Conceptual Architecture of Building Energy Management Open Source Software (BEMOSS)

W. Khamphanchai, A. Saha, K. Rathinavel
M. Kuzlu, M. Pipattanasomporn and S. Rahman
Virginia Tech – Advanced Research Institute, Arlington, VA

B. Akyol and J. Haack
Pacific Northwest National Laboratory, Richland, WA

Abstract— **The objective of this paper is to present a conceptual architecture of a Building Energy Management Open Source Software (BEMOSS) platform. BEMOSS is an open source operating system that is expected to improve sensing and control of equipment in small- and medium-sized commercial buildings, reduce energy consumption and help implement demand response (DR). It aims to offer: scalability, robustness, plug and play, open protocol, interoperability, cost-effectiveness, as well as local and remote monitoring. In this paper, four essential layers of BEMOSS software architecture -- namely User Interface, Application and Data Management, Operating System and Framework, and Connectivity layers -- are presented. A laboratory test bed to demonstrate the functionality of BEMOSS located at the Advanced Research Institute of Virginia Tech is also briefly described.**

***Index Terms*— Building energy management, open source, and demand response.**

I. Introduction

In the United States, buildings consume over 40% of the country's total energy consumption [1], and 90% of these buildings are either small-sized (<5,000 sqft) or medium-sized (5,000-50,000 sqft). For small- and medium-sized buildings, heating consumption is the dominant end use, followed by lighting, plug loads and cooling [2]. Specifically, Heating, Ventilation, and Air-Conditioning (HVAC), lighting and plug loads account for almost 90% of all consumption in buildings.

The U.S. Department of Energy (DOE) targets to save $2.2 trillion in energy-related costs by reducing building energy use by 50% compared to the 2010 baseline [3]. The study [4] shows that due to the lack of building monitoring and control, significant portion of the energy consumed in buildings is wasted. This is because existing Building Automation Systems (BAS) are still cost-prohibitive and being used mostly in large buildings. BAS are not popular in most small- and medium-sized buildings due to lack of awareness of benefits, lack of inexpensive packaged solutions, and sometimes due to the owner not being the tenant and so finding no incentive to invest in these systems [4].

Recently, more and more commercial products for home/building automation systems have become available to tackle these problems. Some examples of these products are SmartThings [5], Staples Connect™ [6], GE Brillion™ [7], Lowe's Iris [8], Revolv [9] and etc. These solutions allow homeowners or small building owners to monitor or control specific compatible devices. However, there are still certain limitations for users or developers of that particular platform including incompatibility between vendors, limited number of supported devices, standards, communication technologies or data exchange protocols. Even though advanced users or developers can sign up to develop applications or add new devices to the platform, they are limited by the need to rely on specific tools provided by the platform. This makes the development ecosystem/vendor specific for that particular platform.

Some open source building automation solutions are available such as Freedomotic [10], OpenRemote [11], and openHAB [12] etc. These platforms provide open, flexible architectures, and hardware/protocol agnostic tools for developing residential or commercial automation. The intelligence of buildings can be enhanced to implement automated rules, scripts, or event triggering for a specific device(s). However, the solution to make devices coordinate and communicate autonomously and seamlessly together in order to achieve the global objective of a building's owner (e.g. perform DR when the price of electricity is high) has not yet been addressed.

The knowledge gaps discussed above prompted the development of a web-based Building Energy Management Open Source Software (BEMOSS) platform for optimizing electricity usage and implementing demand response (DR) in small- and medium- sized buildings. This opens up demand side ancillary services markets and creates opportunities for building owners, which in turn can help accelerate development of market-ready products like embedded Building Energy Management (BEM) systems and device controllers for HVAC, lighting and plug loads. This also enables utilities and independent system operator (ISOs) to actively leverage DR as a partial substitute for generation reserve or transmission upgrade. This paper discusses the core

This work was supported in part by the U.S. Department of Energy under Grant# DE-EE-0006352.

W. Khamphanchai, A. Saha, K. Rathinavel, M. Kuzlu, and M. Pipattanasomporn are with Virginia Tech – Advanced Research Institute, Arlington, VA 22203 USA (e-mails: kwarodom@vt.edu, avijit@vt.edu, kruthika@vt.edu, mkuzlu@vt.edu, mpipatta@vt.edu). S. Rahman is professor and director of Virginia Tech – Advanced Research Institute, Arlington, VA 22203 USA (e-mail: srahman@vt.edu).

