Web Services for Energy Management in a Smart Grid Environment

Adnan Afsar Khan

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the Doctorate in Philosophy degree in Computer Science

Ottawa-Carleton Institute for Computer Science
Faculty of Engineering
University of Ottawa

L'Université canadienne
Canada's university

Abstract

Smart grid is an emerging technology that aims to empower the current power grid with the integration of two-way communication and computer technology. This thesis deals with energy management in smart grid with focus on the smart home and the Intelligent Transportation System (ITS). The smart home contains a network that connects home elements like smart appliances, HVAC (heating, ventilation, and air conditioning), thermostat, smart meter, sensors, solar panels and energy storage. ITS integrates computer and communication technologies for advanced traffic management and communication among road infrastructure, vehicles and users. A web service describes a collection of operations that are accessible via the Internet. Web services can also provide security and interoperability. Due to the rising cost of energy, more and more residential consumers are interested in controlling temperature or appliances in order to reduce energy consumption. In this thesis, we propose an approach that uses Web services to remotely and efficiently interact with smart home devices in order to manage energy consumption, in a smart grid environment. Consequently, the user is able to adjust the temperature, control appliances or read energy consumption values quickly, remotely and securely. A smart home with a wireless network based on ZigBee and XMPP (eXtensible Messaging and Presence Protocol) is simulated. The advantage of XMPP is that it provides near real-time communication and security. There is a central computer that can communicate with all home elements. Business Process Execution Language (BPEL) is used to implement the Web service on a central computer. Furthermore, quality of service is offered. Therefore, different levels of security and an access control mechanism are provided. An algorithm is proposed that can sell stored energy back to the grid from smart home. Another algorithm is proposed that can facilitate demand response. Moreover, dynamic programming is used to minimize energy consumption. Also, a broadcasting algorithm is presented that can be used by an electric vehicle to find the most suitable charging station in ITS. Simulation and analytical study is undertaken to demonstrate the performance and advantage of the proposed approach and algorithms.

Acknowledgements

My sincere gratitude goes to my thesis supervisor, Professor Hussein T. Mouftah, for his invaluable advice and encouragement throughout my research activities. I earnestly thank Professor Hussein T. Mouftah for his financial support, patience, and guidance during the pursuit of my Doctoral studies. Sometimes, he would give me time beyond normal hours (e.g. weekends and evenings). It has been an honour to research under the guidance of such distinguished and renowned professor. This thesis could not have been completed without his help.

I would like to thank my colleague, Rami F. Z. Henry, for his help with the smart home simulator. Furthermore, I would like to acknowledge the University of Ottawa for providing me with the opportunity to undertake my Doctoral degree with financial assistance.

Finally, I would like to give my utmost appreciation to my parents for their constant emotional and financial support during my doctoral studies. Furthermore, I owe special thanks to my wife, Sahar Ullah, for her constant encouragement, patience and care. She also proofread part of this thesis. Her support in household matters helped me to focus on my studies. Also, I would like to thank my parents-in-law and relatives who encouraged me during my studies at the University of Ottawa.

Table of Contents

List of Tables

List of Figures

List of Acronyms

6LoWPAN	IPv6 over Low-Power Wireless Personal Area Networks
ADSL	Asymmetric Digital Subscriber Line
AI	Artificial Intelligence
AMI	Advanced Metering Infrastructure
API	Application Programming Interface
BPEL	Business Process Execution Language
BPL	Broadband over Power Line
CDS	Connected Dominating Set
CEMA	Comfort Control and Energy Management Algorithms
CI	Confidence Interval
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSt	Charging Stations
CSV	Comma Separated Values
DG	Distributed Generation
DPWS	Devices Profile for Web Services
DR	Demand Response
DSM	Demand-Side Management
ECDs	Energy Consumption Displays
FTTH	Fibre To The Home
GAs	Genetic Algorithms
GIS	Geographic Information System
GPS	Global Positioning System
HOTP	Hash-Based Message Authentication Code One Time Password
HTTP	Hypertext Transfer Protocol
HVAC	Heating, Ventilation, and Air Conditioning
IDE	Integrated Development Environment
ITS	Intelligent Transportation Systems

J2EE	Java 2 Platform Enterprise Edition
JAR	Java Archive
JDK 5.0	Java 2 Platform Standard Edition 5.0 Development Kit
JSP	Java Server Pages
LTP	Lean Transport Protocol
MAC	Media Access Control
MANET	Mobile Ad hoc wireless Network
microFIT	micro Feed-In-Tariff
NES	Neighbor Elimination Scheme
OAS	Office Automation System
OBPM	Oracle Bpel Process Manager
OC4J	Oracle Containers For J2EE
OSGi	Open Services Gateway Initiative
P2P	Peer-To-Peer
PBSM	Parameterless Broadcasting from Static to Mobile networks
PEV	Plug-In Electric Vehicles
PHEV	Plug-In Hybrid Electric Vehicles
PLC	Programmable Logic Controller
PMUs	Phasor Measurement Units
QFD	Quality Function Deployment
QoS	Quality Of Service
QoSS	Quality Of Security Service
RBAC	Role Based Access Control
REST	Representational State Transfer
SAML	Security Assertion Markup Language
SASL	Simple Authentication and Security Layer
SCADA	Supervisory Control And Data Acquisition
SG	Smart Grid
SMC	Soap-Message Compression
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol

SSL	Secure Sockets Layer
SWE	Sensor Web Enablement
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TOU	Time-Of-Use
TTL	Time for data To Live
UDDI	Universal Description, Discovery and Integration
WS	Web Service
WSDL	Web Service Description Language
WSIF	Web Services Invocation Framework
WSN	Wireless Sensor Network
XACML	eXtensible Access Control Markup Language
XML	eXtensible Mark-Up Language
XMPP	eXtensible Messaging and Presence Protocol

List of Symbols

$c[i, e]$	Value of optimal solution for items 1,2, . . . , i and maximum energy e
E	Maximum energy consumption (threshold value)
E_{max}	maximum energy consumption of the house
e_i	Energy consumption of appliance i
enH	Amount of energy needed by home during peak-hours
$enStore$	Amount of energy stored in storage
p	Peak hours
t	Current time
v_i	Priority value of the appliance i
λ	Rate of people arriving the room per hour
μ	Rate at which people leave the room per hour
ρ	Traffic intensity

Chapter 1

Introduction

1.1 Background

A **sensor** is a device that measures physical parameters. A sensor node typically consists of five main parts: sensors (one or more) that gather data from the environment; a central unit in the form of a microprocessor that manages the tasks; a transceiver that communicates with the environment; a memory unit that is used to store temporary data or data generated during processing; and a battery that supplies all parts with energy [VER08]. It can sense the environment, perform computation on sensed data and exchange information with neighbouring elements. Each sensor has a limited sensing region, processing power and energy. A **wireless sensor network** (WSN) consists of sensors that are spatially distributed over a geographical area. The sensors monitor the environment and report events (e.g., detecting intrusion) wirelessly to a sink (or base station), which is connected to a central server [MUR04]. This thesis deals with WSN-based energy management in smart grid with focus on the smart home and the Intelligent Transportation Systems (ITS).

Smart home, *connected home*, *digital home*, *adaptive house*, and *aware home* are terms used to represent future homes. Although these terminologies share much of the common concept of future homes, each terminology has a slightly different emphasis. *Connected home* emphasizes the connection among digital devices in a home. *Smart home* focuses on home automation services that can control and administer digital devices locally or remotely. *Digital home* focuses on sharing digital media such as music, picture or video and providing digital content service to homes in liaison with the Internet. *Adaptive house* has been developed to use neural networks to control temperature, heating, and lighting without previous programming by the residents. It

continuously monitors the environment and observes actions taken by the residents (e.g., how they use lights, how they adjust the thermostat). From these data, it infers patterns in the home and uses reinforcement learning, to predict future behaviour [CHA08]. *Aware home* refers to the residential environment that has the ability to recognize the information of a home and its surroundings and the information of the actions of the residents [JAN01]. All of these terminologies are interchangeably used in practice. In this thesis, *smart home* is used to represent the future home. In [BER99], a smart home is defined as a dwelling incorporating a communications network that connects the key electrical devices and services, and allows them to be remotely controlled, monitored or accessed. A smart home contains different elements like a washer, dryer, lighting, HVAC (heating, ventilation, and air conditioning system), sensors, thermostat and smart meter. A homeowner on vacation can use a mobile phone to communicate with a central computer in the home in order to arm a home security system, trigger appliances, control lighting, and program a home theatre or entertainment system. A smart home is also useful to monitor and assist the elderly and disabled. Wearable sensors can be used to measure blood pressure and any anomalies can be reported wirelessly to a hospital server. An important device in a smart home is a **smart meter**. The meter stores energy consumption data and transmits them automatically and wirelessly to the utility provider. The advantage is that nobody needs to go on site to check the meter readings. Furthermore, smart homes may have a solar panel or windmill and energy storage to obtain and store renewable energy. Thus, a smart home has many benefits like energy conservation, security, safety, centralized control and remote access of home environment and devices, comfort, and independent living for the elderly and disabled.

Smart grid (also called *intelligent grid* or *future grid*) is a term referring to the next generation power grid in which the electricity distribution and management is upgraded by incorporating advanced two-way digital technology and communication capabilities for improved control, efficiency, reliability and safety. It covers the generation (through various power plants), transmission, and distribution of electricity to consumers (e.g., residents, commercial buildings, industry). A smart grid (SG) delivers electricity from suppliers to consumers using digital technology to save energy, reduce cost and increase transparency. Furthermore, it reduces emission of greenhouse gases and fosters demand response. A smart grid also contains a

networked sensor inside a transformer or along wires to quickly locate and report a problem wirelessly or prevent it from happening in the first place. The transmission system delivers electricity from power plants to distribution substations while the distribution system delivers electricity from distribution substations to consumers. A smart grid also includes distributed and renewable generation resources like residential solar panels, small windmill and plug-in electric vehicles. These environment-friendly resources will enable a home (or small business) to generate, consume, store and then sell energy to their neighbours or back to the grid. The real-time, two-way communications available in a smart grid will enable the suppliers to observe the energy consumption of a house and send information (e.g., price signals) to the consumers through a smart meter. Therefore, when consumers shift their energy demands to off-peak hours or supply green energy (e.g., solar energy) to the grid, they are able to decrease their energy bills while also decreasing the load on the grid. Thus, smart grid enables the consumer to be more involved in order to optimize their energy consumption. The communication between smart home elements and outside users (e.g., utility provider, network operator, etc.) is usually done wirelessly [SGR].

A **mobile ad hoc wireless network** (MANET) is a collection of wireless mobile hosts that form a temporary network without any fixed infrastructure or centralized administration. Two mobile hosts can communicate if they are within transmission range of each other. **Broadcasting** is the task of sending a particular message from the source node to all other nodes in the network. **Plug-in Electric Vehicle** (PEV) has an electric engine that is topped up (i.e. charged) from the power grid. It has no gasoline engine and usually uses Lithium-ion batteries. PEVs are becoming popular as their operating cost is low and they are environment-friendly. ITS are advanced applications that aim to provide services in the field of road transport. It integrates computer and communication technologies for traffic management and communication among infrastructure, vehicles and users.

Service oriented architecture (SOA) has recently become very popular. The basic principles of SOA can be described in the following manner. First, it is necessary to provide an abstract

definition of the service. Second, those who are the providers of services need to publish details of their services so that those who want to use them can understand more precisely what they do and can obtain the information necessary to connect to the services in order to use them. Third, those who require services need to have some way to find what services are available that meet their needs. In order to make this approach work well, standards must be defined to govern what and how to publish, how to find information, and the specific details about how to bind to the service. A **Web service** (WS) describes a collection of operations that are accessible via the Internet through standardized XML (Extensible Mark-up Language) messaging. It can be implemented in any language on any platform. Web services define a standardized mechanism for location, description and communications with applications. Web services use XML to code and decode data and SOAP (Simple Object Access Protocol) to transport it using existing protocols, like HTTP (Hypertext Transfer Protocol). The main advantage of web services is its interoperability as it uses vendor, platform, and language independent XML technologies and the ubiquitous HTTP as a transport. The client only requires the WSDL (Web service description language) definition to effectively exchange data with the service – and neither part needs to know how the other is implemented. These benefits allow integration of disparate applications and data formats with relative ease. It can be accessed by applications running on different platforms. Furthermore, an application can expose its functionalities as Web service and other applications can communicate with it via Web service. Furthermore, Web services can easily be accessed over the Internet as it uses SOAP and HTTP for communications. SOAP can get around a firewall. Web services technologies are a set of technologies based on XML standards that help describe, access, and interact with Web services. Web Services Framework (or Technology) consists of three platform elements: WSDL, UDDI (Universal Description, Discovery and Integration) and SOAP. Web services technology is a standards-based, XML-centric realization of SOA. When a client wants to use the Web service, it searches for the specific service in UDDI. UDDI returns a WSDL file which describes the Web service and its location. Clients use this information in the WSDL file to form a SOAP request to the designated computer offering the service. The designated computer performs the required operation and returns the result via SOAP. As Web services are accessed over the Internet and by remote users (or applications), they need to be secured. There are various efficient security specifications for Web services.

1.2 Motivation and Objective

Due to the rising cost of energy during peak hours, more and more people are interested to reduce their energy cost by decreasing energy consumption, shifting their load to off-peak hours or using renewable energy. Therefore, the user (e.g., residential consumer) wants to undertake operations like control temperature, control appliances or use stored energy. The residential consumer also likes to interact with home elements remotely for comfort like remotely adjusting temperature or turning on the coffee-maker before arriving home. If an unauthorized person has access to the system, then he can damage the home elements, invade privacy or cause financial harm by manipulating stored energy. Consequently, the consumer likes to carry out these operations quickly, remotely and securely. The user may face problems communicating with home elements in a unified way if the elements contain heterogeneous technologies. Furthermore, there may be interference or signal attenuation if the elements were communicated directly via a wireless device from outside the home. Moreover, a PEV often needs to charge (or change) its battery and therefore the driver needs to be able to find the nearest or most convenient charging station.

The objective of this thesis is to provide a solution to overcome the above mentioned challenges. The Internet accessibility, security and interoperability feature of Web services can be used for efficient communications with home elements and also for integration of different home elements. In this thesis, an approach is proposed that uses Web services to remotely and efficiently interact with smart home elements in a smart grid environment. The aim is also to facilitate demand response, minimize the use of energy and sell stored energy back to the grid. The goal is to analyze the performance (e.g., completion time), advantage and limitations of this communication. Furthermore, the purpose is also to offer good quality of service. Therefore, different levels of security and access control mechanisms are to be provided. Furthermore, the aim is to enable a PEV to reliably find the most convenient charging station in ITS.

The proposed approach differs from the existing home energy management systems and the dissimilarities are discussed here. An important difference is that this system considers the SG

environment and energy management involving demand response and selling of energy back to the grid. An example of a home energy management system is the "Smart Home Monitoring System" from Rogers Communications Inc. [SHM11]. The user of this system can use a web portal to remotely read and control room temperature or appliances. A web portal connects to a central home controller and the user is able to control home elements via this controller. The system does not use WS. In the proposed system, a Web service is used that enables the user to interact with different home elements and directly initiate operations like control temperature. Some papers [ASA11], [GLO09], [PRI08] mention running WS on sensors instead of a central computer. Due to the limited capability of the sensor and the size of XML messages, it is not convenient to run WS on a sensor. In the proposed system, the Web service runs only in the central computer. Furthermore, in some papers WS is not used to communicate with end devices like sensors. They are used to communicate with other systems like a surveillance system [PER08] and room booking system of a hotel [CHU08]. The authors in [XU10], [AIE06] talked about using a separate controller to communicate with sensors. WS are used to communicate with the controller. In our approach, there is no middleware or additional controller. In addition, some papers [KAM11] used Representational State Transfer (REST) full Web services instead of SOAP-based Web services. Although REST has its advantages, our proposed approach uses SOAP as it is still a widely used protocol and compatible with many applications. SOAP also has better support for security and quality of service than REST [WEE05]. ZigBee protocol has better support for interoperability than 6LoWPAN (IPv6 over Low-power Wireless Personal Area Networks), which is used in [KAM11].

1.3 Thesis contributions

The main contributions of this thesis are as follows:

- It proposes an approach that uses Web services to remotely and efficiently interact with different smart home elements in a smart grid environment. Unlike some existing home automation systems, the Web service runs only in the central computer instead of resource-constrained sensors.

- An algorithm to sell stored energy back to the grid from smart home is proposed.
- An algorithm is proposed that can facilitate demand response and reduce energy consumption.
- A dynamic programming technique is presented to minimize the use of energy.
- A broadcasting algorithm is presented that can be used by an electric vehicle to find the nearest, cheapest or most convenient charging station in ITS.
- An application of Extensible Messaging and Presence Protocol (XMPP) to provide near real-time messaging service and security between the central computer and the appliances.
- A technique to implement Web service using Business Process Execution Language (BPEL). Furthermore, Web Services Invocation Framework (WSIF) is used to integrate the Java-based simulator that simulates the smart home.

1.4 Thesis Outline

The rest of this thesis is organized as follows: Chapter 2 presents a survey of the related work done in the area of the smart home, smart grid, Web services, XMPP, security and broadcasting. Furthermore, the chapter consists of related work done on Web services security and usage of Web services in the smart home. In addition, energy management systems and recent developments in smart grid deployment are discussed. Chapter 3 describes the basic characteristics of the proposed system. This chapter specifies the architecture and functionalities of the proposed system. Chapter 4 deals with the interaction of the proposed system with the smart grid. Furthermore, the chapter presents the algorithms to sell energy and minimize the use of energy. Chapter 5 consists of the implementation details and quality of service provided by the system. In addition, it describes the security, access control mechanism and differentiated service. Chapter 6 shows the performance evaluation and analysis of results. Chapter 7 discusses a broadcasting algorithm that can be used to find suitable charging station. Chapter 8 concludes this thesis and contains suggestions for future research.

Publications

Book Chapter

1. A. Chehri, H. Mouftah, A. Khan, "Adaptive Synthesis of Domestic Home Energy Management in Smart Grid Environment", Energy Efficiency, Book edited by Moustafa Eissa, InTech Open Access Publisher, Rijeka, Croatia, 2012, ISBN 979-953-307-780-1, in press.

Journal Papers

1. A. A. Khan and H. T. Mouftah, "Secure Cloud Services for Energy Management of Smart Home in Smart Grid Environment", Elsevier Journal of Future Generation Computer Systems Special Issue on Internet of Things and Cloud Services, November 2014, submitted.

Conference Papers

1. A. A. Khan and H. T. Mouftah, "Energy Optimization and Energy Management of Home Via Web Services in Smart Grid", Proceedings Annual IEEE Electrical Power and Energy Conference (EPEC), London, Canada, October 2012, pp. 14-19.
2. A. A. Khan and H. T. Mouftah, "Secured Web Services for Home Automation in Smart Grid Environment", Proceedings IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Montreal, Canada, April-May 2012, pp. 1-4.
3. A. A. Khan and H. T. Mouftah, "Web Services for Indoor Energy Management in a Smart Grid Environment," Proceedings 22nd Annual IEEE International Symposium on Personal, Indoor, Mobile and Radio Communications (PIMRC), Toronto, Canada, September 2011, pp. 1036-1040.

4. Adnan Afsar Khan and Hussein T. Mouftah, “Secured Web Services for Energy Management of Smart Home in Smart Grid”, a poster presented at 2012 WiSense Workshop, University of Ottawa, Ottawa, September 2012.
5. Adnan Afsar Khan and Hussein T. Mouftah, “Web Services for Smart Homes Energy Management in a Smart Grid Environment”, a poster presented at 2011 WiSense Workshop, University of Ottawa, Ottawa, September 2011.
6. Adnan A. Khan and Hussein T. Mouftah, “Using Web Services for Accessing Smart Home Elements in Smart Grid Environment”, a poster presented at 2010 WiSense Workshop, Queen’s University, Kingston, May 2010.

Chapter 2

Literature Review

2.1 Introduction

This chapter presents a survey of the related work done in the area of the smart home, smart grid, Web services, security, XMPP and broadcasting. In Section 2.2, related work on the smart home is discussed. This section focuses on devices and technologies that are used in the smart home. Furthermore, different types of smart homes and their advantages or drawbacks are discussed. Section 2.3 presents the related work done on the smart grid. This section also presents the components of the smart grid, advantages of making the grid smarter and upcoming challenges in implementing the grid. In addition, energy management systems and recent developments in smart grid deployment are discussed. Section 2.4 deals with the related work done on Web services. This section focuses on the usage of Web services in the smart home. Furthermore, it presents XMPP and Web services along with its securities and quality of service. Moreover, Section 2.5 discusses the broadcasting algorithms in the mobile network. Finally, Section 2.6 contains a summary of the chapter.

2.2 Smart Home

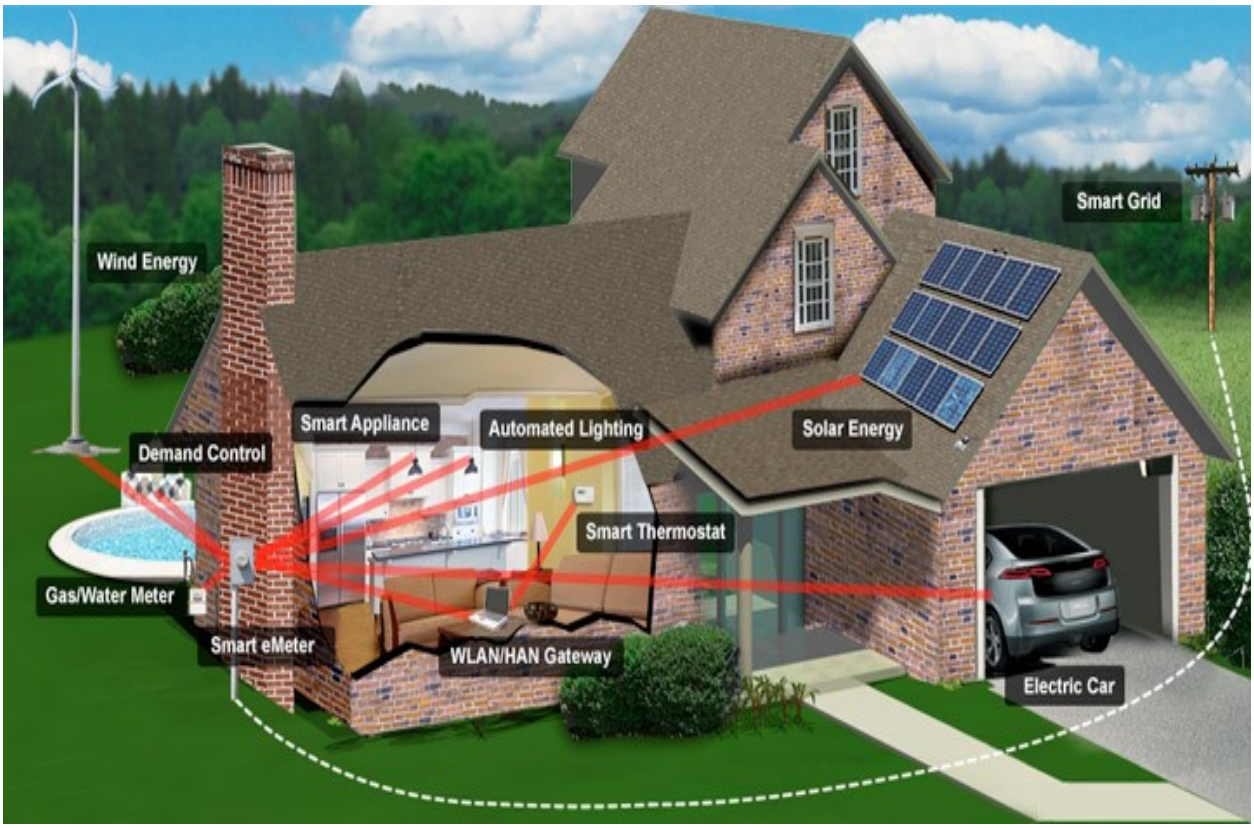

Figure 2.1: Smart Home

Figure 2.1 illustrates the smart home along with its devices. In [DEW01], smart home devices are categorized into active and passive devices. Active devices include control panels and switches which home occupants will directly interact with and use. Passive devices include sensors and receivers which home occupants have no direct contact over and that function to empower the living experience of the occupants. On the contrary, authors of [ROD03] divided smart home devices into the following three categories based on the interaction techniques: information appliances; interactive household objects; and augmented furniture. Information appliances often use touch screens, and the form of interaction is very similar to those of PCs and handheld devices. Interaction of household objects is incorporated into the form of the object. Thus, cups are augmented with temperature/motion sensors. Interaction of augmented furniture is enhanced

through sensors detecting actions with the furniture. Examples of some smart home devices are the smart thermostat, smart washer/dryer, smart sofa, smart window, smart meter, sensors that measure physical parameters like temperature, etc. Smart Pillows [PAR03] [JIA04] are designed to personalize bedtime. They can play people's favourite music and read bed time stories. Once an individual begins sleeping, the smart pillow can automatically check the quality of sleep and will gradually decrease the music volume and eventually turn off. In addition, smart pillows can check body temperature, blood pressure level and—if there is a need—it can automatically report to a hospital through the Internet. Smart Refrigerators [BUT99] [JIA04] are equipped with an inventory application. These smart refrigerators are capable of keeping track of the individual food items based on their RFID tag. It can track the expiry dates of the food items and also the quantities of food items. It can even prepare shopping lists [JIA04] for the food items to buy once they are consumed.

The paper in [MEI08] discusses the residential thermostat. Design and implementation of user interfaces tend to be poor in current thermostats. Current plans are to make them compatible with some existing wireless home control systems such as ZigBee and Z-wave. The paper also discusses the situation where every occupant has their own personal sensor which they carry with them. This requires some carefully thought out control strategies. For example, what happens if two occupants with very different thermal requirements are in the same zone - who gets priority, or are the two requirements averaged? Modern thermostats need to make these decisions. The paper in [WOO07] discusses advanced energy consumption displays (ECDs) in the home. These displays may be 'central' (one per home) or 'local' (specific to the location where an individual energy-use event takes place). Methods for motivating energy-saving behaviours and for presenting energy-use information on these two display types are discussed in the article. Consideration is given to the energy units to be displayed; the method of display; how to display the information temporally; and how the display information might be categorised (i.e., by fuel, by appliance, by room, etc.). The facility to set goals via an energy consumption display is identified as a key method for motivating consumers to save energy.

LAN technologies connect different smart devices at the smart home [SSN09]. These technologies can be classified into three main groups: wireless IEEE standards 802.x, wired Ethernet, as well as in-building power line communications like BPL and X10 [INT06]. Wireless IEEE standards include Wi-Fi (IEEE 802.11), WiMAX (IEEE 802.16), ZigBee (IEEE 802.15.4), Z-wave and Bluetooth (IEEE 802.15.1). ZigBee is an open global standard providing wireless networking based on the IEEE 802.15.4 standard. The layers defined by the ZigBee Alliance are the network layer, the application framework layer, and the application profile layer. The security features implemented within the ZigBee stack are very flexible as it can be implemented in any of the layers. ZigBee profiles provide target applications with the interoperability and inter-compatibility required to allow similar products from different manufacturers to work. The key features of ZigBee include low power consumption, reliability, easy deployment, security, low cost and product interoperability [PEN10]. These features make ZigBee very suitable to use in wireless sensor networks. Authors of [SHI04] mentions that RFID tags could be attached to all objects of a smart home in order to make various kinds of ubiquitous services possible. This enables us to predict that it will also become possible for people to control all the objects having RFID tags through a smart home system. Gateways to manage the systems are needed for intelligent control of smart homes remotely and providing access from the home to external services. A traditional architecture for the smart home is server-centric. To solve the problems caused by server-centric architecture and to support a dynamic environment, authors of [WU07] proposed a service-oriented architecture (SOA) for smart home environments based on the Open Services Gateway Initiative and mobile-agent technology.

2.2.1 Examples of the smart home

In [HEN11], a centralized modular energy consumption control system, along with algorithms, is proposed for efficient and economical comfort control in smart homes. The energy consumption control system manages energy consumption in all aspects of a typical residence. One of the facets is concerned with control when the HVAC operates for each room separately. This is in contrast to a typical HVAC system that deals with the whole residential floor. Another facet is concerned with controlling the lighting in each room so as to not exceed a certain input value. The communication network is based on ZigBee where sensor nodes send data to a smart

thermostat which does all the required calculations and activates the modules required for comfort control and energy management, if needed. A Java-based discrete event simulator is then written up to simulate a floor of a typical Canadian single-family dwelling. The simulation assumes errorless communications and proceeds to record certain room variables and the ongoing cost of operation periodically. These results from the simulator are compared to the results of the well-known simulator, created by Design Builder, which describes typical home conditions. The environment in both simulations is kept exactly alike. The conclusion from this analysis is that the Comfort Control and Energy Management Algorithms (CEMA) are feasible, and that their implementation incurs significant monetary savings.

The paper in [TSE09] exploited the context-aware capability of WSN to achieve energy conservation in intelligent buildings. It therefore proposes an intelligent and personalized energy-conservation system by wireless sensor networks (iPower). This system consists of some WSNs connected to a control server, some power-line control devices and some user identification devices. A WSN is deployed in each room to monitor the usage of electric appliances and to help the control server determine if there are electric appliances that can be turned off to reduce unnecessary energy consumption. The iPower system is quite intelligent and can adapt to personal need by automatically adjusting electric appliances to satisfy users' requirements. The design and implementation details of iPower are reported in this paper. Furthermore, the paper discusses X10, which is a communications protocol for remote control of electrical devices. It is designed for communications between X10 transmitters and X10 receivers which communicate on standard household wiring. Transmitters and receivers generally plug into standard electrical outlets although some must be hardwired into electrical boxes. Transmitters send commands such as "turn on," "turn off" or "dim," preceded by the identification of the receiver unit to be controlled. This broadcast goes out over the electrical wiring in a building. Each receiver is set to a certain unit ID and reacts only to commands addressed to it. Receivers ignore commands not addressed to them. The sensor network works as follows: When sensor nodes detect a low temperature or a high brightness in a likely unoccupied room, they can report to the server that the electric appliances in that room (e.g., air conditioners or lights) could be turned off. The server then sends an alarm signal to notify people in the room that the electric appliances could

be turned off shortly. If there are still users in that room, they can signal the system that these appliances should not be turned off by triggering some events (by speaking, changing the light reading of any sensor, or moving any furniture attached with sensors). If there is no such intentional event made by a human being detected in a predefined amount of time, the server will turn off the electric appliances in the room through some power-line control devices. In particular, each user can create a profile to describe his/her favourite temperature and brightness. Such users are considered priority users and need to carry user identification devices (e.g., a badge) so that the system can retrieve their profiles. When there are priority users in a room, the server will adjust the air conditioners and lights in that room according to the profiles of these users. If there is smart furniture in the room, they can help detect the existence of people in the room. For example, if there is a person sitting on a smart chair, the system will keep on reporting that someone is in the chair. The paper in [CHA08] discusses monitoring the elderly and disabled with numerous intelligent devices. Sensors can be implanted into their home for continuous mobility assistance and non-obtrusive disease prevention. Modern sensor-embedded houses—or smart houses—can not only assist people with reduced physical functions, but help resolve the social isolation they face. They are capable of providing assistance without limiting or disturbing the resident's daily routine thereby giving him or her greater comfort, pleasure and well-being. Ever since the eighties, the elderly have benefitted from devices that signal assistance services. Miniature transmitters can be worn around the neck or wrist, or carried in a pocket, allowing an individual to signal danger or request help simply by pressing a button. In effect, the device is an emergency telephone connected to a professional service center or a family member. The major targets are improving comfort, dealing with medical rehabilitation, monitoring mobility and physiological parameters and delivering therapy. In conclusion, the paper mentioned some key challenges faced by electronic home health care including privacy and confidentiality, accessible design and reimbursement. The main design difficulty is validating the alarms triggered by an older person living alone.

In [RED06], the authors propose an efficient multi-zone HVAC control system. Most residential heating, ventilating and air-conditioning (HVAC) systems utilize a single zone for conditioning air throughout the entire house. While inexpensive, these systems lead to wide temperature

distributions and inefficient cooling due to the difference in thermal loads in different rooms. The end result is additional cost to the end user because the house is over-conditioned. To reduce the total amount of energy used in a home and to increase occupant comfort there is a need for a better control system using multiple temperature zones. Typical multi-zone systems are costly and require extensive infrastructure to function. Recent advances in wireless sensor networks (WSNs) have enabled a low cost drop-in wireless vent register control system. The register control system is regulated by a master controller unit which collects sensor data from a distributed wireless sensor network. Each sensor node samples local settings such as occupancy, light, humidity and temperature. Then it reports the data back to the master control unit. The master control unit compiles the incoming data and then actuates the vent registers to control the airflow throughout the house. The control system also utilizes a smart thermostat with a movable set point to enable the user to define their given comfort levels. The new system can reduce the run time of the HVAC system, thus decreasing the amount of energy used and increasing the comfort of the home occupants.

The paper in [HAG08] discusses digital objects that have networking and computing capabilities and are present in the network. These objects advertise their presence and capabilities in the form of services so that they can be discovered and, if desired, exploited by the user or other networked devices. It is beneficial for these devices to be equipped with some sort of intelligence and self-awareness to enable them to be self-configuring and self-programming. One way to relieve this load is by employing artificial intelligence (AI) techniques to create an intelligent "presence" where the system will be able to recognize the users and autonomously program the environment to be energy efficient and responsive to the user's needs and behaviours. These AI mechanisms should be embedded in the users' environments and should operate in a non-intrusive manner. The paper shows how computational intelligence, which is an emerging domain of AI, could be employed and embedded in our living spaces to help such environments to be more energy efficient, intelligent, adaptive and convenient to the users. Computational Intelligence will be able to learn the user behaviour in a personalized and unobtrusive manner and adapt to the changes in the environment and the user behaviour in a life-long manner while dealing with a large amount of uncertainties available in these environments and thus delivering

more efficient energy systems. Furthermore, “fuzzy logic” attempts to mimic the way of human thinking to reason in an approximate way rather than a precise way. GAs (genetic algorithms) are particularly suitable for solving complex optimization problems and hence for applications that require adaptive problem-solving strategies. In addition, GAs are inherently parallel since their search for the best solution is performed over genetic structures (building blocks) that can represent a number of possible solutions. The paper in [KUS07] describes practical solutions for constructing the ubiquitous network for building and home control systems. A hybrid network, which consists of the RF wireless network based on the ZigBee specifications and the power/pipe line communication network with high frequency band dispersed-tone method, are developed. An appliance control system was constructed on the proposed hybrid network and installed into actual buildings and homes for evaluation. High performance and high reliability have been confirmed through the year-long field test. The PLC (programmable logic controller) is a method to utilize the domestic power line as a communication cable. The pipe line communication is similar to the PLC. A pair of the refrigerant pipes of the air-conditioner is utilized as communication media instead of the power line in the PLC.

In [TSO06], the authors proposed functions such as intelligence entrance guard's management, home security, environmental monitor and light control for a smart sensor network. It can be implemented by a smart network integrating multimedia access service, image processing, security and sensor, and control technologies. It specifies details of the development process of a smart home network in Taiwan based on ZigBee technology with the combination of the smart home appliance communication protocol, SAANet. The authors in [VAI11a] propose a robust, lightweight and efficient user authentication scheme based on strong-password approach in order to provide secure remote access in home network environments. The proposed user authentication scheme uses the HOTP (Hash-based Message Authentication Code One Time Password) algorithm, hash-chaining technique along with low cost smart cards. The scheme satisfies several security requirements including stolen smart card attack and forward secrecy with lost smart cards as well as functional requirements including no verification table and no time synchronization. The security of the proposed scheme is verified using non-monotonic cryptographic logic (Rubin logic). While comparing with existing representative schemes in

terms of security requirements and functional requirements, it can be seen that even though the proposed scheme has slightly higher computational overheads than the existing representative schemes, it is a more robust authentication mechanism and has better security properties.

2.2.2 Advantages and Drawbacks

There are many advantages of a smart home such as energy efficiency, security, upgraded home appliances, on-demand video programming, safety mechanisms [VEN03], centralized control, remote access, and convenience [PRA00]. Furthermore, the smart home is also beneficial in health care such as independent living and monitoring for the elderly and disabled, safety, remote access, and privacy [RAS07] [HEN07] [JAK07] [LEI07]. Despite these identified benefits, there still exists some drawbacks or challenges of using smart home technologies. These challenges are not purely technical. Rather, they raise cross-cutting issues in technical, social, and design domains [ABO03]. Another important issue is standardization [DIG05]. Venkatesh [VEN03] mentioned six challenges of smart home technologies including the following: consumers' unawareness of the benefits of smart homes; cost and hassle for running additional wires; complexity of technology for most household users; lack of incentive for Internet providers to push networking technology; potential privacy issues; and interface issues. Authors of [GRE04] found many important aspects needed to be considered for smart homes such as cost, reliability, security/privacy/safety, ease of use, flexibility, convenience, maintaining independence or keeping active, future proof technology and some other aspects. Furthermore, there is difficulty in validating the alarms triggered by an older person living alone in a home designed to assist the elderly [CHA08]. The paper in [CER06] mentions that wireless communications in indoor environments (e.g., buildings) is still quite unpredictable when using low-power consumption RF transceivers. They are particularly unpredictable in cluttered environments common inside buildings with many interfering electromagnetic fields such as those produced by elevators, machinery and computers.

2.3 Smart Grid

Smart Grid Conceptual Model of "National Institute of Standards and Technology" [IEE] provides a high-level framework for the smart grid that defines seven important domains: Bulk Generation, Transmission, Distribution, Customers, Operations, Markets and Service Providers. Each individual domain is itself comprised of important smart grid elements that are connected to each other through two-way communications and energy/electricity paths. The distribution domain delivers the electricity to and from the end customers in the smart grid. The customer domain of the smart grid is where the end-users of electricity (home, commercial/building) are connected to the electric distribution network through the smart meters. The operations domain manages and controls the electricity flow of all other domains in the smart grid. The service provider domain of the smart grid handles all third-party operations among the domains.

2.3.1 Benefits

The smart grid eliminates some of the major drawbacks of the current electric grid [SMG08]. There have been three major blackouts in the United States over the past ten years. Blackouts can cause severe problems for traffic lights, credit card transactions, security systems, the economy etc. In many areas of the United States, the only way a utility knows there is an outage in an electrical grid is when a customer calls to report it; that is, there is no system to report the problem automatically. Furthermore, the current grid struggles greatly to meet energy demands because electricity must be consumed the moment it is generated [SMG08]. There is no energy storage. The components of a grid are predominantly dumb conductors and are not controllable [SAN10]. The equipment is old and it is unable to deal with the increasing load [ERO10a].

Table 2.1: The difference between an electric grid and smart grid [SMG08]

Characteristic	Today's Grid	Smart Grid
Enables active participation by consumers	Consumers are uninformed and non-participative with power system	Informed, involved, and active consumers - demand response and distributed energy resources.
Accommodates all generation and storage options	Dominated by central generation- many obstacles exist for distributed energy resources interconnection	Many distributed energy resources with plug-and-play convenience focus on renewables
Enables new products, services and markets	Limited wholesale markets, not well integrated - limited opportunities for consumers	Mature, well-integrated wholesale markets, growth of new electricity markets for consumers
Provides power quality for the digital economy	Focus on outages - slow response to power quality issues	Power quality is a priority with a variety of quality/price options - rapid resolution of issues
Optimizes assets & operates efficiently	Little integration of operational data with asset management - business process silos	Greatly expanded data acquisition of grid parameters - focus on prevention, minimizing impact to consumers
Anticipates and responds to system disturbances (self-heals)	Responds to prevent further damage- focus is on protecting assets following fault	Automatically detects and responds to problems - focus on prevention, minimizing impact to consumer
Operates resiliently against attack and natural disaster	Vulnerable to malicious acts of terror and natural disasters	Resilient to attack and natural disasters with rapid restoration capabilities

The benefits of smart grid are discussed in many papers. The main benefits of the smart grid are given below.

- More efficient energy routing and thus an optimised energy usage and a reduction of the need for excess capacity [SSN09].
- Quality power that meets 21st century needs [SAN10].
- Better monitoring and control of energy and grid components [SSN09].
- Improved data capture and thus an improved outage management [SSN09].
- Handling increased consumer demand without adding infrastructure [SMG08].
- Two-way flow of electricity and real-time information allowing for the incorporation of green energy sources and demand-side management [SSN09].

- Enabling active participation by consumers in demand response [SAN10].
- Resilient to natural disasters, physical attacks and cyber-attacks [SMG08].
- Ability to integrate new products, services, technologies and markets. It is flexible [SAN10].
- Highly automated, responsive and self-healing energy network with seamless interfaces between all parts of the grid [SSN09].
- Reduction of greenhouse gases which is mainly due to utilizing wind and solar energy [RAC09].
- Storage for intermittent energy sources, small-scale generation, micro-grid [ERO10b].
- Supports many distributed energy generation sources [SMG08].
- Reduce power needed during peak times [SSN09].
- Lower Operation and Maintenance Costs [SSN09].
- Economical as it provides best value through innovation and efficient energy management [SAN10].

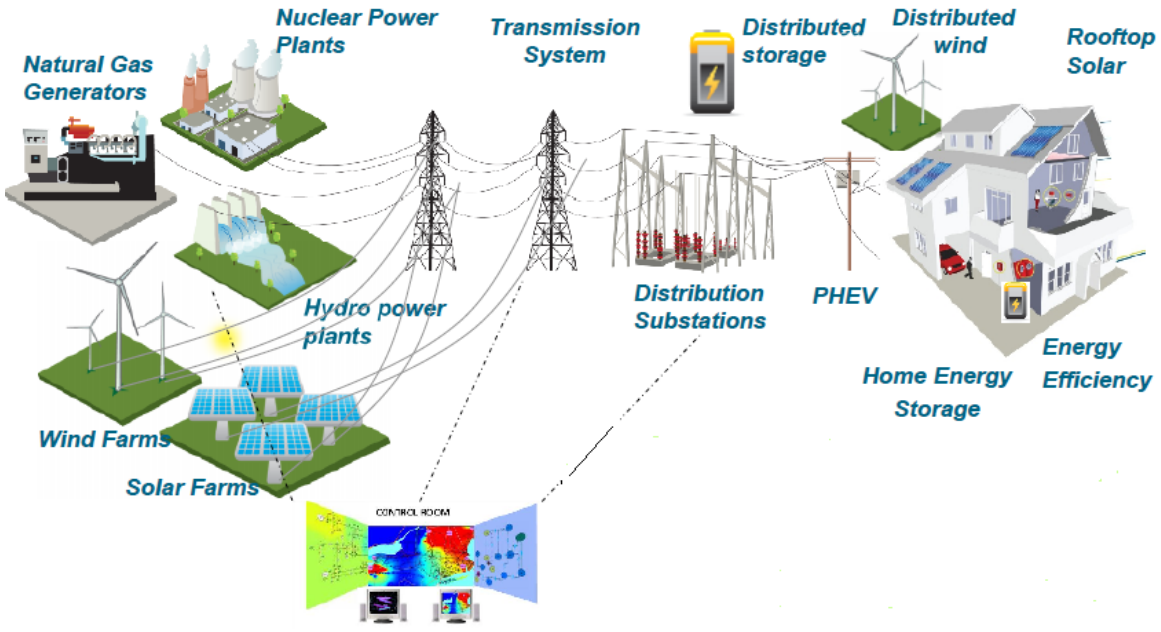

Figure 2.2: Smart Grid Elements and its Control System [GRI10]

The elements of a smart grid are depicted in the Figure 2.2. The scope of the smart grid extends from centralized bulk generation to distributed generation (DG); from transmission systems to distribution systems; from utility control centers to end-user home-area networks; from bulk power markets to demand response service providers; and from traditional energy resources to distributed and renewable generation and storage [SAN10]. Unlike electricity, energy can be economically stored with the application of smart grid technologies. Demand response greatly reduces energy consumption [SMG08]. The smart grid is capable of delivering quality power that is free of sags, spikes, disturbances and interruptions [SMG08]. The distribution of energy production from renewable sources increases the resilience of the grid in the face of widespread disturbances (e.g., blackouts) and reduces demand for centralized power plants [SAN10]. This also reduces congestion and the need for new infrastructure [SMG08]. Furthermore, since the energy generated is usually closer to the loads [IPA09], it can be supplied more quickly, reliably and economically. Wind or solar power is not expensive to integrate nor does it require dedicated backup generation or storage. Developments in tools such as wind forecasting also aid in integrating wind power and dealing with the intermittent nature of wind [MIL09]. SG is increasingly resistant to attack and natural disasters as it becomes more decentralized and reinforced with security protocols [SMG08]. Specific firewalls and interfaces are used for data sharing and communications in order to provide security.

