Reference Design for Residential Energy Gateways

Technical Advisory Committee/Industry Presentation

Wednesday, October 6, 2010
University of California Berkeley, Room 240
Sutardja Dai Hall

Daniel Arnold
Graduate Student Researcher
UC Berkeley, Dept. of Mechanical Engineering
Presentation Outline

- Introduction
- Gateway Overview
- Reference Design Specifics
- OSGi Software Framework Introduction
- Gateway in the OSGi framework
- User Interface
- Outstanding Issues
- Future Work
Gateway Overview

What is a Gateway?

- A device capable of communicating with appliances within the home
- A device capable of communicating with the smart meter, or other metering devices
- A device capable of communicating with the outside world (internet)
- It enables the resident to manage their energy usage more efficiently
Gateway Overview

Residential Energy Gateway Reference Design

- Demand Response Center
- Upper Level/Neighborhood Gateway
- Aggregator
- Electric Utility
- Resident User Interface
- Appliances
  - refrigerator
  - television
  - washer/dryer
  - dishwasher
- HVAC
- Local Generation
  - solar PV
  - small wind

Outside World

Home Energy Network (HEN)
Why do we need a Residential Energy Gateway?

- Loads within the home are essentially unmanaged, substantial savings could be reaped
- The nature of future residential loads (PHEVs) is not fully known and management of these loads is important as to not overtax the grid
- Implementation of residential demand response
- Increase level of home automation
- Educate/Involve consumer in home energy management
Why do we need a reference design?

- Current Gateways lack communications over multiple mediums (ZigBee, Wi-Fi/Ethernet, Zwave)
  - HEN elements communicating over ZigBee will not communicate directly with Wi-Fi/Ethernet HEN (Home Energy Network) elements
  - This is burdensome to the consumer
- Allows smart appliances of different manufacture to be a part of the HEN
- This provides the consumer with more options
Gateway Project Overview

- Phase 2 (Jan 2010 – Oct. 2010): Develop a working prototype and simulate/test Gateway functionality.
  - Control strategies based on demand response
  - Refine web-UI, database, registration
Reference Design Specifics

Centralized vs. Distributed System

- Should the Gateway physically reside within a single device (such as a computer, or router-like device)?
- Should the Gateway reside within individual components of the Home Energy Management System (such as appliances and the advanced meter)?
- A distributed system could realize most, if not all, of the desired Gateway functionality.
The Advantages of a Centralized System

- Optimization within the residence is a possibility.
- It is not necessary to require appliances to communicate with the outside world, individually.
- There is a central user (resident) interface, rather than having separate interfaces on each appliance.
- A single user interface would greatly increase user education possibilities
Reference Design Specifics

Hardware Requirements:

- Target platform will be computationally “modest”: less computing power than standard PC
- Target platform should be inexpensive as to not discourage the consumer
- Must be capable of supporting necessary communications media: Wi-Fi/Ethernet, ZigBee, Zwave, etc.
- Must be capable of web page hosting
- Some data storage required

DEcision: Which hardware platform suits our needs for development?
Reference Design Specifics

Target Prototype Platform:

Router-like device

Netbook
Reference Design Specifics

Router-like device:

Advantages
- Relatively inexpensive: $50 - $150 (so a Gateway with the same parts could eventually cost a similar amount)
- Commercially successful
- Could support different operating systems depending on embedded processor
- Physical characteristics suitable for mass production

Disadvantages
- Considerable internal complexity
- Very limited memory storage capacity
- Considerable effort required to construct prototype in this fashion (create PC board with processor, programming drivers, etc.)
- Access and support for modification of COTS internet router unavailable
- User interface is not clearly defined
Reference Design Specifics

Netbook:

Advantages

- Can support standard OS (Windows, Linux, etc.)
- Price: $250 - $500, this price is suitable for development as a prototype
- User interface is clearly defined
- Can easily interface with external devices (WiFi, USB, Bluetooth, etc.)

Disadvantages

- Much less capable than a standard PC
- Price: $250 - $500, not suitable for mass production
- Limited memory storage (although has more than a router-like device)

CHOICE: Netbook for familiarity and for ability to emulate a router during design process
Reference Design Specifics

Operating System

- If mass quantities are produced, a royalty free OS would be a logical choice, such as Linux or freeBSD
- Cisco routers utilize VxWorks as an OS (proprietary)
- Netbooks can utilize Microsoft Windows or Linux
- Some communications protocols, such as ZigBee, may not be compatible with Linux
- Given this constraint, the application software should be written in a way that it is easily portable from one OS to another

CHOICE: Arbitrarily choose OS for development, however, write code that is OS independent
Software Application Language

- Compiler type languages: C, C++, C# and Java
- Scripting Language: PHP, Javascript, Python
- Scripting languages are easier to use, but lack the organization and execution efficiency of compiler-type languages
- Of the compiler languages, C++, C# and Java are object oriented (OO)
- C++ probably has the most efficient execution and smallest footprint, although Java is the most portable

CHOICE: Java, for the factors listed above and the presence of a large developer's community and widely available packages for mathematics, GUIs and networking.
Communication Protocols

- Perhaps the most sensitive part of the project, as all interested parties will need to adopt uniform communication protocols to communicate with the Gateway.