B. Akyol and J. Haack are with Pacific Northwest National Laboratory (PNNL), Richland, WA 99354 (e-mails: bora@pnnl.gov, jereme.haack@pnnl.gov).

concept of the BEMOSS operating system, its software architecture and lab experiment to demonstrate BEMOSS functionalities.

II. BEMOSS CONCEPT

This section discusses key features of BEMOSS, target buildings, types of load controllers, as well as communication technologies and protocols supported by BEMOSS.

A. Key Features

The proposed BEMOSS operating system has following features:

Application: BEMOSS is designed to have an open architecture to make it easy for hardware manufacturers to develop Application Programming Interface (API) to seamlessly interface their devices with BEMOSS. Software developers can also contribute to adding additional BEMOSS functionalities and applications.

Usability: BEMOSS is designed to provide interoperability among various standards and protocols, thus allowing integration of heterogeneous devices in a plug & play and scalable fashion.

Advanced Monitoring: BEMOSS provides real-time monitoring of building energy consumption and device status through a web interface and mobile apps.

Advanced Control: BEMOSS provides advanced algorithms for sensing and control of equipment, helps reduce energy consumption and implement demand response (DR) that can offer different grades of comfort and savings.

Cost-effectiveness: Implementation of BEMOSS is deemed to be cost-effective as BEMOSS is built upon a robust open-source operating system that can operate on a low-cost embedded system, such as Raspberry Pi and BeagleBone.

B. Target Buildings and Types of Load Controllers supported by BEMOSS

BEMOSS mainly targets small- (<5,000 sqft) and medium-sized (between 5,000 and 50,000 sqft) commercial buildings. BEMOSS controls HVAC, lighting and selected plug loads. Load controllers with communication features for each of these target load categories are considered for BEMOSS integration. For HVAC systems, smart thermostats and RTU/VAV controllers are the most popular control devices. For lighting loads, dimmable ballasts and Wi-Fi light switches can be used. Smart plugs can be used as controllers for miscellaneous plug loads. In addition to these controllers, BEMOSS also supports power/energy meters and sensors (e.g., occupancy, light, temperature and humidity).

C. Communication Technologies and Protocols supported by BEMOSS

Various technologies are available that allow communications between BEMOSS and associated load controllers in a building. These include wired technologies, like: Power Line Communication (PLC), Ethernet and Serial (RS-485); as well as, wireless ones, like: ZigBee, Wi-Fi, Z-wave and EnOcean.

Currently, BEMOSS supports the following prevalent communication technologies: Ethernet (IEEE 802.3), Serial (RS-485), ZigBee (IEEE 802.15.4) and Wi-Fi (IEEE 802.11), as summarized in Table I.

TABLE I.
COMMUNICATION TECHNOLOGIES SUPPORTED BY BEMOSS [13]

Technology	Standard/ Protocol	Max. Theoretical Data Rate	Coverage Range
Wired Communication Technologies			
Ethernet	IEEE 802.3	10 Mbps- 1 Gbps	up to 100 m
Serial	RS-485	100 kbps – 35 Mbps	up to 1,200 m
Wireless Communication Technologies			
ZigBee	ZigBee	250 kbps	up to 100 m
	ZigBee Pro	250 kbps	up to 1,600 m
WiFi	802.11x	2-600 Mbps	up to 100 m

Devices communicating using the same communication technology may utilize different data exchange protocols. For BEM, there are many protocols that are popular or becoming popular. These are open standard protocols like: BACnet, Modbus, KNX, M-bus, Web, OpenADR and Smart Energy; as well as proprietary protocols like: LonWorks and DALI.

Currently, BEMOSS supports the following protocols: BACnet, Modbus, Web, OpenADR, ZigBee API and Smart Energy protocols, as summarized in Table II.

TABLE II.
DATA EXCHANGE PROTOCOLS SUPPORTED IN BEMOSS

Data exchange protocol	Application	Allow communications over:				
		Power line	Ethernet	Serial	WiFi	ZigBee
1. BACnet (IP)	Building automation		X		X	
BACnet (MS/TP)				X		
2. Modbus (RTU)	Legacy device communications			X		
Modbus (TCP)			X		X	
3. Web (e.g., XML, JSON, RSS/Atom)	Numerous applications		X		X	
4. ZigBee API	Home/building automation					X
5. OpenADR	Demand response		X		X	
6. Smart Energy	Smart grid	X			X	X

III. BEMOSS ARCHITECTURE

Fig. 1 illustrates BEMOSS software architecture, which comprises the following four layers: User Interface (UI) layer, Application and Data Management layer, Operating System and Framework layer, and Connectivity layer. Each layer is described below.