2.3.2 Components of the Smart Grid

From a technical components' perspective, the smart grid is a highly complex combination and integration of multiple digital and non-digital technologies and systems [SSN09]. The main components of a smart grid are the following:

i) new and advanced grid components,

ii) smart devices and smart metering,

iii) integrated communications technologies,

iv) systems for monitoring, advanced control and decision support.

2.3.2.1 New and advanced grid components

In [AMI05], the author proposes a way to make an electric grid smarter. He mentions that to add intelligence to an electric power transmission system, we need to have independent processors in each component and at each substation and power plant. These processors must have a robust operating system and be able to act as independent agents that can communicate and cooperate with others, forming a large distributed computing platform. Each agent must be connected to sensors associated with its own component or its own substation so that it can assess its own operating conditions and report them to its neighbouring agents via the communications paths. Thus, for example, a processor associated with a circuit breaker would have the ability to communicate with sensors built into the breaker and communicate those sensor values using high bandwidth fibre communications connected to other such processor agents. Furthermore, superconducting power cables will be introduced that will reduce line losses and will carry more power than traditional copper-based cable [SMG08].

2.3.2.2 Smart devices and smart metering

Demand response (DR) is a term used for programs designed to encourage end-users to make short-term reductions in energy demand in response to a price signal from the electricity hourly market, or a trigger initiated by the electricity grid operator. Price signals are prices that are set for a specific time period in the future. According to Time-Of-Use (TOU), energy rates differ in peak, moderate peak and off-peak hours. During peak hours, consumers are charged more [ERO10a]. Typically, DR actions would be in the range of 1 to 4 hours and include turning off or dimming banks of lighting, adjusting HVAC levels, or shutting down a portion of a manufacturing process during peak hours to decrease energy costs. Alternatively, onsite generation can be used to displace a load drawn from the power grid. Onsite generation incorporates renewable, micro-grid energy sources such as residential solar panels. The demand response increases awareness of energy usage; enhances reliability by decreasing stress on the grid; and support the use of renewable energy resources, distributed generation, and Advanced Metering Infrastructure (AMI). AMI provides utilities with the ability to detect problems within their systems and operate them more efficiently. Furthermore, price signals can be relayed to "smart" home controllers or end-consumer devices like thermostats, washer/dryers and

refrigerators. The devices, in turn, will process the information based on consumers' learned wishes and will power accordingly [SMG08]. In addition, utilities can directly control certain devices (water heaters, air conditioners, electric heating, pool pumps, etc.) during times of critical demand in exchange for some sort of monitory rate reduction or rebate. This more traditional method is called "demand control" [SAI09]. These will reduce the time consumers need to participate in order to decrease their energy consumption as it will be done automatically once the consumers set their preferences. Furthermore, energy gateways can be used in smart homes. The advantage of such gateways is that it can integrate more devices and consumption points to be parts of a Home Energy Management System. Other advantages are that it facilitates remote access and control of devices in real-time, displays more detailed energy consumption data in real-time and controls load automatically [RON10].

2.3.2.3 Integrated Communications Technologies

Information provided by smart sensors and smart meters needs to be transmitted via a communication backbone. This backbone is characterized by a high-speed and two-way flow of information. This communication is between different components of the grid. Integrated communications connect components to open architecture for real-time information and control. Thus, allowing every part of the grid to both 'talk' and 'listen' [SMG08]. The integrated communications technologies of the smart grid are discussed in [SSN09]. Different communication applications and WAN technologies form the communication backbone. The choice of WAN technologies will depend on factors such as reliability, affordability, security and the network infrastructure that is already available. It is likely that utilities will rely on several network technologies when they build smart grids. Some examples of WAN technologies that can be used in a smart grid are ADSL (Asymmetric Digital Subscriber Line), Cable Modem, FTTH (Fibre to the Home), Power line communications like BPL (broadband over power line), cellular services, Wimax, and satellite services. Wireless communications for the smart home have low power and low data rate. Examples are ZigBee, low-power Wi-Fi and Z-wave. Z-wave is usually embedded in smart appliances [ERO10b]. ZigBee does not support IP and therefore 6lowpan is used to transfer IP packets over ZigBee.

2.3.2.4 Systems for monitoring, advanced control and decision support

In [SSN09], it is mentioned that sensors are used at multiple places along the grid, e.g., at transformers and substations or at customers' homes [SHA09]. They play an important role in remote monitoring and they enable demand-side management. Spread over the grid, sensors and sensor networks monitor the functioning and the health of grid devices, monitor temperature, provide outage detection and detect power quality disturbances. Control centres can thus immediately receive accurate information about the actual condition of the grid. Consequently, maintenance staff can maintain the grid in real time in the case of disruptions rather than rely on interval-based inspections. Smart meters at homes provide the possibility to read energy consumption of a home both locally and remotely [SID06]. Furthermore, it provides the means to detect fluctuations and power outages, permit remote limitations on consumption by customers and permit the meters to be switched off. This results in important cost savings and enables utilities to prevent electricity theft. Electricity providers get a better picture of customers' energy consumption and obtain a precise understanding of energy consumption at different points in time. As a consequence, utilities are able to establish demand-side management (DSM) and to develop new pricing mechanisms. The authors of [SAI09] mentioned some other methods of detecting or locating fault. Voltage sensors at the end of the line can also be used to locate fault.

Another feature of the smart grid is peer-to-peer communication to switch critical loads from one feeder to another. These types of switching technologies automatically isolate the faulted section and determine if the other loads can be picked up from another circuit. A Global Positioning System (GPS) can also be used to geographically represent locations of load. Furthermore, in case of power line carrier based meter reading infrastructures, the noise levels of the communication can be monitored to determine that a device on the distribution system, such as an insulator or arrester, is about to fail. The presentation in [LOC10] discussed how geocoding, thematic mapping and Geographic Information System (GIS) can be used to monitor the grid. The grid can be displayed as an interactive map. It could facilitate the localization and prevention of failure as potential failure could be graphically displayed on a map. Web (Web 2) based systems like Google Earth, satellite images can also be used for this purpose.

In a conventional grid, a supervisory control and data acquisition (SCADA) system is used to monitor and control the grid. Phasor measurement units (PMUs) are being used for a monitoring and controlling system in a smart grid. The general objective of these PMU installation activities is to eventually make a transition from the conventional supervisory control and data acquisition (SCADA)-based measurement system to a more advanced measurement system that will utilize synchronized measurements from geographically distant locations and increase the situational awareness by monitoring a wide area of the power system in real time. The measurements made by PMU are more accurate than that of SCADA. PMU measures voltage, current and other units. Thus, PMU provides a better monitoring, protection and control system. Furthermore, there will be a new and advanced level of controllability in the smart grid. FACTS (flexible ac transmission systems) technology will enable system operators to route power flows along the most efficient paths and find the best power production mixes and schedules [SAN10].

In [SSN09], it is mentioned that another key component area of smart grids comprises decision support and control systems. The data volume in smart grids will increase tremendously compared to traditional grids. Thus, one of the main challenges for utilities is the integration and management of the generated data and making the data available to grid operators and managers in a user-friendly manner to support their decisions. Artificial intelligence (AI) methods as well as semi-autonomous agent software minimizes data volume [SAN06]. New methods of visualisation enable integration of data from different sources and present them in a clear and concise form. GIS provide geographic, spatial and location information along the smart grid. Control systems monitor and control essential elements of the smart grid. New developed subsystems are capable of making information available in the whole grid and thus provide better power management. VERDE (Visualizing Energy Resources Dynamically on Earth) is a system that will provide wide-area grid awareness, integrating real-time sensor data, weather information and grid modeling with geographical information [SMG08]. All of this information enables utilities to efficiently monitor and control the grid. Furthermore, intelligent electronic devices, substation automation system, AI methods use the collected information to give decisions. Controllable technologies for supply, demand, power flow, and storage provide the means to implement decisions made by smart control algorithms [SAN10].

The paper in [ERO11a] discusses the potential applications and the challenges of employing wireless multimedia sensors and actor networks for the smart grid. It states that collecting multimedia content could highly improve the safety and security of the grid. For example, a bird collision with a wind turbine is a significant problem affecting the healthy operation of the turbine. For instance, upon detection of a collision by the acoustic sensors i.e. microphone, image sensors may wake up to capture images of the blades and the tower to identify the faulty component, e.g. blade, pitch, gear box. Meanwhile, during the restoration, actors can shut down the faulty turbine.

PHEV (Plug-in Hybrid Electric Vehicles) is mentioned as an excellent application for the smart grid in [SMG08]. These vehicles offer consumers the opportunity to shift use of oil and gasoline to electricity and to power a car from the grid. Furthermore, PHEVs reduce greenhouse gases. PHEV can be associated with the smart grid for storing power during off-peak periods and selling it back to the grid when the grid requires it in peak periods. Parked PHEVs with Vehicle-to-Grid (V2G) capability can provide energy to the grid and alleviate localized distribution system overload problems [IPA09]. The authors in [VAI11b] proposed multi-domain network architecture for V2G infrastructure which includes comprehensive hybrid public key infrastructure (PKI) using hierarchical and peer-to-peer cross-certifications. By system analysis and performance evaluation, it is shown that the proposed mechanism is better than the traditional hierarchical model, and it is more efficient than the hybrid PKI scheme using an explicit certificate. Smart grids do not end at the substation but interact with the city.

The smart city consists of dwellings that adjust their energy consumption to market prices, electric vehicles that charge (and discharge) throughout the city. Smart services use networked sensors and actuators deployed in the city. Thus, allowing the authorities to monitor the environment in real-time, to react immediately and to establish automated control processes. These services rely heavily on information and communication technologies. A set of such services and the intelligent infrastructure form the basis of smart cities. Each appliance (e.g., the refrigerator, kettle, toaster, washing machine) has its own energy fingerprint, or “appliance load signature” that a smart meter can read. The smart meter transmits power-usage information to the

utility as frequently as every 15 minutes. Anyone who gets hold of this data gets a glimpse of exactly what appliances you use and how often you use them. Criminals can use this information to identify the best times for a burglary or to identify high-priced appliances to steal. To counter this, Toshiba developed a system that would hide a smart meter address before sending energy-usage data to utilities [BLE10].

2.3.3 Drawbacks or Challenges

The smart grid has some drawbacks that it must take into consideration or it has to deal with some challenges. Many of the smart-grid applications are new with limited technical standards and no established industry business practices [IPA09]. It will be expensive to empower the current grid with technologies in order to make it smart. The interoperability among standards and backward compatibility will be an issue due to lack of agreement about common standards. Additionally, there is a security concern especially regarding cyber-attacks. Optimization is required in terms of managing, storing and utilizing data [ERO10b]. Furthermore, the electricity grid will face a significant challenge by demand generated by the electric vehicles which will be on the roads soon. Electric vehicles will increase the load on the grid and most probably during peak hours and in populated locations [ERO10a]. The intermittent nature of wind and solar energy generation poses certain operational challenges for the transmission grid. Furthermore, a wind farm creates significant ramping. Many large renewable resources are located where limited transmission capacity exists and are far from load centers. Therefore, additional ancillary services (e.g., spinning reserves and regulation) are required to maintain reliability and operational requirements of the grid [IPA09]. Existing applications and components of the grid need to be changed or updated. Many of these changes are incremental with respect to the existing capabilities and might not be an economical and operationally acceptable option [IPA09]. The smart meter only measures aggregated energy consumption of a house and therefore it is difficult to understand which appliance uses most of the energy. There is a concern of how much a consumer is willing to pay for a cost related to DSM (e.g., smart meter, thermostat) to make energy savings. There are some misconceptions among consumers that utilities will limit the power they use and will make a profit; consequently, customers may need to get up around 3am

to do laundry. Another issue is that if the consumer is able to reduce the bill to zero by load-shifting and selling energy back to grid then it will affect the profit made by the utility provider.

Although it is a challenging task to provide intelligence to the current grid, the smart grid will provide many advantages. The overall benefits of SG will outweigh the costs and challenges needed to bring the change. It is important for the utility provider, residential and industrial consumer, and other stakeholders to understand the overall benefits of SG and work together to bring about this change.

2.3.4 Recent developments in smart grid deployment

Currently, smart meters are being deployed to homes in Canada, the United States and Europe. Furthermore, several utility companies employed Time-Of-Use (TOU) rates to encourage consumers to shift their loads to off-peak hours. Utilities have AMI and are able to read the meters remotely. Consumers are being educated about the advantage of smart meters and TOU rates. Some web applications, like "Google PowerMeter" [POW], have been developed that will enable the consumer to view electricity consumption from the previous day. "Microsoft Hohm" [HOH09] is a free Web service that helps make smarter decisions about energy saving. Consumers can log into the Microsoft Hohm site, and start off by entering the zip code. Using this simple location information, Hohm uses algorithms to start predicting home energy consumption. Users can enter as much information as they want about home size, water heater brand, etc. to make the energy prediction of their home as accurate as possible. Some energy management applications have also been developed. Rogers Communications Inc. has introduced a "Smart Home Monitoring System" [SHM11] in 2011. The user of this system can use a web portal or smartphone to remotely read and control temperature, lighting and appliances. Web portal connects to a central home controller and the user is able to control home elements. The system does not use WS. "Intelligent Home Energy Management" [INT10] is a wall-mounted device that allows a residential consumer to view and control various electrical appliances. It is developed by Intel Corporation. It provides current and historical reports on energy consumption. Users can remotely view and control thermostats, appliances and security systems from a mobile phone or PC. Apple Inc. has also developed a "Smart Home Energy management

system". "Tivoli Monitoring for Energy Management" [TIV] system provides visibility into key energy consumption and environmental metrics for IT, facility, and enterprise asset infrastructures. It also identifies areas where energy consumption and costs can be reduced. It is a product of IBM Corporation. Furthermore, smart appliances can accept signals from the utility that instruct it to go into an energy-saving mode or turn off during peak hours. The feed-in-tariff (microFIT) program allows a homeowner to sell renewable energy back to the grid for a guaranteed price. A local distribution company helps the consumer with connection to the grid [MIC10].

2.4 Web services in the smart home

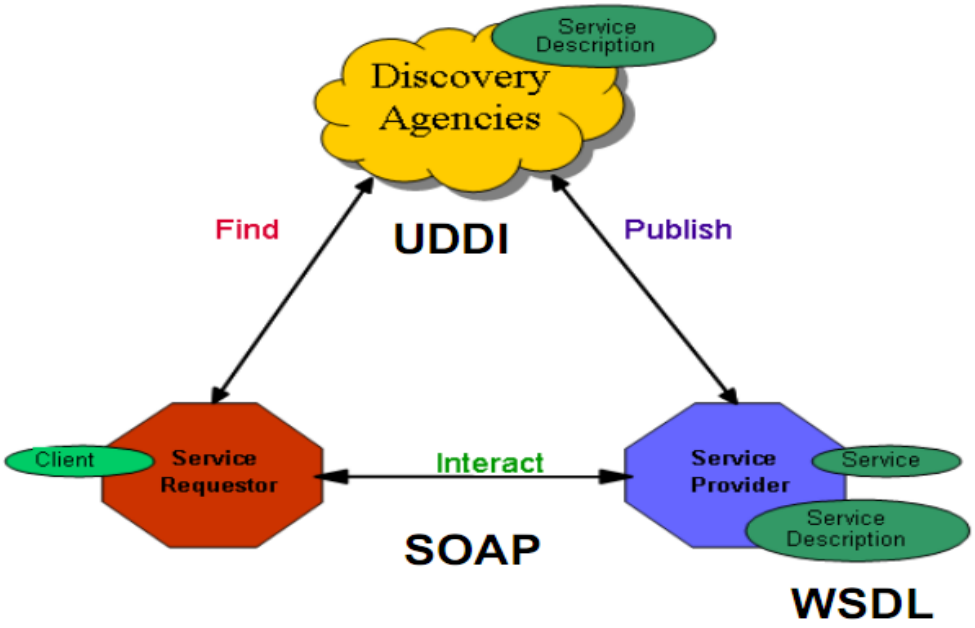

Figure 2.3: Web Services Framework (realization of SOA)

The Web services framework is shown in Figure 2.3. The paper in [CUR02] discusses the Web services framework in some detail. Web Services Framework consists of WSDL, UDDI and SOAP. All elements of this framework are based on XML and their structure is described in this paper. WSDL is an XML based language that describes Web services. It specifies the operations that the service provides along with the message and type format. It contains details about

communication protocol and how to access service over this protocol. It also specifies location of Web services. The next element is UDDI. It is a directory of Web services. A UDDI repository stores the WSDL file. It enables service providers to publish their Web services and it offers users a unified and systematic way to find service providers. SOAP is used to access it, which is another element of Web services framework. SOAP is used for communications among Web services. It is based on XML and it works on existing protocol like HTTP. SOAP is platform independent and it can be implemented in many programming languages.

2.4.1 Web Services in smart home

The paper in [ASA11] analyzes various Demand-Side Energy Management scenarios that become available with sensor network Web services. The authors assume a smart home with a wireless sensor network where the sensors are mounted on the appliances and able to run Web services. The web server retrieves data from the appliances via the Web services running on the sensor nodes. These data can be stored in a database after processing, where the database can be accessed by the utility, as well as the inhabitants of the smart home. They showed that their implementation is efficient in terms of running time. Moreover, the message sizes and the implementation code are quite small which makes it suitable for the memory-limited sensor nodes. Furthermore, they show some application scenarios that provide energy saving for the smart home. SOAP protocol is used to communicate with Web services. In [XU10], a new system is proposed which can share the home appliance abilities as Web Service. By this system, a lot of related applications can be developed easily and can give end users friendlier, flexible control interfaces. Furthermore, functions of different appliances can be integrated to provide more complex services to the user, which is known as a Home Appliance Mashup. A separate controller is used to communicate with sensors. WSs are used to communicate with the controller.

The paper in [PER08] deals with interoperability issues among Heterogeneous Systems in the smart home. Heterogeneous systems in the smart home environment consist of home entertainment, surveillance, access control, energy management, home automation, assistive computing and healthcare. Interoperability among heterogeneous systems involves not only system interconnectivity with multiple entities but also joins execution of tasks. The paper

describes interoperability issues that need to be considered and presents a solution based on Web services technology, particularly SOAP, to solve the interoperability problem in the smart home environment. It proposes the utilization of the Ethernet cloud as basic connectivity interoperability in the smart home environment. Regarding network interoperability, TCP (Transmission Control Protocol) is utilized to perform message exchanges between heterogeneous systems in the smart home environment. TCP hides the details of actual interactions between communicating heterogeneous systems from users. The use of TCP here is justified as there is always one distinct approach in the field of a smart home environment and that is to incorporate TCP based networking into embedded consumer devices as well as appliances. The home entertainment system and surveillance system communicates with each other via Web services that are implemented at the core. Both systems are capable of processing SOAP messages. The authors of [AIE06] argue that the Web service stack should be implemented on smart home devices equipped with sufficient computational power to solve the issues of openness, scalability and heterogeneity. Communication and coordination could be achieved using Web services technology and WSNotification in particular. They discuss a scenario where groups of sensors and actuators communicate with a controller, which in turn communicates with the centralized monitor of the home (usually a PC). Sensors and actuators do not support XML and Web services, but their controllers do. Web services are used as the standard for communication between the centralized controller and its clients. A Service-Oriented Middleware for Heterogeneous Sensor Networks (Sensor Web 2.0) is presented in [SEN08]. Sensor Web combines sensors and sensor networks with SOA. Services are defined for common operations including data query, retrieval and aggregation, resource scheduling, allocation and discovery. The Open Sensor Web Architecture Middleware is built upon a uniform set of operations and standard sensor data representations. Sensor Web Enablement (SWE) method is defined which includes specifications of interfaces, protocols and encodings that enable discovering, accessing, and obtaining sensor data as well as sensor-processing services. Sensor networks can be discovered, accessed and controlled over the Internet. The authors of [GLO09] present an approach to integrate WSNs seamlessly into business process (i.e., SOA) environments using Business Process Execution Language (BPEL) and Web Services while using only very few resources on the sensor nodes. Their objective is to make sensor nodes fully

capable of interacting with standard Web services in the Internet without using any proprietary middleware. Furthermore, it is pointed out that existing research targeted at middleware systems for WSNs is still too specific to be actually used in the industry. The major benefit of the approach is twofold. On the one hand, services offered by the WSN can be used seamlessly in enterprise-level business processes without the need for hand-crafted code for data conversion. On the other hand, these services can quickly be composed to higher level applications by simply modifying the business process. XML is required to interact with standard Web services. The downside of XML is that it is very verbose resulting in large overhead in terms of size. The resulting length of network messages – which are used to interact with Web services and are comprised of HTTP, TCP/IP and XML data – easily surpasses the maximum packet length and bandwidth of radio interfaces available on typical sensor nodes. Apart from message length, the amount of code required to implement HTTP, TCP/IP and an XML parser also far exceeds the capabilities of sensor nodes. To overcome this, a solution is presented that comprises of SOAP-message compression (SMC), the Lean Transport Protocol (LTP) (transport protocol for Web service) and an integration of both SMC and LTP into a BPEL engine. The authors of [PRI08] propose a Web service based approach to enable an evolutionary sensornet system where additional sensor nodes may be added after the initial deployment. Web services are used to expose the functionality and data provided by the sensor nodes. The functionalities are described programmatically (i.e., WSDL) and data is represented in a structured format (i.e., XML). Most of the message formats and packet exchanges can be programmatically and automatically generated from the programmatic description. Therefore, all relevant applications can access the functionality of sensor nodes programmatically via XML. A key challenge in using Web services on resource constrained sensor nodes is the energy and bandwidth overhead of the XML data formats and the WSDL used in Web services. These overheads are evaluated and optimization techniques like low latency messaging and sleep modes are discussed to overcome the challenge. The TCP/IP overheads can be reduced by using persistent TCP connections; disabling delayed TCP acknowledgements; and using link layer retransmissions instead of TCP retransmissions in case of lost packets. Furthermore, XML parser on the sensor can be designed to only parse the simple messages that are specified in its own WSDL. The XML format can be optimized by replacing some of the method names and argument names with very compact tags. The sensors

sometimes go to sleep mode to conserve battery-power. Sleep modes are supported by WS-Eventing and persistent TCP. A home energy management application that uses sensors is used to demonstrate the proposed approach.

The paper in [CHU08] proposes a design scheme of intelligent building OAS (Office Automation System) based on Web services that can offer considerate services and convenience to users in this building. The function of booking rooms via a hotel management module can be packaged as a Web service, so that other enterprise applications can invoke it. Taking design and implementation of booking ticket module in intelligent building OAS as an example, the paper introduces development process of Web services in detail. The paper adopts Apache Axis to develop Web services based on J2EE. Then it uses AdminClient (i.e., a deployment tool of Apache Axis) to deploy it in Tomcat according to the description of the deployment file [LIN05]. At the same time it produces the corresponding WSDL file. At last the WSDL file is registered in a UDDI registry. Web services are invoked through JSP (Java Server Pages). The authors in [YAZ09] present an IP-based sensor network system where nodes communicate their information using Web services, allowing direct integration in modern systems. The system uses two mechanisms to provide a good performance and low-power operation: a session-aware power-saving radio protocol and the use of the HTTP Conditional GET mechanism. They showed that Web services are a viable mechanism for use in low-power sensor networks and that Web service requests can be completed well below one second and with low power consumption, even in a multi-hop setting. An Application Framework for web-based smart homes is presented in [KAM11]. A 6LoWPAN-based wireless sensor network is included inside the home environment, addressing issues such as device discovery and service description. The recent progress in embedded IPv6 makes it possible to run IPv6 applications directly on sensor nodes. Web techniques, such as HTTP caching and push messaging, facilitate the efficient operation of a fully web-based smart home. Through a technical evaluation, they show the benefits of directly web-enabling embedded sensors in terms of performance and energy conservation. The development of a web-based graphical application abstracts home automation procedure for typical residents. REST has been used for web-based interaction with household appliances. Furthermore, Web Application Description Language (WADL) is used to model the resources (services) provided by

an embedded device. In [PAR09], the design and development of a flexible smart home architecture using a peer-to-peer (P2P) approach is proposed. The authors specifically focus on two distinct aspects of their proposed architecture. First, they analyze how the different home devices and services can be represented as individual peers in order to have a decentralized system. Second, they investigate the distribution of application workflow logic among the peers to develop a flexible home architecture with autonomous behaviour of the peers. They analyze the suitability of Devices Profile for Web Services (DPWS) to realize the proposed P2P-like architecture for the smart home. They further showed how to distribute the application workflow logic among the peers and yet achieve the same global behaviour of the system. Their experimental results show that DPWS provides tools and techniques, in particular its discovery and eventing mechanism, which can be leveraged to provide flexibility and autonomy in the overall architecture. DPWS is based on a SOA paradigm and with clearly stated and differentiated client and server roles. Both roles, however, can simultaneously be implemented in the same component, thus enabling P2P-like architectures. The proposed approach not only provides the flexibility of adding or removing new or existing devices to and from the home network, it also ensures scalability and removes the burden from the central entity usually encountered in a traditional server or gateway based solutions.

In [XU09], the authors propose an ontology-based framework to facilitate the automatic composition of appropriate applications for smart homes. This is done to make the applications configurable and adaptive. Ontology is an explicit specification of conceptualizations which organizes the semantic (i.e., meaning-related) information as the knowledge base of the specific application domains. The proposed system composes appropriate services depending upon the user profile and available equipment in each individual household automatically. A generic knowledge representation is used to facilitate this composition. Moreover, it dynamically adjusts the environment parameters to match the customer needs and to encompass the available resource. Customers are able to specify their usual behavior templates as a different mode via customized function template editing. The system consists of a smart home knowledgebase (ontology), a household database, a service manager and a plan deployment component.

2.4.2 Security and Quality of service provided by Web service

The paper in [KIM05] suggests a priority assignment method for providing differentiated Web services on the web server not at the network level but at the application level. For implementing this method, the Web services quality factors were analyzed to extract the factors required for assigning priorities. The suggested method assigns the priority dynamically in order to satisfy the service level agreement as much as possible. The paper in [LIU09] discusses the design of SOA-based Web service systems for the satisfaction of quality of service requirements using Quality Function Deployment (QFD). QFD is a major quality management system used to determine product development characteristics from customer requirements. It can trace customer needs in terms of quality of service requirements to manage design attributes. It can be used to determine technical targets of Web service design attributes based on their impact on satisfaction of quality of service requirements. It shows how to apply the QFD for the development of SOA-based Web service systems to improve their customer satisfaction. The paper in [ALA09] shows that the secure transfer of sensitive data among subsystems of the smart home can be achieved by using secure Web services for communication purposes and is demonstrated by prototyped implementation. The pharmacy subsystem uses a Web service provided by the smart home to forward the prescription data from the pharmacy into the home system.

The paper in [MAA09] introduces a Web services-centric solution to handle privacy concerns. In addition the solution uses policies to control how Web services handle privacy concerns and penalizes those Web services that are not binding to these policies. Moreover the solution uses trust and reputation to select the most trustworthy Web services. The physical level represents a smart home with all its constituents such as a microwave, TV sets and lights. Each constituent binds to a group of wireless sensors for monitoring and data collection purposes. The logical level represents the software applications that control the smart home's constituents and take appropriate actions as need be. This control is driven by scenarios such as temperature adjustment, light switching, etc. Software applications are built-upon Web services that are sometimes put together to constitute composite Web services. SAML (Security Assertion Markup Language) [LOC] is a specification language that defines how to specify security credentials, which are represented as assertions. The standard purpose of using SAML is to realize Web

Single Sign-On. XACML (eXtensible Access Control Markup Language) [OAS04] is an extension to SAML that focuses on access control rights. XACML defines how to express access policies. Furthermore, it specifies a request/response protocol between a policy decision and a policy enforcement point. XACML is considered the better way to implement role- based access control (RBAC) [HAI06] which restricts the WS accessibility according to predefined security policies and rules. The authors of [PAC08] address some deficiencies in BPEL. Significant omissions from BPEL are the specification of activities that require interactions with users to be completed, called human activities, and the specification of authorization information associating users with human activities in a BPEL business process and authorization constraints, such as separation of duties on the execution of human activities. To overcome it, the authors introduced a new type of BPEL activity to model human activities. They also developed RBAC BPEL—a role-based access-control model for BPEL—and a language to specify authorization constraints. The authorization information is encoded using XACML. They have proposed an algorithm to evaluate whether a request by a user to perform an activity in a BPEL process can be granted or not. The algorithm verifies whether the execution of a BPEL process will complete without violation to the authorization constraints. Finally, they have proposed an implementation of access-control enforcement on the BPEL engine that does not require any modification to existing engines.

The paper in [YAM09] proposes a metadata for quality of security service for SOA. The proposed metadata provides different levels to describe the available variations of the Authentication, Authorization and Privacy features that are related to SOA security. This metadata is flexible and editable and is divided into four basic levels: High, Moderate, Low and Guest. These levels allow the varied security requirements of the service provider and consumer to be satisfied. A Web service for Quality of Security Service (QoSS) is then constructed to encapsulate the suggested metadata in order to assist the service consumer and provider to achieve a QoSS agreement meeting both of their requirements. The QoSS agreement will perform as an enforced policy for managing the interactions between the service provider and consumer. The service of QoSS is located inside a complete framework for securing SOA.

The paper in [ERR06] presents the core architecture and features of WSDiffServ, a practical middleware for differentiated Web services responsiveness from a service provider's perspective. The architecture leverages the profiles of the service users to prioritize incoming requests coupled with strict and adaptive scheduling strategies to regulate the service throughput at a desirable level and improve the response time for the preferred classes of requests.

The authors of [MOU10] propose a new approach for the dynamic enforcement of WS security. Security policies are specified as aspects and the aspects are then integrated in the BPEL process at runtime. The paper in [REK09] proposes a security approach based on the PKI infrastructure and UDDI functioning to provide adequate security for Web services. After finding the requested Web service, a client contacts the WS provider to negotiate the service access procedure. These first contacts between clients and providers are usually and commonly not protected (encrypted) yielding enough room for hackers to intrude into these unprotected messages. The proposed approach overcomes this vulnerability.

2.4.3 XMPP

The paper in [HOR09] presents a XMPP-based WSN, with web-service like access, using XMPP and Atom feeds, also interfacing with UPnP. A new approach based on XMPP and OSGi (Open Services Gateway initiative) technology to home automation on the web is proposed in [HOR10]. An application of a fault-detection mechanism for Internet-connected smart refrigerators is demonstrated. It shows the feasibility of using an XMPP server and XMPP client via a refrigerator for message notification. The paper in [LOV00] mentions that the toaster was one of the first devices to be connected to the Internet for remote control. This was done back in 1990. The authors discuss about connecting a thermostat to the Internet. Therefore, it is very convenient to connect electrical appliances to the Internet. The authors in [WEI10] introduce the smart home Cloud model, which is based on the present Cloud architecture and modifies the traditional service layer to provide efficient and stable services for the smart home. Smart home nodes and Cloud server form a peer-to-peer network, which can help the Cloud server to reduce bandwidth pressure when transmitting higher quality audio/video signals. The smart home gateway describes their services in WSDL and registers them to the

Cloud service directory so that other homes can search and consume the service. The authors in [WAR09] discuss the technical and management perspective of integrating Web services into a smart grid.

2.4.4 Advantages

There are many advantages of using Web services in a smart home. Smart homes have relied on the Service Oriented Computing (SOC) approach to simplify its design, shorten the development time and reduce cost [ALA09]. In a smart home, Web services could provide interoperability among heterogeneous systems that need to work together and perform their tasks efficiently. On the syntactic interoperability, it is evident that SOAP could be the ideal solution as a defined structure for message exchanges. SOAP is very well suited for providing interoperability among heterogeneous systems due to the standard way of data representation, and the format is extensible to deal with changing requirements [PER08]. Web services could solve the issues of openness, scalability and heterogeneity [AIE06]. Web services mechanism has the capability of loose coupling, fine encapsulation and high grade integration [CHU08]. Web services provide the functionalities that allow users to remotely interact with the systems empowering the smart homes [MAA09]. Secure Web services could be used to transport private and confidential data [ALA09]. Web services could bind to a policy in order to control privacy. Web services can also provide an access control mechanism. Furthermore, Web services allow for building systems both in a client-server and a peer-to-peer fashion [AIE06]. The P2P architecture enables a system to be decentralized. Decentralized systems are more flexible, scalable and avoid single-point of failure encountered in a centralized server [PAR09].

2.5 Broadcasting in a mobile network

A protocol to broadcast messages in highly mobile networks is described in [HAH03]. The protocol increases the availability of data by using data replication. Nodes maintain a local copy of the most recent state of all objects. Nodes also maintain a field called TTL (time for data to live). The TTL is decremented by each node that holds a copy of the data. A node *u* discovers

new node *v* (*v* is not in *u*'s current neighbor list) if it receives a hello or data message from *v*. Whenever a node discovers a new neighbor or receives a new message, it sends an advertise message as long as TTL has not expired. If TTL = 0, the data is deleted. Instead of sending a whole message, a node sends a short (advertise) message about its data. Upon receiving the advertised message, the nodes that need the data send a request message. If at least one neighbor requests data, the data is flooded. When the full message is relatively short, this protocol is inefficient. Many advertising strategies are used. One strategy is to give data that has not been sent before a higher priority compared to data that has already been sent. Another strategy is to give data that has been sent a few times a higher priority over data that has been sent even more times. The advantage of this method is that it increases data reliability as data is advertised more than once. The downside is that there is message overhead due to some additional advertise and request messages.

A broadcasting method based on a *neighbor elimination scheme* (NES) is described in [PEN00, STO02]. This scheme provides a simple way to eliminate redundant messages. In NES, a node does not need to rebroadcast a message if all of its neighbors have been covered by previous transmissions. In this method, when a node receives a message to broadcast, it sets a timeout period before transmitting. That is, nodes do not retransmit immediately, but wait for timeout duration while monitoring their neighborhood. If all neighbors become covered during the monitoring period then the node does not transmit. Otherwise, if the node's timeout expires and there are still uncovered neighbors, the node transmits. *Timeout = (C / numberUncovered)*, where *C* is a constant and *numberUncovered* is the number of neighbors that have not received the packet, based on the node's knowledge. Flooding algorithms for dynamic networks with dynamic edges and fixed nodes are proposed in [ODE05]. Among the assumptions they use are that the network stays connected at all times, and evolves slower than the message transit time between adjacent nodes. The CounterFlooding algorithm requires one input *T* and intercepts two types of events: connectivity-driven events and message-driven events. When a message is first received at a node *u*, the counter *c* is reset to 0, and the message is retransmitted. If *u* is notified of a neighborhood change with the arrival of a connectivity-driven event, it broadcasts the message and increments the counter *c* by one until it reaches maximum *T*. The authors of

[SOE06] relax the flooding algorithm's assumptions by removing the requirement that the network stays connected at all times, and extended the algorithm to solve the problem where dynamic nodes are also involved. This is achieved by introducing a delay between the actual connectivity change and its corresponding event.

2.6 Summary

This chapter presented a survey of the related work done in the area of the smart home, smart grid, Web services, XMPP, security and broadcasting. This chapter is divided into four main sections. The first, second, third and fourth section focuses on the smart home, smart grid, Web services and broadcasting respectively. In addition, the second section discusses energy management systems and recent developments in smart grid deployment. Furthermore, the third section consists of related work done on Web services security and usage of Web services in the smart home. In addition, the fourth section has literature review on broadcasting in mobile networks. Moreover, there is a conclusion at the end of each section discussing the advantages or challenges of work presented in that particular section.

There are quite a few existing home energy management systems. However, the proposed system overcomes some of the drawbacks of existing system. The system in [SHM11] enables user to remotely read and control home devices via web portal. A web portal connects to a controller and the user is able to control home elements via this controller. The system does not use WS. In the proposed system, a web service is used that enables the user to directly interact with different home elements and directly call operations like control temperature. Some papers like [ASA11], [SLE11], [GLO09], and [PRI08] mention running WS on sensors instead of a central computer. Due to the limited capability of the sensor and the size of XML messages, it is not convenient to run WS on a sensor. In the proposed system, the web service runs only in the central computer. Furthermore, the authors in [XU10], [AIE06] talked about using a separate controller to communicate with sensors. WS are used to communicate with the controller rather than sensors. In our approach, there is no middleware or additional controller.

Chapter 3

Web Services for Home Automation

3.1 Introduction

This chapter describes the basic characteristics of the proposed system. The overview of the proposed approach is given in Section 3.2. Section 3.3 contains the architecture or system model. Section 3.4 presents the operations provided by the system. Security is mentioned in Section 3.5. Finally, Section 3.6 contains the summary of the chapter.

3.2 Overview of the proposed approach

This thesis proposes an approach that uses Web services to remotely and efficiently interact with smart home devices to manage energy consumption in a smart grid environment. A smart home with a wireless network based on ZigBee and XMPP is simulated. The smart home contains different kinds of elements like smart appliances, HVAC, sensors and thermostat. BPEL is used to implement the secure Web service on the central computer. The central computer or gateway is able to communicate with all these elements directly or via a thermostat. The system provides operations to read the temperature of a room, light intensity of a room, energy consumption of a room as well as energy consumption of an entire home. Furthermore, there are functions to adjust light intensity, control temperature and control smart appliances.

3.3 Architecture

In this section, the model of the system is described. The components of the smart home and the network connecting the elements are explained. Furthermore, the Web service and the way a user (e.g. residential consumer) interacts with the home are discussed.

3.3.1 Smart Home

A typical single-family home with one floor is simulated [HEN11]. The smart home simulator is further developed from [HEN11]. The smart home contains a washer, dryer, dishwasher, coffeemaker, HVAC, light sensors, temperature sensors, a smart (central) thermostat and central computer. The thermostat is directly attached to the computer. HVAC operates for each room separately. In each room of the smart home, there are two sensors. If one sensor fails, the reading of the other sensor is used. If both sensors are active, then the average of both readings is computed. The sensors measure temperature and light intensity of each room every twelve seconds. Sensors send data to thermostat periodically. A floor plan is shown in Figure 3.1.

Figure 3.1: Floor plan of the smart home [HEN11]

3.3.2 Network

Sensors send data to the thermostat via ZigBee. The central computer is able to communicate with all these elements directly or via the thermostat. XMPP-protocol is used for the communication between the central computer and the home appliances. It is assumed that the home appliances are connected to the Internet as XMPP requires TCP for connection. The advantage of using ZigBee is that it is low in cost, supports device interoperability and provides security. The other features of ZigBee include reliability, easy deployment and support for a large number of nodes. The advantage of XMPP is that it provides near real-time messaging service between the computer and the appliances. It also provides authentication and data encryption.

3.3.3 Web Service

BPEL is used to implement the secure Web service on central computer. The user (e.g. residential consumer) can access the Web service over the Internet. The user must pass security credentials in order to invoke the Web service. The user is able to access or control all home elements via the secure Web service. Figure 3.2 illustrates this communication and high level model of the system.

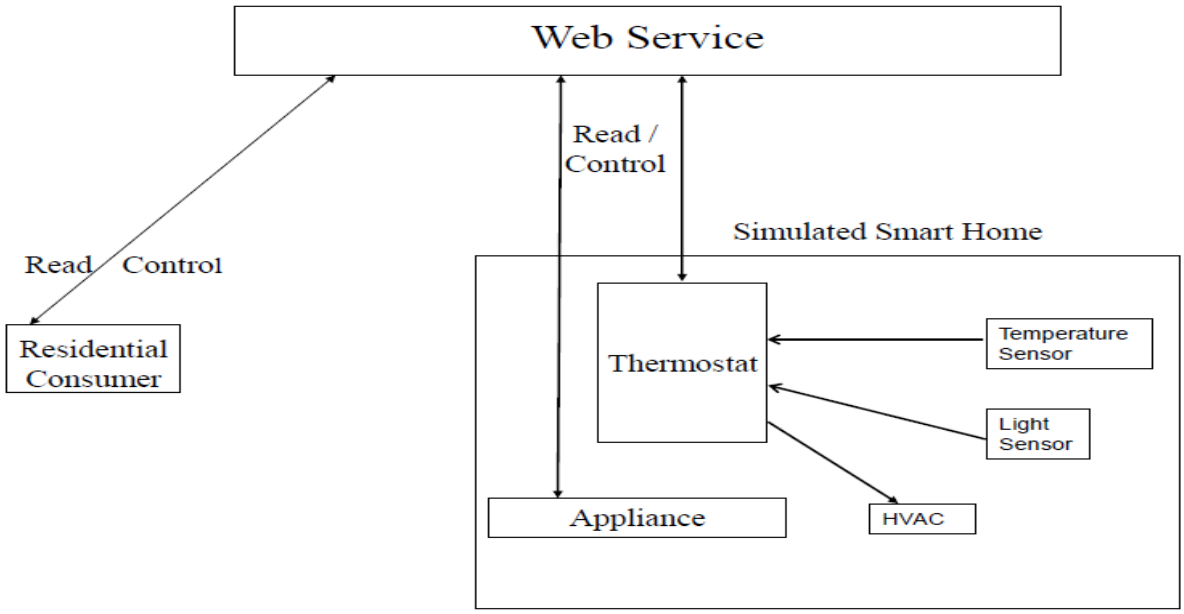

Figure 3.2: System Model

3.4 Operations

The user (e.g. residential consumer) is able to use the Web service to read the temperature of a room, light intensity of a room, energy consumption of a room as well as energy consumption of the entire home. Furthermore, the user is able to adjust light intensity, control temperature and control smart appliances. In this section, each of the above mentioned operations are described in detail.

To simplify the interaction between a user and Web services, the central computer provides a web interface. If the user does not have any client to invoke the Web service directly using SOAP, then the user can access the web based interface over the Internet and enter required information to access or control the smart home elements. The interface is shown in Figure 3.3.

