Study on Network Management of Smart Home System - Combination of direct and indirect management techniques

Pham Van Cu

School of Information Science
Japan Advanced Institute of Science and Technology
September, 2016

Master's Thesis

Study on Network Management of Smart Home System - Combination of direct and indirect management techniques

1410222 Pham Van Cu

Supervisor : Professor Yasuo Tan
Main Examiner : Professor Yasuo Tan
Examiners : Associate Professor Yuto Lim
Professor Yoichi Shinoda

School of Information Science
Japan Advanced Institute of Science and Technology

August, 2016

Acknowledgement

First and foremost, I would like to express my deep gratitude to Professor Yasuo Tan, my research supervisors, for all of his assistance and guidance. I can not finish my research without his encouragement, guidance and support with his patient and knowledge. I am really proud to be his student.

I wish to offer my special thanks to Associate Professor Yuto Lim, my second supervisor, for his valuable comments and advices not only to my research but also presentation skills and academic writing skills. He always points out my problems and gives me great suggestions.

I would also like to thank Professor Yoichi Shinoda for his straight comments at my midterm defence. Without his comments, I am still an innocent kid in science.

I wish to say sincere thanks to Professor Satoshi Tojo and Associate Professor Nguyen Le Minh, who taught me and gave me a chance to be here, at JAIST.

I am particularly grateful for the warmly welcome and kindly support given by my labmates, members of Professor Tan laboratory. Specially Mr. Makino Yoshiki for allowing me to use your framework, Mr. Marios Sioutis for your advices and encouragement, Mr. Yang Zhengguo for your guidances.

Finally, my family is the biggest motivation to go on my study. My dear parents, your unconditional love made me today. This thesis is dedicated to you.

List of Figures

List of Tables

Contents

This dissertation was prepared according to the curriculum for the Collaborative Education Program organized by Japan Advanced Institute of Science and Technology and Ho Chi Minh National University

Terms and Definitions

To understand and clarify the terms used in the study, the following are hereby defined:

Internet of Things

[ITU-T Y.2060]: A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies.

Home Gateway

[ITU-T Y.2070]: An always on, always connected device which acts as the central point connecting the devices on the home network to the applications on the wide area network, and monitors and performs actions on data flows within the home network as well as on bidirectional communication flows between the home network and the wide area network.

Home Network

[ITU-T J.190]: A short-range communications system designed for the residential environment, in which two or more devices exchange information under some sort of standard control.

Management Platform

[ITU-T Y.2070]: A platform which has common functions providing the interface and the management for the home network applications, and the virtual device management and the resource management for the home gateway and the devices.

Managed Agent

[ITU-T Y.2070]: A software running on the device to set the configuration information and to collect the information of the device. The managed agent gets the information from the resource management function on the management platform for the configuration of the device and sends the collection of the internal status of the device to it for various home network services including remote management and fault diagnosis.

Abbreviations and acronyms

This study uses the following abbreviations and acronyms:

HGW	Home Gateway
MP	Management Platform
IP	Internet Protocol
HN	Home Network
CoAP	Constrained Application Protocol
M2M	Machine to Machine
IoT	Internet of Things
CPU	Central Processing Unit
VPN	Virtual Private Network
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
HTTP	Hypertext Transfer Protocol
RAM	Random Access Memory
EHD	ECHONET Lite message header
TID	Transaction ID
EDATA	ECHONET Lite Data
ESV	ECHONET Lite service
OPC	Number of processing properties
EPC	ECHONET Lite Property
PDC	Property data counter
EDT	Property value data

Chapter 1

Introduction

1.1 Smart home systems overview

In accordance with the Moores Law, the number of components per integrated circuit at minimal cost doubles approximately every two years. In the vision of the Internet of Things (IoT), this gain will decrease power consumption, to integrate whole systems on a tiny chip, and in particular to minimize unit costs. Due to this, myriads of IoT devices deployed in homes, office buildings, factories, whole cities, and other environments of interest.