A. User Interface (UI) Layer

The BEMOSS UI layer has two components: user interaction and user management.

1) User Interaction

BEMOSS UI consists of: (1) web browser interface, and (2) mobile interface. The BEMOSS web application resides in a central server, while BEMOSS mobile apps reside in end user devices.

BEMOSS web interface: BEMOSS web UI is a dashboard

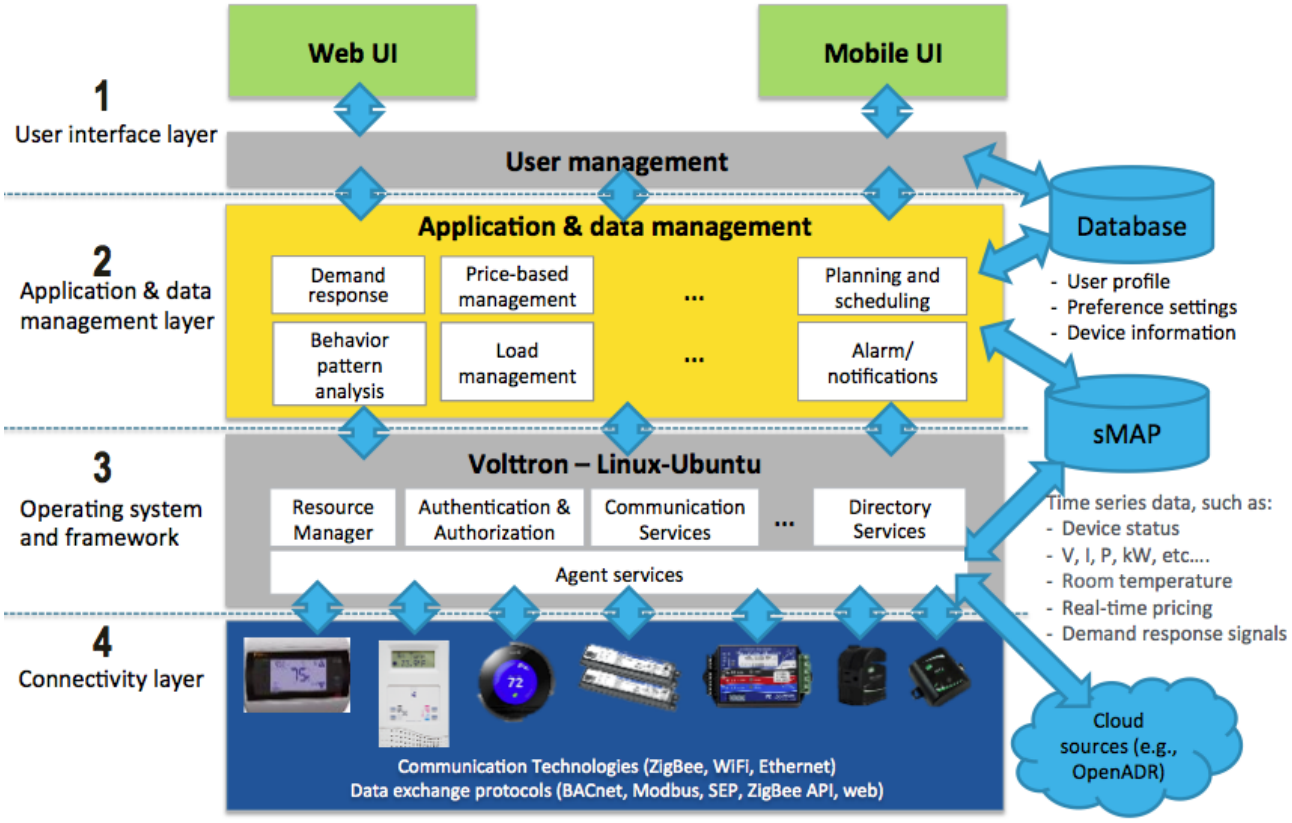

Fig. 1. BEMOSS software architecture.

type interface with visuals and graphs to show current settings of devices in each zone. Authenticated users can also control these devices through the interface. BEMOSS web interface is implemented using the model-view-controller (MVC) architecture [14]. The ***model*** includes data management and data objects from time-series database, and relational database. The ***view*** refers to what is presented to users and how users interact with BEMOSS. The view comprises HTML, CSS and JavaScript [15][16]. The ***controller*** acts as the interface between the model and the view. The controller updates the model when there is a change of the system state or when a user requests for a change. Then, the controller communicates the change in the model to the view. It also handles requests and responses, controls database connections, and handles communications with the Application and Data Management layer.