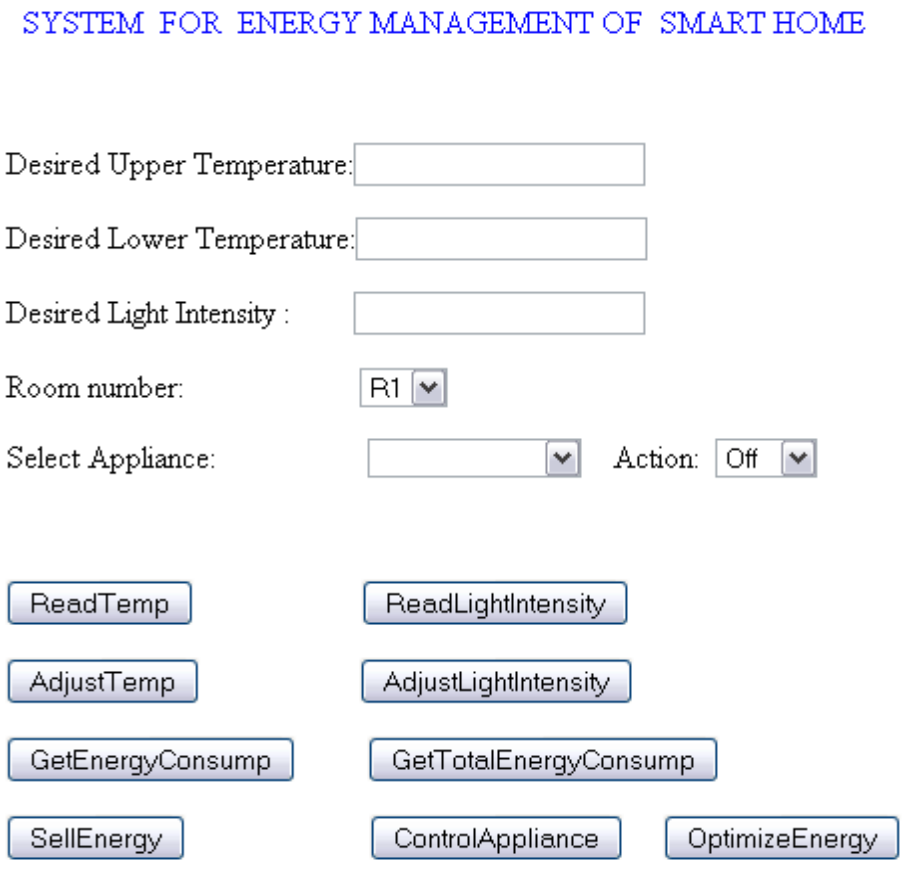

Figure 3.3: Web-based Graphical User Interface

3.4.1 Function to control temperature of a room

To control the temperature of a room, the user specifies the desired temperature range by entering the desired upper temperature and lower temperature. Furthermore, the user specifies the room number and selects the operation “adjustTemp”. The Web service is invoked with these parameters and it calls the “controlTemp” function of the simulated smart thermostat. Sensors in every room send the temperature data to the thermostat periodically. Therefore, a smart thermostat knows the temperature of every room. The smart thermostat uses the current temperature of the selected room, along with the desired temperature range, to find out whether the current temperature is within the desired range. If the current temperature is out of range, then the thermostat controls the HVAC in the room to change the temperature of that room to the desired value. Finally, the user is informed about the new temperature. The following sequence diagram shows the exchange of messages that takes place among the user, the Web service and the thermostat. These communications take place during the operation to control the temperature.

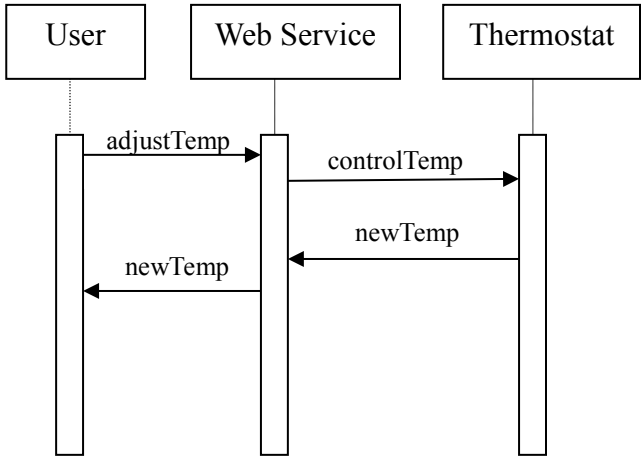

Figure 3.4: Sequence diagram of the control temperature process

control temperature

// following parameters are initialized by user
upperTemp, lowerTemp, room, Func
String oper = upperTemp + lowerTemp + room + Func

// connecting to web (bpel) server
jndi.put (Context.PROVIDER_URL, "opmn:ormi://../orabpel");

// putting parameters in string xml.
String xml="..<para1>" + oper + "</para1>.."

//invoking method named "process" of web service. *nm* contains *xml*
deliveryService.request("smartHome","process", nm);

//web service uses WSIF to call simulator's function
ctrl (room, upperTemp, lowerTemp);

Print "the new temperature of the room"

Figure 3.5: Java-code snippet of the operation to control temperature.

The following activity diagram is used to illustrate the function to control temperature. It shows the order of actions that are taken. It also shows alternate actions and the decisions upon which some actions are based upon.

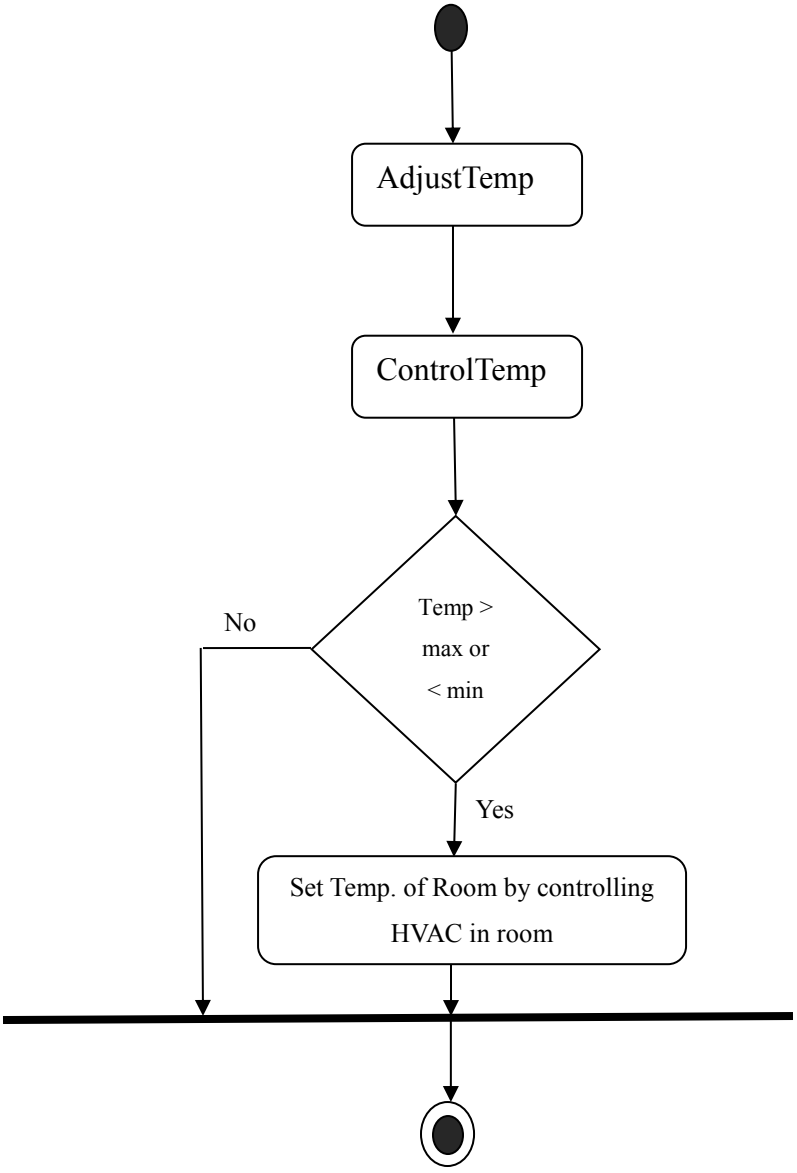

Figure 3.6: Activity diagram of the control temperature process

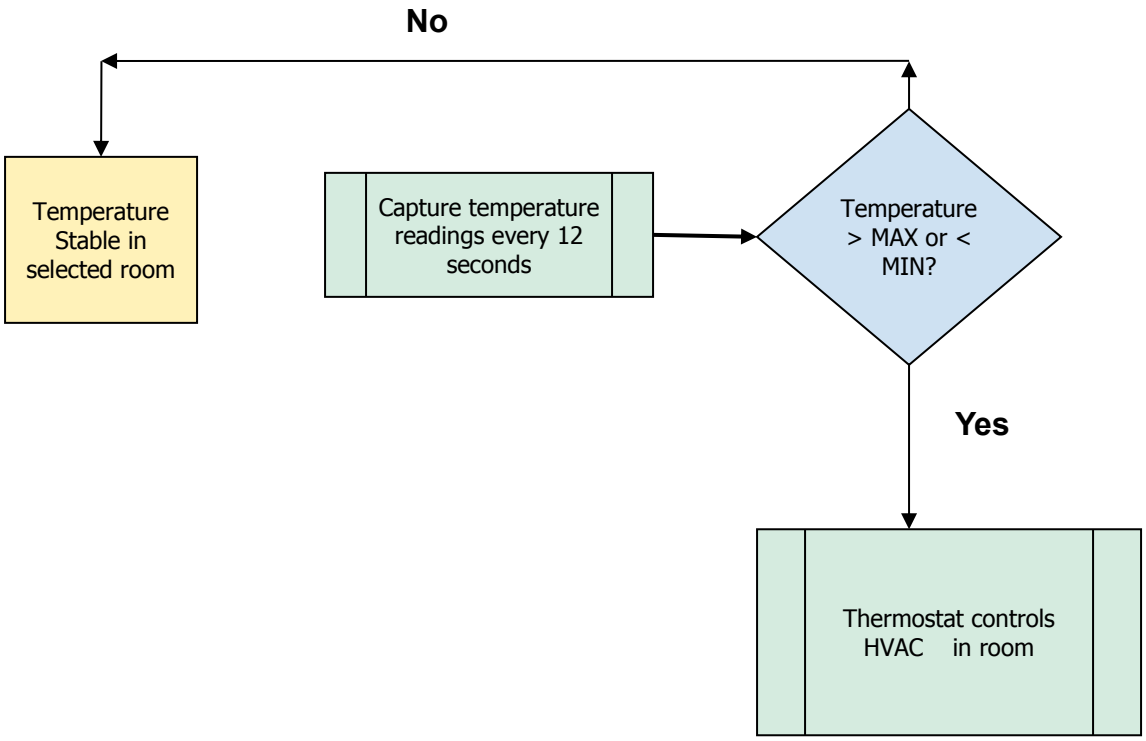

Figure 3.7: Procedures taken by thermostat to control temperature

The thermostat controls the temperature of a room on request and also whenever the temperature goes out of desired range. Figure 3.7 shows the procedure taken by the thermostat to control the temperature of a room. The thermostat receives the current temperature of every room from corresponding sensors every twelve seconds. If the current temperature of a room is more than one degree Celsius away from the user-inputted or pre-defined range of temperatures, then the thermostat controls the HVAC in the room to change the temperature of that room to the desired value. The thermostat carries out these tasks periodically to keep the temperature of the room within the desired range. This is done for every room.

3.4.2 Function to control light intensity of a room

To control light intensity of a room, the user specifies the desired light intensity. Furthermore, the user specifies the room number and selects the operation "adjustLightIntensity". The Web service

is invoked with these parameters and it calls the “controlLightIntensity” function of the simulated smart thermostat. Sensors in every room send information about light intensity to the thermostat periodically. Therefore, the smart thermostat knows the light intensity of every room. If the current light intensity of the selected room is above or below the desired value, then the thermostat activates the window shades or adjusts the light intensity of the light source in order to change the light intensity of the room to the desired value. Finally, the user is informed about the new light intensity. The following sequence diagram shows the exchange of messages that takes place among the user, the Web service and the thermostat. These communications take place during the operation to control light intensity.

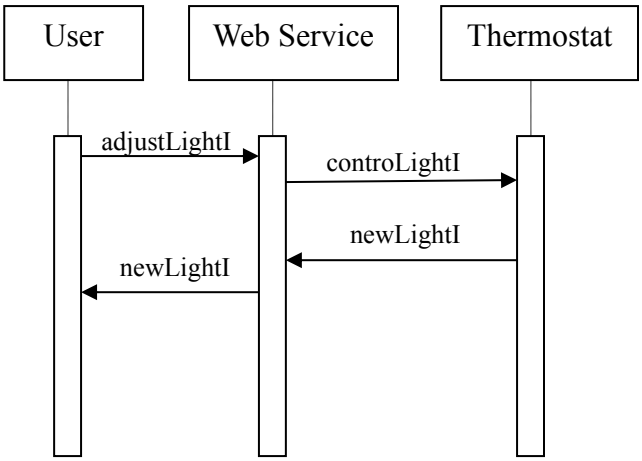

Figure 3.8: Sequence diagram of the control light intensity process

control light intensity

// following parameters are initialized by user
desLightI, room, Func
String oper = desLightI + room + Func

// connecting to web (bpel) server
jndi.put (Context.PROVIDER_URL, "opmn:ormi://../orabpel");

// putting parameters in string xml.
String xml="..<para1>" + oper + "</para1>.."

//invoking method named "process" of web service. *nm* contains *xml*
deliveryService.request("smartHome","process", nm);

//web service uses WSIF to call simulator's function
ctrlLightI (room, desLightI);

Print "the new light intensity of the room"

Figure 3.9: Java-code snippet of the operation to control light intensity.

The following activity diagram is used to illustrate the function to control light intensity. It shows the order of actions that are taken. It also shows alternate actions and the decisions upon which some actions are based upon.

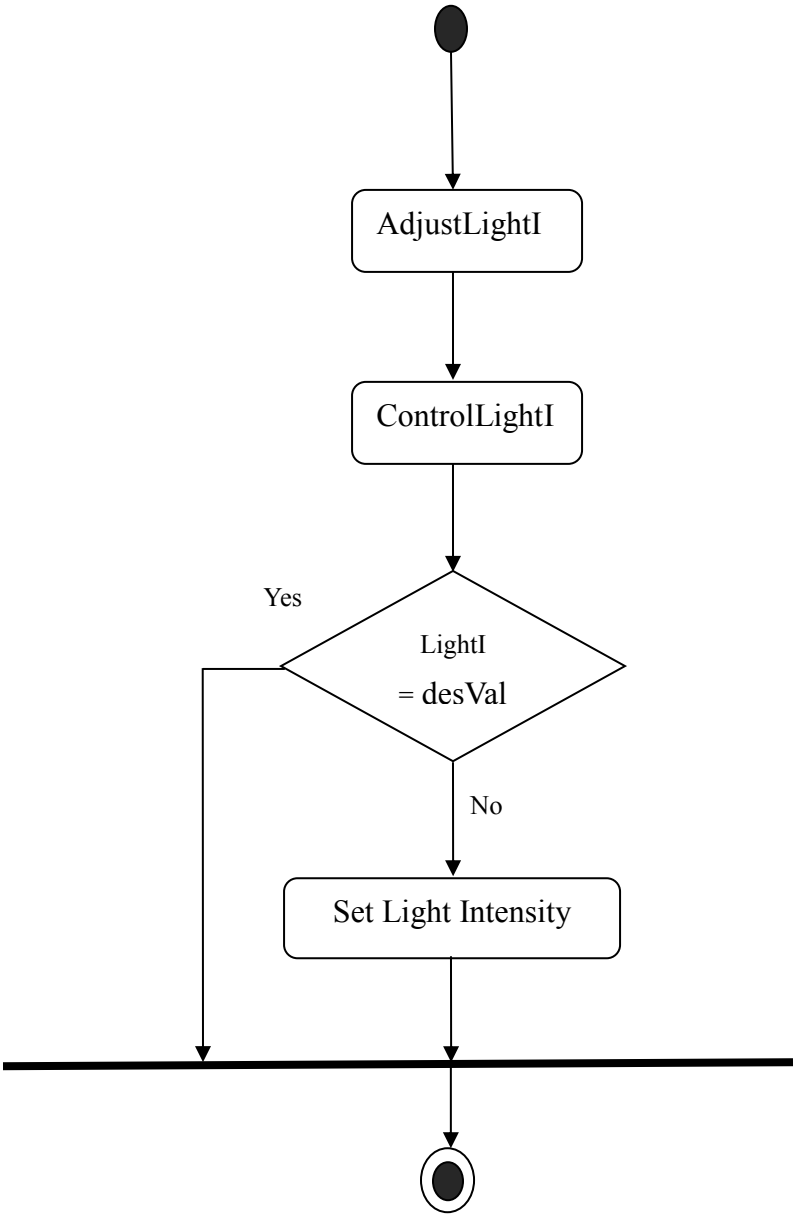

Figure 3.10: Activity diagram of the control light intensity process

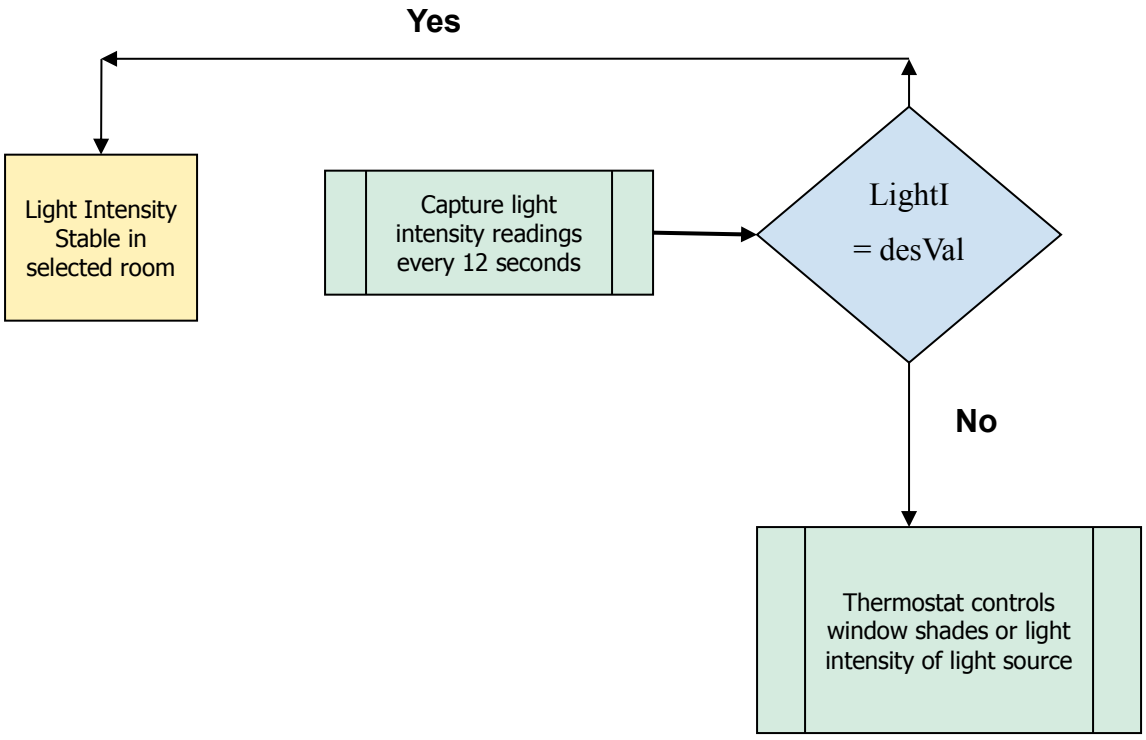

Figure 3.11: Procedures taken by thermostat to control light intensity

The smart thermostat regulates light intensity of a room on request and also whenever light intensity is above or below the desired value. Figure 3.11 shows this procedure. Sensors in every room send the current light intensity to the thermostat every 12 seconds. If the current light intensity of a room is above or below the desired value, then the thermostat activates the window shades or adjusts the light intensity of the light source in order to change the light intensity of the room to the desired value. The thermostat carries out these tasks periodically to keep the light intensity of the room within the desired range.

3.4.3 Function to control appliance

A user can turn the appliance on or shut it off. The user can also put the appliance in energy saving (low) mode. A user specifies the appliance (e.g., washer, dryer) it wants to control, the action (e.g., on/off/low), the room number and the operation "ctrlAppl". The Web service is invoked and the information is passed to it. The Web service sends user-specified action to the

user-selected appliance in the room via XMPP. Finally, the user is informed whether the action is successful or not. The following sequence diagram shows the exchange of messages that takes place among the user, the Web service and the smart appliance. These communications take place during the operation to control appliances.

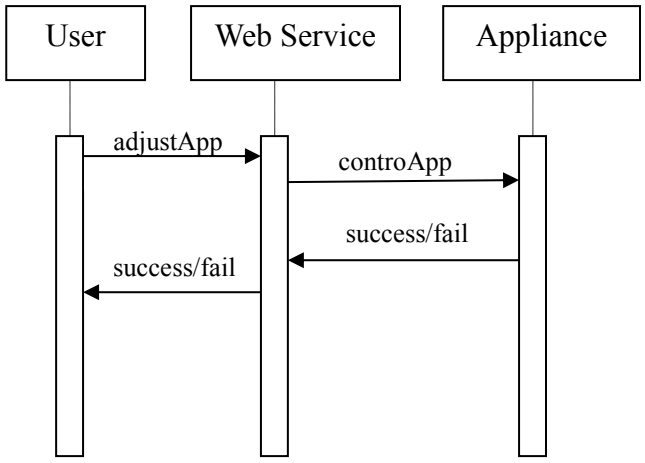

Figure 3.12: Sequence diagram of the control appliance process

```
control appliance
// following parameters are initialized by user
appName, action,  room, Func
String oper = appName + action + room + Func

// connecting to web (bpel) server
jndi.put (Context.PROVIDER_URL, "opmn:ormi://../orabpel");

// putting parameters in string xml.
  String xml="..<para1>" + oper + "</para1>.."

//invoking method named "process" of web service. nm contains xml
  deliveryService.request("smartHome","process", nm);

//web service uses WSIF to call simulator's function
   ctrlApp (appName, room, action);

Print "operation is successful"
```

Figure 3.13: Java-code snippet of the operation to control appliance.

The following activity diagram is used to illustrate the function to control appliances. It shows the order of actions that are taken. It also shows alternate actions and the decisions upon which some actions are based upon.

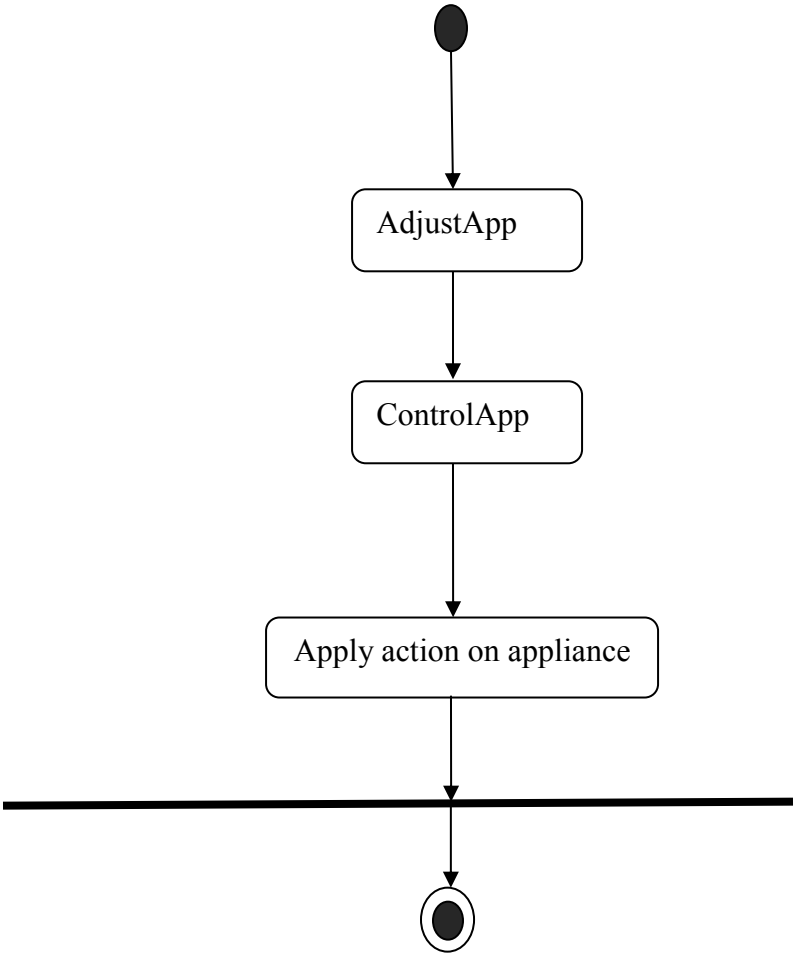

Figure 3.14: Activity diagram of the control appliance process

3.4.4 Function to read temperature

If a user wants to read the temperature of a room, he specifies the room number and the operation "readTemp". The Web service is invoked and it calls the "getTemperature" function of the smart thermostat. Sensors in every room send the current temperature to the thermostat periodically. The thermostat replies back the "current temperature of the room". Finally, the user is informed

about the current temperature. The following sequence diagram shows the exchange of messages that takes place among the user, the Web service and the smart thermostat.

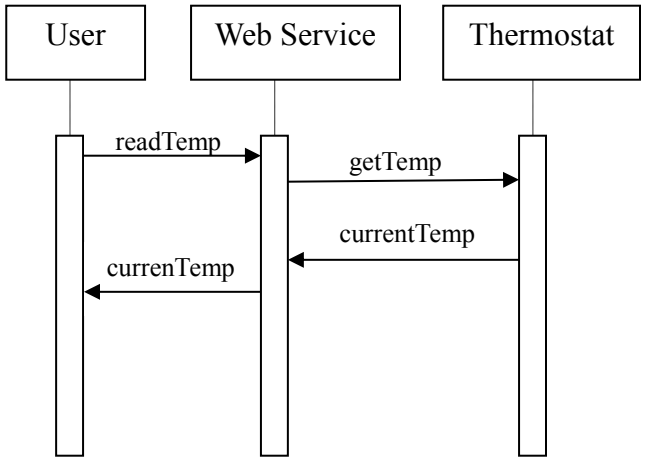

Figure 3.15: Sequence diagram of the read temperature process

read temperature

// following parameters are initialized by user
room, Func
String oper = room + Func

// connecting to web (bpel) server
jndi.put (Context.PROVIDER_URL, "opmn:ormi://../orabpel");

// putting parameters in string xml.
String xml="..<para1>" + oper + "</para1>.."

//invoking method named "process" of web service. *nm* contains *xml*
deliveryService.request("smartHome","process", nm);

//web service uses WSIF to call thermostat's function
readTemp (room);

Print "current temperature of room"

Figure 3.16: Java-code snippet of the operation to read temperature.

The following activity diagram is used to illustrate the function to read temperature. It shows the order of actions that are taken.

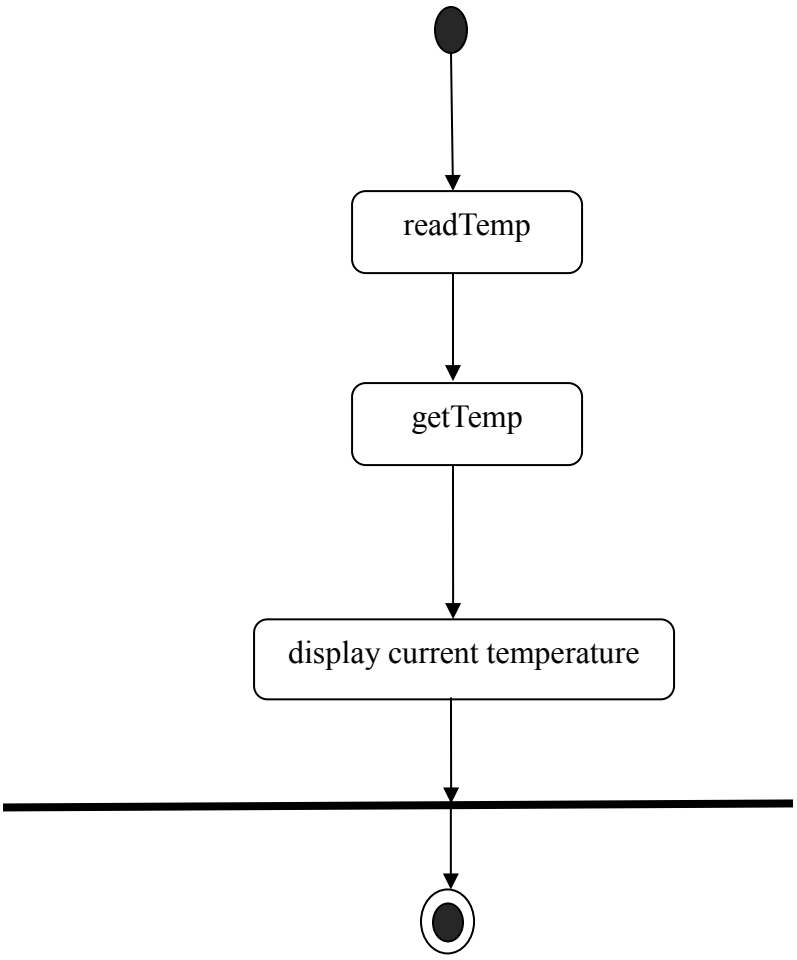

Figure 3.17: Activity diagram of the read temperature process

3.4.5 Function to read light intensity

If a user wants to read light intensity of a room, he specifies the room number and the operation "readLightIntensity". The Web service is invoked and it calls the "getLightIntensity" function of the smart thermostat. Sensors in every room send current light intensity to the thermostat periodically. The thermostat replies back the "current light intensity of the room". Finally, the user is informed about the current light intensity. The following sequence diagram shows the exchange of messages that takes place among the user, the Web service and the smart thermostat.

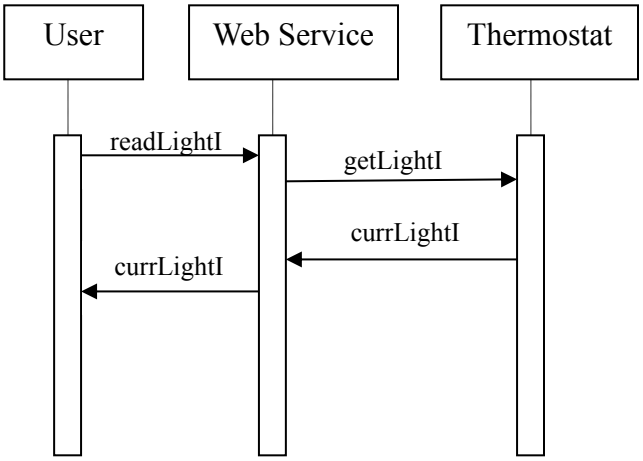

Figure 3.18: Sequence diagram of the read light intensity process

read light intensity

// following parameters are initialized by user
room, Func
String oper = room + Func

// connecting to web (bpel) server
jndi.put (Context.PROVIDER_URL, "opmn:ormi://../orabpel");

// putting parameters in string xml.
String xml="..<para1>" + oper + "</para1>.."

//invoking method named "process" of web service. *nm* contains *xml*
deliveryService.request("smartHome","process", nm);

//web service uses WSIF to call thermostat's function
readLightIntensity (room);

Print "current light intensity of room"

Figure 3.19: Java-code snippet of the operation to read light intensity.

The following activity diagram is used to illustrate the function to read temperature. It shows the order of actions that are taken.

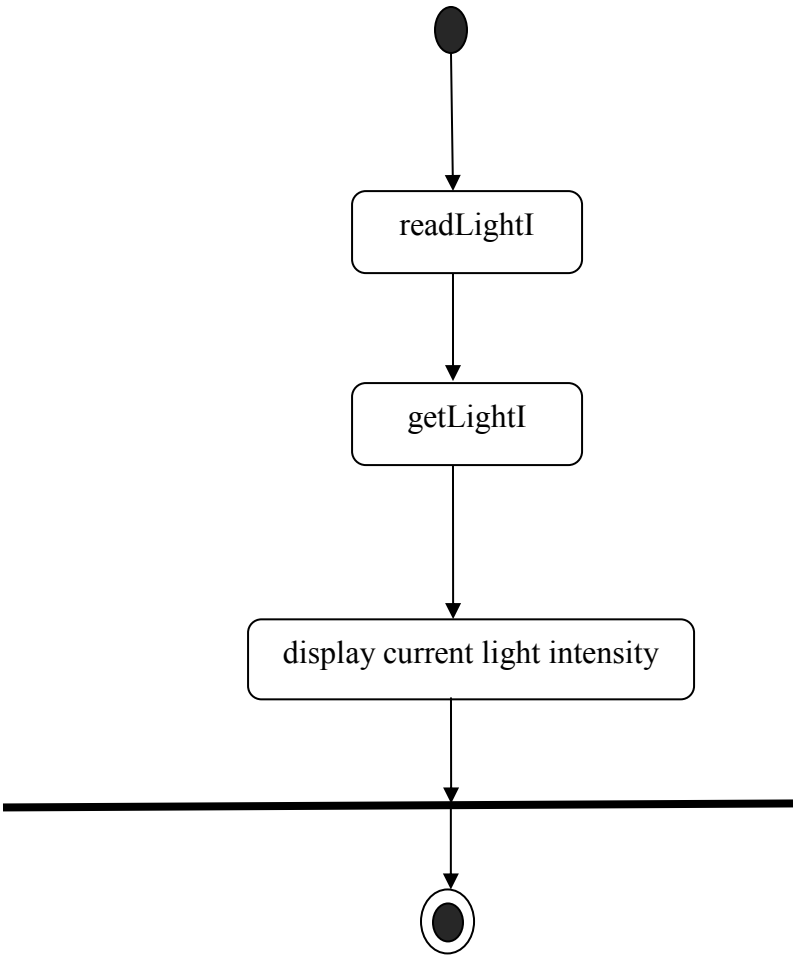

Figure 3.20: Activity diagram of the read light intensity process

3.4.6 Function to read energy consumption of room

If a user wants to read energy consumption of a room, he specifies the room number and the operation "getEnergyConsump". The Web service is invoked and it calls the "getEnergy" function of the smart thermostat. Sensors in every room send various data (such as temperature, light intensity) to the thermostat periodically. The thermostat uses these data to calculate the energy consumption of the room. The thermostat sends the "energy consumption of the room" to the Web service. Furthermore, the Web service communicates with all the appliances in the room

via XMPP in order to obtain their energy consumption. The Web service sends the messages to all appliances in the room asking for their energy consumption. Every appliance records its energy consumption. Then each appliance in the room sends its energy consumption to the Web service, which calculates the total energy consumption of the room by using energy consumption data from the thermostat and the appliances in the room. Finally, the user is informed about the energy consumption of the room. The following sequence diagram shows the exchange of messages that takes place among the user, the Web service, the smart thermostat and the appliances.

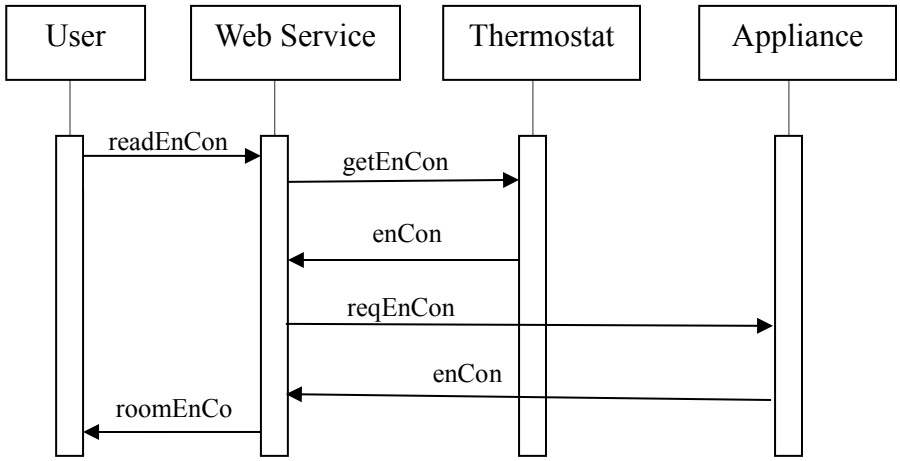

Figure 3.21: Sequence diagram of the "read room energy consumption" process

```
read room energy consumption

// following parameters are initialized by user
room, Func
String oper =  room + Func

// connecting to web (bpel) server
jndi.put (Context.PROVIDER_URL, "opmn:ormi://../orabpel");

// putting parameters in string xml.
  String xml="..<para1>" + oper + "</para1>.."

//invoking method named "process" of web service. nm contains xml
  deliveryService.request("smartHome","process", nm);

//web service uses WSIF to call thermostat's function
   enTh = getEnConsump (room);

//web service uses WSIF to call all room appliance's function
   enApp = reqEnConsump ();

totalRoomConsump = enTh +  enApp

Print "total energy consumption of room"
```

Figure 3.22: Java-code snippet of the operation to read room energy consumption.

The following activity diagram is used to illustrate the function to read room energy consumption. It shows the order of actions that are taken.

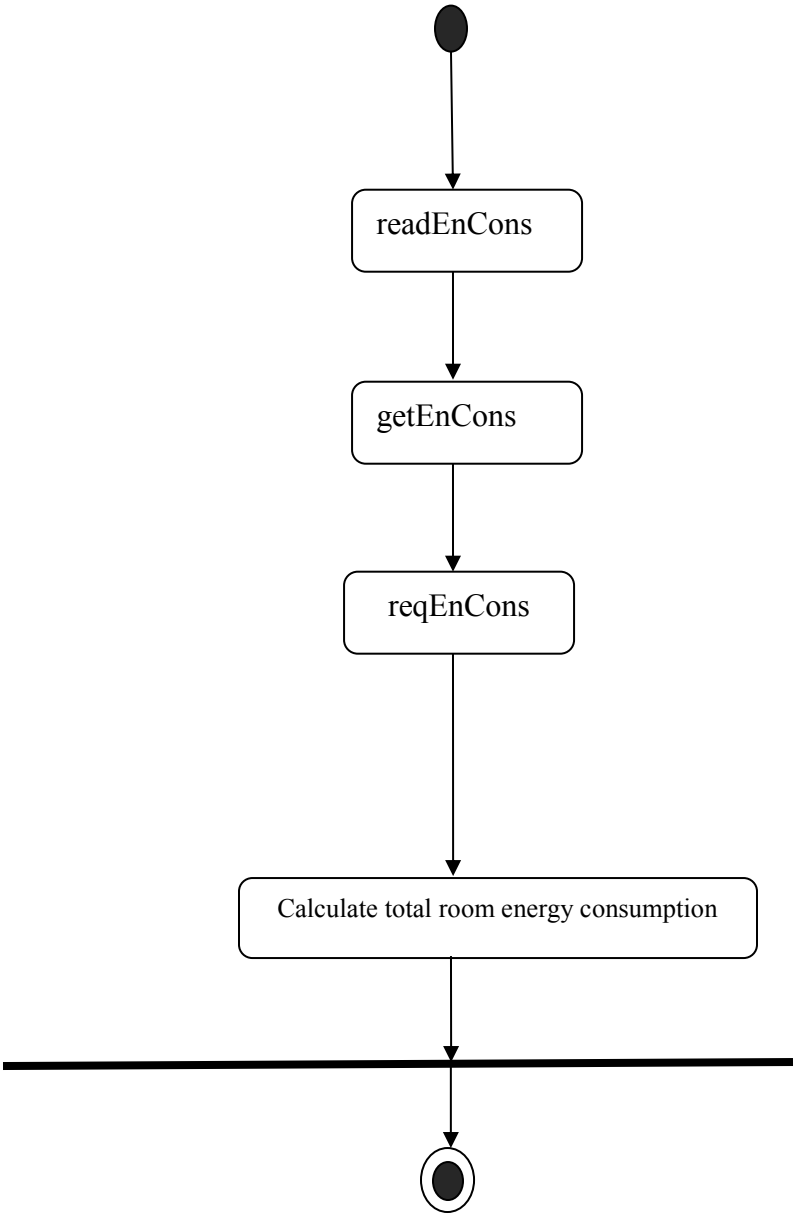

Figure 3.23: Activity diagram of the "read room energy consumption" process

3.4.7 Function to read energy consumption of home

If a user wants to read total energy consumption of a home, he specifies the room number and the operation "getTotalEnergyConsump". The Web service is invoked and it calls the "getTotalEnergy" function of the smart thermostat. Sensors in every room send various data (such

as temperature, light intensity) to the thermostat periodically. The thermostat uses these data to calculate the energy consumption of the home. The thermostat sends the "energy consumption of the home" to the Web service. Furthermore, the Web service communicates with all the appliances in the home via XMPP in order to obtain their energy consumption. The Web service sends the messages to all appliances in the home asking for their energy consumption. Every appliance records its energy consumption. Then each appliance of the home sends its energy consumption to the Web service. The Web service calculates the total energy consumption of the home by using energy consumption data from the thermostat and the appliances in the home. Finally, the user is informed about the total energy consumption of the home. The following sequence diagram shows the exchange of messages that takes place among the user, the Web service, the smart thermostat and the appliances.

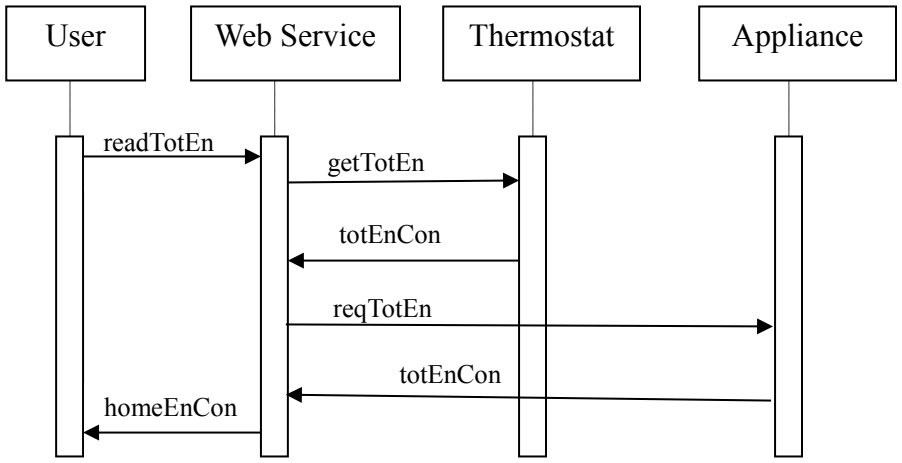

Figure 3.24: Sequence diagram of the "read home energy consumption" process

read home energy consumption

// following parameters are initialized by user
String oper = Func

// connecting to web (bpel) server
jndi.put (Context.PROVIDER_URL, "opmn:ormi://../orabpel");

// putting parameters in string xml.
String xml="..<para1>" + oper + "</para1>.."

//invoking method named "process" of web service. *nm* contains *xml*
deliveryService.request("smartHome","process", nm);

//web service uses WSIF to call thermostat's function
enTh = getTotEnConsump ();

//web service uses WSIF to call all appliance's function
enApp = reqTotEnConsump ();

totalHomeConsump = enTh + enApp

Print "total energy consumption of home"

Figure 3.25: Java-code snippet of the operation to read home energy consumption.

