An Open Source Extensible Smart Energy Framework

SBIR Phase 1: Final Scientific Report

3/24/2017

Submitted By:

V-SQUARED
2362 SW Madison St.
Portland, OR 97205

Linda Rankin
linda@vee-squared.com

Acknowledgements:
Professor Robert Bass, Annie Clark, Tylor Slay of Portland State University
Conrad Eustis, Portland General Electric

This material is based upon work supported by the U.S. Department of Energy, Office of Science under Award Number SC0015791.

Summary

Aggregated distributed energy resources are the subject of much interest in the energy industry and are expected to play an important role in meeting our future energy needs by changing how we use, distribute and generate electricity. This energy future includes an increased amount of energy from renewable resources, load management techniques to improve resiliency and reliability, and distributed energy storage and generation capabilities that can be managed to meet the needs of the grid as well as individual customers. These energy assets are commonly referred to as Distributed Energy Resources (DER).

DERs rely on a means to communicate information between an energy provider and multitudes of devices. Today DER control systems are typically vendor-specific, using custom hardware and software solutions. As a result, customers are locked into communication transport protocols, applications, tools, and data formats. Today's systems are often difficult to extend to meet new application requirements, resulting in stranded assets when business requirements or energy management models evolve.

By partnering with industry advisors and researchers, an implementation DER research platform was developed called the Smart Energy Framework (SEF). The hypothesis of this research was that an open source Internet of Things (IoT) framework could play a role in creating a commodity-based eco-system for DER assets that would reduce costs and provide interoperable products.

SEF is based on the AllJoyn™ IoT open source framework. The demonstration system incorporated DER assets, specifically batteries and smart water heaters. To verify the behavior of the distributed system, models of water heaters and batteries were also developed. An IoT interface for communicating between the assets and a control server was defined. This interface supports a series of "events" and telemetry reporting, similar to those defined by current smart grid communication standards.

The results of this effort demonstrated the feasibility and application potential of using IoT frameworks for the creation of commodity-based DER systems. All of the identified commodity-based system requirements were met by the AllJoyn framework. By having commodity solutions, small vendors can enter the market and the cost of implementation for all parties is reduced. Utilities and aggregators can choose from multiple interoperable products reducing the risk of stranded assets.

Based on this research it is recommended that interfaces based on existing smart grid communication protocol standards be created for these emerging IoT frameworks. These interfaces should be standardized as part of the IoT framework allowing for interoperability testing and certification. Similarly, IoT frameworks are introducing application level security. This type of security is needed for protecting application and platforms and will be important moving forward. Recommendations are that along with DER-based data model interfaces, platform and application security requirements also be prescribed when IoT devices support DER applications.

Purpose of the Research

Aggregated distributed energy resources are the subject of much interest in the energy industry and are expected to play an important role in meeting our future energy needs by changing how we use, distribute and generate electricity. This energy future includes an increased amount of energy from renewable resources, load management techniques to improve resiliency and reliability, and distributed energy storage and generation capabilities that can be managed to meet the needs of the grid as well as individual customers. These energy assets are commonly referred to as Distributed Energy Resources (DER).

The Smart Energy Framework (SEF) research effort is targeted towards enabling thousands, or even millions, of smart devices connected via one or more communication networks, which would provide a utility or an aggregator the ability to control DERs in aggregate to respond to grid conditions or power needs. Related to this work are standards that have been defined and are being used, such as OpenADR[1] and IEEE 2030.5 (formerly known as Smart Energy Profile 2.0) that define the characteristics of the communication network and associated data models. However, integration of asset devices into these DER systems today commonly use customized software and hardware.