BEMOSS mobile interface: BEMOSS also provides a mobile interface to monitor and control selected loads in the building. The mobile interface follows the same standards as the web interface. Android [17] and iOS [18] mobile development platforms are chosen, as they are the most widely used among mobile platforms.

2) User Management

In BEMOSS, role-based access control is implemented to allow different levels of access to different individuals. For example, building engineers will have full authority to adjust set points and schedules of loads in buildings, while tenants will have limited access to view current status and historical load data, or control selected loads in specific zones. In BEMOSS, this role-based access control is achieved using access control lists.

B. Application and Data Management Layer

This layer comprises application management and data management, as described below.

1) Application Management

This layer embeds algorithms to allow monitoring and control of hardware devices interfaced with BEMOSS. Examples of possible applications include demand response, price-based management, operation monitoring, energy consumption analysis, load control based on local conditions, alarming notifications, planning and scheduling, data visualization and web services.

2) Data Management

For intelligent decision making, BEMOSS requires data to be collected continuously with time. These ***time-series data*** are usually difficult to manage due to the amount of data accumulated over time. As a result, BEMOSS needs a powerful time-series database that is open source and scalable. After analyzing a few choices, including sMAP [19], InfluxDB [20] and OpenTSDB [21], the BEMOSS team selected sMAP(Simple Measurement and Actuation Profile) for storing BEMOSS time-series data. This is because sMAP performs better compression and extraction with time-series data than the other choices. This feature is crucial to the BEMOSS storage system, where within a fraction of time, the large amount of data has to be collected from multiple devices for archiving and processing.

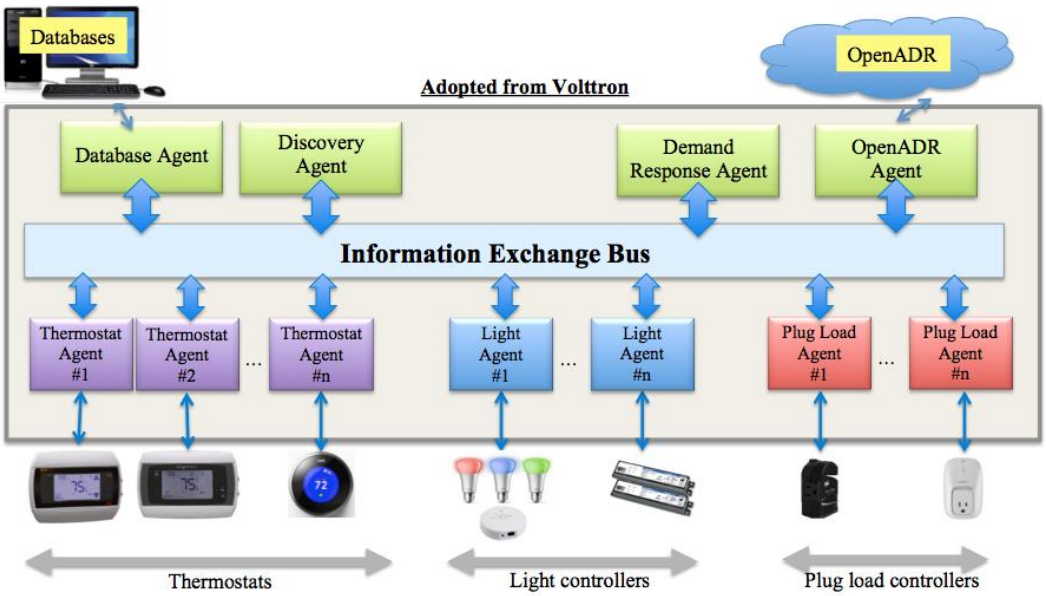

Fig. 2. BEMOSS agent services in the Operating System and Framework layer.

In addition, there is also a need to store ***metadata*** for identifying users, devices, and process controls. Since sMAP cannot store metadata, another database system is required. A relational database management system (e.g., SQLite [22], PostgreSQL [23]) can be used to satisfy the need to store the metadata.