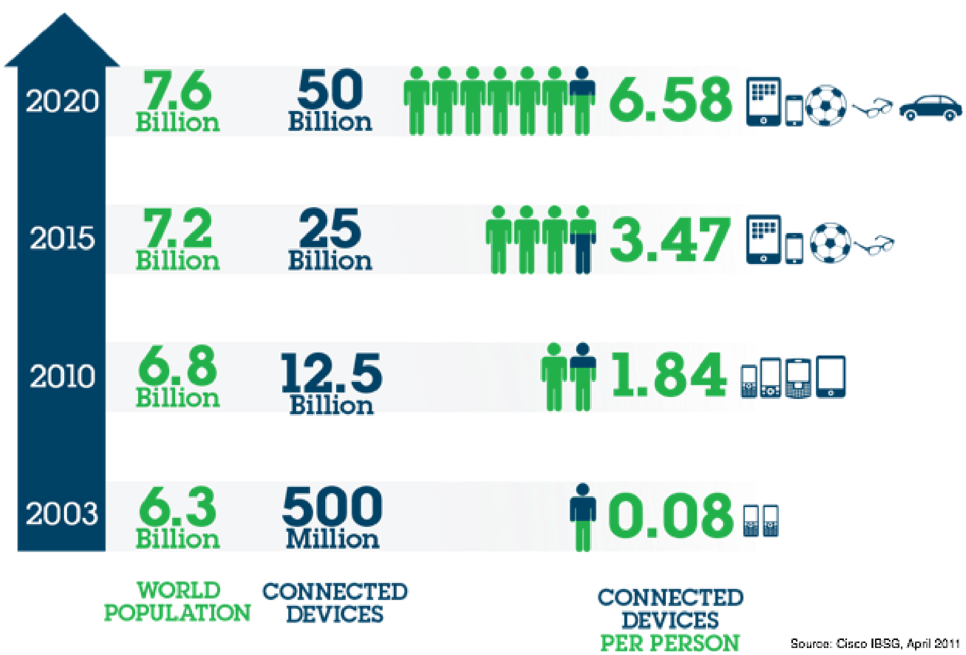

Figure 1.1: Number of connected devices per person

Smart home includes the control of lighting, heating, ventilation, air conditioning, appliances, entertainment and home security devices to improve convenience, comfort, energy efficiency, and safety. With the IoT revolution, home appliances are not single function devices but also have abilities to interoperate and interconnect with the global information and communication infrastructure. For example, the air-conditioner now can exchange data with temperature sensors to automatically adjust the temperature and the operation status can be monitored by users via the Internet. Management a large number of interconnected devices are quite challenging due to there is no network administrator or technician in every house.

1.2 Requirements in smart home network management

In the most instances, devices in smart homes are constrained IoT devices and heterogeneity is one of fundamental characteristics of the IoT due to they are based on different hardware platforms and networks. They can interact with other devices or service platforms through different networks as described in [1]. The enormous number of devices and their heterogeneity pose many requirements for the management architecture. Traditional management functionalities such as remote control, monitoring and maintenance are considered of paramount significance for the operation of things in the IoT. However, these management capabilities need to evolve to cater for the unique characteristics of the IoT. Considered requirements for management which are referred from [2] are depicted in the Table 1.1

Table 1.1: Requirements on the Management of Networks with constrained devices

Requirement	Description
Support multiple device classes within a single network	Larger networks usually consist of devices belonging to different device classes,communicating with each other. Hence, the management architecture must be applicable to networks that have a mix of different device classes
Management scalability	The management architecture must be able to scale with the number ofdevices involved and operate efficiently in any network size and topology.,This implies that, e.g., the managing entityis able to handle large amounts of device monitoring data and the management protocol is not sensitive to the decrease ofthe time between two client requests.
Support low latency devices	Provide mechanisms to manage time-sensitive or dangerous devices with low latency.
Protocol extensibility	Provide means of extensibility for the management protocol, i.e.,by adding new protocol messages or mechanisms that can deal with changing requirements on a supported message and data types effectively, without causing interoperability problems or having to replace/update large amount of deployed devices.
Self-monitoring	Provide self-monitoring (local fault detection) feature, for fast fault detection and recovery.

1.3 Objective and contributions

Currently, many research efforts focus on the management of IoT devices and Home Network (HN). One such approach is direct management. In [4], by implementing LWM2M - a remote device management standard, IoT devices are directly managed, but their architecture can not support very constrained devices with limited in CPU or memory.

Another approach focuses on indirect management [5] by using a light version of SNMP [6] and NETCONF [7], In that approach, very constrained devices can be managed, but less constrained devices are also managed in the same manner with very constrained devices.

The characteristics and management requirements of IoT devices as well as a common set of management functionalities are described in [8]. Therefore, only direct management or indirect management may not cover all of characteristic of devices and network. For example, indirect management is realistic for sleeping devices, low power lossy area network, heterogeneous networks. However, devices have more computing power and network ability can benefit from direct management.

The objective of this study is to propose a management architecture that is adaptable to the heterogeneity of devices in the smart home. Combining of indirect and direct management to benefit from their advantages is our approach.

The contributions of this study is divided into:

- Propose an architecture for management.
- Propose an intelligent home gateway for management.
- Implement a prototype to validate the feasibility of the proposed architecture.

1.4 Outline of this document

The structure of this document is follow by:

- In chapter 1, I introduce the smart home system devices characteristics and requirements for management architecture that apply for managing the smart home system. The objective and contributions are described at the end of chapter 1.
- In chapter 2, the OmniRAN architecture and network reference model are addressed as the related work.
- In chapter 3, I will explain the background information regarding to the heterogeneity of devices, current management approaches and a management architecture. Devices are based on different hardware platforms, communication technology and the power budgets. The problem of device heterogeneity concerns a wide range of aspects, but in this research we focus on the heterogeneity in terms of communication models (i.e., device to device, device to gateway, device to cloud, etc.) and device's characteristics(i.e., memory and processing capabilities, strategies for using power to communication). The qualitative analysis of indirect management and direct management are also addressed. The characteristics of two management approaches and example of concrete systems are also described. From this analysis, the idea of combining direct and indirect management approaches are considered.
- In chapter 4, the qualitative analysis of the Home Gateway (HGW) implementation patterns are described. These implementation patterns are (i) tunnelling data from the HN to the MP via the HGW, (ii) translating HN data at the home gate then data is managed by other management protocols, (iii) using application in the HGW to manage the local network then passing the data to the MP. The proposed pattern based on (iii) is depicted.
- In chapter 5, I introduce some basic knowledge of the implementation and experiments. I implemented a prototype of the proposed architecture by combining ECHONET Lite for indirect management and CoAP for direct management.
- Finally, I evaluate my proposed architecture by analysing the packets. The quantitative analysis of the implementation patterns are also made. I will investigate other protocol combinations as my future works.