- Possible communications protocols include: Ethernet, WiFi, IEEE 802.15.4 (ZigBee), Zwave, Pager, Cellular networks, radio frequency communications.

- ZigBee is possibly the best-known standard built on IEEE 802.15.4.

- PG&E, SDG&E and SCE meters support communications based on ZigBee.

- Although ZigBee adoption could be difficult considering membership and licensing fees.

- Given the need to gather internet based information, the Gateway must include standard internet and wireless communications.

CHOICE: ZigBee & Wi-Fi/Ethernet for development
ZigBee Communications on a PC

- Unlike Wi-Fi/Ethernet, ZigBee comms. are not available on a standard PC (Netbook)
- Must demonstrate connectivity over ZigBee as to show Gateway communications functionality
- Many different ZigBee specifications exist: S.E.P. 1.0, Health Care, Telecommunications
- S.E.P. 2.0 still in development
- Need to find ZigBee hardware available for rapid deployment and development given the scope of this project
ZigBee Communications on a PC

- Telegesis USB dongle provides connectivity over ZigBee
- Does not utilize any aforementioned specification
- COTS product with documentation/firmware support
- Firmware provides bridge from USB to COM port in Windows
- Dongle accepts serial “AT” commands from any terminal software

October 6, 2010
PI: Professor David Auslander
GSR: Daniel Arnold
Software Framework: Open Services Gateway Initiative (OSGi)

- Writing the application software completely from scratch is not feasible given the time constraints on this project.
- OSGi is a software framework which supports a dynamic module system for Java
- Software originally intended for home automation market
- Software framework is incorporated into reliable IDEs, which includes support for creating OSGi bundles
- OSGi supports a run-time environment in which bundles can be installed, uninstalled, etc. independently of one another
- OSGi bundles are created using a relatively simple Java interface
- OSGi software framework is widely (but not fully) supported in various industries (a full list is available here: http://www.osgi.org/About/Members)
OSGi Software Framework

Introduction

OSGi terminology

- **Bundle** – similar to JAR file in JAVA, are visible to the user in the OSGi runtime
- **Service** – a JAVA object, not visible to the user, how information is passed from one bundle to another
- **Registration** – exporting a service from a bundle to the OSGi framework
- **Consumption** – importing a service from the OSGi framework into a bundle
OSGi Software Framework

Introduction
Why is this advantageous?

- Bundle B requires no knowledge of Bundle A, vice versa
- Bundle A only registers Service X with the framework
- Bundle B only consumes Service X from the framework
- Consumed services can be “returned” to the framework
- Registered services can be “recalled” from the framework

Bottom Line: Bundles A and B can be operated on independently
Gateway in the OSGi Framework

Gateway Bundles

- Wi-Fi/Ethernet bundle: registers *NetService* in the OSGi framework
- ZigBee bundle: registers *ZigBeeService* with the OSGi framework
- Open Automated Demand Response (OpenADR) bundle: registers *OpenADRService* with the OSGi framework
- Control bundle: consumes *ControlService, OpenADRService*, contains control logic to actuate appliances
- Web UI bundle: controls lifecycle of Gateway web user interface, provides the resident with “opt-in” or “opt-out” capability, allows for the installation of new simulated appliances
- Utility bundle: provides support code for all Gateway services
Gateway in the OSGi Framework

Gateway Services

- NetService: facilitates a connection over JAVA network sockets with a simulated appliance
- ZigBee bundle: facilitates a connection over generic ZigBee stack with a simulated appliance
- OpenADRService: connects to Akuakom *Demand Response Actuation Server* (DRAS), receives and parses OpenADR event information
- ControlService: spawned from consumption of NetService or ZigBee service, provides methods to actuate simulated appliances
Gateway Bundles and Services – Initial Service Registration

Core Bundles
- WiFi Bundle
- ZigBee Bundle
- Control Bundle
- OpenADR Bundle

External Bundles
- WiFi Appliance
- ZigBee Appliance

- Net Service
- ZigBee Service
- OpenADR Service
Gateway Bundles and Services – Initial Service Consumption

- WiFi Bundle
- ZigBee Bundle
- Control Bundle
- OpenADR Bundle
- Net Service
- ZigBee Service
- OpenADR Service
- WiFi Appliance
- ZigBee Appliance

Core Bundles

External Bundles
Gateway Bundles and Services – Secondary Service Registration

Core Bundles
- WiFi Bundle
- ZigBee Bundle
- Control Bundle
- OpenADR Bundle

External Bundles
- Net Service
- ZigBee Service
- Control Service
- OpenADR Service
- WiFi Appliance
- ZigBee Appliance
Gateway Bundles and Services – Secondary Service Consumption

Core Bundles
- WiFi Bundle
- ZigBee Bundle
- Control Bundle
- OpenADR Bundle

External Bundles
- Net Service
- ZigBee Service
- Control Service
- OpenADR Service
- WiFi Appliance
- ZigBee Appliance
Gateway in the OSGi Framework