C. Operating System and Framework Layer

In this layer, VOLTTRON™, a distributed agent platform developed by Pacific Northwest National Laboratory (PNNL) [24]-[26], is chosen as the software platform for BEMOSS. VOLTTRON™ is designed to run on small-form-factor computers and is capable of interfacing with legacy devices, maintaining security and managing platform resources, and servicing for applications. VOLTTRON™ platform enables the deployment of intelligent sensors and controllers in residential/commercial buildings and the smart grid. Distributed agents using peer-to-peer communications in VOLTTRON™ cooperate to bring computation closer to data to enable distributed control decisions and data analysis. Intelligent agents residing in VOLTTRON™ are designed to have most of these capabilities: reactive, pro-active, social, mobility, veracity, benevolence, rationality, and learning/adaptation.

With respect to VOLTTRON™ architecture and design, the platform consists of communication services (CS), resource manager (RM), authentication and authorization (AA), directory services (DS), agent instantiation and packaging (AIP) and information exchange bus (IEB) modules. For the proposed BEMOSS software architecture, VOLTTRON™ provides connectivity and abstraction capabilities to BEMOSS.

In terms of ***connectivity***, devices with different communication technologies and different data exchange protocols are able to integrate into the VOLTTRON™ platform. VOLTTRON™ includes drivers for interfacing with MODBUS BACnet based devices. Drivers can be written for devices which do not use those protocols based on the availability of an Application Programming Interface (API) for that device. These drivers then publish data to VOLTTRON™ enables a device to communicate, be monitored and controlled by an assigned intelligent agent with assistance from the Connectivity layer described in the next subsection. A device can also response to an event triggered by an associated agent or an external environment specified in the Application and Data Management layer.

In terms of ***abstraction***, VOLTTRON™ allows applications to communicate with devices via its message bus instead of requiring them to use the particular device protocol. Drivers can be added to the platform to increase the number of supported devices. This enables BEMOSS developers to develop an agent to control a device or coordinate multiple devices regardless of communication technologies or data exchange protocols used.

Fig. 2 shows important types of BEMOSS agents available/developed in VOLTTRON™. Each type of agents has different functionalities as described below.

- *Discovery agent:* This agent searches for a new device that can be detected by BEMOSS, identifies device model, classifies device type, and specifies appropriate API class for a device. Discovery agent also instantiates a suitable control agent for each device.
- *Control agents:* Control agents include thermostat agents, lighting load agents and plug load agents. These agents are generated automatically to monitor, communicate and control hardware devices after they were discovered by the Discovery agent.
- *Database agent:* This agent communicates and interfaces with the sMAP database to store time-series data and with the relational database to store metadata.
- *OpenADR agent:* This agent receives demand response request from a utility or an aggregator (e.g., EnerNOC [27]) through a web service on the cloud. It then notifies selected agents of a DR event.

- *Demand response agent:* This agent communicates and coordinates with control agents. This is to reduce peak demand consumption during a certain period, according to the price signal or the demand reduction signal received from the OpenADR agent.

D. Connectivity Layer

This layer takes care of the communication between the Operating System and Framework layer and all physical hardware devices. This encompasses different communication technologies, data exchange protocols and device functionalities. For this layer to work properly, physical communication mediums should be properly setup and configured. In this section, communication technologies and their corresponding setups for connectivity layer, data exchange protocols and message interpretation functions, and supported functionalities of hardware devices are discussed.

1) Communication Setup

The description below discusses how hardware devices that use different communication technologies can be interfaced with BEMOSS:

Wi-Fi: A Wireless Local Area Network (WLAN) can be setup using existing wireless infrastructure in a building or by using wireless routers to setup a new wireless network. Both Wi-Fi hardware devices and BEMOSS must be configured to join the same network.

ZigBee: To enable BEMOSS to communicate with ZigBee devices, BEMOSS must be connected with a ZigBee coordinator to allow a personal area network (PAN) to be formed. Then, all ZigBee devices must be configured to join associated PAN(s).

Ethernet: All Ethernet devices are connected via Ethernet cables and switches. To be interfaced with BEMOSS, Ethernet devices must be configured such that they are in the same subnet with BEMOSS. Routers can be used to route messages between devices in different subnets.

Serial (RS-485): Serial devices are connected using physical wires. These devices may also be interfaced with BEMOSS via a router (for example, a BACnet IP to MS/TP router).