Chapter 2

Related Works

Currently, many researches are focusing on the heterogeneity of IoT devices. OmniRAN (P802.1CF [3]) is an example of these researches. OmniRAN is a recommended practice for network reference model and functional description of IEEE 802 access network. OmniRAN is stands for:

- OMNI: Open Mobile Network Interface
- RAN: Range Area Networks

Due to more networks are coming up by the development of the IoT and IEEE 802 access is becoming more heterogeneous, omniRAN tends to close the gap and tie 802 devices into an family of standards within a heterogeneous IP network. OmniRAN architecture is described as in Figure 2.1.

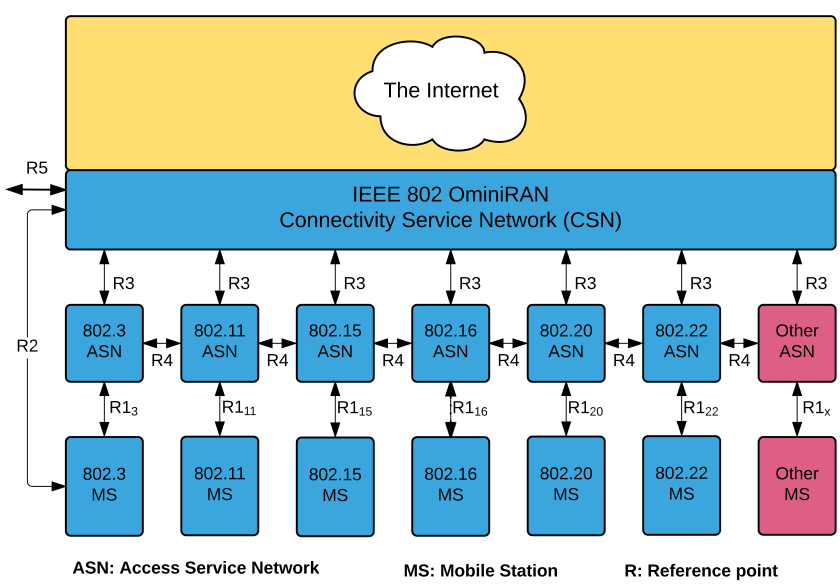

Figure 2.1: OmniRAN Architecture

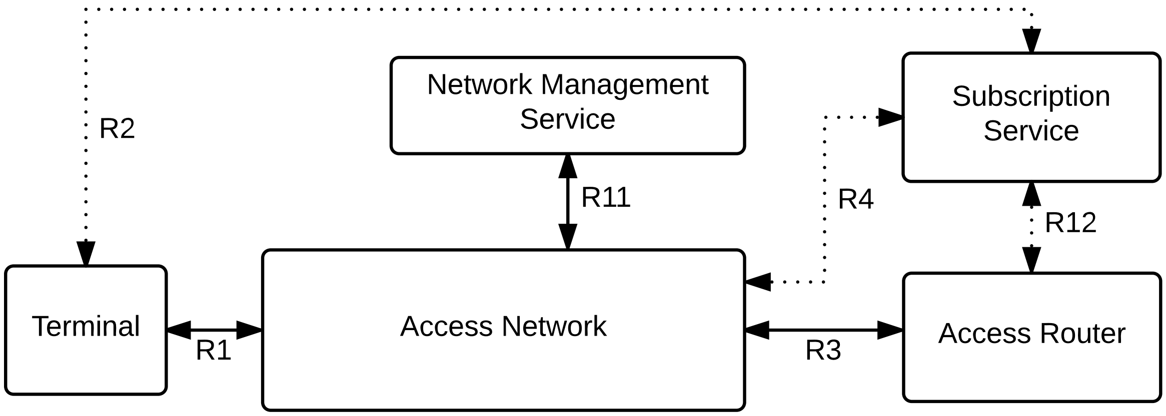

Figure 2.2: Network Reference Model Schematic

In the OmniRAN network reference model schematic, Network Management Service (NMS) is addressed. The NMS plays an important role in Fault Diagnostics and Maintenance (FDM) process and the Reference point 11 (R1) can be mapped to the link between the home gateway and management platform as in the home network as described in ITUT-Y2070 [16].

Chapter 3

Background Research

3.1 Device heterogeneity

Appliances in the HN are heterogeneous devices due to they are different in terms of functionalities, processing power, communication capabilities and power budget. The problem of device heterogeneity concerns a wide range of aspects, but in this research we focus on the heterogeneity in terms of:

- Communication models: Communication protocols and processing power are affected the communication model of devices. The communication models are affected to the management architecture that can be applied for management.
- Device characteristics: Device capabilities and power budgets also impact to the management architecture used to manage devices.

3.1.1 Communication models

In [9], the four basic communication models demonstrate the underlying design strategies used to allow IoT devices to communicate are outlined. The discussion below summarizes key characteristic of each model.

- Device-to-Device model: This model represents two or more devices that directly connect and communicate to each others rather than through an intermediary application server. These device-to-device networks allow devices that adhere to a particular communication protocol to communicate and exchange messages to achieve their function. This communication model is commonly used in applications like home automation systems, which typically use small data packets of information to communicate between devices with relatively low data rate requirements. Residential IoT devices like light bulbs, light switches, thermostats, and door locks normally send small amounts of information to each other (e.g. a door lock status message or turn on light command) in a home automation scenario.