The following activity diagram is used to illustrate the function to read energy consumption of the home. It shows the order of actions that are taken.

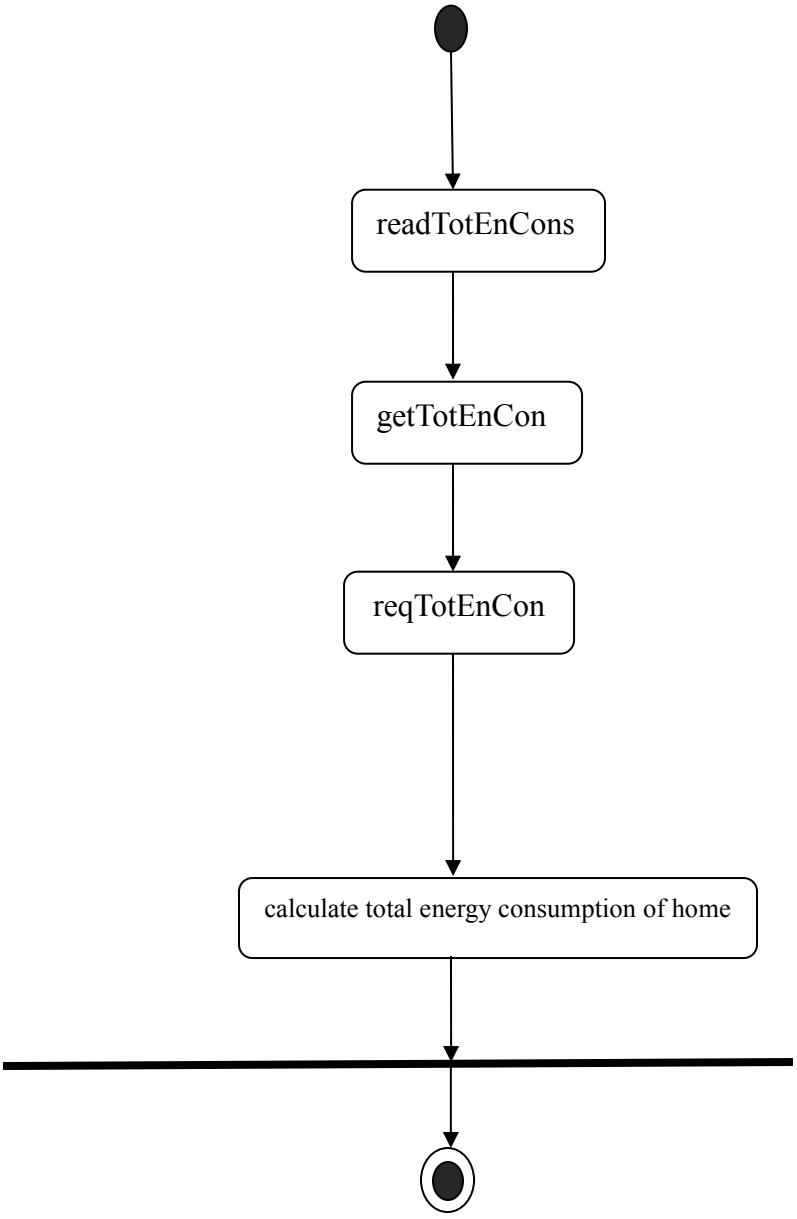

Figure 3.26: Activity diagram of the "read home energy consumption" process

3.5 Security

The system has three stages of security. There is security when the user logs into the system, invokes the WS and interacts with home devices. The user must be authenticated to log into the system. The BPEL process can be invoked by HTTP, SOAP and Java API (Application Programming Interface). Each of these techniques provides a mechanism to pass security credentials. ZigBee is used for communication between the thermostat and sensors. ZigBee has a mechanism to check the data integrity and sender authentication. More details about security are to be provided in Chapter 5.

3.6 Summary

This chapter describes the basic characteristics of the proposed system in detail. The architecture of the system is presented, and the components of the smart home and the network connecting the elements are explained. Furthermore, the Web service and the way the user can interact with the home are discussed. Then, the operations of the system are presented. Moreover, the interaction diagrams are used to illustrate each operation. Code snippets of each operation are also provided. Finally, the security of the system is mentioned. More details about the security, however, are to be provided in Chapter 5.

Chapter 4

Energy Optimization in a Smart Grid Environment

4.1 Introduction

This chapter describes the advanced characteristics of the proposed system. Furthermore, this chapter shows how the system interacts with the smart grid. The smart grid environment is presented in Section 4.2. Section 4.3 describes how the system facilitates demand response. The operation to sell energy is given in Section 4.4. Section 4.5 describes the dynamic programming scheme to minimize the use of energy. Finally, a summary of the chapter is provided in Section 4.6.

4.2 Smart Grid Environment

The user can be a utility provider, grid operator or residential consumer in a smart grid environment. A particular smart home has a central computer running Web service and communicating with all the devices of the home. In the smart grid context, there are several homes. A utility provider can interact with any of the smart homes via associated WS. A smart home can also have an on-site energy generator like residential solar panels or a wind turbine. The renewable energy obtained from it can be stored in batteries or PHEV. Furthermore, the Web service can also control elements of neighbouring homes. These are illustrated in Figure 4.1.

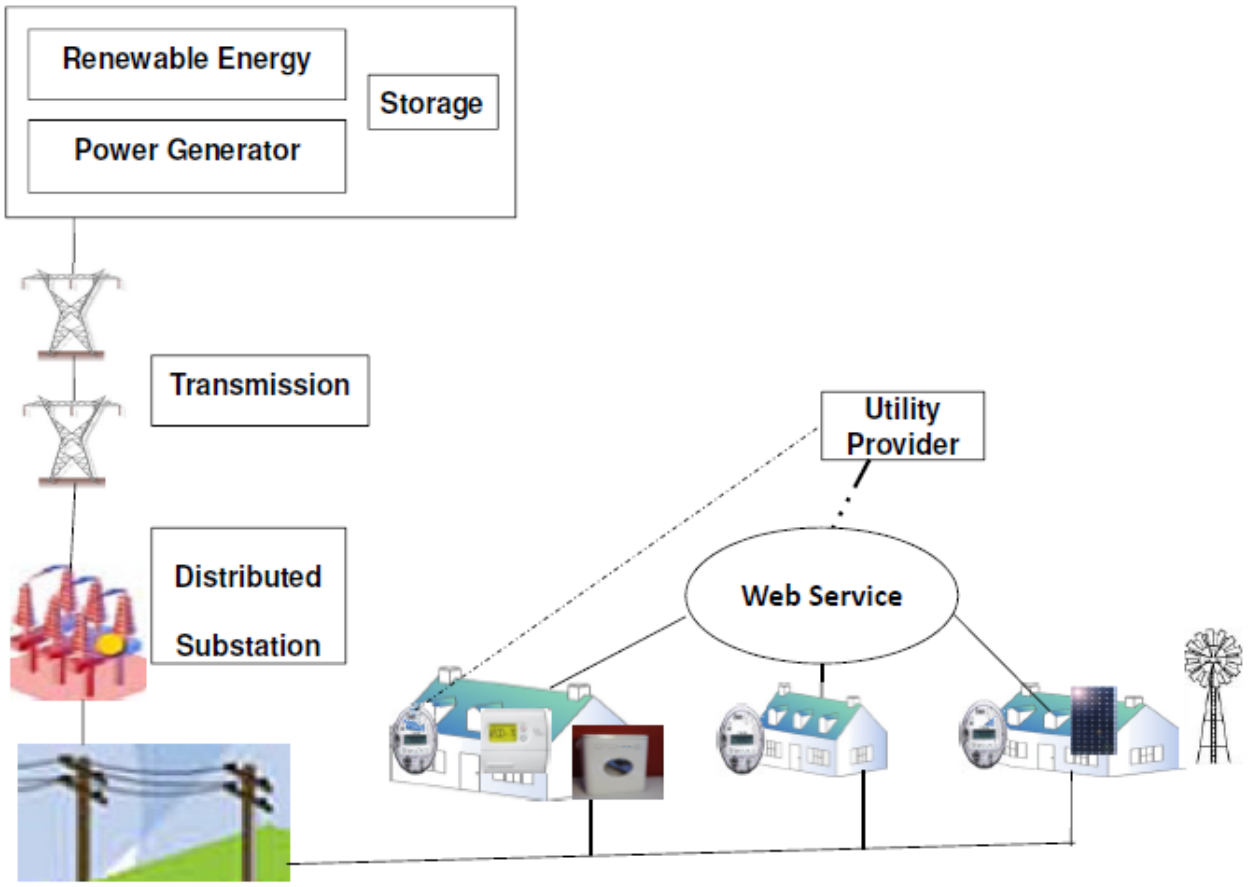

Figure 4.1: Accessing smart home elements via Web services in smart grid environment

4.3 Demand Response

Demand Side Management (DSM) is the process of managing the consumption of energy, generally to optimize available and planned generation resources. Demand response is a subset of DSM. Demand Response (DR) is a term used for programs designed to encourage end-users to make short-term reductions in energy demand in response to a price signal from the electricity hourly market, or a trigger initiated by the electrical grid operator. Price signals are prices that are set for a specific time period in the future. According to TOU, consumers are charged more during peak hours. Typically, DR actions would be in the range of 1 to 4 hours and include turning off or dimming banks of lighting, adjusting HVAC levels, or shutting down appliances

during peak hours to decrease their energy bills or reduce load on the grid. In addition, utility providers can directly control certain devices (e.g., water heaters, air conditioners, electric heating, pool pumps, etc.) during times of critical demand in exchange for some sort of monitory rate reduction or rebate. This more traditional method is called "Demand Control".

The proposed system facilitates demand response. If there is an increase in price during peak hours, a user is able to access the system via WS to reduce temperature or turn off appliances in the home. These actions are performed by calling the operation to control room temperature or an appliance. Thus, energy cost and load on the grid is reduced. The proposed system can also facilitate demand control. The user can allow the utility provider to directly control certain appliances to reduce the load on the grid.

An algorithm is presented to deal with demand response and to reduce the energy cost of the house. When there is an increase in price, the residential consumer needs to decrease the energy consumption. Therefore, the user can set a value (E_{max}) for the maximum energy consumption of the house during that period. This value can be estimated by observing past energy consumption and energy bills or consulting with the utility provider. The algorithm works in the following way. When there is an increase in price during peak hours, it calculates the total energy consumption of the house. If the total energy consumption is greater than E_{max} , then the algorithm tries to turn off (or put into low energy mode) the device that has the highest energy consumption. If a device can be turned off immediately (e.g. air conditioner) then its minimum energy value (min_i) is 0. If a device cannot be turned off immediately but has a lower power mode (e.g. washing machine) then its minimum energy value (min_i) is set to its energy consumption in lowest power mode. If the total energy consumption is still greater than E_{max} , then the algorithm tries to turn off (or put into low energy mode) another device that now has the highest energy consumption. The devices are numbered from 1 to n (assuming there are n devices). The pseudo-code of the algorithm is given below:

```
{E_i  = energy usage of device i}
{To = total energy usage of home }
{t = current time}
{p = peak hours}
{E_max = maximum total allowable energy consumption of a home during peak hours }
{index = id of device that consumes maximum energy}
{min_i = minimum energy value of device i} (either 0 (if switched off) or energy in low mode)
{max = energy usage of a device that has the highest consumption}
{n = total number of electrical devices in the home}

if   t ∈ p
  To  ← 0

  for i = 1 to n
    To  ← To + E_i

  while   To > E_max
     max = E_1
    index = 1

    for i = 2 to n
       if   E_i > max
          max = E_i
         index = i

     E_index = min_index


     To  ← 0

    for i = 1 to n
      To  ← To + E_i
```

Figure 4.2: Algorithm for demand response

The following activity diagram is used to illustrate the algorithm for demand response. It shows the order of actions that are taken. It also shows alternate actions and the decisions upon which some actions are based.

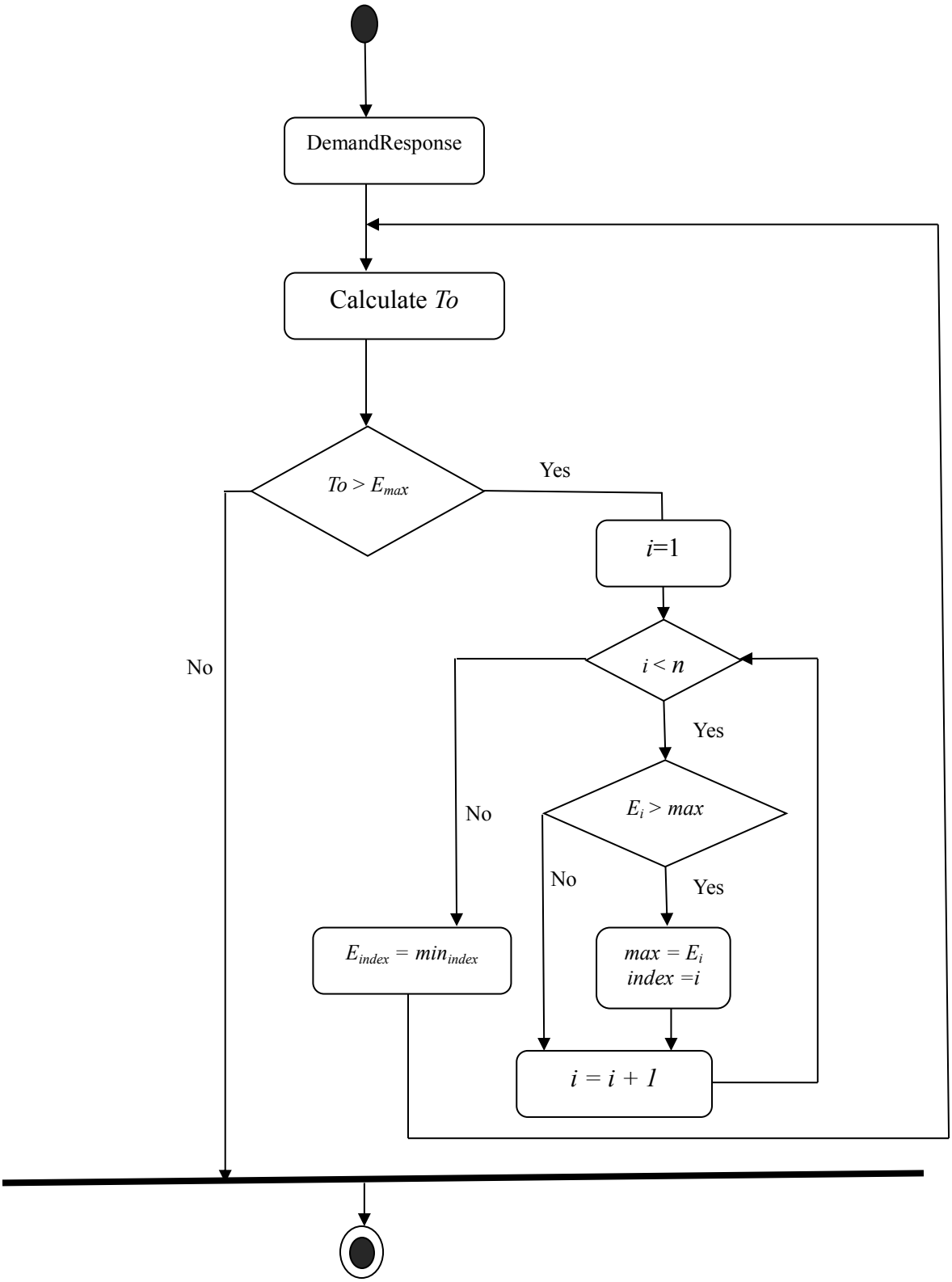

Figure 4.3: Activity diagram of "demand response" algorithm

4.4 Selling Energy back to the grid

If a house can store energy, then the energy from the storage device can be used to reduce energy cost or sold back to the grid. An algorithm is presented that sells the stored energy back to the grid and reduces the energy cost of the house. The algorithm works in the following way. Energy is obtained from the grid during off-peak hours. Energy is obtained from the local storage device when it is peak-hours. Average energy needed by the house during peak hours can be estimated by observing past energy consumption. If request to sell energy is made during off-peak hours, then the user can sell energy provided that the amount of stored energy is greater than the energy needed by the house during peak-hours. If the request to sell energy is made during peak hours, then the user can sell energy provided that the amount of stored energy is at least double than the energy needed by the house during peak-hours. This ensures that there is enough stored energy to supply the home during current and subsequent peak-hours. Thus, the algorithm ensures that the energy cost is reduced by using stored energy during peak hours and also profit is made by selling energy. The pseudo-code of the algorithm is given below:

Algorithm to sell energy

{enH = amount of energy needed by home during peak-hours*}*
{enStore = amount of energy stored in storage }
{t = current time*}*
{p = peak hours*}*

if $t \in p$
  excessEn ← *enStore – 2*enH*
else
  excessEn ← *enStore – enH*

if *excessEn > 0*
 enStore ← *enStore – excessEn*
 Print *"excessEn* amount of energy is sold*"*
else
 Print "Not enough energy to sell"

Figure 4.4: Algorithm to sell energy

The system provides a function to sell stored energy back to the grid and reduce the energy cost. This function uses the above algorithm to sell energy. It is assumed that the simulated house has solar panels to generate renewable energy and store it in a storage device like batteries or PHEV. It is also assumed that an electrical sensor is mounted on the storage device. The electrical sensor measures the amount of energy stored and periodically sends this information to the thermostat. If a user wants to sell energy, he specifies the operation "sellEnergy". The Web service is invoked and it calls the "getStorEn" function of the smart thermostat. The thermostat replies back the "amount of stored energy". The Web service estimates the peak hour by considering the current time and the current season. The peak-hours during the summer are from 11 a.m. to 5 p.m. In the winter, the peak hours are from 7 a.m. to 11 a.m. and from 5 p.m. to 7 p.m. The Web service calculates the excess energy by applying the algorithm to sell energy and using information from the thermostat. Finally, it sells energy or informs the user that there is insufficient energy to sell. The following sequence diagram shows the exchange of messages that takes place among the user, the Web service and the smart thermostat.

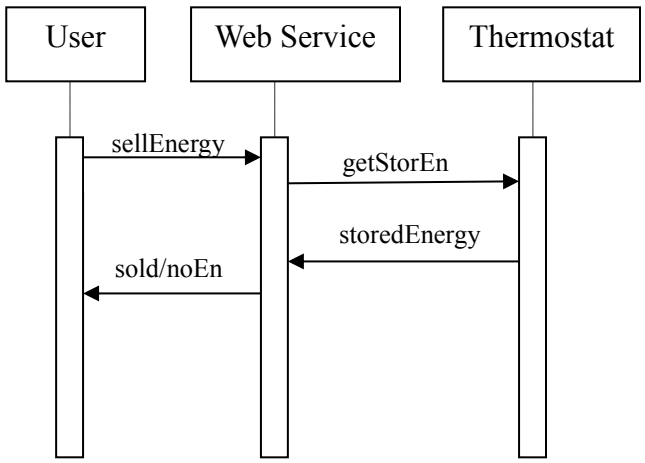

Figure 4.5: Sequence diagram of the "sell energy" process

The following activity diagram is used to illustrate the function to sell energy. It shows the order of actions that are taken. It also shows alternate actions and the decisions upon which some actions are based.

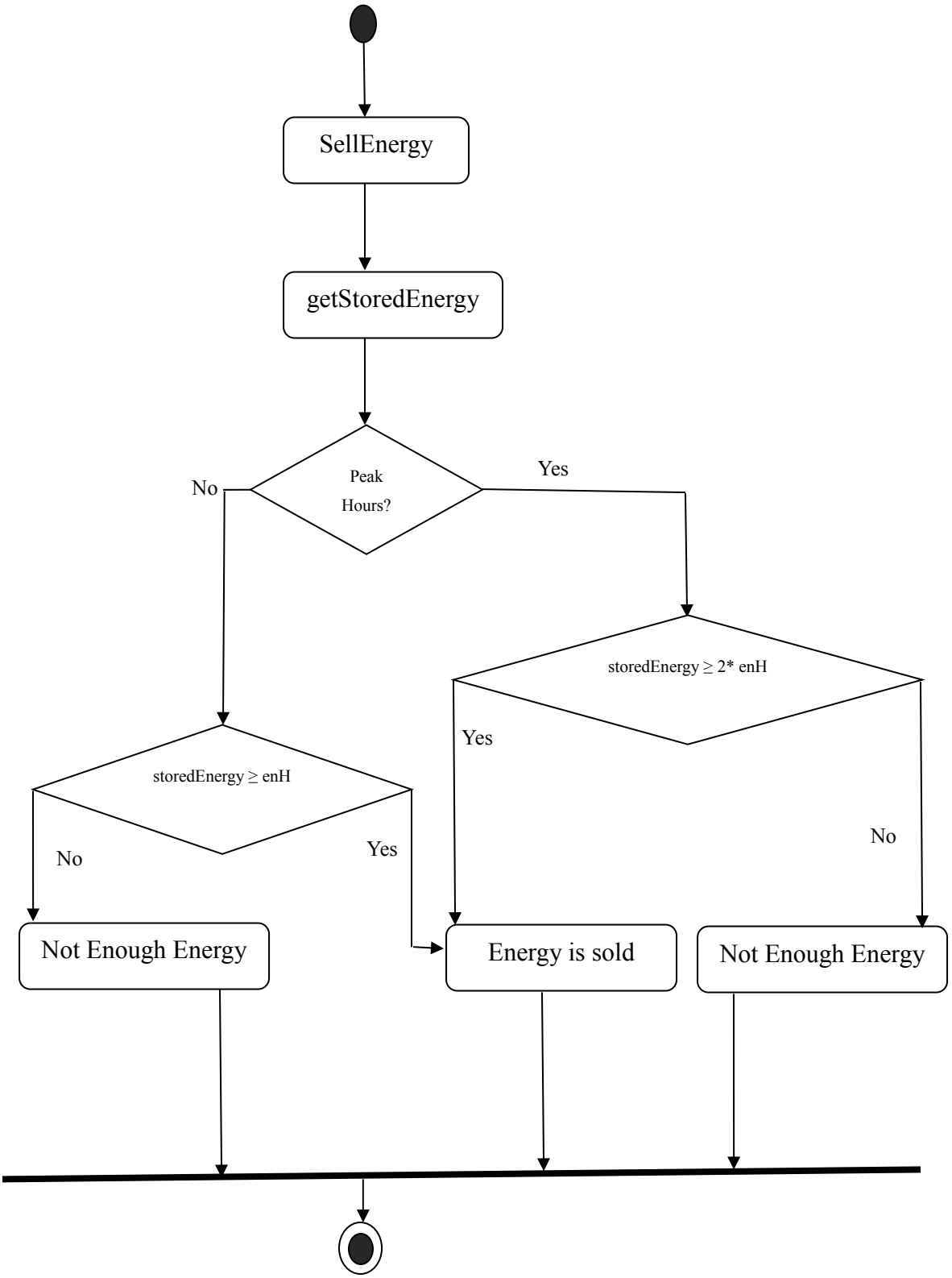

Figure 4.6: Activity diagram of the "sell energy" process

On the contrary, the system can automatically and periodically communicate with the thermostat to get the amount of stored energy and then calculate excess energy. If there is significant excess energy, it can inform the user or utility providers via email. If there is excess energy and utility providers immediately need energy, then they can contact the consumer.

4.4.1 Analysis of the algorithm to sell energy

Consider a residential smart home. The house consumes an average of 30kWh daily according to the US energy information administration [UEA]. Summer peak-hours are generally from 11 am to 5 pm. On average, the daily energy consumption of the house during peak hours is 10 kWh and daily energy consumption of the house during off-peak hours is 20kWh. According to Hydro Ottawa, the peak hour rate is given as 9.3 cents/kWh while the off-peak rate is 4.4 cent/kWh. Therefore, monthly energy cost during peak hour is about 28 CAD (i.e. 10 x 9.3 x 30) and monthly energy cost during off-peak hour is about 26 CAD (i.e. 20 x 4.4 x 30). Hence, total monthly energy cost without using stored energy is about 54 CAD. Now, we assume a house that has solar panels and a wind turbine. On average, the house can generate and store 40kWh of renewable energy daily [UEA]. Daily, energy is obtained from the storage during peak hours. Applying the above algorithm and assuming that the energy is sold during off-peak hours, it is seen that the total cost of monthly energy consumption is about 26 CAD (i.e. 20 x 4.4 x 30). Thus, it is reduced by half. Furthermore, about 600 kWh (i.e. (40 -10 -10) x 30) of energy can be sold back to the grid monthly. This is illustrated in Figure 4.7. Therefore, the residential consumer is able to reduce his energy cost significantly and make a profit by selling energy. Furthermore, the emission of greenhouse gases by the grid can be reduced if multiple houses provide green energy to it.

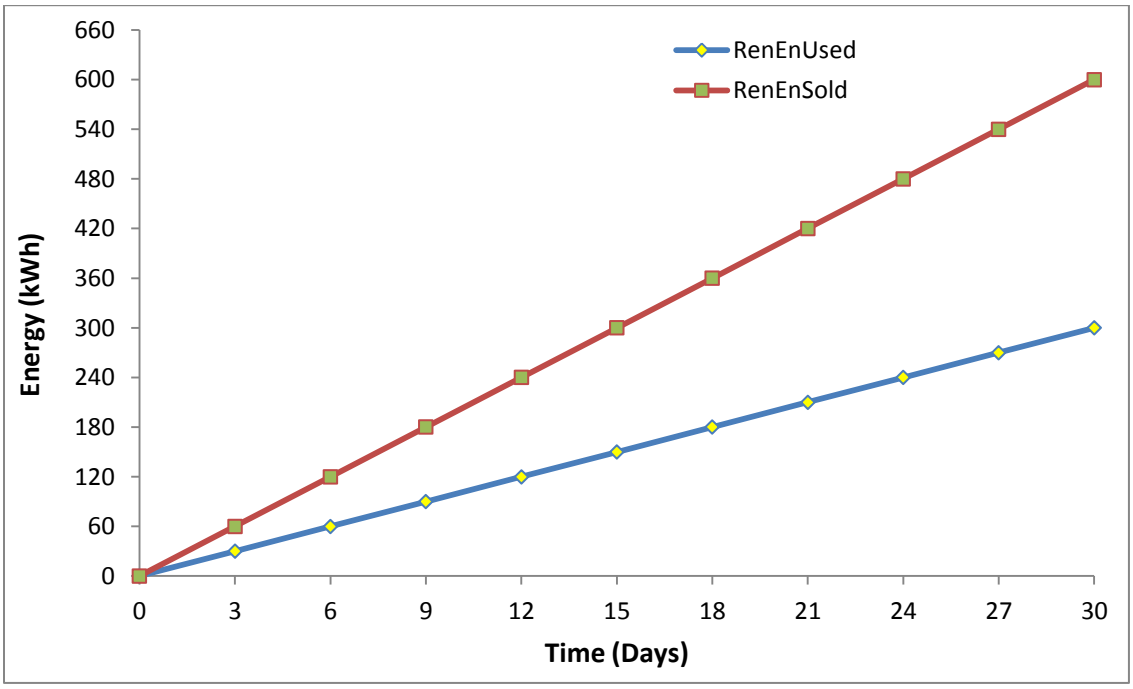

Figure 4.7: Amount of renewable energy sold and the amount of renewable energy used

4.5 Minimizing Energy using Dynamic Programming

As energy costs are rising, it is becoming imperative to minimize the consumption of energy. A dynamic programming scheme can be used for this purpose.

4.5.1 Dynamic Programming

Dynamic programming is a method for solving optimization problems [DPR]. The idea is to compute the solutions to the sub-problems once and store the solutions in a table, so that they can be reused later. It typically consists of the following steps:

I. The first step is to break the problem into smaller problems. The next thing is to find a relation between the structure of the optimal solution of the original problem and the solutions of the smaller problems.

II. The next task is to express the solution of the original problem in terms of optimal solutions for smaller problems.

III. The next action is to store the solutions of sub-problems in a table. Then, use the table to compute the value of an optimal solution in a bottom-up fashion.

IV. Use the computed information to construct an optimal solution.

4.5.2 Problem Statement

Suppose a residential consumer wants to use several appliances but at the same time he wants to keep the energy consumption less than a certain threshold (E) value. The value of E may depend on the amount of stored energy or on peak hours. In peak hours, E may be less in order to further reduce the cost of energy. The consumer cannot use all the appliances if the energy consumption has to be less than E. In this case, the user specifies his preference for each appliance. A priority value is assigned for each appliance such that the appliance that the user wants to use the most will have the highest priority value. Therefore, the problem is to find a set of appliances that have the highest total value while their consumption is less than E.

4.5.3 Dynamic Programming Model

The problem, which is mentioned in the previous section, can be solved by using a dynamic programming algorithm based on [DPR]. To solve such a problem, the dynamic programming algorithm is set up in the following way. The optimal solution consists of a set of appliances that have the highest total value while their maximum energy consumption is E. Let n be the total number of appliances. $v(j)$ refers to the priority value of the appliance j and $e(j)$ is the energy consumption of appliance j. Let S be the optimal solution. The optimization problem can be formulated as follows:

$S \subseteq \{1,2,...n\}$ such that

$$\text{maximize} \sum_{j \varepsilon S} v(j) \qquad (4.1)$$

$$\text{subject to} \sum_{j \varepsilon S} e(j) \leq E \qquad (4.2)$$

Let i be the highest-numbered appliance in an optimal solution S for maximum energy consumption E. v_i refers to the priority value of the appliance i and e_i is the energy consumption of appliance i. If $S`$ is $S$ - $i$, then $S`$ is an optimal solution with maximum energy consumption E - e_i. The value to the solution S is v_i plus the value of the subproblem $S`$. If we define $c[i, e]$ to be the value of optimal solution for items 1,2, . . . , i and maximum energy e, then we can express this fact in following equations:

$$c[i,e] = 0 \quad \text{if } i = 0 \text{ or } e = 0 \qquad (4.3)$$

$$c[i,e] = c[i-1, e] \quad \text{if } e_i > e \qquad (4.4)$$

$$c[i,e] = \max [v_i + c[i-1, e-e_i], c[i-1, e]\} \text{ if } i>0 \text{ and } e \geq e_i \qquad (4.5)$$

The value of the solution for i appliances either includes i^{th} appliance, in which case it is v_i plus a subproblem solution for (i - 1) appliances and the energy consumption excluding e_i, or does not include i^{th} appliance, in which case it is a sub-problem's solution for (i - 1) appliances and the same energy consumption. The algorithm takes as input the maximum energy E, the number of items n, and the two sequences ($v_1, v_2, \ldots, v_n$) and ($e_1, e_2, \ldots, e_n$). It stores the $c[i, j]$ values in the table, that is, a two dimensional array, $c[0 \ldots n, 0 \ldots e]$. The first row of c is filled in from left to right, then the second row, and so on. At the end of the computation, $c[n, e]$ contains the maximum value of solution. To find out the actual appliances that are in the optimal solution, we start at $c[n, E]$ and trace backwards. Therefore, if $c[n, E]$ is not equal to $c[n-1, E]$, then n^{th}

appliance is in the optimal solution and we continue tracing with $c[n\text{-}1, E\text{-}e_n]$. The algorithm is given below.

```
FOR e = 0 TO E
    c[0, e] = 0
FOR i =1 to n
      c[i, 0] = 0
      FOR e = 1 TO E
        IF e_i ≤ e AND ( v_i + c[i-1, e-e_i] > c[i-1, e] )
            c[i, e] = v_i + c[i-1, e-e_i]
        ELSE
            c[i, e] = c[i-1, e]

K = E
FOR i = n downto 1
    IF  c[i, K] ≠ c[i-1, K]
        output i
        K = K-e_i
```

Figure 4.8: Dynamic Programming algorithm

4.5.4 Analysis

Table 4.1: Energy consumption and priority values of appliances

Appliance	Priority Value	Energy Consumption
Dryer	15	1
Washer	10	5
Coffeemaker	9	3
Dishwasher	5	4

The following example is used to analyze the above mentioned dynamic programming algorithm. A user wants to use the dryer, washer, coffeemaker and dishwasher. According to the preference of the user, the priority values for the dryer, washer, coffeemaker and dishwasher are set as 15, 10, 9 and 5 respectively. The dryer, washer, coffeemaker and dishwasher consume 1, 5, 3 and 4 amount of energy respectively. These values are given in the above table. Suppose E is 8. Therefore, the sequence of priority values and energy consumption are (15, 10, 9, 5) and (1, 5, 3, 4) respectively. Considering the above values, a straight forward (or greedy) method first selects the dryer as it has the highest priority value and then it selects the washer as it has the second highest priority value. The greedy method cannot select the coffeemaker or dishwasher as the remaining energy is not sufficient to power any other appliance. Therefore, the greedy method tells the user to use the dryer and washer. Their combined priority value is 25. On the contrary, the above dynamic programming algorithm tells the user to use the dryer, coffeemaker and dishwasher. We see that their combined priority value (i.e., 29) is more than any other combination of appliances while keeping total consumption less than E. We can see this from the above algorithm where $c[4, 8]$ is 29. Therefore, the dynamic programming algorithm helps the user to select the appliances he prefers to use, while keeping consumption less than E. The following table shows the values of $c[i, j]$:

Table 4.2: Values of solutions ($c[i, j]$)

	j=0	j=1	j=2	j=3	j=4	j=5	j=6	j=7	j=8
i=0	0	0	0	0	0	0	0	0	0
i=1	0	**15**	15	15	15	15	15	15	15
i=2	0	**15**	15	15	15	15	25	25	25
i=3	0	15	15	15	**24**	24	25	25	25
i=4	0	15	15	15	24	24	25	25	**29**

c [4,8] is the last value in the table and it contains the maximum priority value (i.e. 29). This is the value of optimal solution. To find out the actual appliances that are in the optimal solution, we start at $c[4, 8]$ and trace backwards using Figure 4.8 (the last for loop). The dryer, washer,

coffeemaker and dishwasher are numbered as appliance 1, 2, 3 and 4 respectively. Table 4.3 shows the values used in backtracking. From these values, we see that the dryer, coffeemaker and dishwasher are in the optimal solution.

Table 4.3: Values used during backtracking

	condition	**Appliance no**	**K**
i=4	$c\,[4,8] \neq c\,[3,8]$	4	4
i=3	$c\,[3,4] \neq c\,[2,4]$	3	1
i=2	$c\,[2,1] = c\,[1,1]$		1
i=1	$c\,[1,1] \neq c\,[0,1]$	1	0

A more practical example is given below. A user wants to use the dryer, washer and coffeemaker. The dryer, washer and coffeemaker consume 1.19, 0.89 and 0.40 kilowatt-hours respectively. According to the preference of the user, the priority values for the dryer, washer and coffeemaker are set as 30, 20 and 15 respectively. These values are provided in the following table. Suppose E is 1.5 kilowatt-hours. Considering the above values, a straight forward (or greedy) method tells the user to operate only the dryer as it has the highest priority value (i.e. 30) and remaining energy is not sufficient to use any other appliance. On the contrary, the dynamic programming algorithm tells the user to use the washer and coffeemaker as their combined priority value (i.e. 35) is more than that of the dryer and combined consumption is less than E. Therefore, the dynamic programming algorithm helps the user to select the appliances he prefers to use, while keeping consumption less than E. Similarly, a dynamic programming algorithm can be applied to minimize energy consumption of several appliances in a building.

Table 4.4: Priority values and energy consumption of appliances

Appliance	Priority Value	Energy Consumption (kWh)
Dryer	30	1.19
Washer	20	0.89
Coffeemaker	15	0.40

4.5.5 Function to minimize energy

The system provides a function to minimize energy. This function uses the dynamic programming algorithm mentioned in Section 4.5.3. The user selects this function when he wants to operate several appliances together but wants to reduce the consumption of energy. The function keeps the energy consumption less than a certain threshold value, *E*. To use this function, the user specifies the appliances he wants to use according to his preference. The system assigns a priority value for each appliance such that the appliance which the user wants to use the most will have the highest priority value and so on. Then the user selects the operation "minimizeEnergy". The Web service is invoked. The following table shows the cycle duration and energy consumption of four different types of appliances. The energy consumption of the dryer, washer and dishwasher is given in kWh per cycle for an average load. The coffee maker is assumed to be used for making 2 cups of coffee. The system may calculate *E* depending on the amount of stored energy or peak/off-peak hour. The Web service uses the priority values, energy consumption values (from the following table) and *E* as an input to the dynamic programming algorithm. The dynamic programming algorithm returns a set of appliances that have the highest total value while their consumption is less than *E*. Finally, the user is informed about the appliances he can use. Thus, the function enables the user to reduce energy consumption while letting him use the appliances he prefers to use.

Table 4.5: Energy consumption and cycle durations of the appliances [STA08]

Appliance	Energy Consumption (kWh)	Cycle Duration (min)
Dryer	2.46	60
Washer	0.89	30
Dishwasher	1.19	90
Coffeemaker	0.4	10

The following sequence diagram shows the exchange of messages that takes place among the user and the Web service.

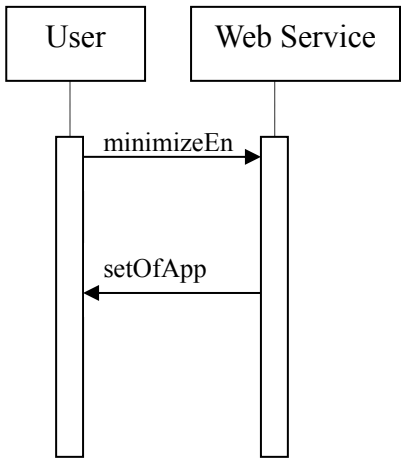

Figure 4.9: Sequence diagram of the "minimize energy" process

The following activity diagram is used to illustrate the function to minimize energy. It shows the order of actions that are taken.

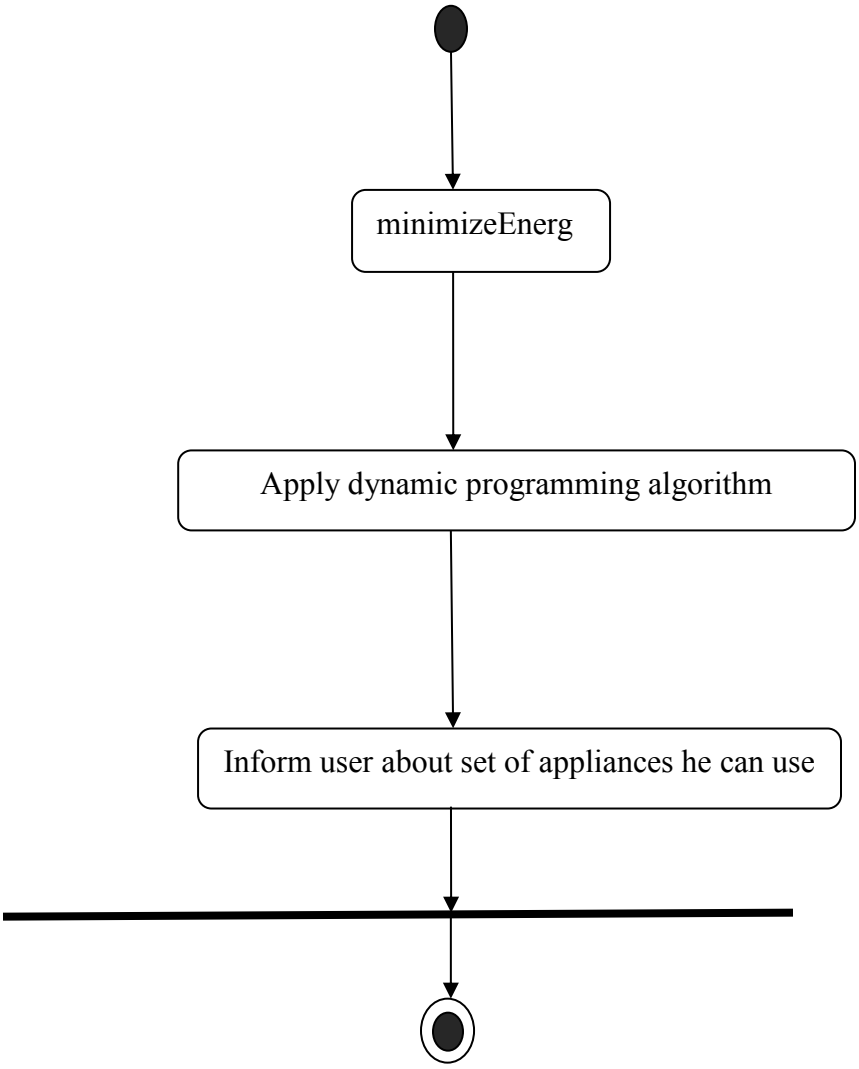

Figure 4.10: Activity diagram of the "minimize energy" function

4.6 Summary

This chapter described the advanced characteristics of the proposed system in detail. Furthermore, the interaction of the system with the smart grid was discussed. It was also described how the system supports demand response. In addition, the operations to sell energy and to minimize energy were presented. Moreover, the interaction diagrams were used to illustrate each operation. An algorithm to sell energy and dynamic programming scheme were also provided. Analysis of these operations was also presented.

Chapter 5

Web Services Implementation and Quality of Service

5.1 Introduction

This chapter describes the implementation of the proposed system and the quality of service offered by it. The implementation details of the proposed system are presented in Section 5.2. The security of the system is discussed in Section 5.3. Section 5.4 contains the quality of service offered by the application. This section describes some features of the system like an access control mechanism, differential service and security. Finally, a summary of the chapter is provided in Section 5.5.

5.2 Implementation

The implementation of the system is described in detail in this section. First, the simulation of the smart home along with ZigBee is discussed. Then, the implementation of the Web service and implementation of XMPP protocol is presented. The components of the system are illustrated in Figure 5.1.