Once physical communication mediums are setup and properly configured, different layers of communication technologies can take care of message transmit/receive functions over physical mediums. Hence, the connectivity layer of BEMOSS only needs to take care of data exchange protocols and message interpretation functions.

2) Data Exchange and Message Interpretation

Currently, BEMOSS supports BACnet, Modbus, SEP, ZigBee API and web service-based data exchange protocols (e.g., XML, JSON). Software components/drivers in the connectivity layer: (a) handle details of data exchange protocols based on the API of hardware devices, (b) provide a simplified API for the upper layer of BEMOSS based on required device functionalities, and (c) interpret messages between these two APIs. An API class is developed in the Operating System and Framework layer for each compatible device, which simplifies the task of agent developers by providing abstraction over details of communication technologies and protocols.

3) Hardware Device Functionality

As mentioned before, BEMOSS primarily targets three types of hardware devices: HVAC, lighting and plug load controllers. These devices have different functional requirements. For example, a thermostat is required to provide room temperature data or set-point control, whereas a plug load controller (e.g., a smart plug) is required to provide ON/OFF control. This is why the Connectivity layer in BEMOSS also needs to address differences in required functionalities of devices in addition to their communication technologies and protocols.

IV. Laboratory Demonstration

A laboratory has been set up at Virginia Tech Advanced Research Institute to demonstrate BEMOSS features and capabilities. It includes a computer to host the BEMOSS operating system and selected hardware devices that use different communication technologies and data exchange protocols. Fig. 3 shows the lab setup with a PC and selected load controllers. Additional load controllers are being integrated into BEMOSS. In the future, BEMOSS is expected to be hosted in a small-form-factor computer, such as Raspberry Pi or BeagleBone.

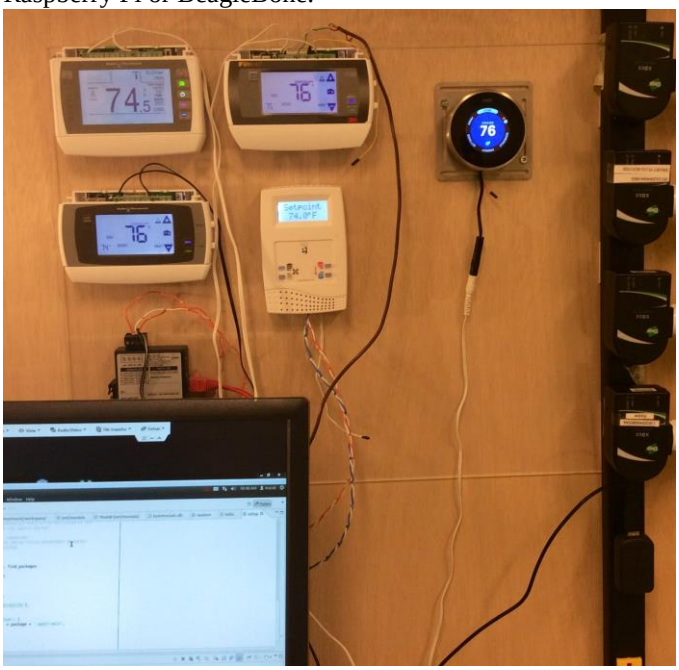

Fig 3. BEMOSS lab setup at VT-ARI.

V. Conclusion

This paper introduces the BEMOSS operating system aiming at improving sensing and control of equipment, reducing energy consumption and enabling demand response of small- and medium-sized commercial buildings. Its core concept and software architecture are presented to showcase how BEMOSS is designed and developed to be a cost-effective, cross-standard, and open-source software platform. The open-source architecture will enable developers with different skill sets to work on different layer(s), thus enabling rapid deployment of BEMOSS. Compatibility across standards will enable vendors to build products suiting customer needs.

Seamless integration and plug-and-play feature of BEMOSS will provide customers with a hassle-free experience. In short, BEMOSS is envisioned to fill a long-awaited gap in energy management in small- to medium-sized buildings.

VI. REFERENCES

[1] Buildings Energy Data Book, Buildings Sector, [Online]. Available: http://buildingsdatabook.eren.doe.gov/ChapterIntro1.aspx?1#4–November 1, 2011. Retrieved: Apr 2014.