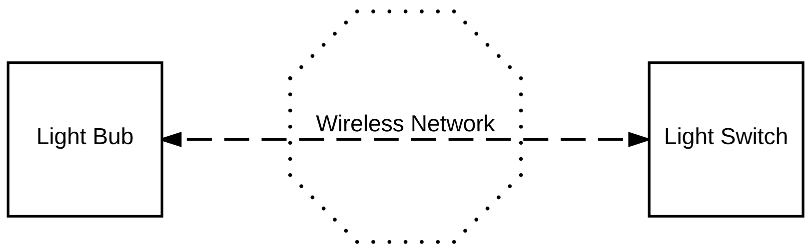

Figure 3.1: Device-to-Device Communication Pattern

- Device-to-Gateway model: In this model, devices connect through the gateway to reach a cloud service. In simpler terms, this means that there is application software operating on a local gateway device, which acts as an intermediary between the device and the cloud service and provides security and other functionality such as data or protocol translation. In other words, this communications model is frequently used to integrate new smart devices into a legacy system with devices that are not natively interoperable with them. A downside of this approach is that the necessary development of the application-layer gateway software and system adds complexity and cost tothe overall system.

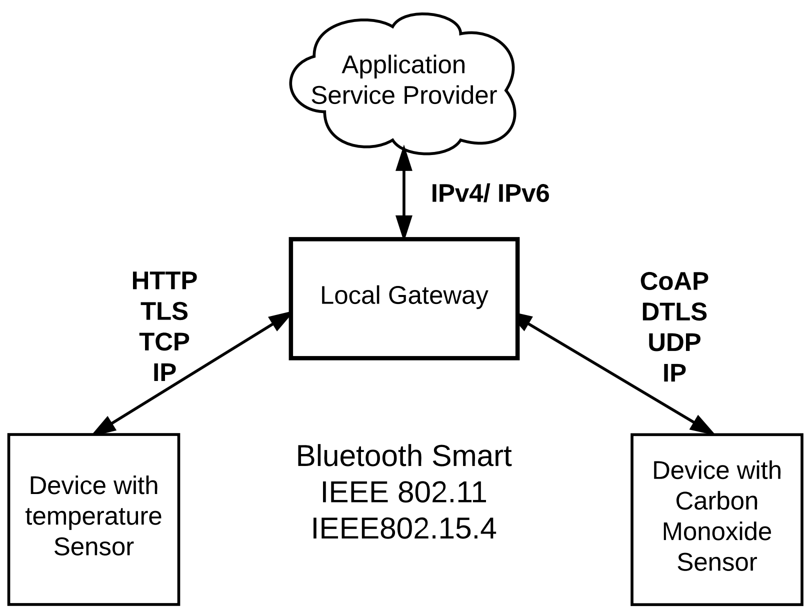

Figure 3.2: Device-to-Gateway Communication Pattern

- In a device-to-cloud communication model, the IoT device connects directly to an Internet cloud service like an application service provider to exchange data and control message traffic. This approach frequently takes advantage of existing communications mechanisms like traditional wired Ethernet or Wi-Fi connections to establish a connection between the device and the IP network, which ultimately connects to the cloud service.This communication model is employed by some popular consumer IoT devices like the Nest Labs Learning Thermostat [10]. In the case of the Nest Learning Thermostat, the device transmits data to a cloud database where the data can be used to analyze home energy consumption. Further, this cloud connection enables the user to obtain remote access to their thermostat via a smartphone or Web interface, and it also supports software updates to the thermostat.

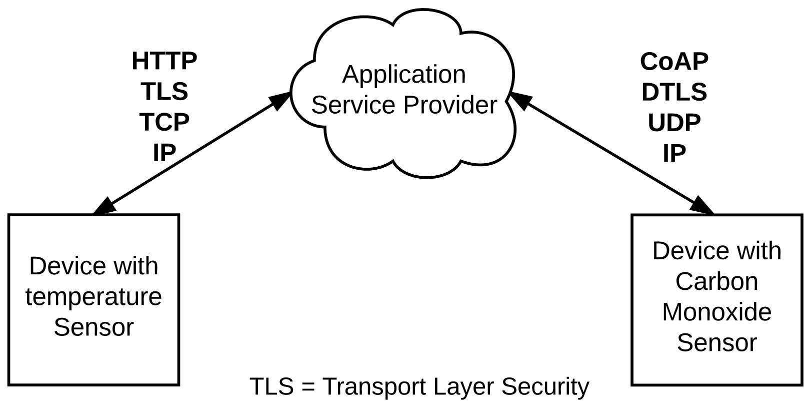

Figure 3.3: Device-to-Cloud Communication Pattern

- Back-End Data Sharing Pattern: This model is an extension of Device-to-Cloud pattern.The back-end data-sharing model refers to a communication architecture that enables users to export and analyze smart object data from a cloud service in combination with data from other sources.

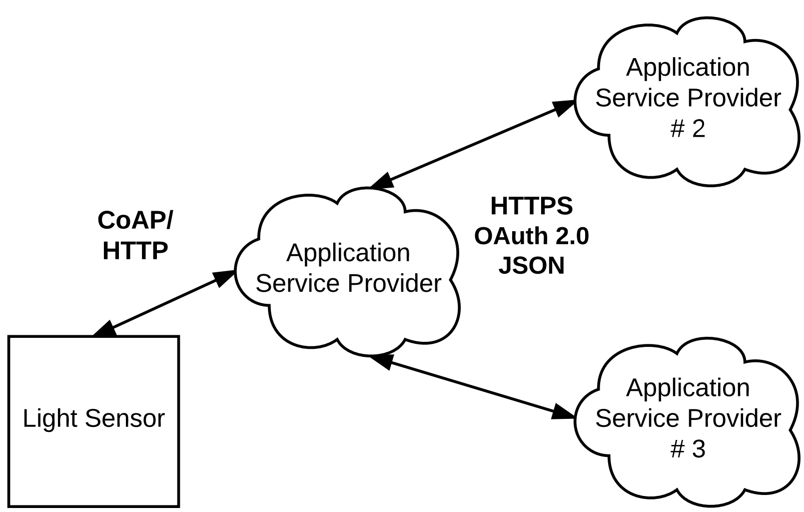

Figure 3.4: Back-End Data Sharing Pattern

3.1.2 Device characteristics

As in [11], devices are classified by:

- Memory and processing capabilities: The existing management technologies utilize the different protocol stacks and the different protocol stacks consumes different amount of memory. The Table 3.1 is the classification of devices according to RAM and storage.

Table 3.1: Classes of constrained devices

Name	Data size (e.g. RAM)	Code size (e.g. Flash)
Class 0, C0	<<10 KB	<<100 KB
Class 1, C1	$\sim$ 10 KB	$\sim$ 100 KB
Class 2, C2	$\sim$ 50 KB	$\sim$ 250 KB

  - Class 0:
    * Devices are very constrained sensor-like motes
    * Devices do not have the resources required to communicate directly with the Internet in a secure manner
  - Class 1:
    * Devices are very quite constrained and cannot easily connect to the Internet
    * Devices can connect to the Internet in a secure manner using a constrained protocol stack (e.g. UDP [12], CoAP[13]).
  - Class 2:
    * Devices are less constrained and fundamentally capable of connecting to the Internet in a secure manner using a full feature and reliable protocol stack (e.g. TCP [14], HTTP [15]).

- Strategies for power usage: Devices also differ in power and the way they consume power for communication. The general strategies for using power for communication can be describe as in Table 3.2

Table 3.2: Strategies of Using Power for Communication

Name	Strategy	Ability to communicate
P0	Normally-off	Reattach when required
P1	Low-power	Appears connected, perhaps with high latency
P2	Always-on	Always connected

  - Normally-off: Devices sleep such long periods at a time that once it wakes up. Devices will reattach to the network as it is woken up.

- Low-power: This strategy is most applicable to devices that need to operate on a very small amount of power but still need to be able to communicate on a relatively frequent basis.
- Always-on: This strategy is most applicable if there is no reason for extreme measures for power saving. The device can stay on in the usual manner all the time.

3.2 Current management approaches

In [2], authors classified options for the management of networks of constrained devices into :

- Hierarchical management: A hierarchy of networks with constrained devices are managed by the managers at their corresponding hierarchy level. That is, each manager is responsible for managing the nodes in its sub-network. It passes information from its sub-network to its higher-level manager and disseminates management functions received from the higher-level manager to its sub-network. Hierarchical management is essentially a scalability mechanism, logically the decision-making may be still centralized.

- Distributed management: A network of constrained devices is managed by more than one manager. Each manager controls a sub-network and may communicate directly with other manager stations in a cooperative fashion. The distributed management may be weakly distributed, where functions are broken down and assigned to many managers dynamically, or strongly distributed, where almost all managed things have embedded management functionality and explicit management disappears, which usually comes with the price that the strongly distributed management logic now needs to be managed.

- Centralized management: A network of constrained devices managed by one central manager. A logically centralized management might be implemented in a hierarchical fashion for scalability and robustness reasons. The manager and the management application logic might have a gateway/ proxy in between or might be on different nodes in different networks, e.g., management application running on a cloud server.

In particular, there are two common approaches for managing devices in the HN: direct management and indirect management. The main difference between them is the involvement of the HGW as described in Figure 3.5.

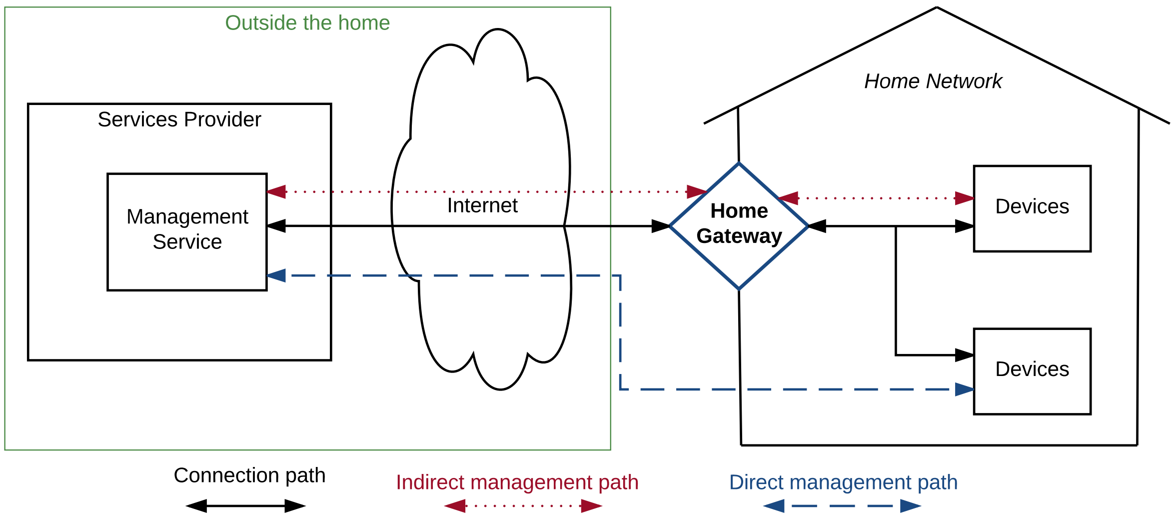

Figure 3.5: Home network services architecture

3.2.1 Direct management

Direct management enables management applications manage devices in the HN directly without any involvement of the HGW.In this approach, the management application and the management agent communicate directly, without the need for intermediate processing of data by the HGW. Thus simplifying the design of the HGW and achieving better performance. For devices that primarily exchange real-time sensory and control data in small but numerous messages, direct management should be preferred due to the aforementioned advantages.