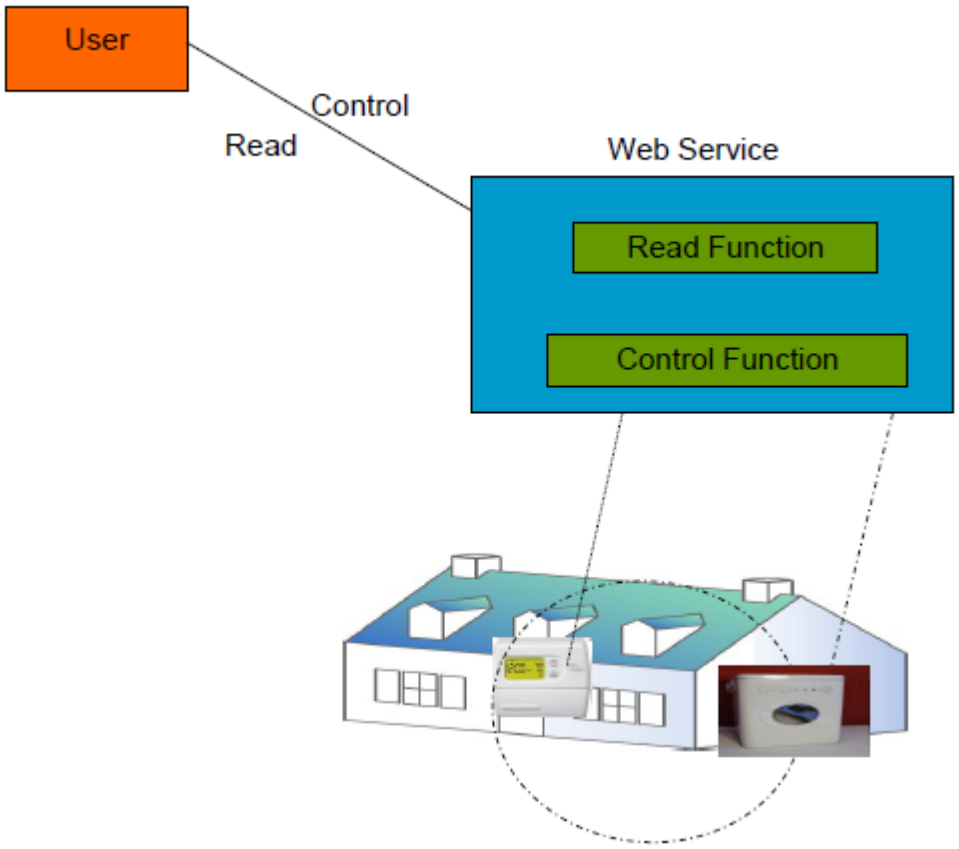

Figure 5.1: Interaction between the user and the smart home via Web services

5.2.1 Simulation of the Smart Home

A discrete-event simulator [HEN11] based on Java Programming language is used to simulate the smart home with a wireless sensor network. A typical Canadian single-family home with one floor is simulated. The model of the simulated home is shown in Figure 5.2. The simulator is further developed from [HEN11]. Appliances are included in the smart home and XMPP is added as a communication protocol. The smart home contains a washer, dryer, dishwasher, coffeemaker, HVAC, light sensors, temperature sensors, a smart (central) thermostat and central computer. HVAC operates for each room separately. There are two sensors in each room of the smart home. If one sensor fails, the reading of the other sensor is used. If both sensors are active, then the average of both readings is computed and used. The sensors measure temperature and light intensity of each room every 12 seconds. Sensors send data to the thermostat periodically via ZigBee. The simulator is designed to handle all typical seasons and can run for variable duration.

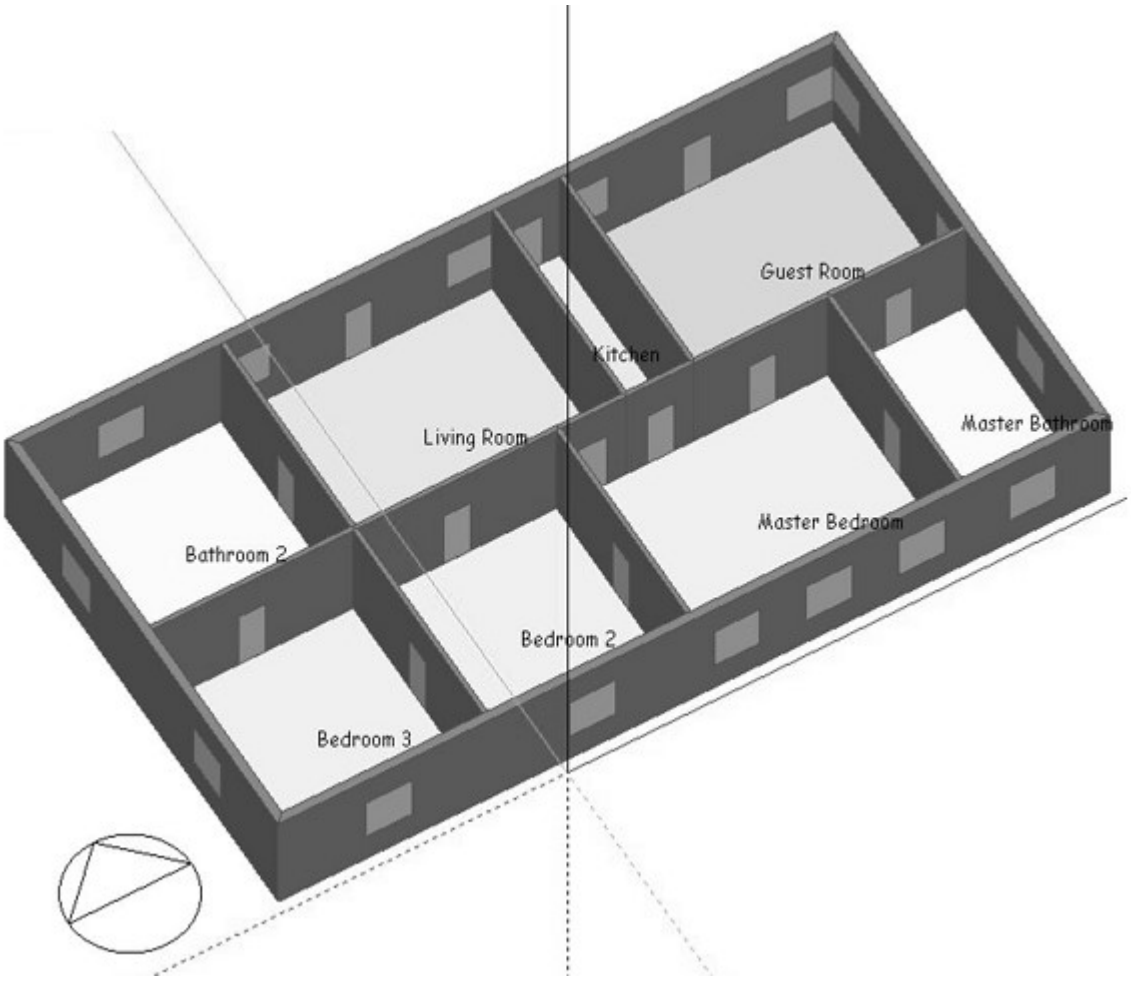

Figure 5.2: Model of the simulated home [HEN11]

ZigBee is used for the following main reasons:

- Low power: ZigBee is very suitable to use in sensor networks as it has low energy consumption. Sensors have low processing power and limited energy. Thus, it enables a sensor to function for a longer period of time.

- Device interoperability: ZigBee profiles provide interoperability and inter-compatibility that are required to allow products from different manufacturers to work. ZigBee defined standard device profiles that behave in well documented ways. Interoperability is

important in a smart home as it contains heterogeneous elements that need to communicate with each other. In the proposed system, the thermostat and sensors have different technologies. However, the sensor communicates with the thermostat via ZigBee.

- Security: ZigBee has a mechanism to check the data integrity and sender authentication. It also can encrypt data. The security features implemented within the ZigBee stack are very flexible as it can be implemented in any of the layers.

5.2.2 Web Services

Business Process Execution Language for Web Services (BPEL or BPEL4WS or WS-BPEL) is an XML-based language used for the definition and execution of business processes using Web services [JUR]. BPEL provides a relatively straightforward way to coordinate several Web services into new composite services called business processes. BPEL builds on the foundation of XML and Web services. It uses an XML-based language that supports the Web services technology stack, including SOAP, WSDL, UDDI, WS-Reliable Messaging, WS-Addressing, WS-Coordination, WS-Security and WS-Transaction. A BPEL process specifies the exact order in which participating Web services should be invoked, either sequentially or in parallel. A BPEL process is basically a Web service that is a composite of existing Web services. With BPEL, one can express conditional behaviours. For example, an invocation of a Web service can depend on the value of a previous invocation. BPEL also enables one to construct loops, declare variables, copy and assign values, define fault handlers, and so on. In a typical scenario, the BPEL business process receives a request. To fulfill it, the process invokes some existing Web services that are able to perform some functions that the process requires. The BPEL process uses the information returned from the participating Web services to serve the request and then responds to the original caller. BPEL is used to implement the secure Web service on the central computer of a smart home. The Web service is based on SOAP protocol. Oracle BPEL Process Manager (OBPM) provides a comprehensive and easy-to-use infrastructure for creating, deploying and managing BPEL business processes. OBPM is used as the server for the secure Web service. The following

code snippet shows how BPEL process invokes the Web services, passes variables and uses returned information to respond back to the caller.

BPEL process

```
<sequence name="main">

<receive name="receiveInput" partnerLink="client" portType="client:switchBpel"
operation="process"  variable="inputVariable" createInstance="yes"/>

<assign name="Assign_1">
   <copy>
     <from variable="inputVariable" part="payload" />
     <to variable="statusRequest" part="parameters" />
   </copy>
</assign>

<invoke name="Invoke_switch" partnerLink="switchWSService" portType="ns1:switchWS"
operation="getstatus" inputVariable="statusRequest" outputVariable="statusOut"/>

<assign name="Assign_2">
  <copy>
    <from variable="statusOut" part="parameters" />
    <to variable="outputVariable" part="payload" query="/client:statusReply"/>
  </copy>
</assign>

<reply name="replyOutput" partnerLink="client"
  portType="client:switchBpel" operation="process" variable="outputVariable" />

</sequence>
```

Figure 5.3: XML-code snippet showing how BPEL process operates

The Web Services Invocation Framework (WSIF) is a simple Java API for invoking Web services [WSI]. WSIF enables developers to interact with abstract representations of Web services through their WSDL descriptions instead of working directly with the SOAP, which is the usual programming model.

Web Services Invocation framework (WSIF) of OBPM is utilized to initiate and call the appropriate Java method of the simulator from the BPEL process [JUR05]. This is illustrated in Figure 5.5. The Java-based smart home simulator is integrated to OBPM as a JAR (Java Archive)

file. The BPEL process invokes a Java method instead of invoking the operation in a web service, say *W*. The way this is done will now be described below.

Schemac is a schema compiler utility provided with OBPM. This utility is used on schema definitions of *W* to generate XML facades. XML facades are a set of Java interfaces, and classes are generated through which one can access and modify XML variables of *W* and map individual XML values to Java variables with get and set methods. A Java method (in a Java class) is written that corresponds to the method in a Web service *W*. Java binding and Java port is added to WSDL file of *W*. Now, when the BPEL process call methods of Web service *W*, the corresponding operation in the Java class will be invoked instead of operation in *W*. Finally, that method will call the appropriate method of simulator. All the operations that are described in Section 3.4 can be called in this manner. Advantage of WSIF is that only the service binding (WSDL) of invoked Web service needs to be modified and the BPEL process remains unchanged. Furthermore, invoking native Java classes is much more efficient than that of invoking Web service operations. The following code snippet shows how BPEL process uses WSIF to call Java function *getCondition* of class *getStatEff* in package *org*.

Using WSIF to call java method

```
<binding name="JavaBinding" type="tns:switchWS">
  <java:binding/>

<format:typeMapping encoding="Java" style="Java">
    <format:typeMap typeName="tns:getstatus" formatType="org.me.Getstatus" />
    <format:typeMap typeName="tns:getstatusResponse"   formatType="org.me.GetstatusResponse" />
 </format:typeMapping>

 <operation name="getstatus">
    <java:operation methodName="getCondition"/>
    <input/>
    <output/>
 </operation>

</binding>


<service name="switchWSService">
  <port name="JavaPort" binding="tns:JavaBinding">
    <java:address className="org.getStatEff"/>
  </port>
</service>
```

Figure 5.4: XML-code snippet showing how BPEL process use WSIF to call Java method

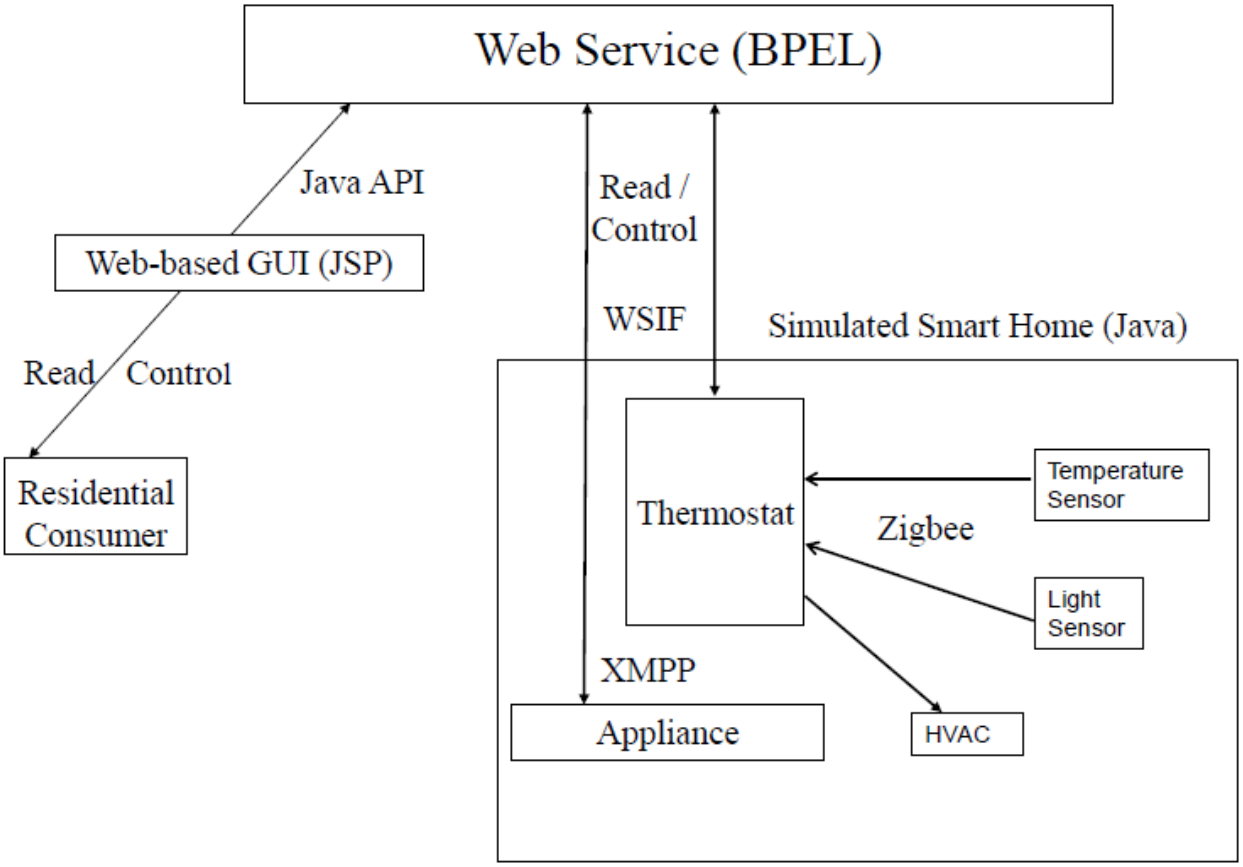

Figure 5.5: The development and communication technologies of the system

5.2.3 XMPP

The eXtensible Messaging and Presence Protocol (XMPP) is developed by the Jabber open-source community and is specified in RFC3920 [SAI04]. It is used for near real-time messaging, presence and request-response services. It typically follows a client-server architecture wherein a client communicates with a server over a TCP connection via an exchange of XML elements. The core transport layer of XMPP is XML streaming protocol. XMPP provides security using Simple Authentication and Security Layer (SASL) and Transport Layer Security (TLS) protocols. The client should use TLS to secure the streams prior to attempting the completion of SASL negotiation. Once the client has completed SASL negotiation, the client can send an unbounded number of XML stanzas over the stream to the server. Servers can also communicate with each other over TCP connections. Servers should use TLS between two domains for the purpose of

securing server-to-server communications. XMPP can exchange any type of data that can be represented in XML. TLS, SASL and XMPP protocols are stacked on top of each other to form layers. TCP is the base connection layer used by all of the protocols stacked on top of it. TLS is often provided at the operating system layer and is stacked on top of TCP. SASL is often provided at the application layer and is stacked on top of TLS. XMPP is the application itself and is stacked on top of SASL. XMPP communication deals with three core stanza types, each with its own semantics. The <presence/> stanza is a basic publish-subscribe mechanism through which several entities can receive information from an entity to which they have subscribed. Presence stanza is used for gathering information about an entity's network availability (that is, whether an entity is online, away or offline). The <message/> stanza is a "push" mechanism whereby one entity can send asynchronous information to another entity. Message stanza is optimized for real-time delivery but it does support storing and late delivery. The <iq/> (info/query) stanza is a mechanism whereby entities can make requests of and receive responses from each other.

XMPP-protocol is used for the communication between the Web service and the home appliances. It is assumed that the home appliances are connected to the Internet as XMPP protocol requires TCP for connection. XMPP client is implemented using Smack library and Openfire server is used as XMPP server. Smack is a Java-based API for communicating with XMPP servers in order to perform real-time communications, including instant messaging and group chat. It is open source and provides the core XMPP functionalities. This API encapsulates the XMPP message protocol within a set of classes such as *Chat*. The Java-based Openfire is an instant messaging and group chat server that uses the XMPP protocol. Each home appliance logs into the Openfire server using smack library. The exchange of messages between two users of Openfire is done using the *Chat* class of smack library. A chat creates a new thread of messages (using a thread ID) between two users. Each appliance uses the smack library to listen to any incoming chats from other users. The messages are processed asynchronously as they arrive and a reply is sent back to the sender. Whenever a Web service wants to communicate with the appliance, it logs into the Openfire server and sends a chat message to the appliance using smack library. It also specifies a listener object when creating a chat so that it can process any reply from

the appliance. When the appliance receives any message, it does the required operation and sends a reply back to the Web service. This kind of communication takes place during the functions *controlAppliance* and *readEnergyConsumption* of a room or a home. Smack API is used to secure the communications between the XMPP client and XMPP server. The connection to the server is secured via TLS encryption and SASL authentication.

5.2.4 Web-based Graphical User Interface

The Web service is used to access or control the smart home elements. The Web service can be invoked directly by passing some specific parameters. Therefore, the user needs to be aware of the specific operation name and the specific parameters he accepts. This way of invoking Web service is meant for an expert user and not meant for normal users. Consequently, the normal user has to use a client application. Such application has a user-friendly interface (e.g. graphical user interface) so a user can easily understand what information to enter and which operations needs to be called to serve his purpose. The application is able to call the Web service (locally or remotely) and pass the user-inputted information to the Web service. To simplify the interaction between user and Web services, the central computer provides a web-based graphical user interface. If the user does not have any client application to invoke the Web service directly, then the user can access the web-based interface over the Internet and enter required information to invoke the Web service. The interface is given in Figure 3.3.

The web-based graphical user interface is developed using JSP (Java Server Pages). OBPM is also used as JSP server. JSP is used to process the user input from the interface and to invoke the BPEL process (i.e., Web service). A Java API [BPE04] is used to invoke the BPEL process from JSP page. This is illustrated in Figure 5.5. The API is provided through a stateless session bean interface by the OBPM. A *Locator* class is used to connect to OBPM. The *Locator* class returns a handle to an *IDeliveryService* instance, which can be used to gain access to BPEL processes deployed on that server. A *NormalizedMessage* class allows the developer to construct an XML message dynamically. *Request* method of *IDeliveryService* instance is used to initiate the BPEL process and the input to the BPEL process is passed as an XML string within *NormalizedMessage*

object. The XML string contains the user input from the interface. The following code snippet is from the JSP page and it shows how the Java API is used to invoke the BPEL process.

Using java API to invoke the BPEL process

```
String xml = "<func xmlns=\"http://services.otn.com\">" + oper + "</func>";

Locator locator = new Locator("lvm", "*****", jndi);

  IDeliveryService deliveryService =
(IDeliveryService)locator.lookupService(IDeliveryService.SERVICE_NAME );

   // construct the normalized message and send to Oracle BPEL Process Manager
   NormalizedMessage nm = new NormalizedMessage( );
   nm.addPart("payload", xml );

   NormalizedMessage res = deliveryService.request("smartHome", "process", nm);

   Map payload = res.getPayload();
```

Figure 5.6: JSP-code snippet showing how Java API is used to invoke the BPEL process

5.3 Security

5.3.1 Security in Web Services

The messages that are exchanged between Web service requesters and providers typically originate deep inside one enterprise and go deep inside another. Mechanisms such as Secure Sockets Layer (SSL) are great for securing (for confidentiality) a direct connection from one machine to another, but they are of no help if the message has to travel over more than one connection. WS-Security: SOAP Message Security provides support for end-to-end message security. SOAP messages can contain security tokens with authentication information. SOAP messages can be encrypted and can contain digital signature information. WS-Security defines a SOAP Security Header format that contains sub-elements for security tokens, signature elements,

and encryption elements. WS-Trust is a specification that enterprises and sites use to build trust relationships. Public key security and Kerberos works only if the certificate authorities are trusted and trustworthy. WS-Secure Conversation uses the mechanisms defined by WS-Trust to efficiently support secure, long-lived interactions between services. WS-Federation extends WS-Trust to allow enterprises to collaborate to provide a single sign-on identification model to customers by sharing their identity information. WS-Security Policy defines assertions to represent security requirements and capabilities in the form of a WS-Policy. WS-Privacy defines how to represent privacy requirements and Web service capabilities; WS-Authorization describes how to express and manage access policies to Web services resources.

5.3.2 Security in OBPM

The Web service is deployed on OBPM. OBPM supports the use of SSL (Secure Sockets Layer) (HTTP/S), J2EE (Java 2 Platform Enterprise Edition) based authentication (HTTP) and BPEL security extensions in order to secure a BPEL process in which an interaction is initiated by an inbound client service request sent to Oracle BPEL Server [BPE07]. Figure 5.7 provides an overview of the transport security and authentication methods available for securing BPEL processes (inbound) and invoking secured services (outbound).

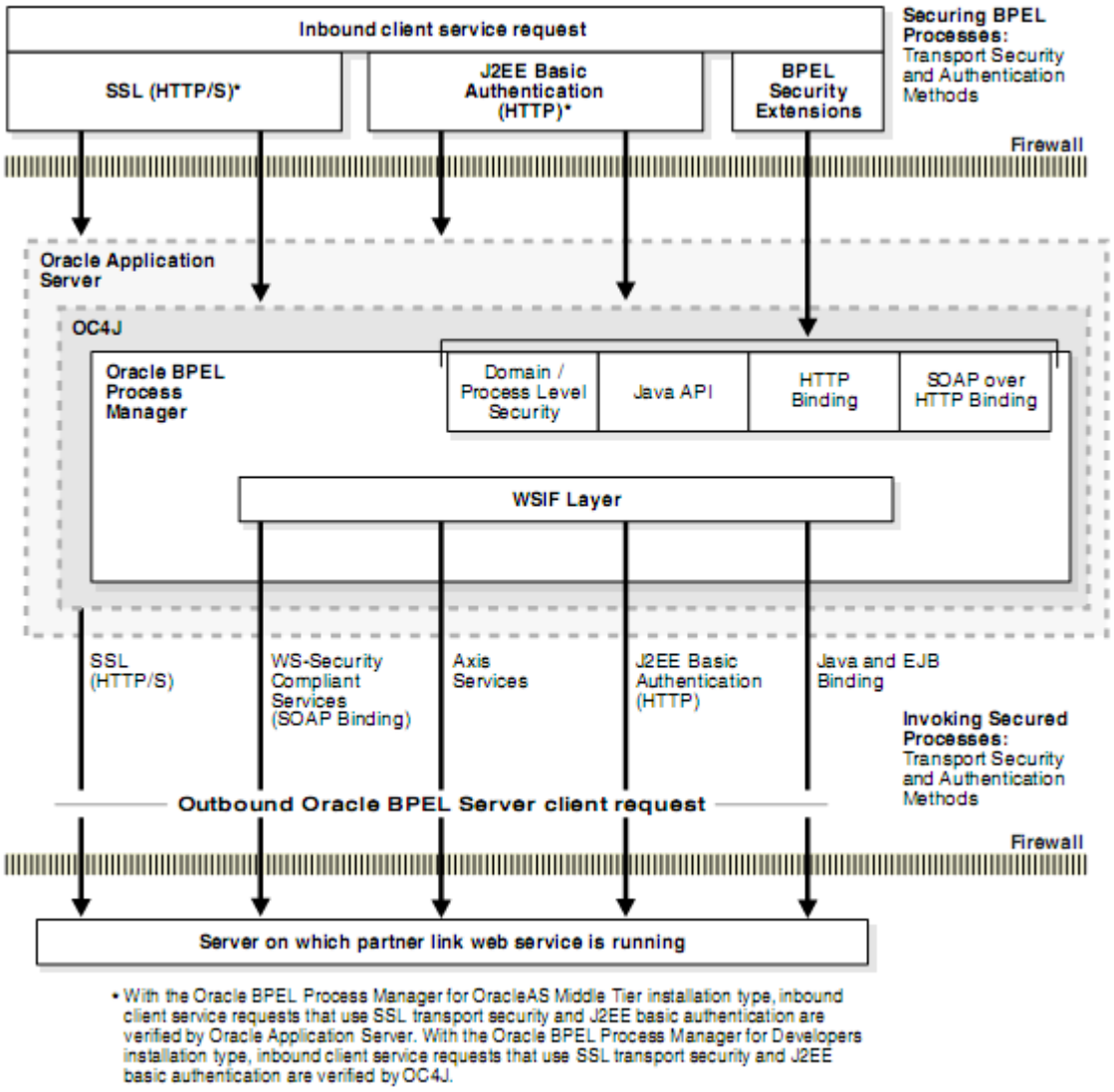

Figure 5.7: Inbound and Outbound Security and Authentication Methods [BPE07]

SSL (HTTP/S) can be used to secure a BPEL process where interaction is initiated by an inbound client service request sent to Oracle BPEL Server. BPEL processes are usually invoked using SOAP over HTTP. User names and password are prone to identification by network packet sniffers. Therefore, the network connection is secured through use of HTTP/S instead of HTTP. If HTTP/S is used as the authentication schema, both the client and server need to be configured to exchange certificates. A successful SSL handshake confirms authentication. The following types of certification authentication can be used:

- Server certificate authentication: In this scenario, the client asks the server for the certificate and authenticates the trustworthiness of the server. The client does not present its certificate unless it is requested by the server to do so.

- Server and client certificate authentication: In this scenario, both the client and server exchange certificates and a successful SSL handshake confirms authentication. This is called client authentication mode. The server (either the standalone OC4J in which Oracle BPEL Process Manager is deployed or Oracle Application Server (and its version of OC4J)) must be configured to request the client's certificate during the SSL handshake and authenticate the trustworthiness of the client. In the context of securing BPEL processes, this means that a client invoking a service presents a valid certificate issued by a mutually-trusted certificate authority.

Furthermore, J2EE basic authentication can be used to secure a BPEL process which is invoked by an inbound client service request. The request is done via HTTP and involves authentication through unsigned tokens, namely a user name and password. Moreover, BPEL security extensions can be used to secure a BPEL process where it is invoked by an inbound client service request. This is described in the next section.

5.3.3 Security and Privacy in the proposed system

It is imperative for the system to provide security. Without security, it poses the following security concerns.

- An unauthorized person can damage the home devices. For example, he can damage the washer by frequently turning it on or off.

- An unauthorized person can harm the home occupant or can cause discomfort. For example, he can set the temperature to be very high.

- An unauthorized person can increase the energy cost of the occupant by turning on lights or appliances unnecessarily.

- An unauthorized person can find out the appliances or devices that are in the house and therefore can attempt to steal them.

- An unauthorized person can find out when an occupant will not be home based on certain characteristics. For example, if the occupant has the habit of drinking coffee before leaving the house, then the intruder can find out whether he is leaving home by observing when the occupant uses the system to turn on the coffeemaker.

- An unauthorized person can sell stored energy and can harm the occupant financially.

The system provides three stages of security. The system has security in the front-end (interface), the middle part (secure web service) and back end (secure access to home devices). Consequently, there is security when the user logs into the system, invokes the WS and interacts with home devices. The following activity diagram is used to illustrate the three stages of security. It shows the order of actions that are taken. It also shows alternate actions and the decisions upon which some actions are based.

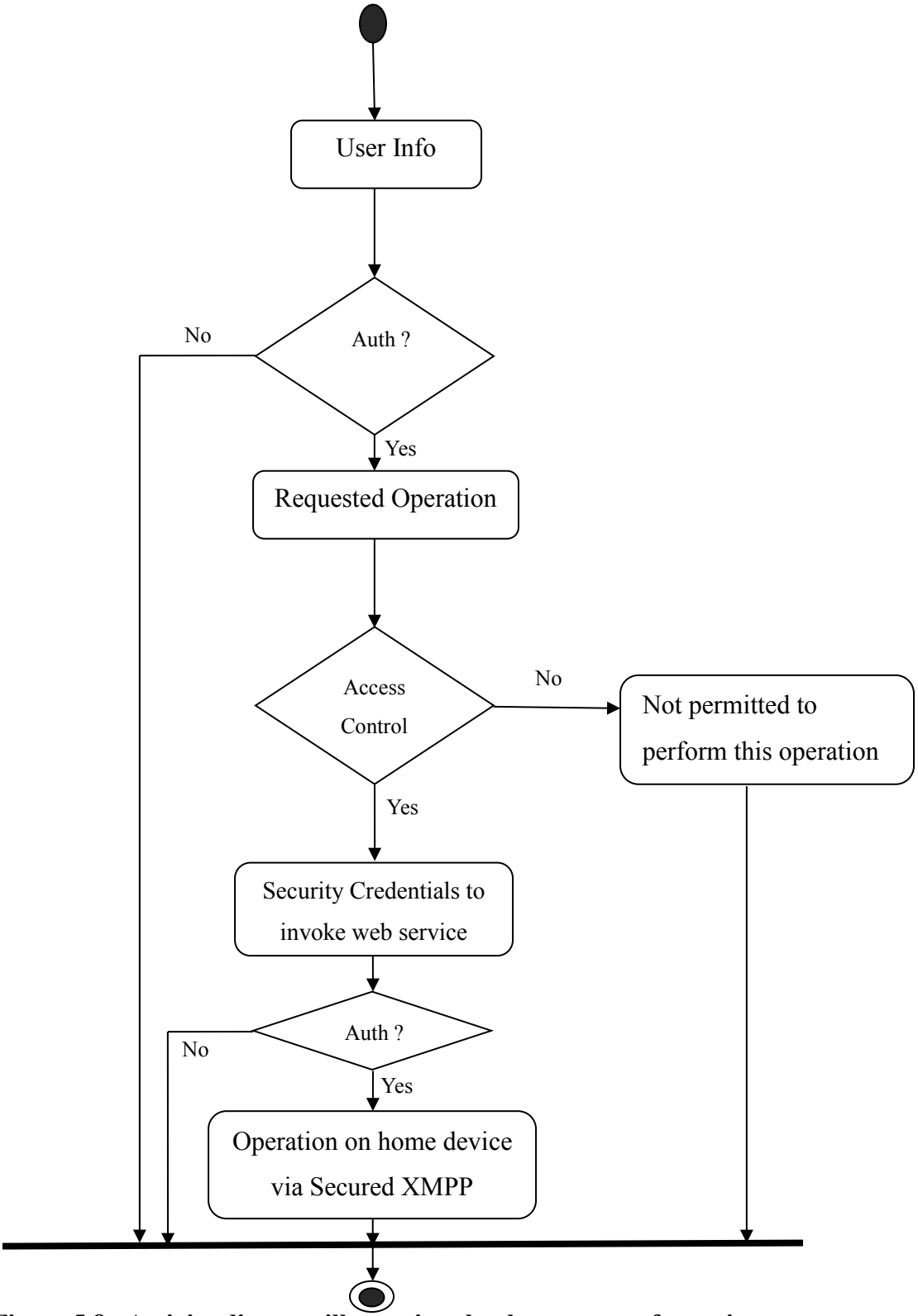

Figure 5.8: Activity diagram illustrating the three stages of security

The user must be authenticated to log into to the system. The Web service is deployed on OBPM. OBPM's API includes (native) BPEL Security Extensions that can be used to create custom security [BPE07]. These extensions are used to secure a BPEL process (i.e. Web service). Therefore, users must be authenticated and authorized to use or invoke a secure BPEL process. Within the Oracle BPEL Server, a message handler framework is used to control and modify inbound and outbound message flows. It provides domain security and process-based security. Domain security enables one to secure all processes running in a specific domain. Process-based security enables one to specify which processes to secure in a specific domain. The following is done to make the BPEL process secure. The configuration file of the OBPM is changed to enable domain security and process-based security. Then, the name of the BPEL process is specified in this file so that security can be applied to the process. In the configuration file of the BPEL process, some properties are included that specifies the username and password required to invoke the BPEL process. Therefore, the BPEL process is now secured and a specified username and password is required to invoke it. The BPEL process can be invoked by HTTP, SOAP and Java API. Regardless of how the BPEL process is invoked, the same security constraints apply. However, the way security credentials are passed depends on the way the process is invoked. The username and password are passed in the normalized message when the BPEL process is invoked by Java API. Security credentials can be passed as a WS-Security compliant SOAP header when process is invoked by SOAP. The following table shows the features that are supported by the Native BPEL Security Extensions.

Table 5.1: Features that are supported by the Native BPEL Security Extensions [BPE07]

Authentication Schemas	Service Access Protocols	User Repository	Customization Permitted	Granularity
Basic authentication (user name and password)	HTTP	Oracle Application Server JAZN repository types: ▪ OID ▪ JAZN XML ▪ Database-based repository ▪ Custom	Custom user repository using the custom validator class	Fine grained: ▪ Individual process level security (for example, `service1` with `username1/password1` and `service2` with `username2/password2`) ▪ Supports domain level protection and all services in that domain with one user name and password
Normalized message properties	▪ Java API ▪ Remote method invocation (RMI)			
WS-Security (in accordance with the *WS-Security Web Services Security Specification*)	SOAP			

Invoking an unsecured BPEL process by Java API is described in Section 5.2.4. A user is also able to invoke a secure BPEL process by Java API. In this case, the username and password are passed in the normalized message. The following code snippet shows how security credentials are passed by adding properties (NormalizedMessage:setProperty(key, value)) to the normalized message object, *nm*.

```
nm.setProperty("secured", "adnan");
nm.setProperty("adnan", "home");
// username is “adnan” and password is “home”.
```

Figure 5.9: Java-code snippet showing how security credentials are passed

Security credentials can be passed as a WS-Security compliant SOAP header when secured process is invoked by SOAP. An example of this is shown in Figure 5.10.

```
<wsse:Security soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next"
            soapenv:mustUnderstand="1"
            xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
            xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<wsse:UsernameToken
   xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd">
   <wsse:Username>Clemens</wsse:Username>
    <wsse:Password Type=
         "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-
profile1.0#PasswordText"> pwForClemens
      </wsse:Password>
 </wsse:UsernameToken>
</wsse:Security>
```

Figure 5.10: Example of passing security credentials as WS-Security SOAP header

ZigBee is used for communication between the thermostat and sensors. ZigBee has a mechanism to check the data integrity and sender authentication. XMPP-protocol is used for the communication between the Web service and the home appliances. This communication is secured via TLS encryption and SASL authentication. Thus, the system provides security and privacy.

5.4 Quality of Service

Quality of service (QoS) is the ability to provide different priority to different users (or applications) or to guarantee a certain level of performance. The goal of QoS is to provide guarantees on the ability of a system to deliver expected results. Some of the metrics used to measure QoS are reliability, latency (delay), accuracy, etc.

5.4.1 Security

One of the important aspects of quality of service is to provide security so that the user can rely on the system. It prevents unauthorized access to the home elements. The system has security in the front-end (interface), the middle part (secure Web service) and back end (secure access to home devices). Security was described in the previous section.

5.4.2 Differentiated Service

The system provides three stages of security as shown in Figure 5.8. There is security when the user logs into the system, invokes the WS and interacts with home devices. A user, who needs security at all three stages, pays more than the user who needs security at one or two stages. Hence, differentiated service is provided as the level of security is configured according to user need or budget. Different residential consumers can ask for different levels of security for their smart home. The basic security is that the user must be authenticated to use the system. The Web service can be made secure or unsecure by configuring the OBPM or the BPEL process. The user interacts with home devices via XMPP. The level of security of this interaction can also be configured. XMPP provides security via TLS encryption and SASL authentication. TLS encryption can be enforced if tougher security is required. The levels of security are defined as follows:

- *Security strength 1*: The user must be authenticated to use the system. However, the BPEL process (i.e. Web service) is not secured. Communication via XMPP is not secured as the TLS encryption is disabled.

- *Security strength 2*: The user must be authenticated to use the system. However, the BPEL process (i.e. Web service) is not secured. Communication via XMPP is secured as the TLS encryption is enforced.

- *Security strength 3*: The user must be authenticated to use the system. The BPEL process (i.e. Web service) is secured by enabling domain security and process-based security of OBPM. Communication via XMPP is not secured as TLS encryption is disabled.

- *Security strength 4*: The user must be authenticated to use the system. The BPEL process (i.e. Web service) is secured by enabling domain security and process-based security of OBPM. Communication via XMPP is secured as TLS encryption is enforced.

5.4.3 Access Control

An access control mechanism restricts access to certain users. A residential consumer has total access and can read or control any smart home elements via Web services. A utility provider or other users are not able to read or control every smart home element. A utility provider can execute all operations except functions to sell energy and control appliances. Other users are able to only read information from home elements. These guest users are able to only read temperature, light intensity and energy consumption of room. However, they are not able to sell energy, and they are not able to control room temperature, light intensity, and appliances. The following table shows the various operations that can be accessed by different types of users. “Yes” means that the particular operation can be accessed by the user and “No” means that the particular operation cannot be accessed by the user.

Table 5.2: Access control mechanism of various operations

Operations	Residential Consumer	Utility Provider	Guest
Read Temperature	Yes	Yes	Yes
Read Light Intensity	Yes	Yes	Yes
Read Room Energy Consumption	Yes	Yes	Yes
Read Home Energy Consumption	Yes	Yes	Yes
Adjust Temperature	Yes	Yes	No
Adjust Light Intensity	Yes	Yes	No
Control Appliance	Yes	No	No
Minimize Energy	Yes	Yes	No
Sell Energy	Yes	No	No

5.5 Summary

This chapter described the implementation of the proposed system in detail. The programming languages or technologies that are used to implement various components of the system are described. The components of the system are the simulated smart home, secure Web service and web-based interface. Moreover, the communication technologies that are used to communicate between various parts of the system are also described. Furthermore, the chapter talked about the security of the system along with the protocols and technologies that are used to secure the system. Then, the quality of service offered by the application is presented. Features of the system like access control mechanisms and differential services are also described.

Chapter 6

Performance Analysis

6.1 Introduction

This chapter presents the simulation results that are obtained by running the system. Furthermore, this chapter contains the analysis and verification of the results. The development environment is presented in Section 6.2. This section contains the configuration of the computer and the software used to develop the simulated system. Section 6.3 describes the system settings that are required before running it. This section specifies the files and the libraries that need to be integrated to the system. The characteristics of the simulator are given in Section 6.4. This section includes the contents of the simulator and the way to run it. Section 6.5 presents the simulation results. Furthermore, this section contains the analysis and verification of results. Finally, a summary of the chapter is provided in Section 6.6.

6.2 Development Environment

The system is implemented on a personal desktop computer that runs Windows XP Professional (Version 2002, SP 2). The processor of the computer is Intel Core 2 Duo (2.33 GHz) and it has a RAM of 1.95 GB. The following software is installed on the desktop computer:

- Oracle SOA suite (Oracle Application Server), version 10.1.3.1.0, is installed. This suite contains the Oracle Containers for J2EE (OC4J), OBPM and Notification Server (OPMN).

- Oracle JDeveloper, version 10.1.3.4, is used. Oracle JDeveloper is an integrated development environment (IDE) for building applications and Web services using Java, XML, and SQL standards. JDeveloper BPEL Designer is integrated with Oracle JDeveloper. JDeveloper BPEL Designer is an IDE for developing BPEL processes. It is also used to compile the BPEL process (i.e., Web service) and then deploy the process to OBPM. Oracle JDeveloper is used to develop the JSP-based interface and also to deploy the JSP to OBPM.

- Netbeans IDE (version 6.1) and Java 2 Platform Standard Edition 5.0 Development Kit (JDK 5.0) (product version number 5.0, developer version number 1.5.0) is used for Java programs.

- Smack API (version 3.2.1) is used as XMPP client and Openfire (version 3.7) is used as the XMPP server.

6.3 System Setup

After installing the software mentioned in previous section, some more actions are required. The following tasks need to be performed before running the system for the first time:

- A Java-based discrete-event simulator is used to simulate the smart home with wireless sensor network [HEN11]. The code of this simulator is integrated to OBPM as a JAR file. This is done by specifying the location of JAR file in the configuration (i.e., server.xml) file of Oracle application server.

- The Java based Smack library is also integrated to OBPM as a JAR file. This is done by specifying the location of JAR file in the configuration file of Oracle application server. Smack is used for communicating with Openfire (i.e., XMPP) server.

- A Java class file (*getStatEff.class*) is put in the system/classes folder of the Oracle application server. A method (named *getCondition*) in this Java file can be invoked from the BPEL process using WSIF.

- The following input files are placed in the home directory of Oracle application server:
  - *RoomProperties.csv*: This is a Comma Separated Values (CSV) file containing the details of the smart home. This file contains aspects of the residential structure, such as the number of rooms, number of outdoor walls in each room, thermal resistance of each wall, and so on. In addition, this file contains the suitable temperature and suitable light intensity for each room.
  - *Appliance.csv*: This is a CSV file containing the locations of the appliances in the smart home.

- Openfire server is started. In Openfire server, user accounts are created for each appliance. Appliances use the corresponding account to log into the server and listen for messages.

- Oracle Application Server is also started.

6.4 Simulator

6.4.1 Classes

The smart home simulator is developed using Java. Packages are simply a group of classes with similar functionalities. The lists of packages [HEN11], along with a brief description of the class, are as follows:

- decision_Handling_Module package: This package holds the classes responsible for the decision making in the program. It contains the following classes:
  - "buildingConstruct.java," the class which has the implementation of all algorithms

- "Door.java," which has the implementation of a door model that indicates the type of door (indoor or outdoor) and its state (open or closed)
- "Room.java," which contains the implementation of one conditioned zone or room.

- input_Output_Manipulation package: This package contains only one class, "CSV_Parser.java," that deals with all input and output CSV files of the simulator.

- Queue_Element_Types package: This package contains twelve classes describing all the types of events that can come into the queue.

- simulator_Backbone package: This package has one interface and two classes. Interfaces are mainly used to allow classes to share global variables or functionalities without actually allowing "global" variables or functionalities. The interface in this simulator is called "SharedConstants.java." All the Queue_Element_Types classes are implementing the Shared_Constants.java file. The two classes in the simulator_Backbone package are DynamicEvent.java, and EventQueue.java, which describe a generic element in the queue of the discrete-event simulator, and the protocol on how to deal with the queue.

6.4.2 Assumptions

The following assumptions [HEN11] are made while simulating the smart home.

- The house is assumed airtight (i.e. no leaks and cracks throughout the home).

- There are no other houses or structures near enough to the tested house to cause any shading.

- Wind chill factors are not included in any calculations.

- House properties are the same for all rooms and throughout the residential structure (same thickness, thermal resistance, color, and type of wall, roof, and windows).

- Windows are never opened.

- The lighting of every room in the home is to be defined as a single, hanging, 100W incandescent light bulb that produces 1750 lux. It is assumed to be always on through the duration of the simulation, albeit not always with full power.