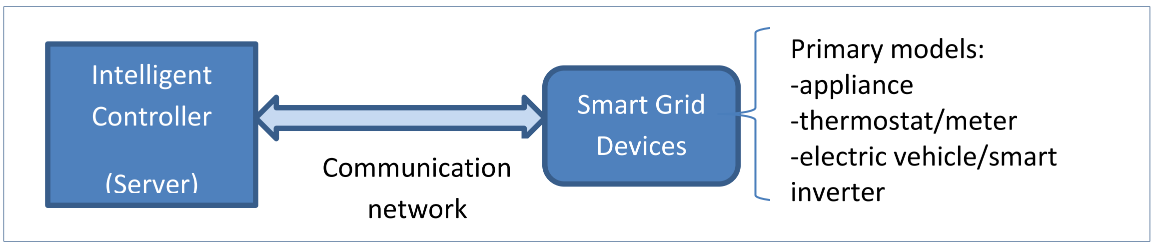

Figure 1 DER Management System, Utility/Aggregator Perspective

A commodity-based ecosystem for DER would enable cost-effective scaling to millions of devices. This implies that the platform, as well as the software and services, use common operating systems (Linux), communication protocols (ex. HTTPS), and hardware (ARM, or x86). A commodity-based ecosystem allows for participation by small vendors, particularly those that are unable to develop and support end-to-end solutions. Utilities and aggregators have more choices as a result, and interoperability between the devices reduces the risk of stranded assets.

Our hypothesis is that emerging open source Internet of Things (IoT) software frameworks could be used as a means to enable vendors to participate in the emerging DER market and could be an important ingredient in the commodity-based ecosystem. IoT frameworks are currently developed for other commodity markets such as home lighting controls, home entertainment and home security systems. Features and data models needed for industrial and health care applications are also being considered. Such a framework would need to meet the following criteria:

- Uses commodity hardware and software. This includes general purpose processors such as ARM or Intel architecture, the various transports such as Ethernet or WiFi and operating systems such

[1] More information about OpenADR, including the OpenADR 2.0 specification is available at www.openadr.org.

as Linux and Windows. Systems based on these commodity solutions are ubiquitous and inexpensive.

- Provides a seamless plug-in for the customer. This implies that connection to the network is automatic and can tolerate customer “abuse”.
- Automatic discovery of the application layer, and integration in to a system or device that acts as an “Intelligent Controller”.
- Supports standard internet security protocols such as Public Key Infrastructure (PKI), and incorporates best practice security measures (this also includes protecting the user privacy).
- A horizontal solution that enables multiple communications providers and multiple smart grid device applications. A horizontal solution is where the software functions are layered with well published interfaces. An example is the network protocol stack.
- Accommodates and/or can be extended to support different emerging smart grid data models such as those defined in Smart Energy Protocol 2.0 (SEP 2.0), and OpenADR.
- Ability to securely provision and upgrade software.
- A robust ecosystem that includes an industry alliance with multiple vendors that provides testing and certification infrastructure.

Our goal was to implement a small distributed energy framework platform based on the system architecture illustrated below using the AllJoyn[2] IoT framework. Our research was to determine the viability of using such a framework for DER.

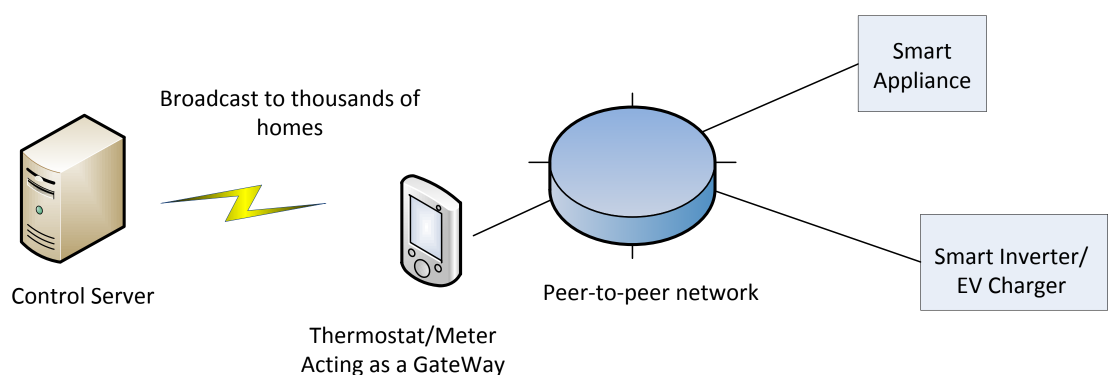

Figure 2 Demonstration Target System Architecture

The remainder of this paper provides details about our approach, results and findings. Finally we close with potential applications and recommendations based on this work.

