HVAC while maintaining the home temperature within the defined range.


Wireless Sensor Control Panel
192.168.1.60 /sensor_ui/index.html
Search

IP address for accessing sensor control panel

Wireless Sensor Control Panel

Status 4 Sensors Connected

Low-level monitoring and controling of sensors

Sensors Configurations:

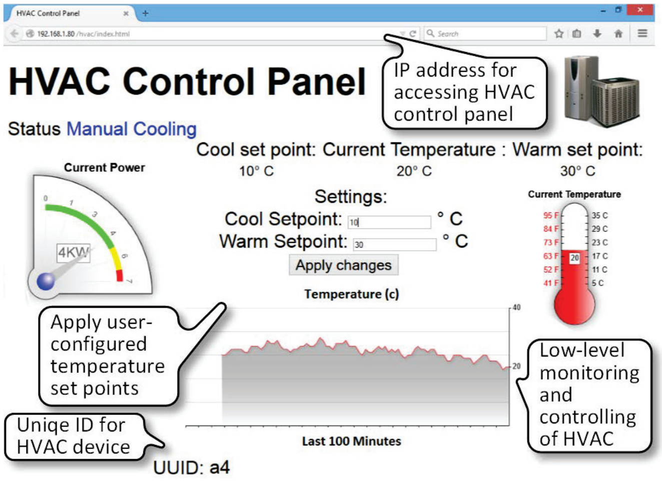
Device ID: 1	Type: BS	Temperature: 24 C	Humidity: 14% RH	On	Turn On/Off	3000	Set Sampling Rate (ms)
Device ID: 2	Type: AP	Temperature: 25 C	Humidity: 15% RH	On	Turn On/Off	3000	Set Sampling Rate (ms)
Device ID: 3	Type: ED	Temperature: 26 C	Humidity: 21% RH	On	Turn On/Off	3000	Set Sampling Rate (ms)
Device ID: 4	Type: ED	Temperature: 32 C	Humidity: 34% RH	On	Turn On/Off	3000	Set Sampling Rate (ms)

Unique ID for sensor network

UUID: ab

Apply user-configured sampling rate

Fig. 8. Sensor Network Control Panel for Managing Sensors' Configurations.



HVAC Control Panel

IP address for accessing HVAC control panel

Status Manual Cooling

Cool set point: 10° C Current Temperature : 20° C Warm set point: 30° C

Settings:

Cool Setpoint: 10 ° C

Warm Setpoint: 30 ° C

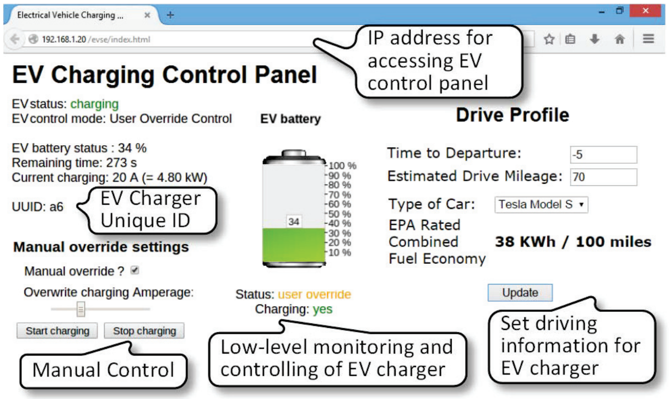
Apply user-configured temperature set points

Low-level monitoring and controlling of HVAC

Uniqe ID for HVAC device

UUID: a4

Fig. 9. HVAC Control Panel for Monitoring and Controlling HVAC Power.



EV Charging Control Panel

IP address for accessing EV control panel

EV status: charging

EV control mode: User Override Control

EV battery status : 34 %

Remaining time: 273 s

Current charging: 20 A (= 4.80 kW)

UUID: a6

EV Charger Unique ID

Manual override settings

Manual override ? ☑

Overwrite charging Amperage:

Status: user override

Charging: yes

Drive Profile

Time to Departure: -5

Estimated Drive Mileage: 70

Type of Car: Tesla Model S

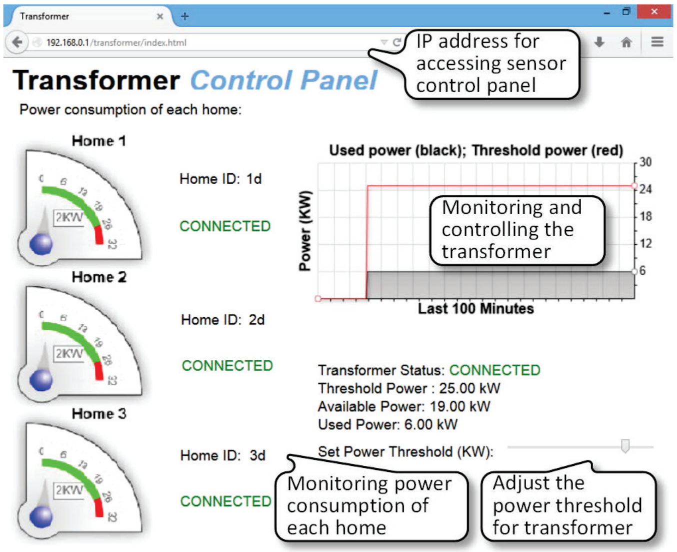
EPA Rated Combined Fuel Economy 38 KWh / 100 miles

Manual Control

Low-level monitoring and controlling of EV charger

Set driving information for EV charger

Fig. 10. EV Control Panel for Monitoring and Controlling Battery Charging.


Transformer
192.168.0.1/transformer/index.html
0 6 12 19 26 32
2KW
Used power (black); Threshold power (red)
30
24
18
12
6
Power (KW)
Last 100 Minutes

Transformer Control Panel

IP address for accessing sensor control panel

Power consumption of each home:

Home 1 Home ID: 1d CONNECTED

Home 2 Home ID: 2d CONNECTED

Home 3 Home ID: 3d CONNECTED

Monitoring and controlling the transformer

Transformer Status: CONNECTED

Threshold Power : 25.00 kW

Available Power: 19.00 kW

Used Power: 6.00 kW

Set Power Threshold (KW):

Monitoring power consumption of each home

Adjust the power threshold for transformer

Fig. 11. Transformer Control Panel for Monitoring and Controlling Power.

B. Microgrid-Level Energy Management

In the microgrid-level energy management, a transformer powers multiple homes. The services like transformer monitoring, and DR may be added to the energy management system. The algorithms for each of the services are implemented in the microgrid-level fog computing platform (Algorithm 3). Fig. 11 shows the user interface for monitoring the power consumption of each home. The user may define certain power threshold for each home. As shown in Fig. 11, the transformer current condition and the total load on it may be monitored. Using this information, the transformer will send DR signals to the home which have violated the threshold so that they may reduce their consumption. Also, the control panel may decide where to get the power from (see Fig. 11).

Both energy management systems have provided the features and met the requirements mentioned in Section I by implementing the cost-effective fog computing platform using off-the-shelf instruments listed in Table I. Also, Table II summarizes the technologies used to provide these features.

TABLE II
SUMMARY OF THE TECHNOLOGIES USED TO IMPLEMENT THE FEATURES REQUIRED FOR THE ENERGY MANAGEMENT PLATFORM.

Feature	Used Technology
Interoperability	WS-Addressing and WS-Transfer enable the devices to communicate with each other.
Interactivity	WS-Eventing and WS-Metadata Exchange enable devices to synchronize each other periodically.
Flexibility	Open hardware/software used for implementing any application or adding new devices.
Scalability	Multiple devices can be connected in a shared network with their own unique IDs.
Ease of Deployment	Control panel web pages leverage HTML, AJAX, and Java scripting in order to provide user-friendly interfaces.
Open Architecture	Raspberry Pi is used as gateway and Linux-based routers and Tiny OS-based TelosB mote sensors are used for connecting and computing.
Plug-n-Play	SOAP-over-UDP is used for discovery and authentication of new devices added to network.
Local/Remote Access	IP addresses are designated for each device and they are also connected to the Internet.
Heterogeneity Abstraction	Service-Oriented Architecture is used to abstract the hardware and communication differences.

V. CONCLUSION

We have seen that energy management is essential for microgrids, homes, and buildings. Hence, in this paper, we have presented a novel energy management which is implemented as a service over fog computing platform. The implementation over fog computing platform provides the flexibility, interoperability, connectivity, data privacy, and real-time features required for energy management. Also, open source software/hardware and the ability to be customized provide the user to add the control as a service to the energy management platform. Therefore, the implementation cost and time-to-market will decrease significantly. To demonstrate the energy management-as-a-service over fog computing platform in different domains, two prototypes of home energy management and microgrid-level energy management have been implemented and experimented.

REFERENCES

[1] M. A. Al Faruque and F. Ahourai, “A Model-Based Design of Cyber-Physical Energy Systems,” *19th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 97–104, 2014.

[2] K. Vatanparvar and M. A. Al Faruque, “Design Space Exploration for the Profitability of a Rule-Based Aggregator Business Model Within a Residential Microgrid,” *IEEE Transactions on Smart Grid*, 2014.