[2] Buildings Energy Data Book, Commercial Sector, [Online]. Available: http://buildingsdatabook.eren.doe.gov/ChapterIntro3.aspx. Retrieved: Apr 2014.

[3] Building Technologies Office, Office of Energy Efficiency & Renewable Technology, Department of Energy [Online]. Available: http://energy.gov/eere/efficiency/buildings. Retrieved: Apr 2014.

[4] "Small- and Medium-Sized Commercial Building Monitoring and Controls Needs: A Scoping Study," October 2012, http://www.pnnl.gov/main/publications/external/technical_reports/PNNL-22169.pdf. Retrieved: Apr 2014.

[5] SmartThings - Home Automation [Online]. Available: www.smartthings.com/. Retrieved: Apr 2014.

[6] Staples Connect powered by Linksys [Online]. Available: www.staples.com/sbd/cre/marketing/staples-connect/. Retrieved: Apr 2014.

[7] GE Brillion Connected Appliances [Online]. Available: www.geappliances.com/connected-home-smart-appliances/. Retrieved: Apr 2014.

[8] Iris Smart Home Management System [Online]. Available: www.lowes.com/cd_Iris_239939199. Retrieved: Apr 2014.

[9] Revolv: Life & Smart Home Awesomation [Online]. Available: http://revolv.com/. Retrieved: Apr 2014.

[10] Freedomotic Open Source Building Automation [Online]. Available: http://freedomotic.com/. Retrieved: Apr 2014.

[11] OpenRemote Open Source Automation Platform [Online]. Available: http://www.openremote.org/. Retrieved: Apr 2014.

[12] openHAB empowering the smart home [Online]. Available: http://www.openhab.org/. Retrieved: Apr 2014.

[13] M. Kuzlu, M. Pipattanasomporn, "Assessment of communication technologies and network requirements for different smart grid applications", in: *IEEE Innovative Smart Grid Technologies (ISGT) Conference*, pp. 1-7, (Washington D.C., USA, 2013)

[14] M. J. Mahemoff, L. J. Johnston, "Handling multiple domain objects with Model-View-Controller", *In Proc. Technology of Object-Oriented Languages and Systems*, pp.28-39, (Melbourne, Australia, 1999).

[15] Twitter Bootstrap Theme [Online]. Available: http://getbootstrap.com/getting-started/. Retrieved: Apr 2014.

[16] jQuery Cross-platform javascript library [Online]. Available: http://api.jquery.com. Retrieved: Apr 2014.

[17] Android Developer Portal Reference [Online]. Available: https://developer.android.com/reference/packages.html. Retrieved: Apr 2014.

[18] iOS developer library – Design Resources [Online]. Available: https://developer.apple.com/library/ios/navigation/. Retrieved: Apr 2014.

[19] Simple Measurement and Actuation Profile [Online]. Available: https://code.google.com/p/smap-data/w/list. Retrieved: Apr 2014.

[20] Influx time series database [Online]. Available: http://influxdb.org/docs/api/http.html. Retrieved: Apr 2014.

[21] OpenTSDB time series database [Online]. Available: http://opentsdb.net/getting-started.html. Retrieved: Apr 2014.

[22] SQLite relational database management system [Online]. Available: https://sqlite.org/index.html. Retrieved: Apr 2014.

[23] PostgreSQL relational database management system [Online]. Available: http://www.postgresql.org/. Retrieved: Apr 2014.

[24] B. Akyol, J. Haack, C. Tews, B. Carpenter, A. Kulkarni, and P. Craig "An Intelligent Sensor Framework for the Power Grid*." In Proc. ASME Conference 2011*, 1485 (2011), DOI=10.1115/ES2011-54619

[25] B. Akyol, J. Haack, S. Ciraci, B. Carpenter, M. Vlachopoulou and C. Tews "VOLTTRON: an agent execution platform for the electric power system." *In Proceedings of the 3rd International Workshop on Agent Technologies for Energy Systems* (Valencia, Spain, June 5, 2012).

[26] J. Haack, B. Akyol, B. Carpenter, C. Tews, and L. Foglesong "VOLTTRON: An agent platform for smart grid." *In Proceedings of the 4th International Workshop on Agent Technologies for Energy Systems*, pp.1367-1368, (Minnesota, USA, May 10, 2013).

[27] EnerNOC Open Source for an Open Grid [Online]. Available: http://open.enernoc.com/. Retrieved: Apr 2014.