[2] More information about AllJoyn is available at www.allseenalliance.org, or www.en.wikipedia.org/wiki/AllJoyn.

Approaches and Findings

Our approach was to build the ingredients for the target platform as shown below using an emerging Internet of Things (IoT) framework called AllJoyn. AllJoyn is an open source software framework that allows devices to communicate with other devices around them. By implementing such a topology, we would be able to gain understanding regarding the strengths and weaknesses of the AllJoyn framework with respect to creating a commodity-based DER framework. Using AllJoyn, we created an interface specific to our DER needs loosely based on OpenADR 2.0. This, in turn, would provide illumination on the extensibility of the IoT framework.

Development Effort

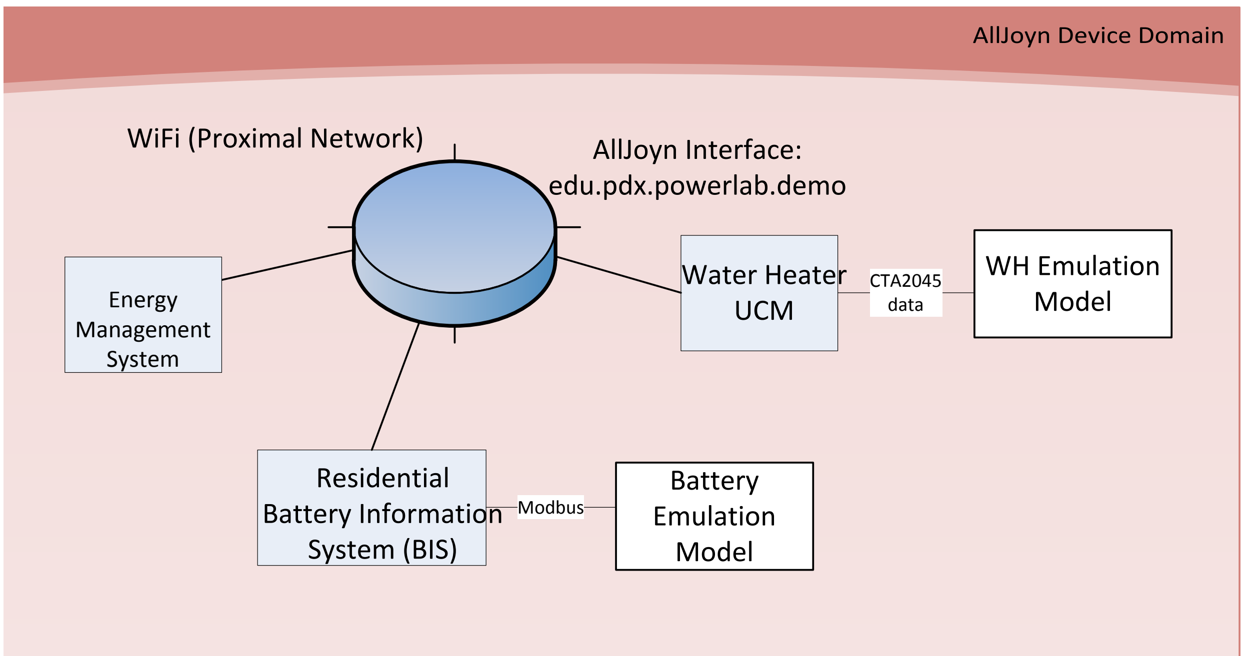

Figure 3 SBIR Phase 1 Research system

During Phase 1, our initial development emphasis was on the architecture, design and development of the proximal network agents. The goal was to determine the feasibility of AllJoyn as a platform for connecting DER assets within the home or commercial establishment and to develop a control interface for these systems. To support software development and understanding, and the eventual deployment of the agents in real systems, emulations of the water heater and residential battery system were also developed.

The Energy Management System, Battery Information System and Water Heater UCM were all AllJoyn agents and shared the following characteristics:

- All agents were developed on Ubuntu Linux (most within a Virtual Machine).
- All agents were based on AllJoyn core version 16.04.00.
- C++ was the language that was used for all agents.
- GitHub private repositories were used for all developed code.