Memory Footprint

- OSGi framework (independent of JRE): ~1.1 MB
- Gateway OSGi bundles (previously shown): ~ 1MB
- JRE6: ~90 MB (on the author’s computer)

The memory presence of the OSGi framework and the Gateway bundles, collectively, is comparatively small.
Gateway User Interface

User Interface Requirements:

- Inform resident of aggregate energy usage information
- Inform resident of individual appliance energy usage information
- Provide ability to actuate appliances individually
- Provide for the ability to install/uninstall new appliances to the HEN
- Display relevant demand response event information
- Provide opt-in/opt-out capability for DR event participation
Gateway User Interface

Website User Interface

Energy Gateway
Energy Management Of The Future

System | Devices | Events

Functions
Add Device
Delete Device

Devices

<table>
<thead>
<tr>
<th>Device</th>
<th>Status</th>
<th>Switch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microwave</td>
<td>Active</td>
<td>[On] [Off]</td>
</tr>
<tr>
<td>Refrigerator</td>
<td>Idle</td>
<td>[On] [Off]</td>
</tr>
<tr>
<td>AC</td>
<td>Idle</td>
<td>[On] [Off]</td>
</tr>
</tbody>
</table>

Current Time: 5:37 PM

© 2010 UC Berkeley. All Rights Reserved.
Powered by Energy...Pure Raw Energy
Gateway User Interface

Website User Interface

Energy Gateway
Energy Management Of The Future

Add Device

- Appliance: Desktop
- Power Usage: Medium
- Interruptible: Yes
- Smart Device: Yes
- Options:
  - Option 1
  - Option 2
  - Option 3

Create
### Gateway User Interface

**Energy Gateway**

**Energy Management Of The Future**

#### Events Table

<table>
<thead>
<tr>
<th>Event</th>
<th>Status</th>
<th>Time Frame</th>
<th>Price</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event 1</td>
<td>Active</td>
<td>3:00 PM-5:00 PM</td>
<td>$0.30 /kWh</td>
<td>Accepted</td>
</tr>
<tr>
<td>Event 2</td>
<td>Near</td>
<td>5:00 PM-9:00 PM</td>
<td>$0.19 /kWh</td>
<td>Opt Out</td>
</tr>
<tr>
<td>Event 3</td>
<td>Far</td>
<td>7:00 PM-10:00 PM</td>
<td>$0.15 /kWh</td>
<td>Join, Deny</td>
</tr>
</tbody>
</table>

© 2010 UC Berkeley. All Rights Reserved.

Powered by Energy...Pure Raw Energy
Outstanding Issues

User Interface

- Need to standardize data transmission from the Gateway to the Web-UI
- Allows different vendors to construct unique user interfaces
- Should allow for data connection to cloud resources (for example: Google PowerMeter)

QUESTION: What remaining functionality must be included?
Data Model

- Information passed from External Bundles (ZigBee/WiFi Appliance) is a JAVA String of the form:

  <value>3650</value>, or <applianceState>OFF</applianceState>

- Data values are time-stamped by the Gateway (server-side)
- This model was adopted for convenience only
- Allows for easy parsing/writing to XML documents

QUESTION: Is there a better way to pass data between the Gateway and appliances (clients)?
Appliance Control

- Control bundle defines some generic methods such as:
  
  \[ \text{turnApplianceOff()} \text{ or } \text{turnApplianceOn()} \]

- Appliances would need to adopt the Gateway data model and listen for specific JAVA strings

- Obviously, thermostats should be actuated differently than washing machines

- Individual appliances might want to respond to more specific control commands.

**QUESTIONS:** What generic control methods should be defined? Can we incorporate individual control methods for specific appliances?
Outstanding Issues

Appliance Registration

- Must ensure that the proper appliance is paired with the proper Gateway
- Registration similar to Bluetooth possible (code conformation)
- This process must be technically simple, as the average consumer might be confused

QUESTIONS: Should this process should be standardized despite the connection media (ZigBee or Wi-Fi)?
Outstanding Issues

Database

- The Gateway must have a mechanism to store non-volatile data (configuration details, passwords, etc.)
- This information must be robust to power failures, etc.
- This provides an opportunity to store aggregate/individual appliance energy usage data as well
- Stored data can strongly influence more sophisticated control strategies

QUESTIONS: Which database software best suits our hardware needs? What granularity of data should be stored? Is there a need to store long-term energy usage data?
Limp-Home mode

- In the event of Gateway failure, the home must continue to operate properly
- The presence of the Gateway should in no way hinder the resident in home operation

QUESTIONS: How do we ensure that individual appliances will function without the Gateway connection?
Questions?

Thank You

For more information, please visit
http://mechatronics.berkeley.edu/gateway.htm

Individual Thanks:

Professor Dave Auslander (UCB)
  Ron Hofmann (CIEE)
  Gaymond Yee (CIEE)
  Kevin Ding (UCB)
  Michael Sankur (UCB)
  Halley Hardiman (UCB)