6.4.3 Characteristics of the simulator

In the smart home simulator, there is some random behaviour. Each outdoor door, and its corresponding room, is modeled as follows. The incoming rate of people at the door is modeled as a Poisson process where arrival rate is λ people per hour. The service rate (i.e. the rate at which people leave the room) is μ people per hour. The duration of stay for each person (i.e. service period) is modeled as an exponential distribution. Each external door with its corresponding room is modeled as an M/M/1 queue [LEO04]. The value of λ is 0.1 arrival/hour and μ is 0.2 departures/hour respectively. The temperature and energy consumption in the room varies when doors are opened and closed. Temperature and energy consumption also varies according to the number of people and the duration of their stay in the room. For an M/M/1 queue to be stable and not overflow, the incoming rate must be less than the outgoing rate (i.e. $\lambda < \mu$) [LEO04]. For M/M/1 queues, the term traffic intensity, denoted by ρ, is defined as λ / μ. For stability, therefore, ρ must be less than 1.

6.4.4 Running the simulator

To run the system, the web based graphical user interface is accessed. The necessary information is entered in the interface and then the required operation is called. A Java API is used to invoke the BPEL process (i.e., Web service) from JSP based interface. The inputs from the interface are passed to the BPEL process as an XML string. The BPEL process uses WSIF to call a Java method (named *getCondition*) inside a Java class (called *getStatEff.class*). This method runs the smart home simulator and implements some operations. Furthermore, it calls relevant functions

from the Java based simulator in order to communicate with smart home devices. In addition, it uses Smack API to communicate with appliances via XMPP. The communications among various components of the system are given in detail in Chapter 3. Using WSIF, the BPEL process initiates the smart home simulator in a thread and set it to run for *20* days (simulation time) in *summer*. Average and confidence interval (CI) of 95% is calculated for all results. In this way, the simulation results are verified. Appendix A contains detailed information on confidence interval.

6.5 Simulation Results

6.5.1 Completion Time

The traffic intensity is increased in increments of 0.2 from 0.2 to 0.8. For each value of traffic intensity, the simulator is run 5 times. In each run, the simulation time is set as *20* days and the season is set as *summer*. In each simulation run, all the operations of the system are called and the completion time of each operation is recorded. The *completion time* of an operation is calculated as the amount of time elapsed from when the Web service is invoked with that operation until the response from the Web service is processed.

For each value of traffic intensity, the completion time of the function to "read temperature of room" is calculated 5 times from 5 different simulations run. Average and CI is calculated for these 5 values of completion times. Each point in the Figure 6.1 represents the average completion times of 5 values (along with 95% confidence interval) measured at that traffic intensity.

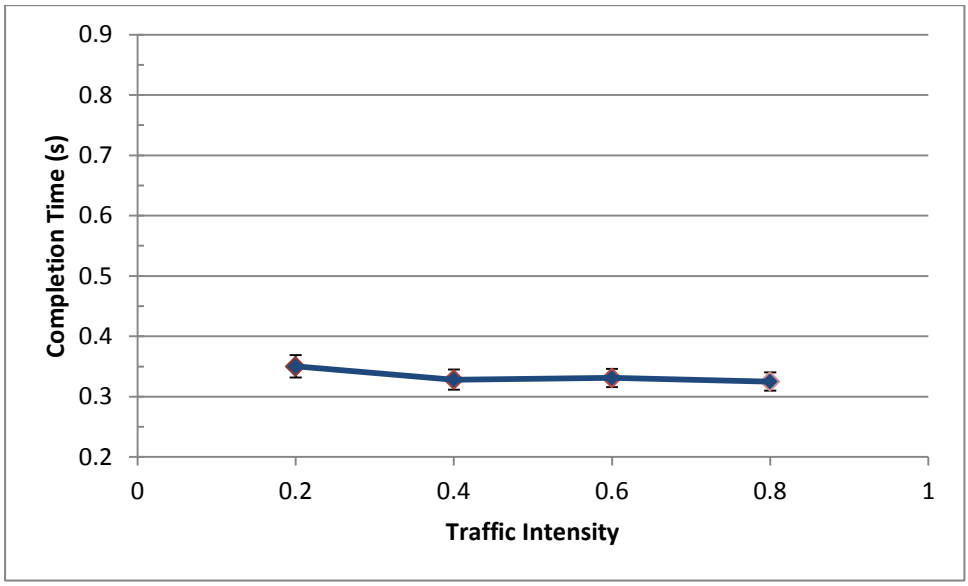

Figure 6.1: Completion time of operation that reads room temperature.

The completion time of other functions are calculated in the same way. Figure 6.2 and Figure 6.3 illustrate the completion time of two other functions.

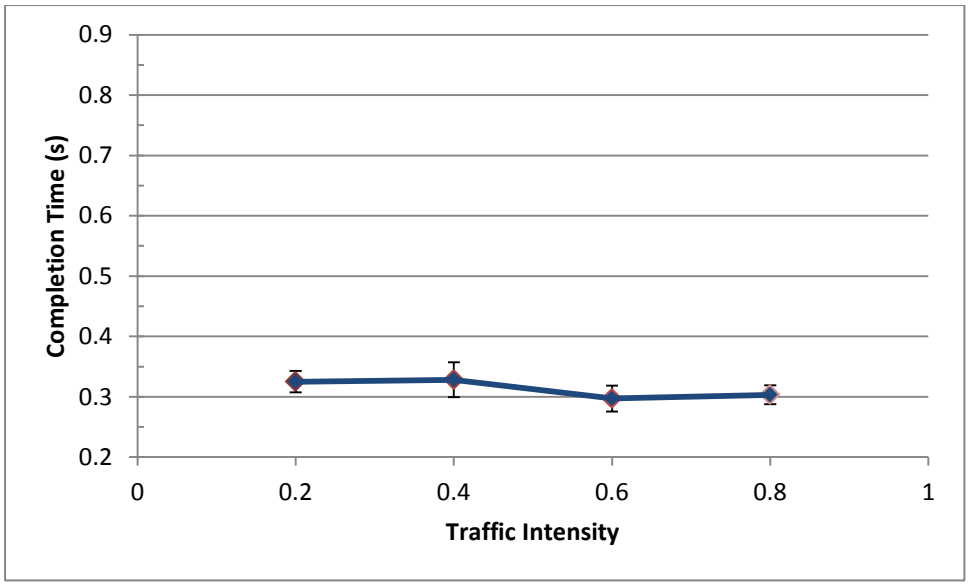

Figure 6.2: Completion time of "minimize energy" operation.

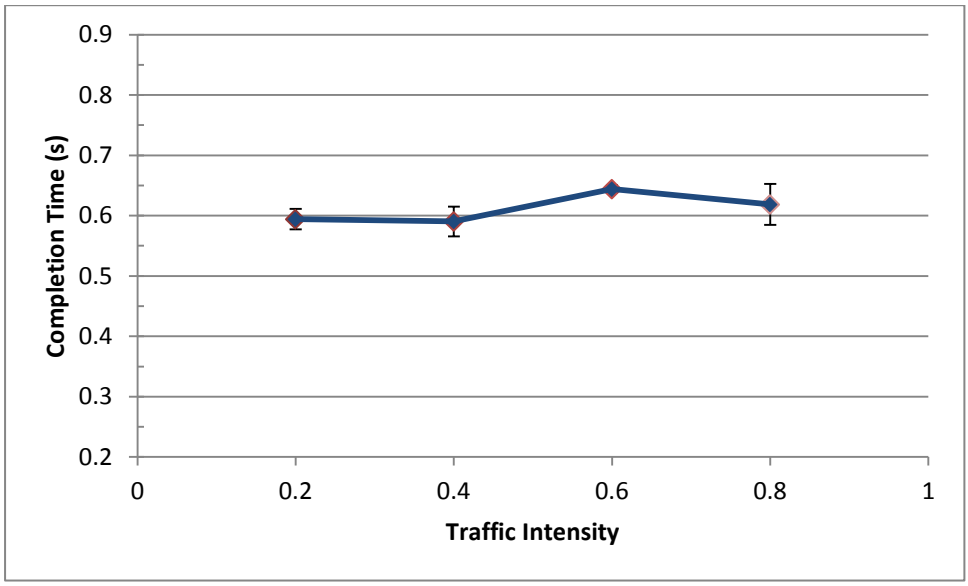

Figure 6.3: Completion time of operation to control appliance.

We can see from the above figures (Figure 6.1, 6.2, 6.3) that the completion time of each of those functions remains fairly constant as traffic intensity changes. The completion time of all the functions of the system remains fairly constant as traffic intensity changes. Traffic intensity reflects the changes in temperature and energy consumption that occurs in a home as people are coming in or going out of the room and doors are opened or closed. The results show that the completion time of functions are consistent as these changes take place inside the house. Furthermore, the results are verified as the confidence interval values are small.

6.5.1.1 Analysis of completion time

In the previous section, average completion time of an operation is calculated for each value of traffic intensity. The overall completion time of an operation is the average of the 4 values calculated at 4 traffic intensities. Figure 6.4 shows the overall completion time (in seconds) of all operations provided by the system.

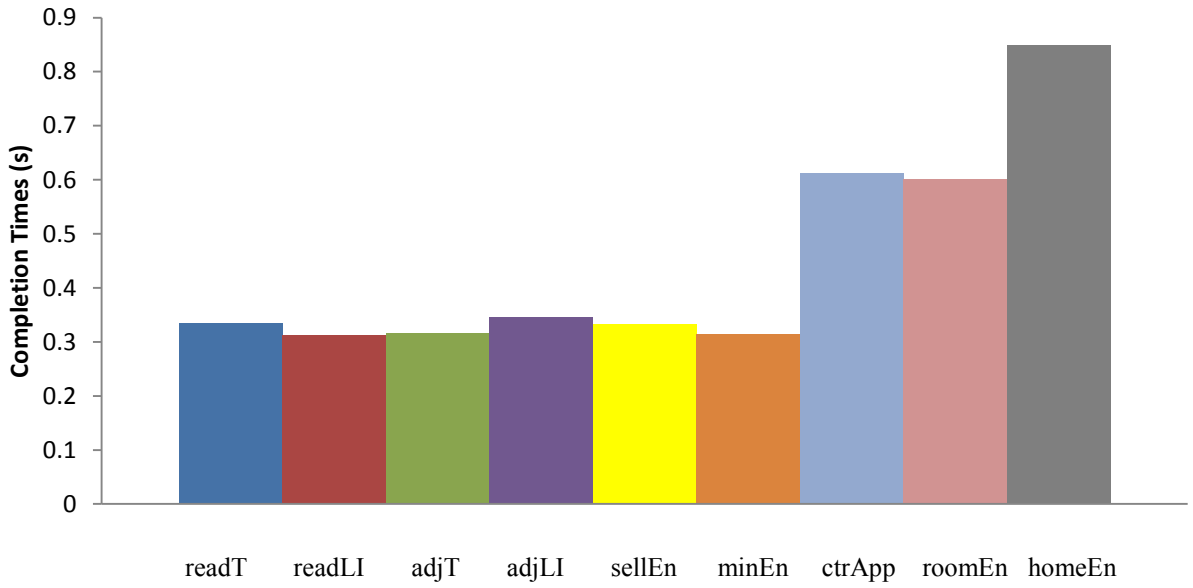

Figure 6.4: Overall Completion time of all operations.

The above bar chart shows the overall completion times of all operations provided by the system. From left to right, the bars shows the overall completion time of functions to "Read Temperature (readT)", "Read Light Intensity (readLI)", "Adjust Temperature (adjT)", "Adjust Light Intensity (adjLI)", "Sell Energy (sellEn)", "Minimize Energy (minEn)", "Control Appliance (ctrApp)", "Read Room Energy Consumption (roomEn)" and "Read Home Energy Consumption (homeEn)" respectively. We can see from the above graph that the completion time of operations (i.e. "Control Appliance", "Read Room Energy Consumption", "Read Home Energy Consumption") that uses XMPP is more than other operations. However, the operations that use XMPP have higher security as XMPP provides additional protection. The completion time of "total home energy consumption" operation is highest compared to other operations. This is because of the time it takes to communicate with all the appliances in all the rooms via XMPP. The completion time of "control appliance" and "get room energy consumption" operations is slightly more than other operations. This is because of the time it takes to communicate with the appliances in the room via XMPP. The overall completion times of rest of the operations are similar. Furthermore, Figure 6.4 shows that the average completion time of all operations is small. In addition, the

results are verified as the confidence interval values are small. Therefore, Web services can be used to perform the above mentioned operations efficiently, remotely and securely.

6.5.1.2 Comparison of completion time with other systems

In [YAZ09], a system is presented that uses Web services for energy management of the home. The system uses Web services to control temperature and light. In order to prove the superiority of our system, the results are compared with the system in [YAZ09]. Figure 6.5 shows the average completion time (in seconds) of the common operations of both systems.

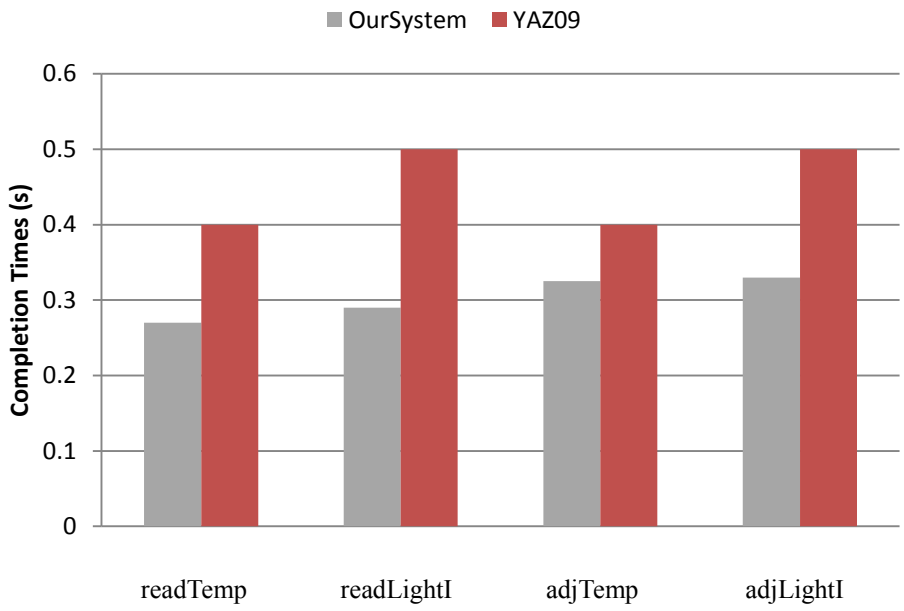

Figure 6.5: Comparison of average completion time with the system in [YAZ09]

Figure 6.5 shows that our system has less completion time than the system in [YAZ09] for common operations. Consequently, our system performs better than the other system and can be used to efficiently communicate with the smart home devices for energy management.

Furthermore, authors of [SLE11] presented a system where Web service is implemented on a device that has a sensor. The data of the sensor is requested by invoking the Web service from the Internet using a browser. The sensor data are provided in XML formats. In order to prove the

superiority of our system, the completion time of an operation of our system is compared with a similar operation of the system in [SLE11]. Figure 6.6 shows the average completion time (in seconds) of the similar operation of both systems.

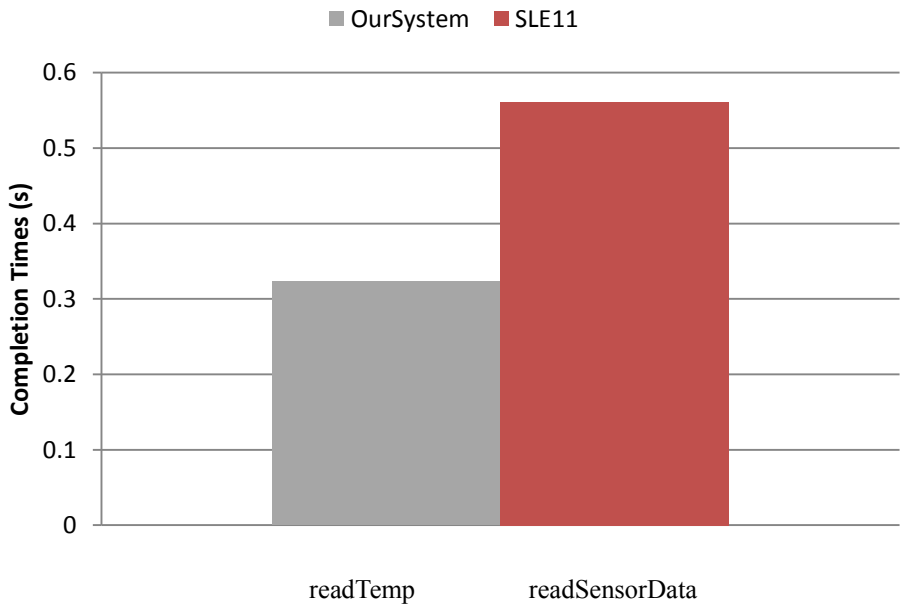

Figure 6.6: Comparison of average completion time with the system in [SLE11]

Figure 6.6 shows that our system has less completion time than the system in [SLE11]. Consequently, our system performs better than the other system and can be used to efficiently communicate with the smart home devices for energy management. The paper in [SLE11] also proposed some methods to reduce the completion time. Those methods are not compared as they are not similar. Furthermore, those methods contain extra processing that may reduce battery life or consume more energy.

6.5.2 Differentiated Service

It is mentioned in Section 4.6.2 that the system provides differentiated service. The system provides three stages of security. There is security when user logs into the system, invokes the WS and interacts with home devices. User interacts with home devices via XMPP. A user, who needs security at all three stages, pays more than the user who needs security at one or two

stages. Therefore, the strength of security is configured according to user need or budget. The levels of security are specified in the following table.

Table 6.1: Levels of Security

Security Strengths	BPEL process (Web service)	Communication via XMPP
1	Not Secured	Not Secured
2	Not Secured	Secured
3	Secured	Not Secured
4	Secured	Secured

In order to study the effect of security on the efficiency of the system, the completion time is measured against the above mentioned security strengths. The security strength is increased from 1 to 4. For each value of security strength, the simulator is run 5 times. In each run, the simulation time is set as *20* days and the season is set as *summer*. In each simulation run, all the operations of the system are called and the completion time of each operation is recorded.

6.5.2.1 Function to control the home environment

The function to control the home environment includes function to read temperature, read light intensity and control light intensity.

Function to read temperature of a room

For each value of security strength, the completion time of the function to "read temperature of a room" is calculated 5 times from 5 different simulations run. The average is calculated for these 5 values of completion times. Each point in the following graph represents the average completion times of 5 values measured at that value of security strength.

Function to read light intensity of a room

For each value of security strength, the completion time of the function to "read light intensity of a room" is calculated 5 times from 5 different simulations run. The average is calculated for these

5 values of completion times. Each point in the following graph represents the average completion times of 5 values measured at that value of security strength.

Function to adjust light intensity of a room

For each value of security strength, the completion time of the function to "adjust light intensity of a room" is calculated 5 times from 5 different simulations run. The average is calculated for these 5 values of completion times. Each point in the following graph represents the average completion times of 5 values measured at that value of security strength.

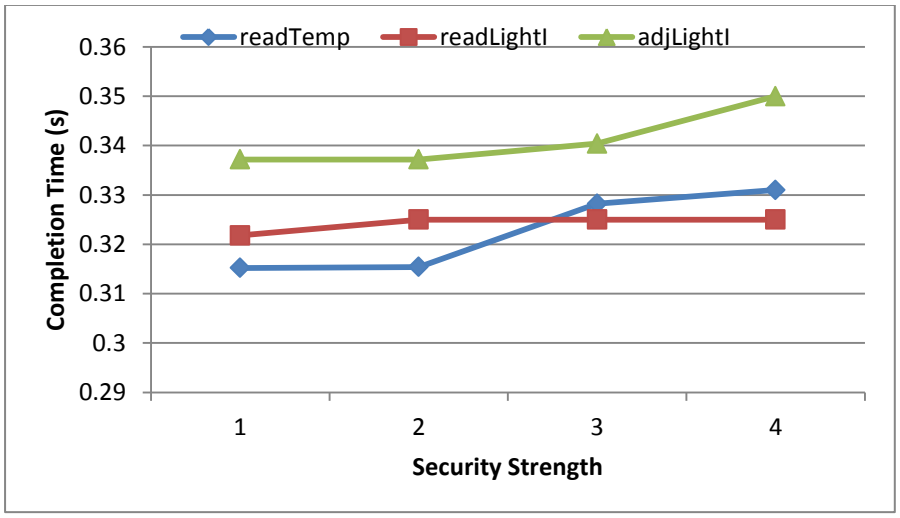

Figure 6.7: Completion time of operations that control the home environment.

6.5.2.2 Function to manage energy

The function to manage energy includes function to sell energy and minimize energy.

Function to sell energy

For each value of security strength, the completion time of the function to "sell energy" is calculated 5 times from 5 different simulations run. The average is calculated for these 5 values of completion times. Each point in the following graph represents the average completion times of 5 values measured at that value of security strength.

Function to minimize energy

For each value of security strength, the completion time of the function to "minimize energy" is calculated 5 times from 5 different simulations run. The average is calculated for these 5 values of completion times. Each point in the following graph represents the average completion times of 5 values measured at that value of security strength.

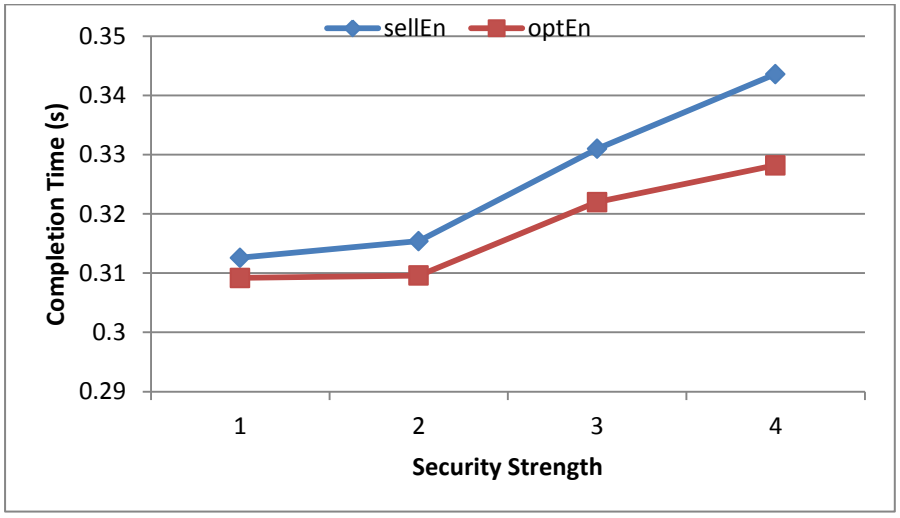

Figure 6.8: Completion time of operation to manage energy.

6.5.2.3 Function that uses XMPP

The functions that use XMPP includes function to control appliance, read energy consumption of room and read energy consumption of home.

Function to control appliance

For each value of security strength, the completion time of the function to "control appliance" is calculated 5 times from 5 different simulations run. The average is calculated for these 5 values of completion times. Each point in the following graph represents the average completion times of 5 values measured at that value of security strength.

Function to read energy consumption of a room

For each value of security strength, the completion time of the function to "read energy consumption of room" is calculated 5 times from 5 different simulations run. The average is calculated for these 5 values of completion times. Each point in the following graph represents the average completion times of 5 values measured at that value of security strength.

Function to read energy consumption of the home

For each value of security strength, the completion time of the function to "read energy consumption of home" is calculated 5 times from 5 different simulations run. The average is calculated for these 5 values of completion times. Each point in the following graph represents the average completion times of 5 values measured at that value of security strength.

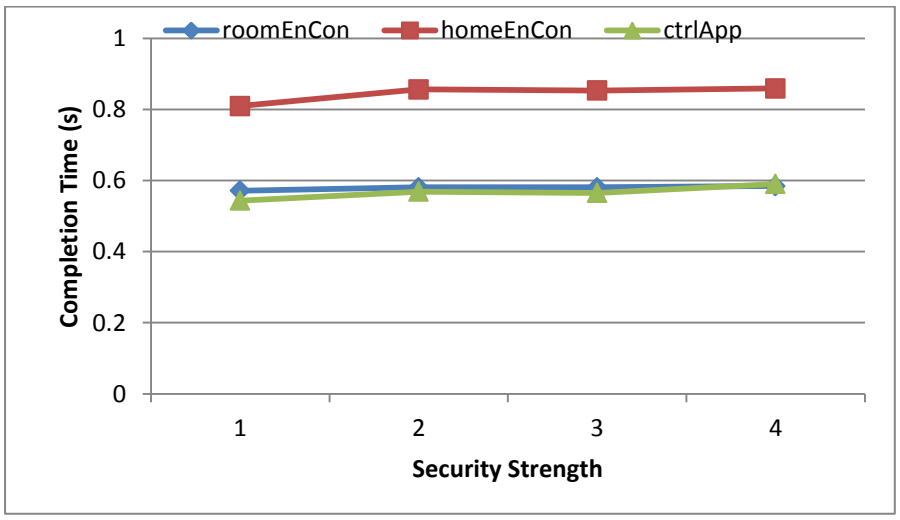

Figure 6.9: Completion time of operations that uses XMPP.

6.5.2.4 Analysis

We can deduce from the above graphs (Figures 6.7 to 6.9) that the completion time of all operations increases as the strength of security increases. Furthermore, the change in completion time of operations that uses XMPP (i.e. Figures 6.9) is different from other operations that do not use XMPP (i.e. Figures 6.7, 6.8).

Figure 6.10 is used to illustrate the difference in the way completion time changes with security for operations that uses XMPP in comparison to the operations that do not use XMPP.

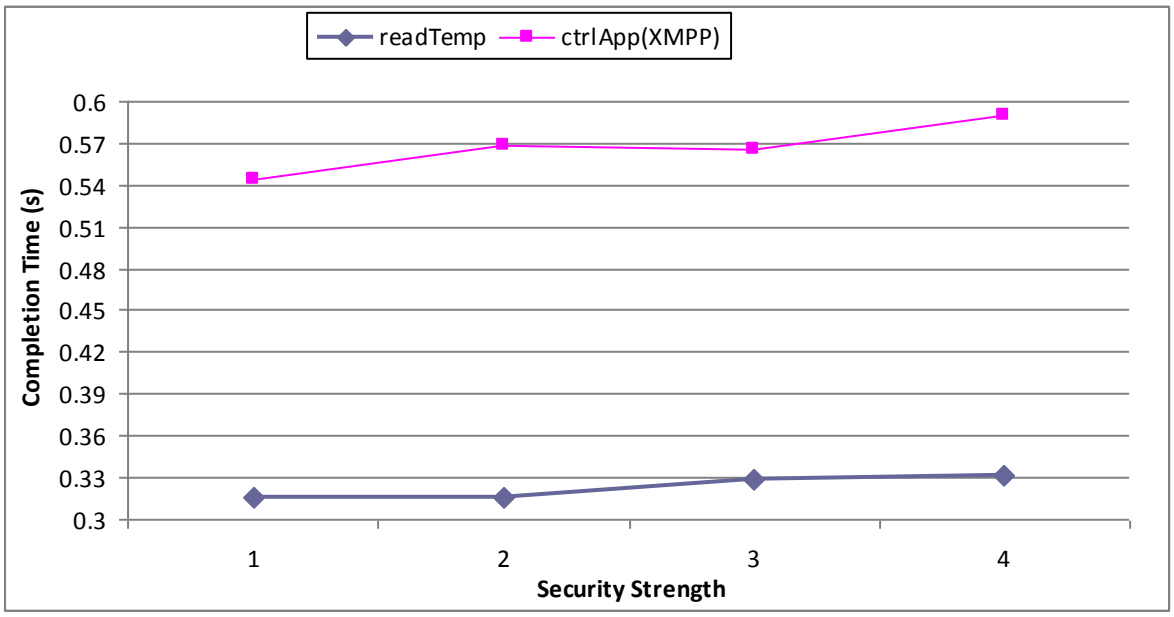

Figure 6.10: Comparison between functions that use XMPP and do not use XMPP

Figure 6.10 shows the completion time of "read temperature" operation and "control appliance" operation as the strength of security increases. XMPP is not used in the "read temperature" operation as the Web service just reads the temperature from the thermostat. The operation to "control appliance" uses XMPP as Web service communicates with the appliance via XMPP. We can see from Figure 6.10 that the completion time of the "read temperature" operation do not change as the security strength changes from 1 to 2. This is because the Web service is unsecured in both cases and the change in security of XMPP has no effect on this operation. The completion time of the "read temperature" operation increases as the security strength changes from 2 to 3. This is because the Web service is made secure when the security strength changes to 3. Furthermore, the completion time of the "read temperature" operation does not change as the security strength changes from 3 to 4. This is because the Web service is secured in both cases and the change in security of XMPP has no effect on this operation. On the contrary, we can see from Figure 6.10 that the completion time of the "control appliance" operation increases as the

security strength changes from 1 to 2. This is because XMPP is made secured when security strength changes to 2. The completion time of the "control appliance" operation does not change as the security strength changes from 2 to 3. This is because there is an increase in time to invoke the Web service when it is made secure. However, it takes less time to communicate with appliances as XMPP is made unsecure. So, overall the completion time doesn't change. Furthermore, the completion time of the "control appliance" operation increases as the security strength changes from 3 to 4. This is because XMPP is again made secure when the security strength changes to 4.

The changes in completion time (with respect to security strength) of operations that do not use XMPP are illustrated in Figures 6.7 and 6.8 and they follow the same pattern as the "Read Temperature" operation of Figure 6.10. These operations are "Read Temperature", "Read Light Intensity", "Adjust Light Intensity", "Sell Energy" and "Minimize Energy". On the contrary, the change in completion time (with respect to security strength) of operations that use XMPP follows the same pattern as the "Control Appliance" operation of Figure 6.10. The completion times of these operations are illustrated in Figures 6.9 and they are "Control Appliance", "Read Room Energy Consumption" and "Read Home Energy Consumption".

6.5.2.5 Overall completion time

In the previous section, the average completion time of an operation is calculated for each value of security strength. The overall completion time of an operation is the average of the 4 values calculated at 4 strengths of security. Figure 6.11 shows the overall completion time (in seconds) of all operations provided by the system.

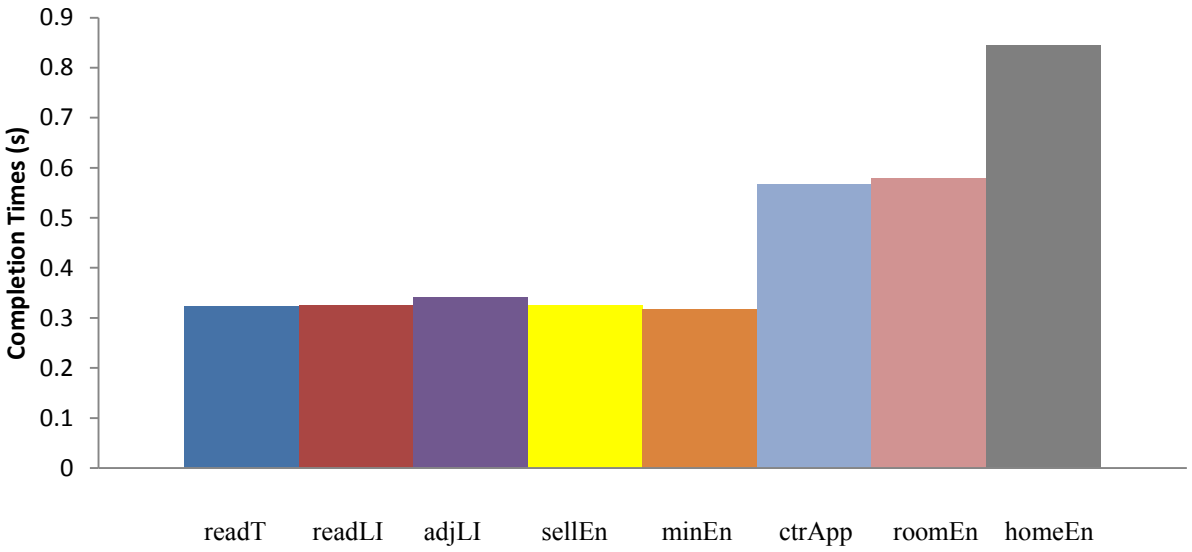

Figure 6.11: Overall Completion time of all operations.

The above bar chart shows the overall completion times of all operations provided by the system. The overall completion times of the operations with respect to security strength are similar to the overall completion times of the operations with respect to traffic intensity (Figure 6.4). From left to right, the bars shows the overall completion time of functions to "Read Temperature (readT)", "Read Light Intensity (readLI)", "Adjust Light Intensity (adjLI)", "Sell Energy (sellEn)", "Minimize Energy (minEn)", "Control Appliance (ctrApp)", "Read Room Energy Consumption (roomEn)" and "Read Home Energy Consumption (homeEn)" respectively. We can see from the above graph that the completion time of operations (i.e. "Control Appliance", "Read Room Energy Consumption", "Read Home Energy Consumption") that uses XMPP is more than other operations. This is because of the time it takes to communicate with the appliances in the room via XMPP. However, the operations that use XMPP have higher security as XMPP provides additional protection. The overall completion times of the operations that do not use XMPP are similar. Furthermore, Figure 6.11 shows that the average completion time of all operations is small as security strength is changed. Therefore, secure Web services can be used to perform the above mentioned operations efficiently at different security strengths (differentiated service).

6.5.3 Energy consumption and cost

A flat rate of 2.3 Canadian cents per kilowatt-hour is used to calculate the energy cost of the home. Energy costs arise from consuming energy. Consequently, costs also give a measure of energy consumption of the house. The energy cost of the home is calculated in summer and winter seasons.

6.5.3.1 Calculating cost of energy consumption of a home

The traffic intensity is increased in increments of 0.2 from 0.2 to 0.8. For each value of traffic intensity, the simulator is run 4 times. In each run, the simulation time is set as *4* days and the season is set as *summer*. For each value of traffic intensity, the total cost of energy consumption of a home is measured 4 times from 4 different simulations run. The average and CI is calculated for these 4 values of cost. Each point in the following graph represents the average completion times of 4 values (along with 95% confidence interval) measured at that traffic intensity.

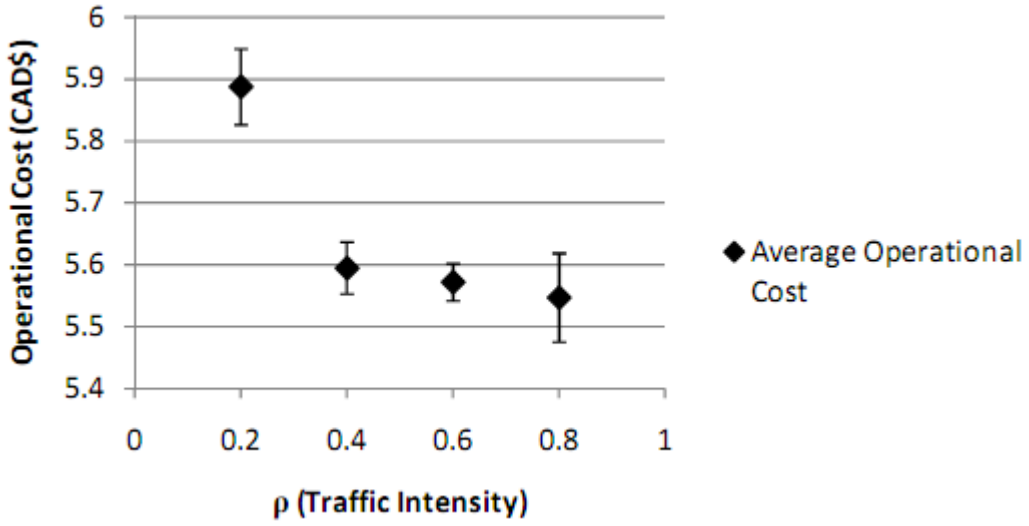

Figure 6.12: Average cost of home in summer [HEN10]

Again, the traffic intensity is increased in increments of 0.2 from 0.2 to 0.8. For each value of traffic intensity, the simulator is run 4 times. In each run, the simulation time is set as *4* days and the season is set as *winter*. For each value of traffic intensity, the total energy cost of the whole

home is measured 4 times from 4 different simulations run. The average and CI is calculated for these 4 values of cost. Each point in the following graph represents the average completion times of 4 values (along with 95% confidence interval) measured at that traffic intensity.

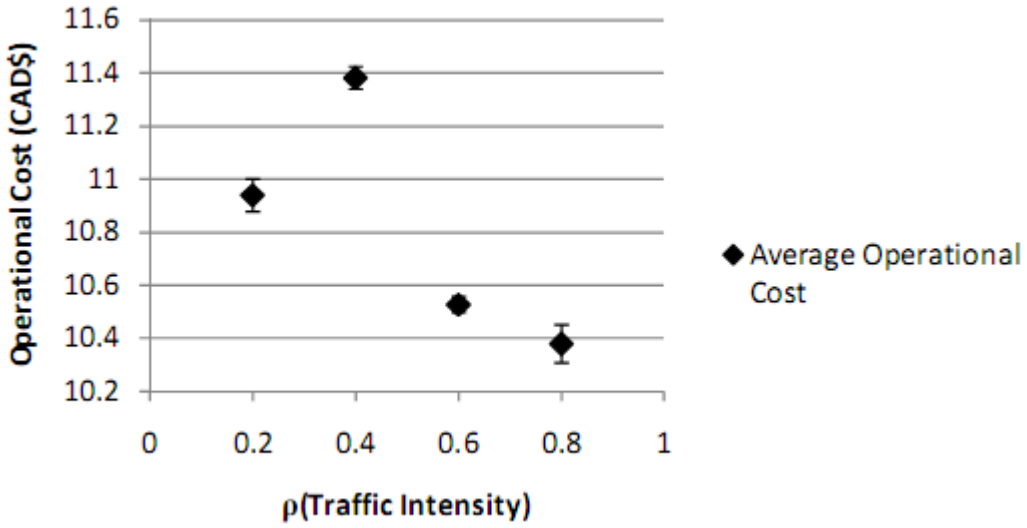

Figure 6.13: Average cost of home in winter [HEN10]

The 95% confidence interval ranges from 0.18% to 2.33% of the average cost at every value of traffic intensity. Therefore, the correctness of the smart home simulator is verified as the confidence interval values are small.

6.5.3.2 Comparing energy cost of the home with another simulator

The smart home simulator has a "temperature control algorithm" to automatically control room temperature. The thermostat periodically receives the current temperature of every room from sensors. If the current temperature of a room is outside the desired range, the thermostat controls the HVAC in the room to adjust the temperature. This keeps the temperature at a comfortable level. Furthermore, if the temperature is within the desired range, then HVAC does not need to run. Therefore, energy consumption and cost is also reduced.

In the smart home simulator of our system, HVAC regulates each room separately. A well-known simulator, called DesignBuilder [DES10], is used to simulate a typical home that has no

"temperature control algorithm". The DesignBuilder simulator has a centralized HVAC which controls the temperature of all the rooms together. In order to verify and prove the superiority of the smart home simulator in our system, the simulator results are compared with the DesignBuilder simulator. The environment in both simulations is kept exactly alike in order to make a fair comparison. Both simulators are run 4 times. Furthermore, both the simulators are run for 4 days in summer and total energy consumption is calculated. A flat rate of 2.3 Canadian cents per kilowatt-hour is used to calculate the costs. Figure 6.14 shows the total energy cost of the simulated homes in summer:

Figure 6.14: Comparison between operational costs of homes (four summer days)

Similarly, the simulators are again compared for winter performance. Figure 6.15 shows the total energy cost of the simulated homes in the winter:

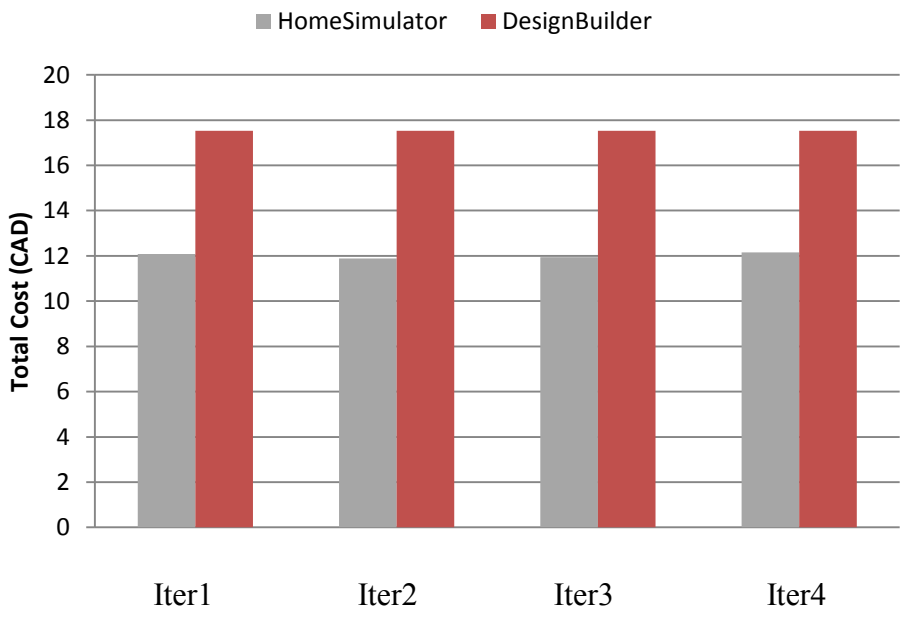

Figure 6.15: Comparison between operational costs of homes (four winter days)

The cost is proportional to the energy consumption. The above table quantitatively proves that the smart home simulator of our system consumes less energy than the DesignBuilder simulator both in summer and winter. Therefore, significant energy and monetary savings has been achieved by using the "temperature control algorithm" and operating HVAC separately for each room.

6.6 Summary

This chapter presents the simulation results. The simulation results are used to evaluate the system. The software used to develop the simulated system is specified. Furthermore, the chapter describes the way in which the simulator is run and the variables that change in each run. Then, the simulation results and the way results are obtained are presented. Graphs are used to illustrate the results. It is shown that the completion time of each function is small. Moreover, it is shown that differentiated service can be applied by changing strengths of security. It is also shown that there is significant energy and monetary savings. Furthermore, the results are analyzed and verified. The results are verified by showing that the confidence intervals (95%) for the results

are small. Moreover, the results are verified by comparing the results with a well-known home simulator, called DesignBuilder [DES10] and systems implemented in [YAZ09] [SLE11]. The comparison shows that the simulator used in our system performs significantly better than the DesignBuilder simulator both in summer and winter. Furthermore, our system performs better than the system in [YAZ09] and [SLE11]. Through simulation and analytical study, it is shown that secure Web service can be used to efficiently and remotely control different smart home elements for energy management.

Chapter 7

Energy Management in Intelligent Transportation Systems within a Smart Grid Environment

7.1 Introduction

This chapter describes a broadcasting algorithm that can be used by an electric vehicle to find a suitable charging station in Intelligent Transportation Systems (ITS). Furthermore, this chapter provides some examples of applying this algorithm in both static and mobile networks. In addition, it contains simulation results to demonstrate the reliability of the algorithm in finding a charging station. Finally, a summary of the chapter is provided in Section 7.3.