- Each agent supported the AllJoyn interface that was defined for the demonstration (named "edu.pdx.powerlab.demo").

AllJoyn Interface

Each agent in an AllJoyn-based system implements one or more interfaces. Agents advertise their presence and what interfaces they support when attaching to the AllJoyn bus. The interface that announces an agents presence is the "About" interface and is required to be supported by all agents. Other interfaces are provided in the core framework; Notification, Control Panel, for example. The interfaces, their capabilities, and definitions are fundamental to the interoperability of AllJoyn agents and therefore testing for compliance to interfaces is a key role of the certification program.

An interface definition collects a group of methods, signals and/or properties and are defined using XML (Extensible Markup Language). The XML for the interface defined for the demonstration (edu.pdx.powerlab.demo) is illustrated in the figure below. The goal of the demonstration interface was to determine if an IoT framework would have sufficient horsepower to support the control of DER assets.

The demonstration interface allows for sending a series of "events", each with an associated "duration", where each event is a "command" to be executed by the assets. These events are annotated as "noReply", but AllJoyn allows for defining a response. The response could be used to indicate OptIn or OptOut, for example. Telemetry provided by the asset is represented as a set of string/value pairs, providing flexibility on the types of data that is reported.

```
# types: 'q'=uint16_t, 'n'=int16_t, 's'=string, 'i'=int32_t, 'u'=uint32_t, 'v'=various
<node>
    <interface name="edu.pdx.powerlab.demo">
      <description language="en">
         Development and demonstration interface for battery and water heater  control.
      </description>
      <method name="EnergyMethod">
        <description language="en">
          An array of one or more energy command events; duration,cmd,rate
          Starttime for the sequence of commands is "now".
          If not defined, Duration defaults to "forever".
         </description>
         <arg name="cmds" type="a(snn)" direction="in"/>
         <description language="en">
           ARRAY[(string)cmd,(int_16)rateInKW,(int_16)DurationInSeconds,]
           Commands: "charge", "discharge", "idle", "shed", "endshed"
         </description>
         <annotation name="org.freedesktop.DBus.Method.NoReply" value="true"/>
      </method>
      <property name=AssetName" type="s" access="read">
         <annotation name="org.freedesktop.DBus.Property.EmitsChangedSignal"
value="False"/>
      </property>
      <property name="Telemetry" type="{sv}" access="read">
         <description language= "en">
           Required telemetry element is "TimeStamp" {su},
           Other keys can be flexible allowing experimentation with information received
           from the asset system. Example KEYS: "Sys_Rated_Energy, Sys_SOC,
Sys_DC_Volts, and Sys_DC_Current"
         </description>
         <annotation name="org.freedesktop.DBus.Property.EmitsChangedSignal"
value="True"/>
      </property>
    </interface>
</node>
```

Figure 4 AllJoyn interface definition for research demonstration

The ability to represent "events" in this manner is similar to the EiEvent services described in OpenADR 2.0b specification. Similarly, the telemetry report is similar to EiReport. OpenADR is also described using XML, although the syntax required by AllJoyn is unique to its bus implementation. Therefore, implementing data models defined by OpenADR would require a translation to AllJoyn syntax and capabilities.

While there is much more development required, it is encouraging that an IoT interface can support more complex data models such as those that would defined by smart grid standards.

AllJoyn Agents

The AllJoyn agents implemented in the system are the Energy Management System (EMS), the Battery Information System (BIS), and a Water Heater UCM (Universal Control Model). Each of these agents supports the demonstration AllJoyn interface (i.e. edu.pdx.powerlab.demo). The research platform also includes emulations of a battery smart storage system as well as a smart water heater, both of which are implemented with the Python programming language.

The EMS is an agent that represents a placeholder for the functionality implemented by an aggregator. It implements the "client" functionality of AllJoyn, responding to the connection announcements broadcast by the BIS and UCM. The EMS can connect to more than one agent, understands when agents become disconnected and stores agent property information. The AllJoyn framework, bus, and the demonstration interface provided sufficient functionality to implement a control application on the EMS that managed one or more battery systems to meet a target load profile.