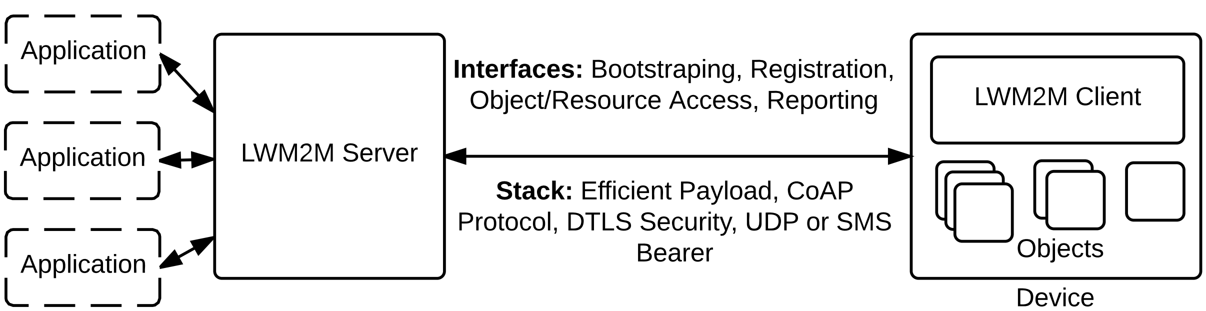

Figure 3.6: Direct management example (OMA LWM2M architecture)

OMA LWM2M [17] is a standard focused on constrained devices. To be applied this architecture, devices must have enough resources to connect to the internet in a secure manner (from Class 1 or above in Table 3.1). Devices must maintain the connection with the management server so it requires more power consumption but the latency is low.

3.2.2 Indirect management

In indirect management, devices in the HN are managed by a hierarchical topology via the HGW. By utilizing the HGW, devices that are really constrained or utilize incompatible communication protocols can connect to the network in a secure manner. Therefore, indirect management enables management services for multiple device classes and communication protocols. In large scale network,the HGW enables management of devices as a group, thus simplifying maintenance and configuration and improving management scalability.

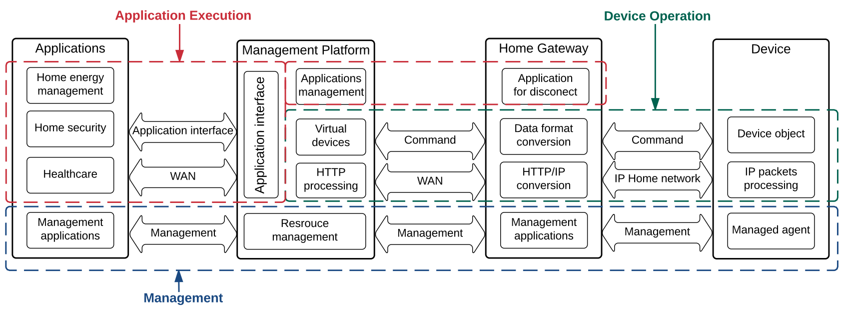

Figure 3.7: Functional architecture for IP based device

The example of indirect management which is referred from [16] is depicted in Figure 3.7. The HGW has a function gather information of the HN resources which the managed agent collects. It also manages the internal status of the device, the network device and the network capacity for management. The MP accesses the HGW placed in home, manages the devices as the function of resource management and enables applications to monitor and control physical devices connected to the HN as logical devices. This architecture enables management services to very constrained (Class 0 in Table 3.1) and low-power devices but the latency between devices and management services is high.

3.3 Summary

There are two common approaches in management: direct management and indirect management and the main difference between them is the involvement of the HGW. Indirect management utilizes the HGW to support very constrained devices and enable local management. In contrast, direct management supports direct end-to-end management thus it requires devices with high capabilities of direct communication and processing power. The merits of direct and indirect management are summarized as in Figure 3.8.

	Direct Management	Indirect Management
Pros	Low latency (Between devices and management services)	Applicable for very constrained devices
	Optimized for less constrained devices	Low cost (device and energy cost)
	No home gateway bottle neck issues	High management scalability
Cons	Cannot support very constrained devices	High latency (Between devices and management services)

Figure 3.8: Merits of direct and indirect management

Devices in the HN are heterogeneous due to they are based on different hardware platforms and communication technologies. In scope of this study, I consider the heterogeneity of devices in terms of communication models and device capabilities. Devices with different profiles should be managed in different approaches to achieve the best performance. Devices and their recommended management approaches are shown in the Table 3.3.

Table 3.3: Devices and corresponding management approaches

	Indirect management	Direct Management
Suitable devices	- Device to Device communication model supported devices - Device to Gateway communication model supported devices - Class 0 devices, Class 1 devices - Normally-off devices, Low-power devices	- Device to Cloud communication model supported devices - Back-end data sharing communication model supporteddevices - Class 0 devices, Class 1 devices - Always-on devices

To handle with device heterogeneity, we need to combine current management approaches to benefit and apply their advantages to manage devices in the smart home. To achieve this goal, the proposed architecture is based on the HN service architecture in ITU-T recommendation [16] as in Figure 3.9.