7.2 Locating most suitable charging station for PEV

Broadcasting is the task of sending a particular message from the source node to all other nodes in the network. Each node also acts as a router and therefore retransmits the packet received from the source. A connected dominating set (CDS) is defined as a set of nodes. In a connected network, the CDS is calculated in such a way that each node in the network either belongs to the CDS itself or has at least one neighboring node that belongs to CDS. The calculation of the CDS itself has negligible overhead as each node uses only local (i.e. neighborhood 2-hop) information. We present a broadcasting algorithm (based on [KHA08]) that can be used to broadcast messages from PEV to other PEVs or charging stations (CSt) in ITS. The algorithm is called PBSM (Parameterless Broadcasting from Static to Mobile networks). This particular algorithm is selected as it can broadcast messages with a low number of transmissions and good reliability (i.e. percentage of nodes receiving the message) in a mobile network. Initially, a PEV which

needs to charge its battery would broadcast a message. The message would contain the amount of charge left, information about battery (type, capacity, purchase date) and ID. A charging station which can charge or change that particular battery will send a reply back to the PEV. The reply will include the location of the charging station, cost of charging, type of location (like restaurant, shopping mall, hotel, parking lot), availability of battery (if replacing the battery is desired), availability of charge, source of energy (like wind, coal, nuclear) and operating hours. In this way, the PEV that needs to charge (or change) its battery can find the nearest, cheapest or most convenient charging station.

It is assumed that the PEVs and charging stations can wirelessly communicate with each other and among themselves, if they are within transmission range. Two PEVs (or charging stations) are called neighbors if they are within transmission range of each other. A charging station is connected to the grid and draws power from it. Furthermore, a charging station can detect the availability of charge that is based on the load of the grid and the transformer capacity [ERO11b]. Consequently, the resilience of the grid is increased. The type of location is useful if the PEV owner wants to eat or shop while the car is being charged. In addition, the source of energy is useful as it will enable the car owner to choose an environmental-friendly station.

7.2.1 Overview of broadcasting algorithm

PEVs (or charging stations) periodically exchange hello messages to update local knowledge up to two hops (direct neighbors and their neighbors). Hello messages from a PEV (or charging station) contain its ID and the list of its direct neighbors. In this way, each PEV (or charging station) gains knowledge of its 2-hop neighbors. CDS [CAR04] is calculated after each hello message round. PEV (i.e. source) initially transmits the message. Upon receiving the message for the first time, each neighboring PEV (or charging station) initializes two lists: receiver list R containing all PEVs (or charging stations) (up to 2-hop distance) believed to have received the packet, and list N containing neighbors in need of the message. A PEV (or charging station) set a timeout waiting period. If a PEV (or charging station) is not in CDS then it selects a longer timeout than a PEV (or charging station) from CDS so that PEVs (or charging station) in CDS react first. For each further message copy received, and its own message sent, every PEV (or

charging station) updates *R, N* and the timeout. At the end of the timeout period it transmits if *N* is nonempty. The message is memorized until *T* hello messages are received. For each hello message received, *N* is updated. PEVs (or charging stations) that are no longer 1-hop neighbors are eliminated from the list, while new neighbors, not present in *R,* are added. Regardless of previous decisions, all PEVs (or charging stations) that so far received the broadcast packet check whether new *N* is nonempty. If so, they start a fresh timeout. PEVs (or charging stations) not in CDS also run timeouts if their *N* lists remain nonempty.

In static networks, the number of transmissions is reduced as only the CDS nodes transmit. Furthermore, some nodes may not need to transmit due to neighbor elimination scheme [PEN00, STO02]. In mobile networks, two nodes discover each other when one node enters the transmission region of the other. Unlike other methods, in this algorithm, two nodes do not transmit every time they discover each other as new neighbors. When a node discovers a new neighbor, it first checks whether the new node is in *R* list. If it is not in *R*, only then does the node consider transmitting. Otherwise, it does not transmit. Consequently, this algorithm has a lower number of transmissions while maintaining good reliability in both static and mobile networks.

7.2.2 Details of broadcasting algorithm

7.2.2.1 Hello message round and Connected Dominating Set (CDS)

Hello messages take place periodically and have two rounds. In the first round, each PEV (or charging station) sends its ID to all of its current neighbors. Upon receiving a hello message, a PEV (or charging station) adds the hello message originator to its neighbor list (if the list does not already contain that PEV or charging station). After the first round, each PEV (or charging station) knows about all of its neighbors (1-hop). In the second round, each PEV (or charging station) transmits the list of IDs of the PEVs (or charging station) from whom the hello messages were just received in the first round. In this way, each PEV (or charging station) learns about its current 2-hop neighbors. Each PEV (or charging station) maintains an *N* list (neighbors who didn't receive messages) and *R* list (PEVs or charging station that have received a message).

After every periodic hello message, the connected dominating set (CDS) is calculated in the following way [CAR04]. Each PEV (or charging station) compares the degree (size of the neighbor list) of its neighbors with its own. The PEV (or charging station) *b*, with a smaller degree, will consider the PEV (or charging station) *a*, with a bigger degree, as a higher ID neighbor. If the degree of two PEVs (or charging stations) are equal, then the PEV (or charging station) with the higher index (i.e. number or letter that is used to identify a PEV or charging station) is considered as a higher ID neighbor. A PEV (or charging station) is an *intermediate* PEV (or charging station) if it has two unconnected neighbors. First, each PEV (or charging station) checks whether it is an intermediate PEV (or charging station). Then each intermediate PEV (or charging station), **A**, determines whether it is in the dominating set in the following way:

Construct subgraph G, consisting of only higher ***Id*** neighbors and existing edges between them.

- If G is empty or disconnected then ***A*** is in the CDS (G is connected if it has only one node).
- If G is connected but there exists any neighbor of ***A*** (not in G) that is not a neighbor of any node from G, then ***A*** is in the CDS.
- Otherwise A is covered and not in the CDS.
- Dijkstra's shortest path algorithm is used to test the connectivity of the subgraph of G.

Consequently, a PEV (or charging station) with a higher degree or index is more likely to be in the dominating set.

7.2.2.2 Broadcasting process

Initially, only the PEV has the message to be transmitted. The hello message takes place and each intermediate PEV (or charging station) determines whether it is in the dominating set or not. After this, the PEV (i.e. source) transmits. The PEVs (or charging stations) react to two events:

a) Receiving and retransmitting broadcast messages between two hello messages:

The following steps are repeated until the next hello message.

- [For any CSt / PEV (CDS or non-CDS)]. If the message is received for the first time, the PEV (or CSt) initializes its *R* and *N* lists as follows. All reported (at the last hello message time) 1-hop neighbors of the sender, and the sender itself, are first included in *R*, the receiver list. Therefore, the *R* list of receivers, consists of PEVs (or CSt) that have received the message (up to 2-hop) based on the available and current local knowledge of the PEV (or CSt). However, when the sender is not in the receiver's current neighbor list, the receiver adds only the sender to its *R* list. This is because the receiver doesn't know the neighbor list of the sender. After that, the *N* (neighbors in need of the message) list will only consist of the 1-hop neighbors of the receivers that are not on the *R* list.
- If the message is received for the first time and list *N* is non-empty, the PEV (or CSt) starts a timeout. If the PEV (or CSt) is in the CDS, it sets a timeout where Timeout = 1/ |N| (|N| is the number of PEVs or CSt in list *N*). If the PEV (or CSt) is not in the CDS, it selects a longer timeout than a PEV (or CSt) from the CDS so that the PEVs (or CSt) in the CDS react first. For non-CDS-PEV (or CSt), Timeout = 5 + 1/ |N|. Alternately, a random timeout can be used.
- [For any CSt or PEV (CDS or non-CDS)]. Whenever a PEV (or CSt) receives a message (not just for the first time), it keeps updating the lists for *N* and *R* which might be needed after the next hello message round. If a PEV (or CSt) is running timeout:
  For a CDS-PEV (or CSt), Timeout = 1/ |N| - any elapsed time.
  For a non-CDS-PEV (or CSt), Timeout = 5 + 1/ |N| - any elapsed time.
  If *N* becomes empty, it cancels retransmission (timeout).
- If two PEVs (or CSt) have the same timeout, then the timeout of the PEV (or CSt) with the smaller index (number or letter used to identify the PEV or CSt) will expire first. When the timeout expires, if there are still neighbors left that are believed not to have received the message (*N* is non-empty), the PEV (or CSt) retransmits. All of its neighbors are included in *R* and all neighbors are now removed from *N*. (*N* is now empty).

b) Sending/receiving hello messages and deciding to broadcast related updates

The following steps take place during a hello message

- A new round of hello messages is sent, and each PEV (or charging station) updates its neighbor lists *N*.

- When a PEV (or charging station) receives a hello message from a neighbor, the sender is checked to see whether it is in the accumulated *R* list. If so, the PEV (or charging station) ignores it for the purpose of transmitting. If not, the PEV (or charging station) then adds it to *N*.

- All PEVs (or charging station) that have already received broadcast packets at any time consider possible new, or very first transmissions by reevaluating decisions (including ones that have previously decided not to transmit (non-CDS), ones that have already retransmitted, and ones running timeout, with nonempty *N* list) as follows.
  - Each charging station or PEV (CDS or non-CDS) that was not running timeout but now has a non-empty *N* list, starts timeout (1/|N|) fresh from the hello message point.
  - Each charging station or PEV (CDS or non-CDS) that was running timeout but now has an empty *N* list, cancels timeout.
  - Each charging station or PEV (CDS or non-CDS) that was already running timeout sets its new timeout as follows:

    New Timeout = 1 / |N| - elapsed time since start of previous timeout

After the hello message, each intermediate PEV (or charging station) again determines whether it is in the dominating set using the updated neighbor list.

The whole process (a and b) is repeated until a fixed number *T* of hello messages occurs after the initial transmission by PEV. *T* is a parameter that determines the lifetime of the message being broadcasted.

Pseudo-Code of the broadcasting process

The following notations are used in pseudo-code:

- Ne (x) means the set containing all 1-hop neighbors of PEV (or charging station) x
- N_x means the N list (neighbors who didn't receive message) of PEV (or CSt) x
- R_x means R list (PEVs or CSt that have received message) of PEV (or CSt) x
- m means data message, h means hello message, # h means no. of hello msg.
- T_x means the timeout of PEV (or CSt) x, CDS means Connected Dominating Set
- eT_x means the elapsed time if PEV (or CSt) x is already running timeout

Do

Repeat

On receiving data message (m) by PEV (or CSt) **x from PEV (or** CSt) **y:**

$R_x \leftarrow R_x + y + Ne(y)$

$N_x \leftarrow Ne(x) - R_x$

If $T_x = 0$ (timeout not running)

$eT_x = 0$

If (x receives msg for first time) or ($T_x \neq 0$)

If (x ε CDS)

$T_x \leftarrow 1 / N_x - eT_x$ ($T_x = 0$ if N_x is empty)

Else

$T_x \leftarrow 5 + 1 / N_x - eT_x$ ($T_x = 0$ if N_x is empty)

On expiration of T_x

If $N_x \neq \emptyset$

x transmits data_msg (m)

$R_x \leftarrow R_x + Ne(x)$

$N_x \leftarrow Ne(x) - R_x$ ($N_x = \emptyset$)

Until next hello message time

At HELLO message time (periodic time)

$\#h \leftarrow \#h + 1$ (number of hello messages)

On receiving hello message (h) by PEV (or CSt) x from PEV (or CSt) **y :**

If y is not in R_x

$N_x \leftarrow N_x + y$

If PEV (or Cst) x has data message (m)

$T_x \leftarrow 1 / N_x - eT_x$ ($T_x = 0$ if N_x is empty)

If ($T_x < 0$)

$T_x \leftarrow 1 / N_x$

After hello message, CDS is again calculated.

While (#h = *T*)

7.2.3 Example showing the operation of algorithm

7.2.3.1 Static Network

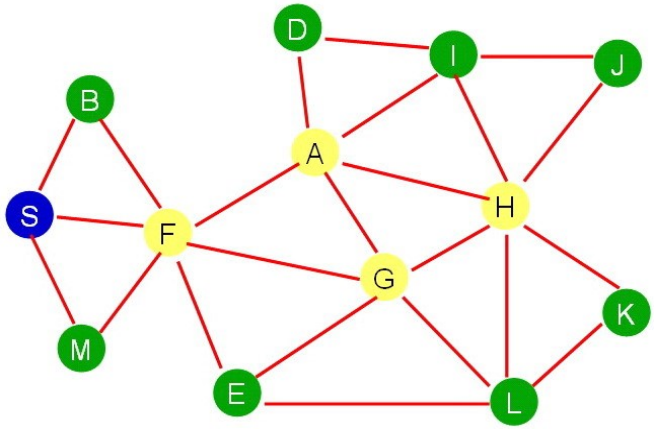

Figure 7.1: Source S and CDS nodes A, F, G, and H in a Static Connected Network

Consider first the behavior of the algorithm in a static vehicular network (see Figure 7.1). Each node in Figure 7.1 and Figure 7.2 is either a PEV or charging station. PEVs (or charging stations) B, M, D, J and K are non-intermediate. The remaining PEVs (or charging stations) in the graph are intermediate. All intermediate PEVs (or charging stations) decide whether or not they are in the CDS. PEVs (or charging station) A, F, G, and H are in the dominating set. PEV S is the source and it transmits. Among the neighbors of S, only PEV (or charging station) F is in the CDS and thus it will retransmit first. F runs a timeout (= 1 / |N|). Figure 7.2(a) shows the timeout at F and the *R*, *N* lists of PEVs (or charging station) B and F. The 2-hop neighbors in the R list are shown in bold.

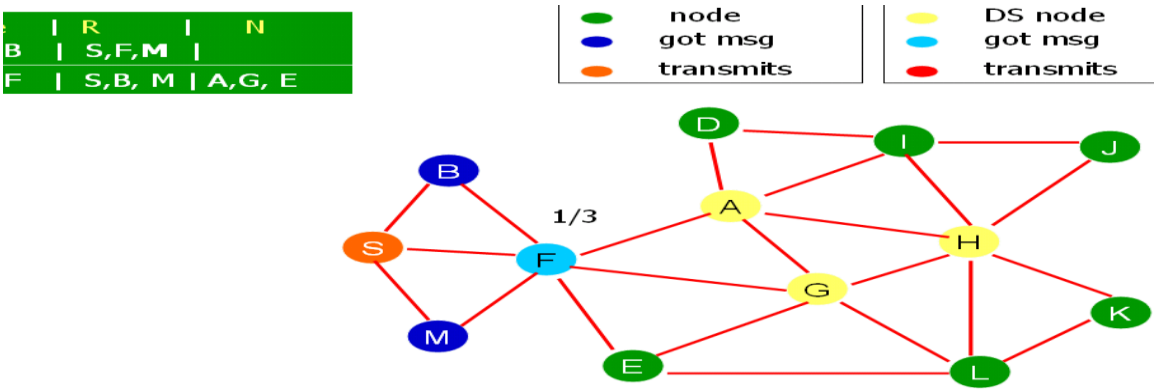

Figure 7.2: (a) Status of PEVs (or charging station) after S transmits.

F transmits. Neighbors A, G, E of F receive the message for the first time. As A and G are in the CDS, they run timeout proportional to their number of uncovered neighbors, 1/3 (neighbors D, I and H of A) and 1/2 (neighbors H and L of G), respectively. As the PEV (or charging station) E is not in CDS, it runs a longer timeout 6 (i.e. 5 + 1/1, one uncovered neighbor L). Figure 7.2(b) shows the *R*, *N* list and timeout after F transmits.

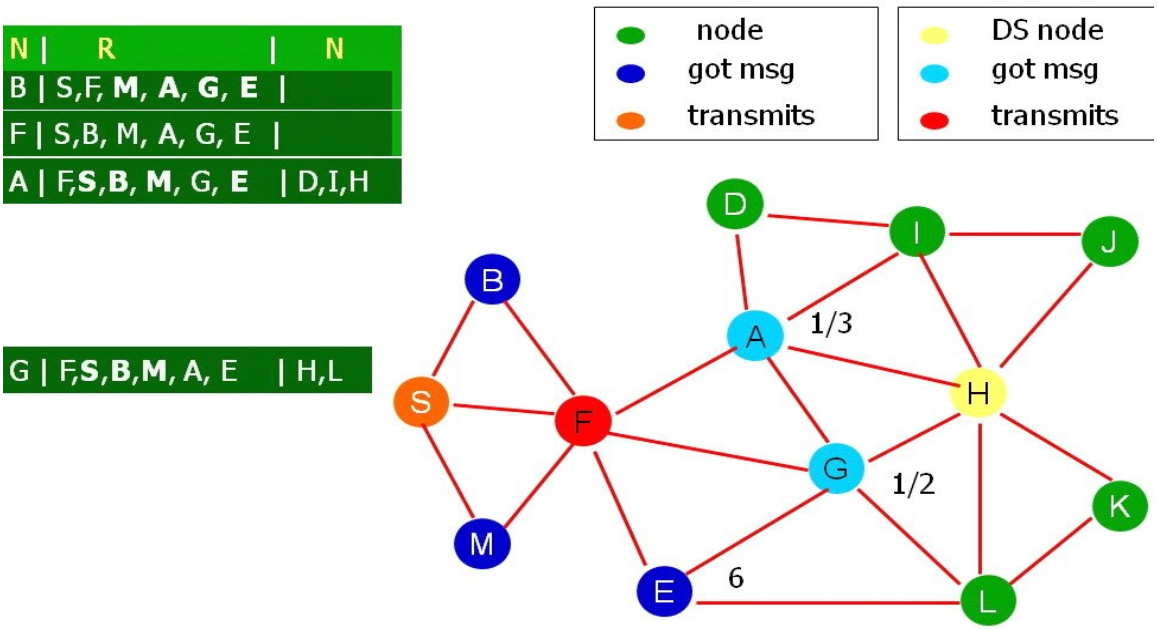

Figure 7.2: (b) Status of PEVs (or charging station) after F transmits.

Since A has a smaller timeout, A transmits. Among the neighbors of A, only F, G and H are in CDS. F has already transmitted. At this moment G extends its timeout to 1/1-1/3 (one uncovered neighbor L left, 1/3 time elapsed). H receives the message for the first time and sets its timeout in proportion to 1/3 (neighbors I, J, K). E reduces its timeout to 6-1/3 (1/3 time elapsed). PEV (or charging station) I (non-CDS) sets a longer timeout 6 (i.e. 5 + 1/1, one uncovered neighbor J). Figure 7.2(c) shows the *R*, *N* list and timeout after A transmits. Only B's *R* list doesn't change.

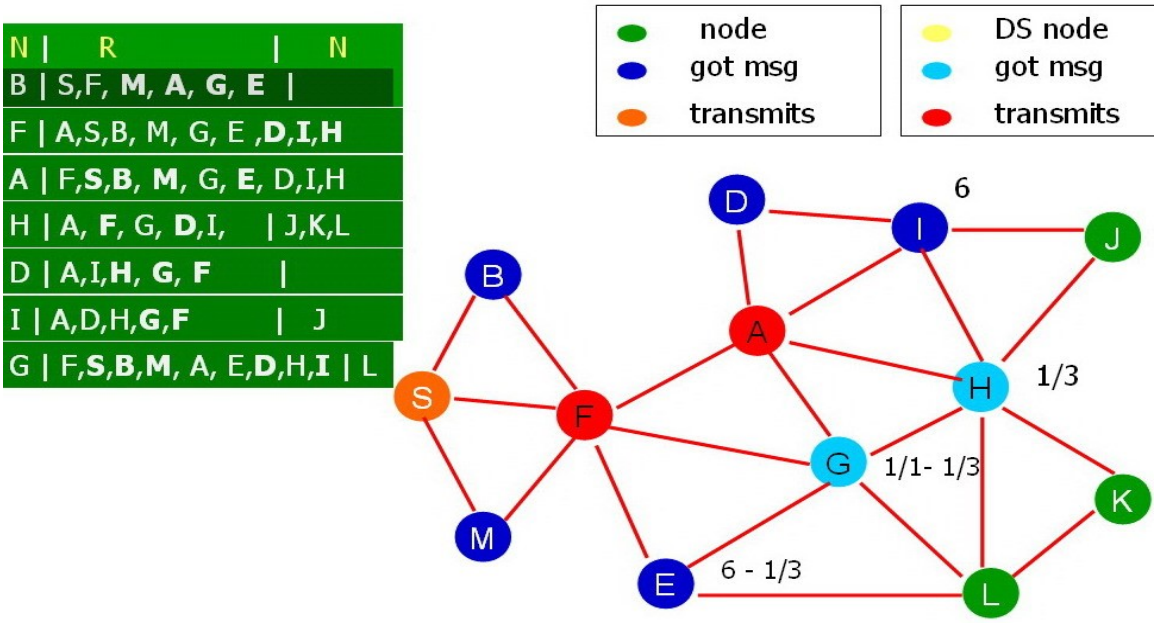

Figure 7.2: (c) Status of PEVs (or charging station) after A transmits.

Since H has a smaller timeout than G, H transmits. Among the neighbors of H, only G and A are in the CDS. A has already transmitted. G and I's *N* lists now become empty and therefore their timeouts stop. E further reduces its timeout to 6-1/3-1/3 (1/3 more time elapsed). PEV (or charging station) L (non-CDS) sets a longer timeout 6 (i.e. 5 + 1/1, one uncovered neighbor E). Figure 7.2(d) shows the *R* and *N* list after H transmits. B, F and D's *R* lists do not change.

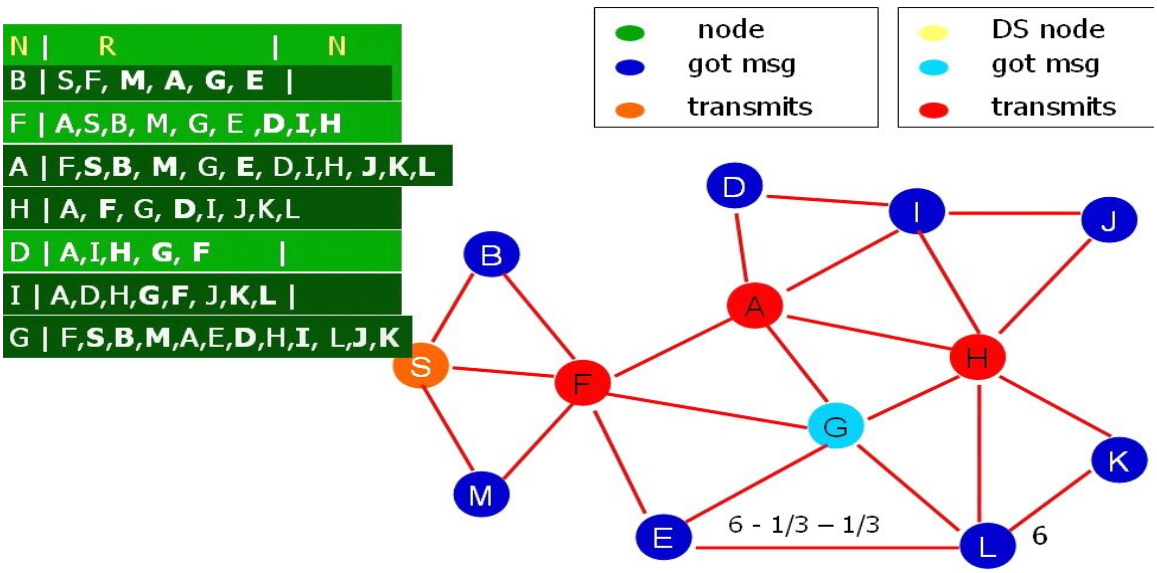

Figure 7.2: (d) Status of PEVs (or charging station) after H transmits.

Since E has a smaller timeout than L, E transmits. Among the neighbors of E, only G and F are in CDS. F has already transmitted and G's *N* list is empty. L's *N* list now becomes empty and therefore its timeout stops. Figure 7.2(e) shows the *R* and *N* list after E transmits.

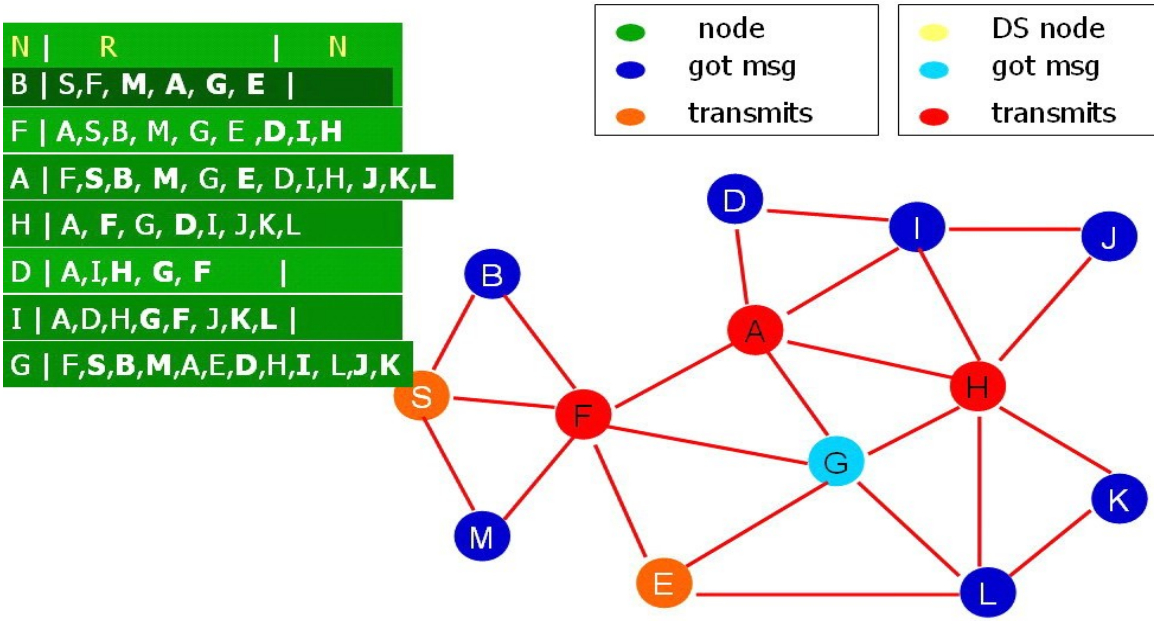

Figure 7.2: (e) Status of PEVs (or charging station) after E transmits.

The proposed method broadcasts based on CDS and neighbor elimination scheme. PEVs (or charging station) in CDS are selected in such a way that if only the PEVs (or charging station) in CDS transmit, all PEVs (or charging station) in the static network will receive the message. Only four charging stations or PEVs (A, F, G, and H) are in the CDS. Although PEV (or charging station) G is in CDS and initially decides to transmit (i.e. runs timeout), it does not transmit because all of its neighbors received the message. On the contrary, the non-CDS PEV (or charging station) E transmits as its *N* list is non-empty according to its local knowledge. Consequently, all of the PEVs (or charging station) in the network receive the message via the retransmissions of only four PEVs (or charging station). In a static network, the number of retransmissions is at most equal to the number of CDS PEVs (or charging station). Suppose, node A and node I are charging stations. Charging stations A and I can send a reply back to the source PEV S. If PEV S is looking for the nearest charging station, then it would select A.

7.2.3.2 Mobile Network

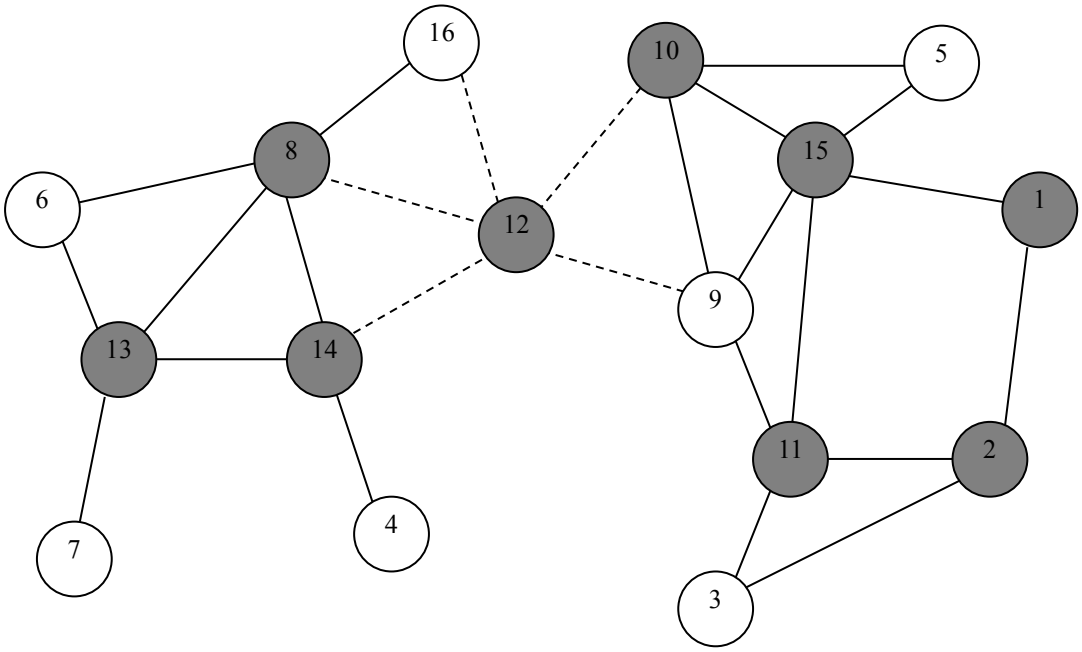

Figure 7.3: PEV joining two disconnected network

The broadcasting algorithm is applied in Figure 7.3. Initially, the graph was divided into two disconnected components as PEV 12 was not there. PEV (Node 14) is the source and it transmits. All PEVs on the left side of PEV 12 receive the message because they are directly or indirectly connected to the source. There is no charging station on the left side of PEV 12. PEVs (or charging station) on the right side of PEV 12 do not receive the message. After sometime, PEV 12 arrives closer to source and other charging stations or PEVs (as shown above). PEV 12 does not have the message. Hello message takes place and PEV 12 is discovered by its neighbors. PEVs 8, 16, and source 14 decided to transmit because their N list contains 12, and they run timeout (1/1). Then CDS is calculated. PEVs (or charging station) 1, 2, 8, 10, 11, 12, 13, 14, and 15 are in the CDS. PEV 8 transmits first. When PEV 14 and PEV 16 receive the message, they remove 12 (12 is a neighbor of sender 8) from their *N* lists. PEV 14 and PEV 16 cancel their transmission as their *N* lists becomes empty. When PEV 12 receives the message from PEV 8, it removes 8, 16, and 14 (8 is the sender and 14 and 16 are its neighbors) from its *N* list. PEV 12 is in CDS and it receives the message for the first time; therefore it runs timeout. The timeout of PEV 12 is ½ as it has 2 neighbors (9, 10) left in its *N* list. Please note that the timeout of a CDS-

PEV = 1 / |N| whereas the timeout of a non-CDS PEV = 5 + 1 / |N| (longer timeout). The left column of the following table shows the PEVs (or charging station) that transmit (in order) and the corresponding right column shows the PEVs (or charging station) whose timeout changes due to the transmission (PEVs (or charging station) in CDS are shown in bold).

Table 7.1: Transmission order and timeout of PEV (or charging station)

Transmission	**PEV** (or charging station)**-Timeout**
8	**12** - ½; **14** - cancels timeout; 16 - cancels timeout
12	9-**(**5+½); **10** - ½
10	9-**(**5+1/1-½);**15** - ½
15	**1** - 1/1; 9 - cancels timeout ;**11** – ½
11	**1** – (1/1-1/2); **2** – 1/1
1	**2** – cancels timeout

The CDS is recalculated after PEV 12 is discovered during the hello message. The CDS-PEVs (or charging station) will transmit first since they have a shorter timeout. All PEVs (or charging station) will receive the message by the retransmission of only CDS PEVs (or charging station). Suppose, node 9 and node 10 are charging stations whereas station 9 has the cheapest cost. Charging stations 9 and 10 can send a reply back to the source PEV 14. If PEV 14 is looking for the economical charging station, then it would select station 9.

7.2.4 Simulation

7.2.4.1 Comparison with other broadcasting protocols for mobile networks

The authors in [VIS02] put forward an adaptive flooding method in which nodes, based on network conditions, can dynamically switch between different flooding techniques. This method is called *vo*. The types of flooding techniques are *plain flooding, scoped flooding* and *hyperflooding*. In *plain flooding*, when a node receives a message for the first time, it retransmits. When a node receives a copy of the same message, it doesn't retransmit. In *scoped flooding,*

when a node receives a message for the first time, it compares the neighbor list of the transmitting node to its own neighbor list. If the receiving node's neighbor list is a subset of the transmitting node's neighbor list, then it does not retransmit the packet. Otherwise, it retransmits. *Hyperflooding* is like plain flooding. The only difference is that when a node receives a data packet or hello message from a node that is not in the current neighbor list, it retransmits even though it has transmitted before. Switching between flooding protocols takes place in two ways. One way is to use *network load* (collisions in Media Access Control layer) and the other way is to use *relative velocity*. The relative-velocity based switching criterion works as follows. Nodes send velocity (speed and direction) information as part of their hello messages. Each node is then able to compute its velocity relative to all of its immediate neighbors. Each node maintains a running average as well as the minimum and maximum value of relative velocity for the past five time windows. Based on the current value of relative velocity and its past history, each node adaptively chooses a low threshold and high threshold value for the current time window. If the current value of relative velocity is higher than the high threshold, the node switches to hyperflooding mode. If the relative velocity is below the low threshold, then scoped flooding is used. Otherwise, the node switches to plain flooding. Thus, different nodes can make different decisions and run different protocol modes. At lower speeds, adaptive flooding switches to the scoped flooding mode in an attempt to reduce redundant retransmissions. As the relative velocity increases, it switches to hyperflooding mode, thus increasing reliability at the cost of additional retransmission.

An encounter based protocol is presented in [COO04]. This protocol is called *cem*. Upon receiving a new message, node u does three things. First, it stores it and sets counter c to 0. Second, it adds the sender to its neighbor list if it is not already in the list. Third, if the current neighborhood contains nodes other than the one sending the message, then u retransmits the message and increases c by 1. An encounter is the discovery of a new neighbor that is not present in previous lists of neighbors (thus a node that has been a neighbor and disappears for just one ‘hello’ cycle is still considered as encountered upon return). Nodes do not maintain a history of neighbor list, but rather experience encounters at intervals. At every encounter, if $c < T$ (T is a

parameter) then the node retransmits the message and increases counter *c* by 1. When *c*=*T*, the node deletes the message. The drawback is that it generates too many transmissions.

The protocols described in [VIS02] and [COO04] are selected for comparison as they consider similar assumptions and are developed for highly mobile scenarios.

7.2.4.2 Environment

The algorithms PBSM, *vo* and *cem* are implemented. The simulator is developed in Visual C++ programming language.

The simulated environment is developed based on the following assumptions:

- A unit disk graph.
- All nodes have an equal transmission radius.
- A one-to-all model for broadcasting is assumed. That is, when a node transmits, all of its neighbors will receive the message unless there is collision.
- There are no obstacles between any two nodes.

A unit disk graph is assumed. In this graph, all edges are less than or equal to a fixed threshold value. In a wireless network, a message sent by a node reaches all of its neighbors within transmission radius. If the transmission range of each node is *r*, then we can represent the wireless network by a unit disk graph in which all of its edges are at most *r*. An edge connects two nodes that are within communication range (i.e. distance between them is $\leq r$). To generate the random unit disk graph, values for *d* and *n* are chosen. *d* represents the desired average node degree of the graph and *n* is the number of nodes. All nodes are randomly generated in a square of side *a*. That is, its *x*- and *y*- coordinates are chosen at random between 0 and *a*. We chose *a*=470 as the computer's viewport is 640 x 480. The generated graph is disconnected most of the time.

The nodes move in a way similar to the random waypoint mobility model [BRO98]. The co-ordinates of their destinations are chosen at random. A node moves to its destination in a straight

line at random speeds between 0 and the maximum speed. Upon reaching the destination or turning point, the node pauses for some time (0 sec) and then selects a new random destination. Under the maximal speed, a node can travel distance *D'* between two hello messages. Let *D'=D*r,* where *r* is the transmission radius and *H* is the time between two hello messages. Therefore, the maximal speed is *D*r /H* (*=D'/H*). The average velocity is maximum speed/2. Consequently, speed and *D'* depend on *D* and *D* is used as a parameter to define mobility. The movement of nodes takes place in each time unit. Movement takes place either while broadcasting is in progress or when there is no broadcast. A movement can change the graph, and a broadcasting method acts with the changed graph. However, a node is not aware of the change until the next hello message time. Thus, some nodes may receive messages although their sender does not know about them (since the nodes have just arrived in the neighborhood), while some believed neighbor nodes may be gone. Few nodes (i.e. charging stations) are static whereas other nodes (i.e. PEV) are mobile.

A simplified IEEE 802.11 Media Access Control (MAC) protocol, based on *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA) scheme, has been implemented. In simplified MAC layer, a node can listen to the transmission from a neighboring node. The time taken to transmit a message from a node to its immediate neighbor is divided into *p* slots. Before transmitting, a node listens to the channel. The node will sense the channel busy if at least one of its neighbors is transmitting and it will sense the channel idle if none of its neighbors is transmitting. After receiving message, a node will wait a random amount of time (*W*). That is, a node will listen in each slot and waits for *W* (not necessarily consecutive) idle slots. After *W* such slots, the node starts transmission in the next slot. Once the node starts transmission, it continues transmitting for *p* next slots without listening to anything. A node will receive messages correctly if it receives unique messages for *p* consecutive slots from the same neighbor. Sending a broadcast message to an immediate neighbor requires fixed time *B'*. The time taken to transmit a message from a node to its immediate neighbor is taken as one unit (i.e., *B'* = 1 unit or 1 sec). *H* is the fixed periodic time between two hello messages. *B* is the number of messages that can be sent between two hello messages. That is, *B = H/B'*. *B*' is divided into 128 slots (i.e. *p*=128), each of duration 1/128. Thus, there are *B*p* slots between two hello messages. Timer is incremented

after p slots. The waiting timeout (W) is a random integer between 0 and 25 for all protocols. In PBSM, the non-CDS nodes add 50 to this timeout. Messages collide when two or more neighboring nodes start transmission simultaneously. Furthermore, messages collide when two or more messages arrive simultaneously at a receiving node. Reliability is reduced due to collision.

7.2.4.3 Simulation Results

The scenario consists of ten different simulations. Each simulation consists of a randomly generated graph and ten hello messages. In each run (simulation), the hello message takes place first. After that, all the broadcasting methods are called. Broadcasting starts at time 0 with the transmission from the source. Hello messages take place at fixed intervals and movements can take place at any time. For each simulation and for each type of flooding, the following metrics are calculated in order to evaluate the method's performances.

$$\textit{Total number of transmissions per node} = \frac{\text{total number of transmission by all node}}{\text{number of node}} \quad (7.1)$$

$$\textit{\% Reliability} = \frac{\text{number of nodes that received message}}{\text{total number of nodes that could have received message}} \times 100 \quad (7.2)$$

In each simulation, the number of nodes (n) is 50 and D (a parameter used to define the speed of movement) is 0.75. In order to obtain a smooth curve, the average of 5 runs (where all parameters are constant) is used to gather each data. A node represents either a PEV or charging station. Furthermore, H (periodic time between two hello messages) is 10. Density (d) indicates the desired average node degree and is incremented by 1 in each simulation. The average node degree of the generated graph is close to density (d).

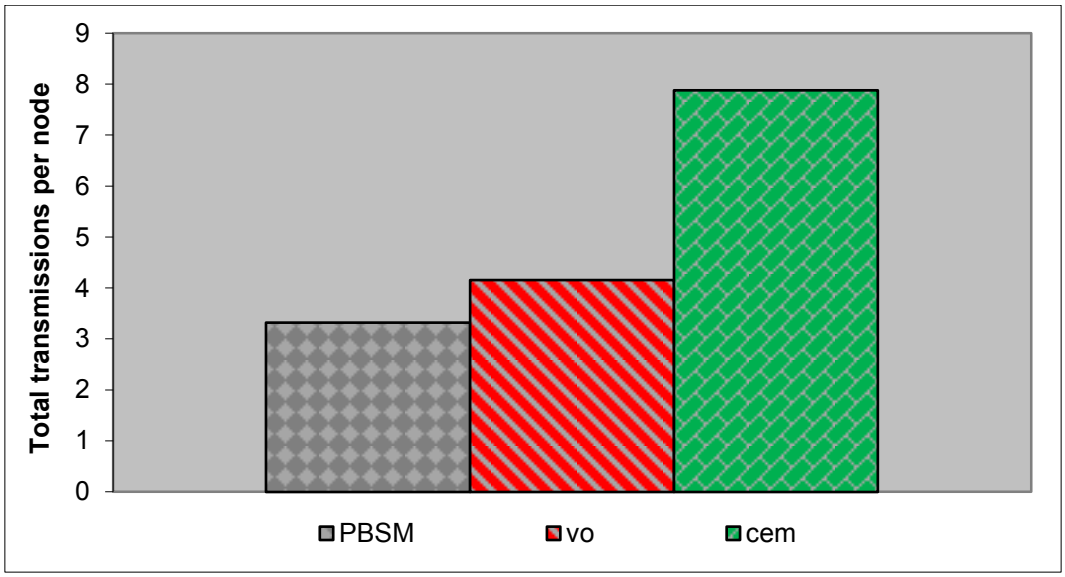

Figure 7.4: Total number of transmissions per node

The above bar graph in Figure 7.4 shows the total transmissions per node (y-axis) against three different flooding techniques (x-axis). The bar graph shows that the total number of transmissions per node for the PBSM method is less than other methods. There are four main reasons for this difference. First, most nodes in the network receive the message via the transmissions of only CDS nodes, and therefore few non-CDS nodes (which run a longer timeout) need to transmit. In the broadcasting methods described in [VIS02] and [COO04], every node considers retransmission when it receives a message for the first time. Second, as PBSM uses neighbor elimination, a node that initially decides to transmit might not transmit if its *N* list becomes empty. Third, the proposed method does not consider broadcasting each time two nodes discover each other as new neighbors, which other approaches do. Each node maintains a receiver list (R) containing nodes that have received the message (up to 2-hops). When a node discovers a new neighbor, it checks whether the new node is in R. If it is not, then the node considers broadcasting. Otherwise, it does not broadcast. Finally, unlike the hyperflooding-based methods described in [VIS02], there is no additional transmission when a node receives a message from a new node that is not in its current neighbor list.