The BIS acts as an AllJoyn server, broadcasting it capabilities and interface when joining the AllJoyn bus. It receives DER commands from the EMS via the AllJoyn demonstration interface and translates them to appropriate Modbus register read/write operations. The registers are implemented in a battery emulation program that also models the battery behavior. The battery emulation program models a physical system that is an 8 kW rate, 60 kWh system intended to be installed “behind the meter”[3]. The BIS also receives information status from the emulated battery and transforms them to telemetry information to be provided to the EMS.

Similarly, the Water Heater UCM is also an AllJoyn server and receives DER commands from the EMS and transforms them to CTA 2045[4] data commands. CTA 2045 is a modular communication interface to be used for energy management (formerly known as USNAP). CTA 2045 defines a communication module, a socket standard, and a protocol for to be implemented by a UCM (Universal Communication module), and a SGD (Smart Grid Device). In the demonstration system the SGD is represented by the emulated water heater.

Both the UCM and SGD in the demonstration system support a subset of the CTA 2045 Basic DR application messages and the Data Link Messages. Basic DR Application messages such as “shed” are transformed to the CTA-2045 byte codes, transferred and acted upon by the SGD that models the state of the water heater. The CTA-2045 data link ACK/NAK exchange is also supported. The intent is that the UCM agent could be ported to physical hardware in the future (similar to the BIS agent) primarily through the addition of the serial CTA-2045 protocol.

Findings

The system demonstrates the following capabilities:

- The EMS is able to connect and control multiple AllJoyn agents.
- Robust connect/disconnect and discovery behavior has been observed.
- AllJoyn is able to connect agents running on the same platform we well as between assets connect in a local WiFi network. Connection is seamless and automatic.
- Energy methods such as “charge”, “discharge”, etc., are broadcast to the assets and reflected in changes in state in the modeled assets.
- Telemetry information on asset state is provided by and sent back to the EMS. This information is stored by the EMS. This makes the “last known state” available at the controller interface in the event that the agent becomes disconnected.

We were able to implement the AllJoyn proximal system in a relatively short amount time. Open source, sample applications, and a strong developer community all were significant in enabling our success.

[3] The battery system is funded and supported by Portland General Electric.
[4] *www.usnap.org/press-releases*

AllJoyn agents are designed to support multiple interfaces; this allows for standard interfaces such as configuration, provisioning to be defined. While not included in our demonstration platform, an AllJoyn Gateway, Security Manager, and XMPP connector are currently available as open source. The Gateway and XMPP connector are intended to be used to connect to an external server.

While not implemented in our system, the AllJoyn Security 2.0 features are worth mentioning. The AllJoyn system provides a security framework for applications to authenticate each other and communicate with encrypted messages (i.e. end-to-end application security). It is up to the application to determine the level of security and types of features that are required. For example, the ability to set up a secure session with a remote application that can be authenticated is an important capability when considering features such as remote configuration and provisioning. The AllJoyn security architecture, includes support for a software managed key store or an external Trusted Platform Module (TPM).