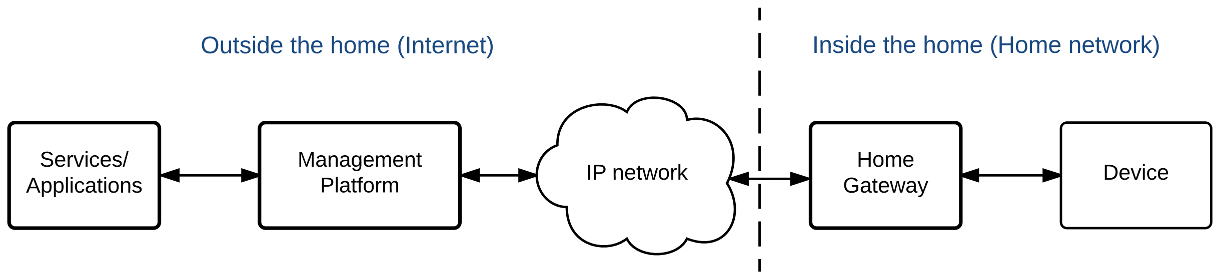

Figure 3.9: Home Network service architecture

In this architecture, devices are managed using indirect management approach with the involvement of the HGW. I extend this architecture by adding direct management approach.

Chapter 4

Design Options and Proposed Architecture

4.1 Overview

Basically, the proposed management architecture is based on the recommended HN service architecture in Chapter 2 with combination of both direct management and indirect management approaches. The management service architecture is depicted as in the Figure 4.1.

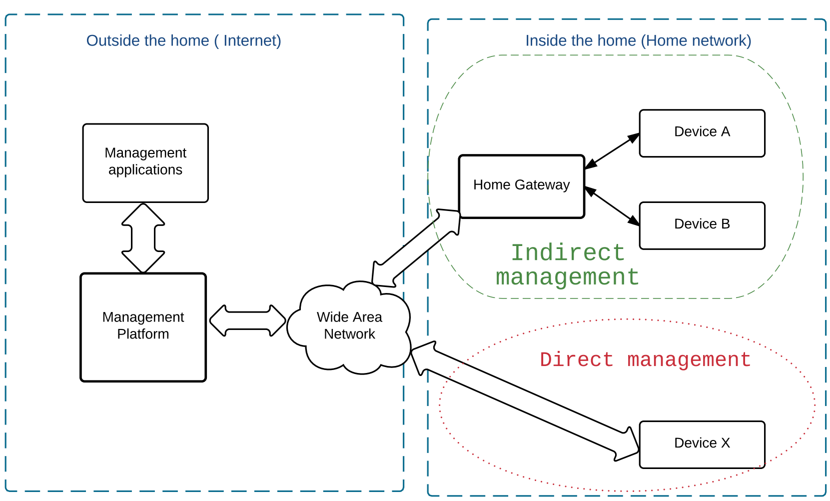

Figure 4.1: Management architecture based on HN service architecture

We considered that indirect management is suitable for very constrained devices and the HGW must carry management tasks. Due to this, devices must communicate to the HGW to exchange the data. The traffic between devices and HGW will be really busy because all devices send packets to the HGW and number of packets is a large number (Due to the constraints in processing power, devices device can not exchange all resources in one packet). Meanwhile, direct management supports end-to-end communication thus the delay time between the MP and device is smaller. However, direct management has limitations to handle the large number of devices connected concurrently.

To present a management architecture, we have to clarify the HGW operation model, how can the MP directly manage devices, how can the HGW locally manage the HN and in which approach the HGW syncs HN's data to the MP, which devices should be indirectly managed or directly managed.

4.2 Home Gateway design options

There are two common approaches to implement a HGW: simple gateway and intelligent gateway as in [18]

- Simple gateway: In general, a simple gateway organizes and packetizes the data for transport over the Internet. It is also responsible for distributing data back to end points in applications where two-way communications is advantageous or required. Using a tunnel to exchange data or making a translation proxy are two sample instances of this approach.

- Intelligent gateway: An intelligent gateway extends the functionality of a simple gateway by providing processing resources and intelligence for handling local applications. Using intelligent applications for local management and communication is one instance of this approach.

4.2.1 Tunnelling approach

In this approach, HGW acts as a forwarder to forward data acquired from devices in the HN to the MP and vice versa without data processing. Because the HGW is released from processing data, the HGW is lightweight, easy to implement and low cost. The data processing and transmitting cost is high due to data processing is executed at the MP and sometime useless data also is transmitted and processed. This approach also lacks of interoperability because data is not in any common format.

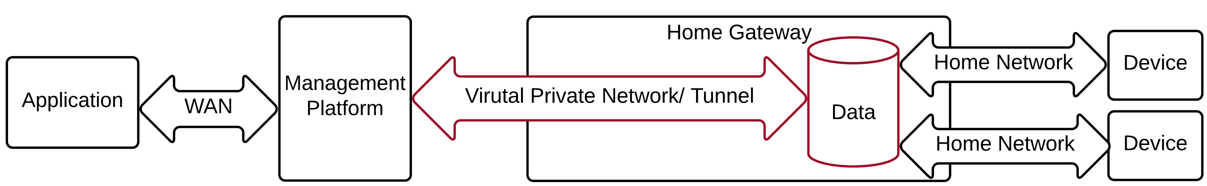

Figure 4.2: Tunnelling approach

Advantages

- Simple and easy to implement

- The HGW is lightweight

Disadvantages

- High cost for data transmission and processing at the MP
- Can not support local management.