VII. BIOGRAPHIES

Warodom Khamphanchai (S'11 - IEEE) received the M.Eng. degrees in Electric Power System Management from Asian Institute of Technology (AIT), Thailand in 2011 and the B.Eng. degree in Electrical Engineering from Chulalongkorn University, Thailand in 2009. He is currently pursuing his PhD degree in the Electrical and Computer Engineering Department, Virginia Tech. His research interests are power systems operation and control, power system optimization, renewable energy systems, distributed microgrids, artificial intelligence and multi-agent systems.

Avijit Saha (S'06 – IEEE) received his B.Sc. degree in Electrical and Electronic Engineering (EEE) from Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh, in 2009. He served as a Lecturer at the department of EEE in Ahsanullah University of Science and Technology, Dhaka, Bangladesh from March 2010 to July 2011. Currently he is working towards his Ph.D. degree in Electrical Engineering in Virginia Tech. His research interests include the areas of smart grid, demand response, energy management systems, embedded systems, and communication technologies.

Kruthika Rathinavel received her Bachelor of Engineering (Honors) degree in Electronics and Instrumentation from Birla Institute of Technology and Science (BITS, Pilani), Goa, India in 2008. She has over four years of experience as a software consultant, when she was associated with CGI (previously Logica), CAI (Tyco Electronics) and NTT Data. She is currently pursuing her MS degree in the Electrical and Computer Engineering Department, Virginia Tech. Her research interests are information storage and retrieval, network security, data mining, software engineering, and machine learning.

Murat Kuzlu (M'11 - IEEE) joined Virginia Tech's Department of Electrical and Computer Engineering as a post-doctoral fellow in 2011. He received his B.Sc., M.Sc., and Ph.D. degrees in Electronics and Telecommunications Engineering from Kocaeli University, Turkey, in 2001, 2004, and 2010, respectively. In 2006, he joined the Energy Institute of TUBITAK-MAM, where he worked as a senior researcher at the Power Electronic Technologies Department. His research interests include smart grid, demand response, smart metering systems (AMR, AMI, AMM), wireless communication and embedded systems.

Manisa Pipattanasomporn (S'01, M'06, SM'11 – IEEE) joined Virginia Tech's Department of Electrical and Computer Engineering as an assistant professor in 2006. She serves as one of the principal investigators (PIs) of multiple research grants from the U.S. National Science Foundation, the U.S. Department of Defense and the U.S. Department of Energy, on research topics related to smart grid, microgrid, energy efficiency, load control, renewable energy and electric vehicles. Her research interests include renewable energy systems, energy efficiency, distributed energy resources, and the smart grid.

Saifur Rahman (S'75, M'78, SM'83, F'98 - IEEE) is the director of the Advanced Research Institute at Virginia Tech where he is the Joseph Loring Professor of electrical and computer engineering. He also directs the Center for Energy and the Global Environment at the university. From 2009-2013 he served as a vice president of the IEEE Power & Energy Society and a member of its Governing Board. He is a member-at-large of the IEEE-USA Energy Policy Committee. Professor Rahman was the chair of the US National Science Foundation Advisory Committee for International Science and Engineering from 2010 to 2013. Between 1996 and 1999 he served as a program director in engineering at NSF. In 2006 he served as the vice president of the IEEE Publications Board, and a member of the IEEE Board of Governors. He is a distinguished lecturer of IEEE PES, and has published in the areas of smart grid, conventional and renewable energy systems, load forecasting, uncertainty evaluation and infrastructure planning.

Bora Akyol (SM'2013 – IEEE) is a chief research scientist at the Pacific Northwest National Laboratory. Bora's work focuses on distributed computing, networking, wireless communications and sensors as well as software architectures and platforms including the VOLTTRON platform that underpins the BEMOSS system discussed in this paper. Bora has a Ph.D. and an M.Sc. from Stanford University. Stanford, CA and a B.S.E.E. from Bilkent University, Ankara, Turkey and has also been granted over 15 patents for inventions related to ASIC design, security, networking and distributed software platforms.

Jereme Haack (SM'2012 – IEEE) is a senior research scientist at the Pacific Northwest National Laboratory. Jereme's research focuses on software agent programming and modeling, information visualization, and cyber security. Previous work includes a project for bio-inspired cyber security. His current focus is on the VOLTTRON platform as detailed in this paper. Jereme has a BS in Computer Science and a BS in Mathematics from Doane College. He is also the president of the local chapter of the IEEE Computer Society.