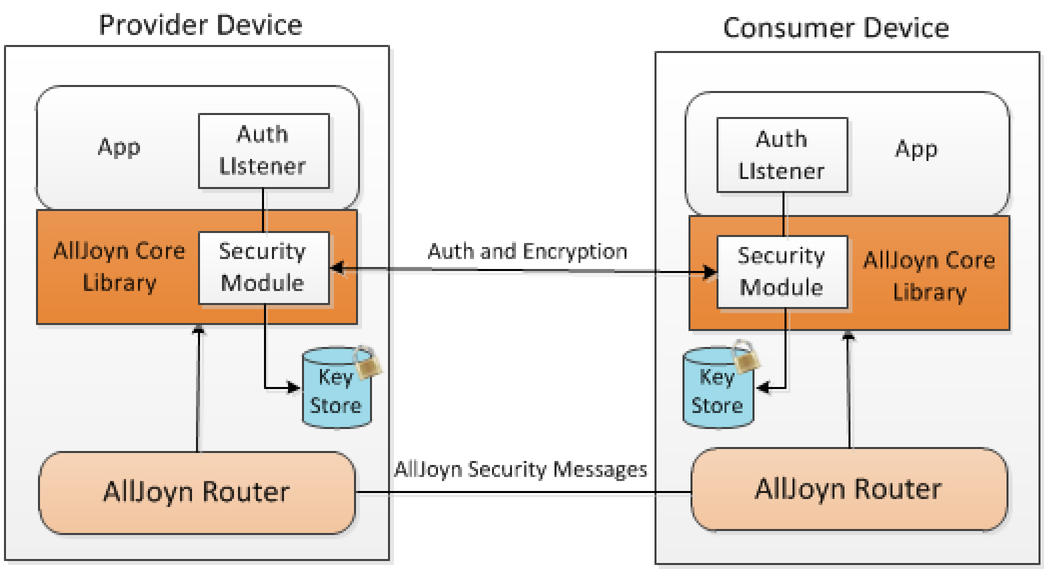

Figure 6 AllJoyn Security Architecture, AllJoyn™ Security, https://allseenalliance.org/framework/documentation/learn/core/system-description/alljoyn-security

AllJoyn appears to have many the features enumerated as important criteria in supporting a robust commodity-based environment (including security), although we did not verify all of them through implementation or code development.

During the research time period, the AllSeen Alliance merged with the Open Connectivity Foundation (OCF)[5]. The ramification of this merger is that the specifications for IoTivity (another IoT framework) and AllJoyn will merge into one. AllJoyn as a separate specification will end with version 16.04. The testing and certification program for OCF will be for the merged specification.

[5] https://allseenalliance.org/allseen-alliance-merges-open-connectivity-foundation-accelerate-internet-things

Applications and Recommendations

Our demonstration system showed the significance of having an open source standard framework available to create distributed control systems easily. Small vendors that do not have the resources to create an end-to-end system can create an agent that will interoperate with other system providers. Having a standard interface for DER along with standard interfaces for configuration, provisioning and security are important enabling ingredients.

IoT frameworks such as AllJoyn are open source frameworks that show promise in the area of supporting DER asset management requirements in this manner. Note that there is an additional potential opportunity to leverage those vendors that already use an IoT framework. Vendors with IoT frameworks in place can extend their application portfolio by adding the DER-specific interface rather than developing a new system from the ground up. Also, when standard interfaces are provided for managing these systems, utilities and energy aggregators can integrate assets from disparate vendors more easily.

There is more work to do, however, before an IoT framework can be effectively used. Even though this work demonstrated the viability of incorporating a DER data model as an IoT interface, much more effort would be required to make it a viable interface. It is clear that directly integrating the XML or data models from current smart grid standards as IoT interfaces is not feasible. It would, however, be of value to have an IoT DER data model that mirrors or reflects that data and functions represented in current smart grid standards. Creation of such an IoT DER interface would benefit from the participation of the organization that supports the smart grid standard (OpenADR alliance, for example).

The process for integrating an IoT DER interface into the standard supported by OCF is fairly well defined. First, the development of an IoT application that demonstrates the interface would be made available as open source. Secondly, the interface would then be reviewed and adopted as a standard interface. Finally, test suites for the interface are made available that provide the opportunity for vendors to certify their products. As mentioned earlier, OCF is developing interfaces for industrial as well as health care settings.

In addition to getting the data models defined and adopted for DER, a set of security requirements for IoT systems to participate in different types of DER would be of value. Current Smart Grid communication standards use DTLS (Data Transport Layer Security), which protect the data transferred between two endpoints (from a Server to a home gateway, for example). However, this does not address the security of the platform or the application that is creating or using the data. Without a security platform reference model for IoT, device vendor applications may be vulnerable to being corrupted[6]. The Security 2.0 features of AllJoyn addressed many of these topics and can be used to drive requirements for IoT implementations.

[6] http://www.zdnet.com/article/internet-of-things-security-what-happens-when-every-device-is-smart-and-you-dont-even-know-it/