[3] DER Group at LBNL, ”WebOpt”. der.lbl.gov/News/3rd-release-distributed-energy-resources-der-web-optimization-tool-webopt., 2014.

[4] Department of Energy (DOE), ”Buildings Energy Data Book”. buildingsdatabook.eren.doe.gov/TableView.aspx?table=1.1.3., 2014.

[5] California Energy Emission, ”Building Energy Efficiency Program”. www.energy.ca.gov/title24., 2014.

[6] Department of Energy, ”US Department of Energy Strategic Plan”. energy.gov/sites/prod/files/2011_DOE_Strategic_Plan_.pdf., 2014.

[7] H. Farhangi, “The Path of the Smart Grid,” *IEEE Power and Energy Magazine*, vol. 8, pp. 18–28, 2010.

[8] Y. Yao, Q. Cao, and A. V. Vasilakos, “EDAL: An energy-efficient, delay-aware, and lifetime-balancing data collection protocol for wireless sensor networks,” *IEEE 10th International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, pp. 182–190, 2013.

[9] K. Han, J. Luo, Y. Liu, and A. V. Vasilakos, “Algorithm design for data communications in duty-cycled wireless sensor networks: A survey,” *IEEE Communications Magazine*, vol. 51, pp. 107–113, 2013.

[10] S. Sengupta, S. Das, M. Nasir, A. V. Vasilakos, and W. Pedrycz, “An evolutionary multiobjective sleep-scheduling scheme for differentiated coverage in wireless sensor networks,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, pp. 1093–1102, 2012.

[11] M. Yang, Y. Li, D. Jin, L. Zeng, X. Wu, and A. V. Vasilakos, “Software-Defined and virtualized future mobile and wireless networks: A Survey,” *Mobile Networks and Applications*, pp. 1–15, 2014.

[12] L. Wang, F. Zhang, J. A. Aroca, A. V. Vasilakos, K. Zheng, C. Hou, D. Li, and Z. Liu, “GreenDCN: A general framework for achieving energy efficiency in data center networks,” *IEEE Journal on Selected Areas in Communications*, vol. 32, pp. 4–15, 2014.

[13] D. Niyato, L. Xiao, and P. Wang, “Machine-to-Machine Communications for Home Energy Management System in Smart Grid,” *IEEE Communications Magazine*, vol. 49, pp. 53–59, 2011.

[14] P. Palensky and D. Dietrich, “Demand Side Management: Demand Response, Intelligent Energy Systems, and Smart Loads,” *IEEE Transactions on Industrial Informatics*, vol. 7, pp. 381–388, 2011.

[15] K. Vatanparvar, Q. Chau, and M. A. Al Faruque, “Home Energy Management as a Service over Networking Platforms,” *IEEE PES Conference on Innovative Smart Grid Technologies (ISGT)*, 2015.

[16] U.S. Department of Energy Energy Efficiency and Renewable Energy Golden Service Center. Financial Assistant Funding Oppurtunity Announcement. 28 March 2013. ”DE-FOA-0000822: ”Turn Key” Open Source Software Solutions for Energy Management of Small to Medium Sized Buildings”., 2014.

[17] F. Jammes and H. Smit, “Service-Oriented Paradigms in Industrial Automation,” *IEEE Transactions on Industrial Informatics*, 2005.

[18] D. Bian, M. Kuzlu, M. Pipattanasomporn, and S. Rahman, “Assessment of Communication Technologies for a Home Energy Management System,” *IEEE PES Innovative Smart Grid Technologies Conference (ISGT)*, pp. 1–5, 2014.

[19] M. Rahman, M. Kuzlu, M. Pipattanasomporn, and S. Rahman, “Architecture of Web Services Interface for A Home Energy Management system,” *IEEE PES Innovative Smart Grid Technologies Conference (ISGT)*, pp. 1–5, 2014.

[20] D.-M. Han and J.-H. Lim, “Design and Implementation of Smart Home Energy Management Systems based on ZigBee,” *IEEE Transactions on Consumer Electronics*, vol. 56, pp. 1417–1425, 2010.

[21] Y.-S. Son, T. Pulkkinen, K.-D. Moon, and C. Kim, “Home Energy Management System based on Power Line Communication,” *IEEE Transactions on Consumer Electronics*, vol. 56, pp. 1380–1386, 2010.

[22] S. Katipamula, R. M. Underhill, J. K. Goddard, D. Taasevigen, M. Piette, J. Granderson, R. E. Brown, S. M. Lanzisera, and T. Kuruganti, “Small-and Medium-Sized Commercial Building Monitoring and Controls Needs: A Scoping Study,” *Pacific Northwest National Laboratory (PNNL), Richland, WA (US), Tech. Rep*, 2012.