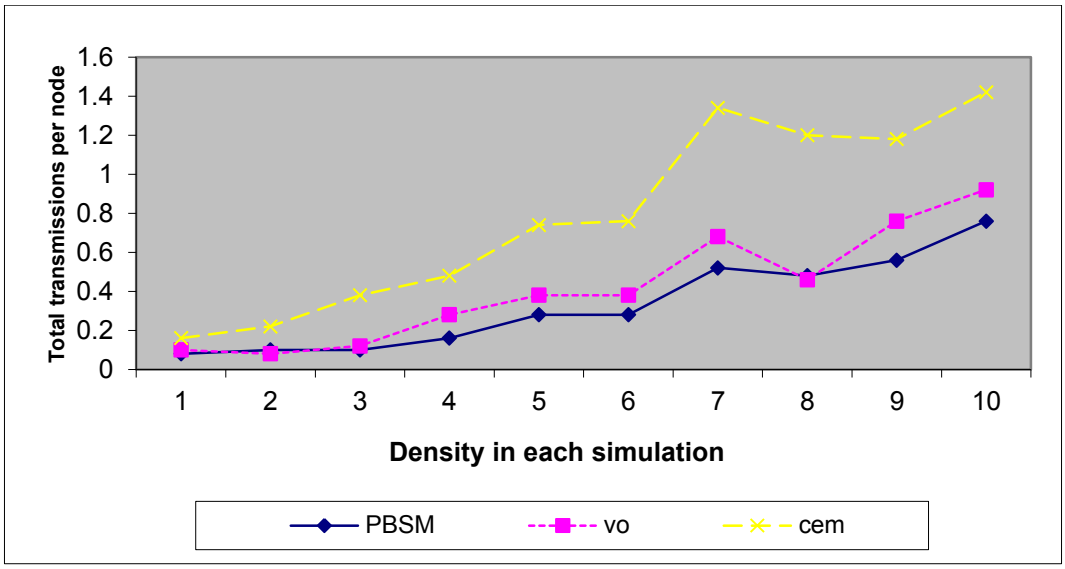

Figure 7.5: Total number of transmissions per node as density increases

The line graph in Figure 7.5 shows the total transmissions per node (y-axis) in each simulation (x-axis). The numbers on the x-axis indicate the value of *d* used in each simulation. As *d* increases, the graph becomes denser, which means that more nodes will receive a message from a single transmission. Therefore, more nodes will decide to retransmit and the number of transmissions of all methods increases as *d* increases. As *d* increases, the percentage of CDS nodes compared to all nodes decreases. Furthermore, in a dense network there is a greater possibility that a new node discovered was previously a 2-hop neighbor. Therefore, if a new node has previously received the message, it will be in the *R* list. Unlike other methods, there will be no additional transmissions in PBSM. Consequently, the difference in the number of transmissions between PBSM and other methods increases as *d* increases. Considering Figure 7.5, the 95% confidence interval for PBSM, *vo* and *cem* are 0.15, 0.18 and 0.29 respectively. The confidence intervals are not shown on Figure 7.5 as they are small.

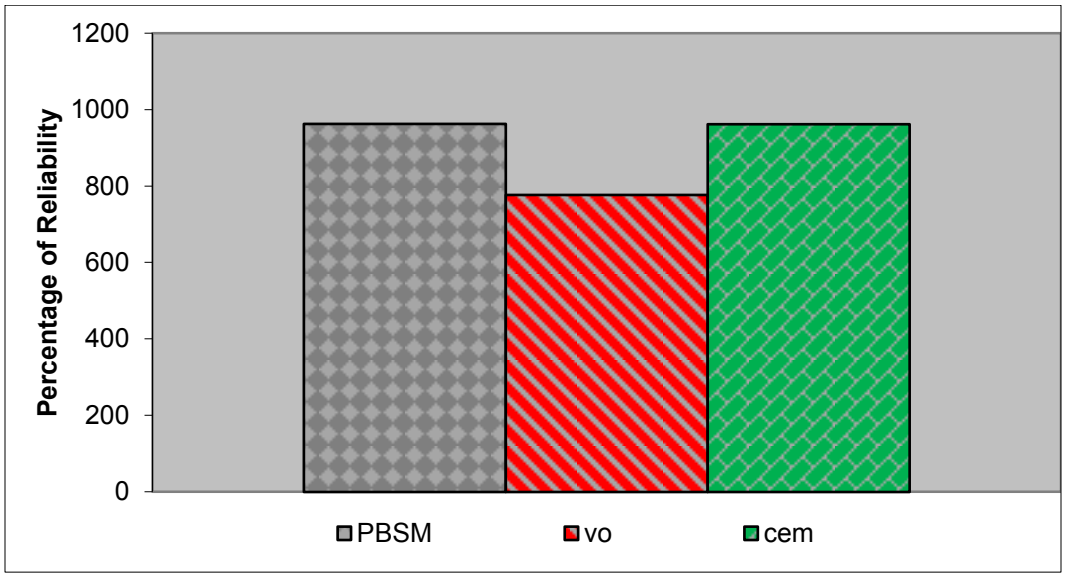

Figure 7.6: Percentage of reliability

The bar graph in Figure 7.6 shows the percentage of reliability (y-axis) against three different flooding techniques (x-axis). The reliability of *vo* is lower since some nodes in *vo* are in scoped flooding mode. These nodes do not transmit when they discover new nodes and therefore the percentage of reliability decreases. Furthermore, the reliability reduces due to collision in a simplified MAC layer. The other methods encounter more collision than PBSM because they generate a greater number of transmissions. Although the CEM method generates the most number of transmissions and encounters the most number of collision, its reliability is close to PBSM because the increased number of transmission make up for loss of reliability due to collision.

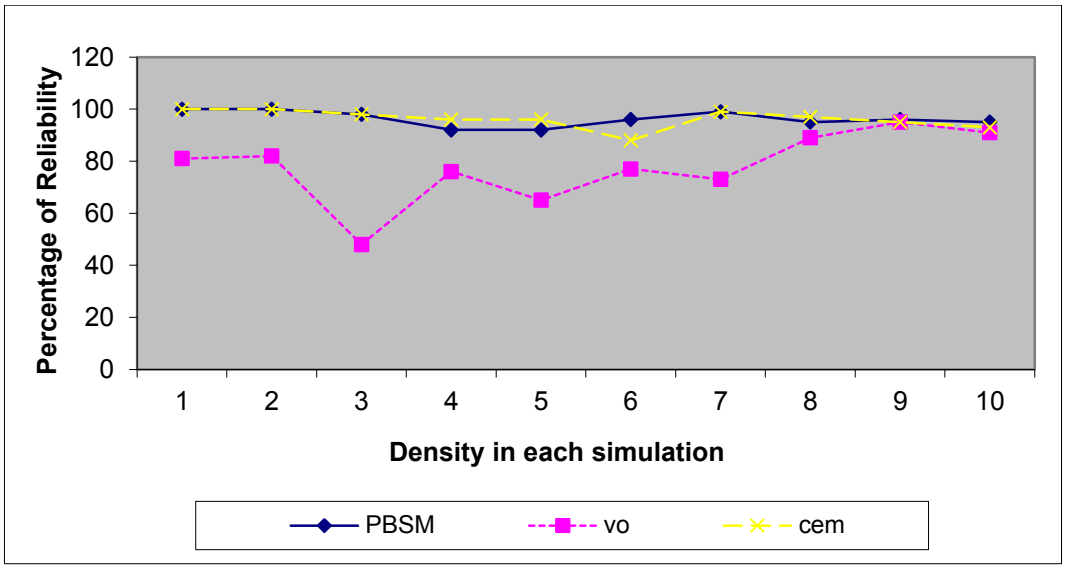

Figure 7.7: Percentage of reliability as density increases

The line graph in Figure 7.7 shows the percentage of reliability (y-axis) in each simulation (x-axis). The numbers on the x-axis indicate the value of *d* used in each simulation. At lower density, a node has few neighbors and therefore it will receive messages from transmission of few neighbors. If there is collision among these transmissions, then a node might not receive messages. Compared to PBSM, *vo* generates more transmission and more collision. Therefore, the reliability of *vo* is less than PBSM at lower densities. As *d* increases, a node will have more neighbors. Therefore, even if there is collision among transmissions from a few neighbors, the node can receive transmissions from other neighbors. Consequently, the reliability of *vo* increases and the difference between this method and PBSM decreases. The percentage of reliability of PBSM remains fairly constant. At lower density, PBSM produces very few collisions. As *d* increases, the number of collision in PBSM will increase. However, as *d* increases, a CDS node will cover more nodes. Therefore, a successful transmission from a CDS node will cover many nodes. This makes up for the loss of reliability due to collision. For a similar reason, the percentage of reliability of CEM also remains fairly constant as successful transmissions from few higher degree nodes will make up for the increased number of collisions as *d* increases.

The results of the simulation showed that the proposed method has the lowest number of transmissions among the methods described and higher reliability. The higher the reliability, the greater is the number of charging stations that received the message. Consequently, PBSM enables the PEV to find more charging stations (approx. 96%) compared to other protocols.

7.3 Summary

This chapter describes a broadcasting algorithm that can be used by an electric vehicle to find a suitable charging station. The main feature of the algorithm is that two nodes do not transmit every time they discover each other as new neighbors. When a node discovers a new neighbor, it first checks whether the new node is in the receiver list. If it is not in the receiver list, only then does the node consider transmitting. Otherwise, it does not transmit. Furthermore, this chapter provides some examples of applying this algorithm both in static and mobile networks. In addition, it contains simulation results to demonstrate the reliability of the algorithm in finding a charging station. The results are verified by comparing with two other broadcasting protocols. It is shown that the applied broadcasting algorithm has the lowest number of transmissions and higher reliability than the other protocols.

Chapter 8

Conclusion and Future Research

8.1 Concluding Remarks

The work presented in this thesis is summarized as follows:

- We propose an approach that uses Web services to remotely and efficiently interact with different smart home devices in order to manage energy consumption, in a smart grid environment. We developed a system that implements this approach.
- A smart home with a wireless network based on ZigBee and XMPP is simulated using Java. The smart home contains different kinds of elements like smart appliances, HVAC, sensors and a thermostat.
- A technique to implement secure Web service using BPEL is demonstrated. The Web service is developed on the central computer of the home. The central computer is able to communicate with all home elements directly or via the thermostat.
- WSIF is used to integrate the smart home simulator with Web service.
- The system provides operations to read the temperature of a room, light intensity of a room, energy consumption of a room as well as energy consumption of an entire home. Furthermore, there are functions to adjust light intensity, control temperature and control smart appliances. Other functions are to sell energy and minimize energy consumption.
- An algorithm to sell stored energy back to the grid from smart home and to reduce energy cost is proposed.
- An algorithm is proposed that can facilitate demand response and reduce energy consumption.
- A dynamic programming technique is presented to minimize the use of energy.

- ZigBee is used for communication between sensors and the thermostat. ZigBee protocol has low power consumption and good support for device interoperability.
- We present an application of XMPP to provide near real-time messaging service between the central computer and the appliances. XMPP also provides authentication and data encryption.
- The system has three stages of security. There is security when a user logs into the system, invokes the Web service and interacts with home devices.
- The proposed system offers quality of service.
- Residential consumers can select different levels of security for their smart home based on their need and affordability. Thus, the system offers differentiated service.
- An access control mechanism is provided that restricts certain users from controlling certain smart home devices via Web service.
- The home elements can be efficiently accessed over the Internet as Web services are easily accessed over the Internet.
- The interoperability feature of Web services enables a user operating a different platform or technology to communicate with the heterogeneous elements of a home in a unified way.
- The operations can be directly invoked via Web services from remote computer.
- By simulation, it is shown that the completion time of each function is small. It is also shown that the system helps consumers to achieve significant energy and monetary savings.
- The simulation results are verified by showing that the confidence intervals (95%) for the results are small.
- The simulation results are verified by comparing the results with a well-known home simulator, called DesignBuilder [DES10] and systems implemented in [YAZ09] [SLE11]. The comparison shows that the simulator used in our system performs significantly better than DesignBuilder simulator both in summer and winter conditions. Furthermore, our system performs better than the system described in [YAZ09] and [SLE11].
- A broadcasting algorithm is presented that can be used by a PEV to find out about the nearest, cheapest or most convenient charging station in ITS.
- The broadcasting algorithm is compared with two other broadcasting protocols. By simulation, it is shown that the applied broadcasting algorithm has the lowest number of transmissions and higher reliability in finding a charging station.

Through simulation and analytical study, it is shown that secure Web service can be used to efficiently and remotely control different smart home elements for energy management. Although web service needs some XML processing, the small completion time of each function proves that secure web service can be used to efficiently and remotely control different home elements. It is also shown that the system performs better than some other similar systems and that it reduces energy consumption significantly. Furthermore, algorithm to sell energy, minimize energy and to find a suitable charging station is analyzed to show their effectiveness.

Although there are quite a few existing home energy management systems, the proposed system has some unique features. The proposed system differs from them in the following way. Unlike some other system, the Web service runs only in the central computer instead of resource-constrained sensors or home devices. Web services do have some overhead due to size of XML messages and XML parser. Therefore, it is not convenient to run Web services on a sensor as it has limited capacity. Some papers offered some techniques to reduce the overhead due to XML. Even if good response time can be achieved by applying these techniques, running Web services on sensors reduces battery life due to extra processing. Most of the complexity of our system is in the centralized computer which communicates with all smart home elements. Moreover, the proposed system has no middleware or additional controller. Furthermore, our system considers the SG environment and energy management involving demand response. The system uses an algorithm for selling energy back to the grid and a dynamic programming technique for minimizing energy. Furthermore, the system provides three stages of security and quality of service in the form of differentiated service and access control mechanism.

8.2 Future Research

The future work could include implementing the approach using real sensors and real smart home elements instead of simulation. Sun SPOT (a Java-based wireless-sensor development kit) [SUN14] could be used to develop the sensors that can communicate wirelessly. The central computer needs to have components in order to process wireless messages and convert them to

Java. In this case, the BPEL process (i.e., the Web services) that is residing in a central computer could use Java to access those messages. Therefore, performance, advantage and limitations of using Web service in this context could be understood more clearly. Furthermore, there could be more gateway computer in order to avoid single-point of failure and distribute the load.

Plug-in Hybrid Electrical Vehicles could also be included in the system and their effects can be analyzed. Furthermore, Web services can be used for communication between different applications along the grid. The functionalities of different applications along the grid can be exposed as Web services. The function of another application can be invoked via Web services instead of calling it directly. This can certainly solve the interoperability issue if two applications are heterogeneous. Therefore, none of the applications need to be changed. Web services can be used as a platform for the incremental addition of new smart grid applications and their integration with other applications. This change can be cost-effective as Web services are inexpensive and easy to implement. Web services can be used to integrate the monitoring and control system, decision support system and systems at different substations along the grid. More ways of using Web services along the grid can be investigated.

Bibliography

[ABO03] G.D. Abowd, K. Edwards and B. Grinter, “Smart Homes or Homes that Smart?”, ACM SIGCHI Bulletin, Vol. 35, No.2, April 2003, pp. 13-13.

[AIE06] M. Aiello, “The Role of Web Services at Home”, Advanced International Conference on Telecommunications (AICT/ICIW), February 2006, pp. 164-170.

[ALA09] J. M. R. Álamo, J. Wong, R. Babbitt, H. Yang and C. K. Chang, “Using Web Services for Medication Management in a Smart Home Environment”, Proceedings of the 7th International Conference on Smart Homes and Health Telematics: Ambient Assistive Health and Wellness Management in Heart of City, Berlin, Heidelberg, 2009, pp. 265–268.

[AMI05] S. M. Amin and B.F. Wollenberg, “Toward a smart grid”, IEEE Power & Energy magazine, Volume 3, Issue 5, Sep-Oct 2005, pp. 34 - 41.

[ASA11] O. Asad, M. Erol-Kantarci and H. Mouftah, “Sensor Network Web Services for Demand-Side Energy Management Applications in the Smart Grid,” Proceedings 8th Annual IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, Nevada, February 2011, pp. 1176-1180.

[BER99] A. Van Berlo, A. Bob, E. Jan, F. Klaus, H. Maik and W. Charles, “Design Guidelines on Smart Homes”. COST 219bis Guidebook. European Commission. 1999.

[BLE10] A. Bleicher, “Privacy on the Smart Grid”, IEEE Spectrum, October 2010, http://spectrum.ieee.org/energy/the-smarter-grid/privacy-on-the-smart-grid, last accessed: Jan 2014.

[BPE04] “Tutorial 7: Invoking BPEL Processes through SOAP and Java”, BPEL Tutorial, Oracle, June 2004, www.oracle.com/technetwork/testcontent/orabpel-tutorial7-invokingbpelproce-131933.pdf, last accessed: Jan 2014.

[BPE07] “Oracle® BPEL Process Manager Administrator's Guide: 10g(10.1.3.1.0)”, Chapter 1, January 2007, http://docs.oracle.com/cd/E12483_01/integrate.1013/b28982/toc.htm, last accessed: Jan 2014.

[BRO98] J. Broch, D. A. Maltz, D. B. Johnson, Y. Hu and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols", Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom), Dallas, Texas, October 1998, pp. 85-97.

[BUT99] S. Butler, "Smart Toilets and Wired Refrigerators". US News and World Report, Volume 126, No. 22, May 1999, pp. 48-58.

[CAR04] J. Carle and D. Simplot-Ryl, "Energy efficient area monitoring for sensor networks", IEEE Computer Magazine, vol. 37, no. 2, February 2004, pp. 40 – 46.

[CER06] A. E. Cerpa, "Sensor Networks Challenges for Intelligent Buildings", In Workshop of Networked Embedded Control for Cyber-Physical Systems (SCADA 2006), Networking and Information Technology Research and Development (NITRD), Pittsburgh, PA, USA, November 2006.

[CHA08] M. Chan and C. Escriba, **"**A review of smart homes—Present state and future challenges", Computer Methods and Programs in Biomedicine, Volume 91, Issue 1, July 2008, pp. 55-81.

[CHU08] G. Chuan and W. Ai-min, "Research on Intelligent Building OAS Based on Web Services", IEEE International Symposium on Computational Intelligence and Design (ISCID), Washington, DC, 2008, pp. 478-481.

[COO04] D. E. Cooper, P. Ezhilchelvan and I. Mitrani, "High Coverage Broadcasting for Mobile Ad Hoc Networks", 3rd International IFIP-TC6 Networking Conference, Athens, Greece, May 2004, pp. 100-111.

[CUR02] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi and S. Weerawarana, "Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI", IEEE Internet Computing, Volume 6, Issue 2, March 2002, pp. 86-93.

[DES10] "DesignBuilder - Building design, simulation and visualisation - Building Simulation. Made Easy", DesignBuilder Software Ltd., May 2010, http://www.designbuilder.co.uk/, last accessed: Jan 2014.

[DEW01] G. Dewsbury, B. Taylor and M. Edge, "The Process of Designing Appropriate Smart Homes: Including the User in the Design", The 1st Equator IRC Workshop on Ubiquitous Computing in Domestic Environments, Nottingham, UK, 2001, pp. 131-146.

[DIG05] "Digital Living: The home of the future", The Economist, September 3 2005, pp. 14

[DPR] "Lecture 13: The Knapsack Problem", http://www.es.ele.tue.nl/education/5MC10/Solutions/knapsack.pdf, last accessed: Jan 2014.

[ERO10a] M. Erol-Kantarci and H. T. Mouftah, "Wireless Sensor Networks for Domestic Energy Management in Smart Grids", 25th Biennial Symposium on Communications (QBSC), Kingston, ON, Canada, May 2010, pp. 63–66.

[ERO10b] M. Erol-Kantarci, "Wireless Sensor Networks for In-Home Energy Management in Smart Grids", Wisense Seminar, Ottawa, Canada, March 2010.

[ERO11a] M. Erol-Kantarci and H. T. Mouftah, "Wireless Multimedia Sensor and Actor Networks for the Next-Generation Power Grid," Elsevier Ad Hoc Networks Journal, vol.9 no.4, June 2011, pp. 542-511.

[ERO11b] M. Erol-Kantarci and H. T. Mouftah, "How Will the Smart Grid Enable Electrification of Transportation", Wisense Seminar, Ottawa, Canada, October 2011.

[ERR06] A. Erradi, S. Padmanabhuni and N. Varadharajan, "Differential QoS support in Web Services Management". IEEE International Conference on Web Services (ICWS), Chicago, IL, Sep. 2006, pp. 781-788.

[GLO09] N. Glombitza, D. Pfisterer and S. Fischer, "Integrating Wireless Sensor Networks into Web Service-Based Business Processes", Proceedings of the 4th International Workshop on Middleware Tools, Services and Run-Time Support for Sensor Networks (MIDSENS), Urbana Champaign, IL, December 2009, pp. 25–30.

[GRE04] W. Green, D. Gyi, R. Kalawsky and D. Atkins, "Capturing user requirements for an integrated home environment", Proceedings of the third Nordic conference on Human-Computer interaction (NordiCHI), Tampere, Finland, 2004, pp. 255-258.

[GRI10] S. Grijalva, "Smart Grid as Distributed Intelligence", Smart Grid V-Summit: Spring 2010.

[HAG08] H. Hagras “Employing Computational Intelligence to Generate More Intelligent and Energy Efficient Living Spaces”, International Journal of Automation and Computing, Volume 5, Issue 1, January 2008, pp. 1-9.

[HAH03] J. Hahner, C. Becker and K. Rothermel, “A protocol for data dissemination in frequently partitioned mobile ad hoc networks”, IEEE International Symposium on Computers and Communications (ISCC), Kemer-Antalya, Turkey, July 2003, pp. 633-640.

[HAI06] D. A. Haidar, F. Cuppens, N. Cuppens-Boulahia and H. Debar. “An extended RBAC profile of XACML”. In Proceedings of 3rd ACM workshop on Secure Web Services (SWS), Alexandria, Virginia, November 2006, pp. 13 - 22.

[HEN07] O.A.B. Henkemans, K.E. Caine, W.A. Rogers, A.D. Fisk, M.A. Neerincx and B.D. Ruyter, “Medical monitoring for independent living: user-centered design of smart home technologies for older adults”. Proceedings of the Med-e-Tel conference for eHealth, telemedicine and health Information and communication technologies, 2007, pp. 368-373.

[HEN10] R. F. Z. Henry, “CEMA: Comfort Control and Energy Management Algorithms for use in Residential Spaces through Wireless Sensor Networks”, Master’s Thesis, University of Ottawa, Ottawa, Canada, 2010.

[HEN11] R. F. Z. Henry and H. T. Mouftah, “Novel algorithms to attain economical comfort control in residential spaces using Wireless Sensor Networks”, IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Niagara Falls, ON, May 2011, pp. 1374-1378.

[HOH09] K. Fehrenbacher, “Microsoft Reveals Its Energy Management Tool: Hohm”, June 2009, http://gigaom.com/2009/06/24/microsoft-reveals-its-energy-managent-tool-hohm/, last accessed: Jan 2014.

[HOR09] A. Hornsby, P. Belimpasakis and I. Defee, “XMPP-based wireless sensor network and its integration into the extended home environment”, 13th IEEE International Symposium on Consumer Electronics (ISCE), Kyoto, Japan, May 2009, pp. 794-797.

[HOR10] M. Horng, M. Hung, Y. Chen, J. Pan and W. Huang “A New Approach based on XMPP and OSGi Technology to Home Automation on Web”, International Conference on Computer Information Systems and Industrial Management Applications (CISIM), Kraków, Poland, October 2010, pp. 487-490.

[IEE] "Smart Grid Conceptual Model", IEEE Smart Grid, http://smartgrid.ieee.org/ieee-smart-grid/smart-grid-conceptual-model, last accessed: Jan 2014

[INT10] D. Adams, "Intel Home dashboard Research Project", OSnews, January 2010, http://www.osnews.com/story/22722/ Intel_s_Home_Dashboard_Research_Project, last accessed: Jan 2014

[INT06] "IntelliGridSM Consumer Portal Telecommunications Assessment and Specification", Electric Power Research Institute (EPRI), Technical Report, 2006.

[IPA09] A. Ipakchi and F. Albuyeh, "Grid of the future", IEEE Power and Energy Magazine, Volume 7, Issue 2, March 2009, pp. 52 - 62.

[JAK07] V.R. Jakkula, D.J. Cook and G. Jain, "Prediction models for a smart home based health care system", 21st IEEE International Conference on Advanced Information Networking and Applications Workshops (AINAW), Niagara Falls, ON, May 2007, pp. 761-765.

[JAN01] S. Jang, S. Lee and W. Woo, "Research Activities on Smart Environment". DBPIA, Vol.28, No. 12, 2001, pp. 85-97.

[JEO09] K. Jeong, "Design and Operation of Smart Homes in USA and Korea", Dissertation, Purdue University, West Lafayette, IN, 2009.

[JIA04] L. Jiang, D.Y. Liu and B. Yang, "Smart Home Research", Proceedings of International Conference on Machine Learning and Cybernetics", Shanghai, China, August 2004, pp. 659 - 663.

[JUR] M. B. Juric, "A Hands-on Introduction to BPEL", Oracle Technology Network, www.oracle.com/technetwork/articles/matjaz-bpel1-090575.html, last accessed: Jan 2014.

[JUR05] M. B. Juric, "Using WSIF for Integration", SOA Best Practices: The BPEL Cookbook, Oracle Technology Network, October 2005, www.oracle.com/ technetwork/articles/juric-095829.html, last accessed: Jan 2014.

[KAM11] A. Kamilaris, V. Trifa and A. Pitsillides, "HomeWeb: An Application Framework for Web-based Smart Homes", In Proceedings of the 18th International Conference on Telecommunications (ICT 2011), Ayia Napa, Cyprus, May 2011, pp. 134-139.

[KHA08] A. A. Khan, I. Stojmenovic and N. Zaguia, “Parameterless broadcasting in static to highly mobile wireless ad hoc, sensor and actuator networks”, Proceedings of 22nd IEEE International Conference on Advanced Information Networking and Applications (AINA), Ginowan, Okinawa, Japan, March 2008, pp. 620-627.

[KIM05] D. Kim, S. Lee, S. Han and A. Abraham “Improving Web Services Performance Using Priority Allocation Method”, Proceedings of IEEE International Conference on Next Generation Web Services Practices (NWeSP), Seoul, Korea, August 2005, pp. 22-26.

[KUS07] N. Kushiro, T. Higuma, M. Nakata, H. Kubota and K. Sato, "Practical Solution for Constructing Ubiquitous Network in Building and Home Control System," International Conference on Consumer Electronics, Las Vegas, NV, January 2007, pp. 1-2.

[LEI07] P. Leijdekkers, V. Gay and E. Lawrence, “Smart homecare system for health tele-monitoring”. Proceedings of the first international conference on the digital society (ICDS’07), Guadeloupe, French Caribbean, January 2007, pp. 3-3.

[LEO04] A. Leon-Garcia and I. Widjaja, "Peer-to-Peer Protocols and Data Link Layer," in Communication Networks - Fundamental Concepts and Key Architecture, S. W., Ed. New York, United States of America: McGraw-Hill, 2004, Chapter 5, pp. 342-345.

[LIN05] Z. Lingyun and L. Jiguang, “The Application of Web Services in the Distributed Component Integrate”, Microcomputer Information, Vol. 21, 2005, pp. 34-36.

[LIU09] X. Liu and L. Zhu, “Design Of SOA Based Web Service Systems Using QFD For Satisfaction Of Quality Of Service Requirements”, Proceedings of IEEE International Conference on Web Services (ICWS), Los Angeles, CA, July 2009, pp. 567-574.

[LOC] “OASIS Security Services (SAML) TC”, OASIS, www.oasis-open.org/committees/tc_home.php?wg_abbrev=security, last accessed: Jan 2014.

[LOC10] J. Peters and B. Thielbar, “Location Intelligence in Advanced Customer-to-Network Relationship Management”, Energy Central Webinar, April 2010.

[LOV00] B. Lovegrove, D. Congdon and S. Schaub, “Internet toasters as a capstone design project”, Proceedings of 30th ASEE/IEEE Frontiers in Education Conference (FIE), Kansas City, MO, Oct. 2000, pp. 14-18.

[MAA09] Z. Maamar, Q. H. Mahmoud, N. Sahli and K. Boukadi, “Privacy-aware web services in smart homes”, Ambient Assistive Health and Wellness Management in the Heart of the City Lecture Notes in Computer Science, Volume 5597, 2009, pp. 174–181.

[MEI08] A. K. Meier, “Residential Thermostats: Comfort Controls In California Homes”, Lawrence Berkeley National Laboratory, Sep. 2008, Available online at: http://escholarship.org/uc/item/2vc7h48t, last accessed: Jan 2014.

[MIC10] “MicroFit Program”, Ontario Power Authority, 2010, http://microfit.powerauthority.on.ca/about-microfit, last accessed: Jan 2014.

[MIL09] M. Milligan, K. Porter, E. DeMeo, P. Denholm, H. Holttinen, B. Kirby, N. Miller, A. Mills, M. O’Malley, M. Schuerger and L. Soder “Wind Power Myths Debunked”, IEEE Power and Energy Magazine, Volume 7, Issue 6, November 2009, pp. 89 - 99.

[MOU10] A. Mourad, S. Ayoubi, H. Yahyaoui and H. Otrok, “New approach for the dynamic enforcement of Web services security”, IEEE Conference on Privacy, Security and Trust (PST), August 2010, pp. 189-196.

[MUR04] C. S. R. Murthy and B.S. Manoj, “Ad Hoc Wireless Networks: Architectures and Protocols”. Prentice Hall, Upper Saddle River, NJ, 2004.

[OAS04] OASIS. Xml access control markup language (XACML), version 2.0. committee draft. In Organization for the Advancement of Structured Information Standards, OASIS, 2004.

[ODE05] R. O’Dell and R. Wattenhofer, “Information Dissemination in Highly Dynamic Graphs”, DIALM-POMC joint workshop on foundations of mobile computing, Cologne, Germany, September 2005, pp. 104 – 110.

[PAC08] F. Paci, E. Bertino and J. Crampton, “An access-Control framework for WS-BPEL”, International Journal of Web Services Research (IJWSR), Vol. 5, Issue 4, January 2008, pp. 20-43.

[PAR03] S. H. Park, S. H. Won, J. B. Lee and S. W. Kim, “Smart Home Digitally Engineered Domestic Life”, Personal Ubiquitous Computing, Vol. 7, Issue 3-4, July 2003, pp. 189-196.

[PAR09] J. Parra, M. A. Hossain, A. Uribarren, E. Jacob and A. E. Saddik, “Flexible Smart Home Architecture using Device Profile for Web Services: a Peer-to-Peer Approach”, International Journal of Smart Home, Vol. 3, April 2009 pp. 39-55.

[PEN00] W. Peng and X. Lu, “On the reduction of broadcast redundancy in mobile ad hoc networks”, in Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing (ACM MobiHoc), Boston, USA, 2000, pp. 129-130.

[PEN10] L.Pengfei, L.Jiakun, N.Luhua and W. Bo, “Research and application of ZigBee protocol stack," IEEE International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), Changsha City, China, March 2010, pp. 1031 – 1034.

[PER08] T. Perumal, A. R. Ramli, C. Y. Leong, S. Mansor and K. Samsudin, “Interoperability among Heterogeneous Systems in Smart Home Environment”, IEEE International Conference on Signal Image Technology and Internet Based Systems (SITIS), Bali, Indonesia, December 2008, pp. 177-186.

[POW] Google PowerMeter, “www.google.org/powermeter/”, last accessed: Jan 2014.

[PRA00] M. Pragnell, L. Spence and R. Moore, “The Market Potential for Smart homes”, Joseph Rowntree Foundation, Ref N40, November 2000, Available online at: www.jrf.org.uk/knowledge/findings/housing/n40.asp, last accessed: Jan 2014.

[PRI08] N. Priyantha, A. Kansal, M. Goraczko and F. Zhao, “Tiny Web Services: Design and Implementation of Interoperable and Evolvable Sensor Networks”, Proceedings of the 6th ACM conference on Embedded network sensor systems (Sensys), Raleigh, NC, November 2008, pp. 253-266.

[RAC09] G. Rackliffe, “Smart Grid Introduction”, PSERC Executive Forum, March, 2009.

[RAS07] U. Rashid, H. Schmidtke and W. Woo, “Managing disclosure of personal Health information in smart home healthcare”, International Conference on Universal Access in Human Computer Interaction: Ambient Interaction, Springer-Verlag, Beijing, China, July 2007, pp.188-197

[RED06] A. Redfern, M. Koplow and P. Wright, “Design architecture for multi-zone HVAC control systems from existing single-zone systems using wireless sensor networks”, SPIE Smart structures, devices, and systems III, Vol. 6414, January 2007.

[REK09] W. Rekik, M. Khemakhem, A. Belghith and J. Fayolle, “PKI and UDDI based trust centre: an attempt to improve web service security”, Proc. IEEE International Conference for Internet Technology and Secured Transactions (ICITST), London, UK, 2009, pp. 1-4.

[ROD03] T. Rodden and S. Benford, “The evolution of buildings and implications for the design of ubiquitous domestic environments”, Proceedings of the SIGCHI conference on Human factors in computing systems, Ft. Lauderdale, Florida, April 2003, pp. 9-16.

[RON10] K. Rönnlund and O. Porri, “How Are the Goals of European Energy Regulators Reached With the Current Smart Metering Deployments?”, Smart Grid V-Summit: Spring 2010.

[SAI04] P. Saint-Andre, ed., “Extensible Messaging and Presence Protocol (XMPP): Core”, RFC 3920, October 2004.

[SAI09] B. Saint, “Rural Distribution System Planning using Smart Grid Technologies”, IEEE Rural Electric Power Conference, Ft. Collins, CO, April 2009, pp. B3 - B3-8.

[SAN06] “San Diego Smart Grid Study Final Report”, The Energy Policy Initiatives Center (EPIC), October 2006, www.sandiego.edu/documents/epic/ 061017_SDSmartGridStudyFINAL.pdf, last accessed: Jan 2014.

[SAN10] E. Santacana, G. Rackliffe, L. Tang and X. Feng, "Getting Smart”, IEEE Power and Energy Magazine, Vol. 8, No.2, March-April 2010, pp.41-48,

[SEN08] “SensorWeb 2.0: Service-Oriented Middleware for Heterogeneous Sensor Networks”, June 2008, www.cloudbus.org/sensorweb/SensorWeb2-0_Release_Notes.pdf, last accessed: Jan 2014.

[SGR] “The Smart Grid”, www.smartgrid.gov/the_smart_grid#smart_grid, last accessed: Jan 2014.

[SHA09] M. Shargal and D. Houseman, “The Big Picture of Your Coming Smart Grid”, March 2009, http://www.smartgridnews.com/artman/publish/ commentary/The_Big_Picture_of_Your_Coming_Smart_Grid-529.html, last accessed: Jan 2014.

[SHI04] N. Shingaki, H. Nojima, M. Kitabata, A. Onozawa and K. Sato, “Location of things in a house: Towards ubiquitous mapping in the home”, International workshop on Ubiquitous, Pervasive Internet Mapping (UPIMap), Tokyo, Japan, September 2004, pp.61-68.

[SHM11] “Home Monitoring”, Rogers, 2011, www.rogers.com/web/content/smart-home-monitoring-features, last accessed: Jan 2014.

[SID06] H. Siderius and A. Dijkstra, “Smart Metering for Households: Costs and Benefits for the Netherlands”, 2006, www.researchgate.net/publication/237456570_Smart_Metering_for_Households_Cost_and_Benefits_for_the_Netherlands, last accessed: Jan 2014.

[SLE11] A. Sleman and R. Moeller, “SOA distributed operating system for managing embedded devices in home and building automation”, IEEE Transactions on Consumer Electronics, Vol. 57, No. 2, May 2011, pp. 945-952.

[SMG08] “The SMART GRID: An Introduction”, U.S. Department of Energy, 2008, http://energy.gov/sites/prod/files/oeprod/DocumentsandMedia/DOE_SG_Book_Single_Pages.pdf, last accessed: Jan 2014.

[SOE06] E. Soedarmadji and R. J. McEliece, “A Dynamic Graph Algorithm for the Highly Dynamic Network Problem”, IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom), Pisa, Italy, March 2006, pp.101-105.

[SSN09] “Smart Sensor Networks: Technologies and Applications for Green Growth”, Organisation for Economic Co-operation and Development (OECD), December 2009, www.oecd.org/dataoecd/39/62/44379113.pdf, last accessed: Jan 2014.

[STO02] I. Stojmenovic, M. Seddigh and J. Zunic, “Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks”, IEEE Transactions on Parallel and Distributed Systems, Vol. 13, Issue 1, January 2002, pp. 14 – 25.

[STA08] R. Stamminger, “Synergy Potential of Smart Appliances”, November 2008, www.smart-a.org/WP2_D_2_3_Synergy_Potential_of_Smart_Appliances.pdf, last accessed: Jan 2014.

[SUN14] “Sun SPOT World”, Oracle, January 2014, www.sunspotworld.com/, last accessed: Jan 2014.

[TIV] "Tivoli Monitoring for Energy Management", IBM, www-01.ibm.com/software/tivoli/products/monitor-energy-management/, last accessed: Jan 2014.

[TSE09] Y. Tseng, Y. Wang and L. Yeh, "iPower: An Energy Conservation System for Intelligent Buildings by Wireless Sensor Networks". International Journal of Sensor Networks, Volume 5, Issue 1, February 2009, pp. 1-10.

[TSO06] Y. Tsou, J. Hsieh, C. Lin and C. Chen, "Building a Remote Supervisory Control Network System for Smart Home Applications", IEEE International Conference on Systems, Man, and Cybernetics, Taipei, Taiwan, October 2006, pp. 1826 - 1830.

[UEA] "Energy Information Administration, independent statistics and analysis", http://www.eia.doe.gov/, last accessed: August 2010.

[VAI11a] B. Vaidya, J. Park, S. Yeo and J. Rodrigues, "Robust one-time password authentication scheme using smart card for home network environment", Elsevier Computer Communications Journal, vol.34, 2011, pp. 326-336.

[VAI11b] B. Vaidya, D. Makrakis and H.T. Mouftah, "Security Mechanism for Multi-domain Vehicle-to-Grid Infrastructure", Proceedings IEEE Globecom Symposium on Communications and System Security, Houston, Texas, December 2011, pp. 1-5.

[VEN03] A. Venkatesh, "Smart Home Concepts: Current Trends". Center for Research on Information Technology and Organizations, I.T. in the Home, Paper 377, 2003.

[VER08] R. Verdone, D. Dardari, G. Mazzini and A. Conti; "Wireless Sensor and Actuator Networks: Technologies, Analysis and Design", London: Academic Press/Elsevier, 2008.

[VIS02] K. Viswanath and K. Obraczka, "An adaptive approach to group communications in multi-hop ad hoc networks", IEEE International Symposium on Computers and Communications (ISCC), Taormina, Italy, July 2002, pp. 559-566.

[WAR09] C.Warmer, K. Kok, S. Karnouskos, A. Weidlich, D. Nestle, P. Selzam, J. Ringelstein, A. Dimeas and S. Drenkard, " Web services for integration of smart houses in the smart grid", Proceedings Grid-Interop Conference, Denver, CO, December 2009.

[WEE05] S. Weerawarana, F. Curbera, F. Leymann, T. Storey and D. F. Ferguson, "Web services platform architecture : SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and more", New Jersey: Prentice Hall, 2005.

[WEI10] Z. Wei, S. Qin, D. Jia and Y. Yang, "Research and Design of Cloud Architecture for Smart Home", Proceedings IEEE International Conference on Software Engineering and Service Sciences (ICSESS), Beijing, China, July 2010, pp. 86-89.

[WOO07] G. Wood and M. Newborough, "Energy-use information transfer for intelligent homes: Enabling energy conservation with central and local displays", Elsevier Energy and Buildings, Vol. 39, Issue 4, April 2007, pp. 495-503.

[WSI] "Web Services Invocation Framework", Apache Software Foundation, http://ws.apache.org/wsif/, last accessed: 2012.

[WU07] C. L. Wu, C. F. Liao and L.C. Fu, "Service-Oriented Smart-Home Architecture Based on OSGi and Mobile-Agent Technology", IEEE Transactions on Systems, Man & Cybernetics: Part C – Applications and Reviews, Vol. 37, No. 2, March 2007, pp. 193 - 205.

[XU09] J. Xu , Y. Lee, W. Tsai, W. Li, Y. Son, J. Park and K. Moon, "Ontology-based Smart Home Solution and Service Composition", Proceedings of the IEEE International Conference on Embedded Software and Systems (ICESS), Zhejiang, China, May 2009, pp. 297 - 304.

[XU10] K. Xu, M. Song and X. Zhang, "Home Appliance Mashup System based on Web Service", IEEE International Conference on Service Sciences (ICSS), Hangzhou, China, May 2010, pp. 94-98.

[YAM09] H. F. E. Yamany, M. A. M. Capretz and D. S. Allison, "Quality of Security Service for Web Services within SOA", IEEE World Conference on Services – I, Los Angeles, CA, July 2009, pp. 653-660.

[YAZ09] D. Yazar and A. Dunkels, "Efficient Application Integration in IP-Based Sensor Networks", Proc. First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings (BuildSys), Berkeley, California, November 2009, pp. 43-48.

Appendix A

Confidence Interval Computation

A confidence interval (CI) is used to quantify the uncertainty in any collected sample of data. It is defined as the estimated range of values within which a generated data lies with a specified probability. This probability is usually set to 95%, which means that we have a confidence of 95% that the collected data lay in a certain (confidence) interval. The end points of the CI are known as the confidence limits. The confidence limits for a normally distributed sample of data *n* are computed as follows:

$$\mu \pm z\frac{\sigma}{\sqrt{n}} \qquad \text{(A.1)}$$

where, μ is the mean value of *n*, σ is the standard deviation of *n*, and *z* is the significance level (described shortly). The above equation states that CI is centered at the mean value of the collected sample data.

The significance level *z* is used to specify the area under the normal distribution curve, shown in Figure A.1, that corresponds to the desired confidence level. Therefore, to find the 95% CI for the shown normal distribution, we need to exclude 5% of the total area under the curve from our computation. This means that we should exclude 2.5% of the area on both sides of the mean, μ. Then, we need to find the area that corresponds to 95% of the sample of data *n*. This area can be found using the z-table that is populated with the values of *z* (or areas) that correspond to the desired confidence level. A partial snapshot of the z-table is shown in Table A.1. To read *z* from this table, we firstly specify the percentage of the sample data needed to achieve a 95% of confidence. This corresponds to 1-0.025 = 0.975. Thus, from the table we can see that *z* = 1.96 (once we locate the 0.975 in the table, we read the first two digits

of *z* from the leftmost column. Then, we get the third digit from the first row of the column where 0.975 is located).

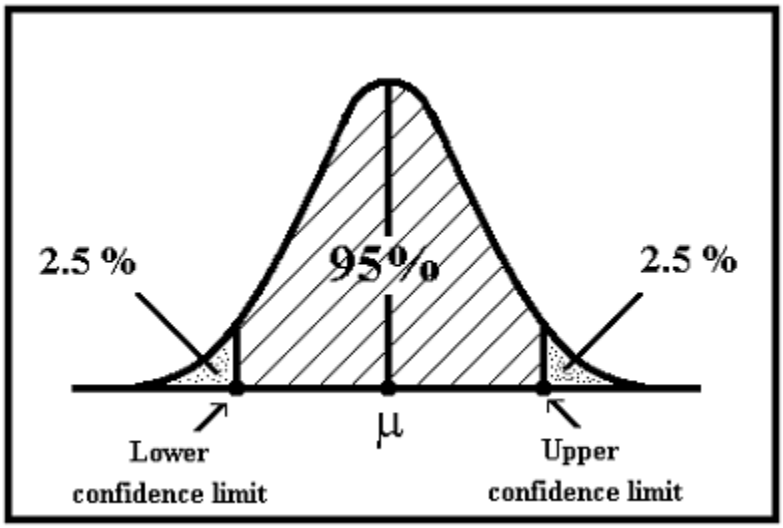

Figure A.1: An illustration of the normal distribution function of the sample data *n*. Only 95% of the area under the curve is considered to compute a CI of 95% centered at the mean value, μ.

Table A.1: z-table

z	0	0.01	0.02	0.03	0.04	0.05	0.06	0.07
1.8	0.96407	0.96485	0.96562	0.96638	0.96712	0.96784	0.96856	0.96926
1.9	0.97128	0.97193	0.97257	0.97320	0.97381	0.97441	0.97500	0.97558
2	0.97725	0.97778	0.97831	0.97882	0.97932	0.97982	0.98030	0.98077