4.2.2 Translation approach

Instead of using a tunnel to directly transmit data to the MP, this approach manages data from the HGW using management protocols or standards. In most instance, there is a difference between two protocols use for managing the HN and the HGW. Due to this, HGW acts as a translation proxy between different protocols or standards. By utilizing the protocols or standards, we can inherit built-in functions for management then simplifying the implementation process. However, by using a translation proxy, only common attributes can be translated and transmitted, it narrows amount of information can be managed.

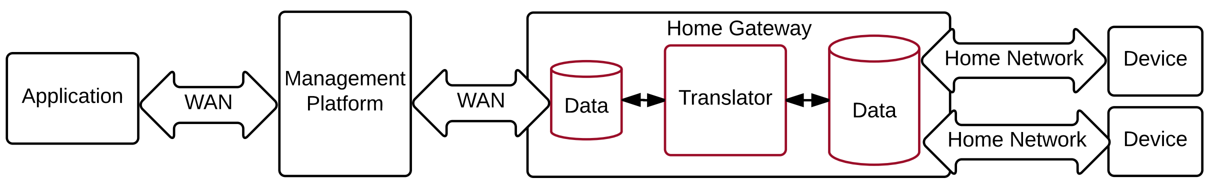

Figure 4.3: Translation approach

Advantages

- Inherit build-in functions for management.
- Improve consistency of the MP.

Disadvantages

- Not all of data can be translated.
- Lack of local management.

4.2.3 Intelligent application approach

In this approach, the HGW could evaluate and filter data from the HN. After evaluating data, HGW could determine whether a critical threshold has been passed. If so, HGW can produce some action to locally handle this problem or ask the MP help by sending the information to the MP. Enabling intelligence in a gateway addresses both interoperability issues on a local level while minimizing the changes required to connect appliances.

Rather than require full intelligence in each appliance, the gateway can provide the base intelligence for all devices.

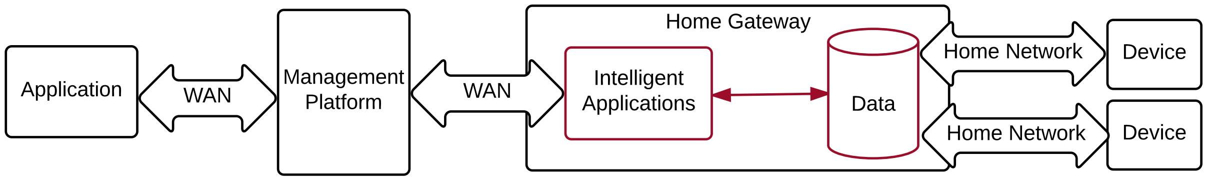

Figure 4.4: Intelligent application approach

Advantages

- Ability to support local management
- The Management Platform is lightweight

Disadvantages

- Implementation cost of the HGW is high (hardware and power cost)
- The HGW is burdened

4.3 Proposed architecture

Our proposed method using intelligent application approach for management. The architecture is describe as in Figure 4.5.

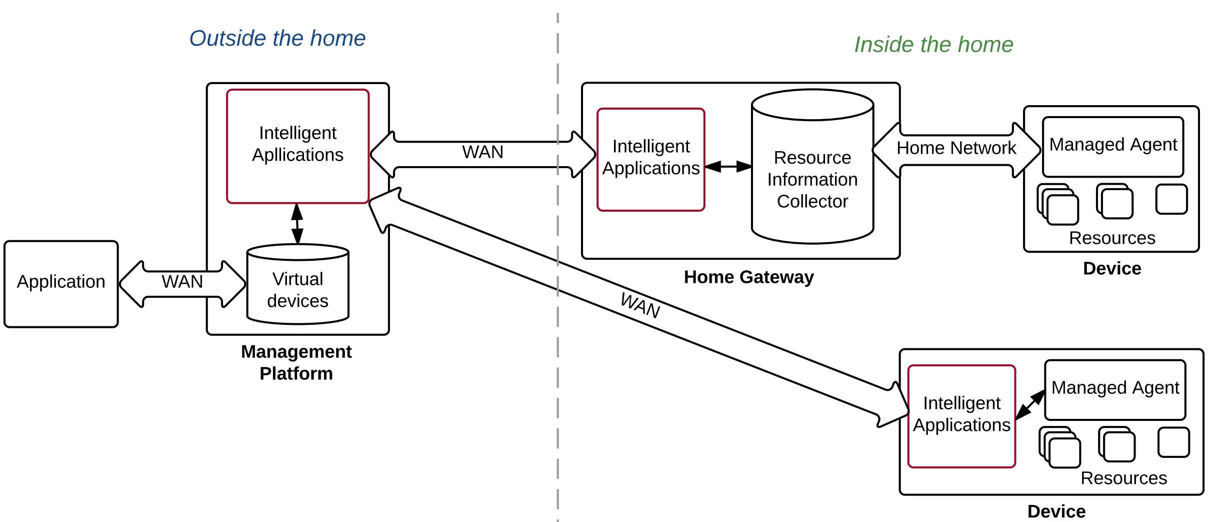

Figure 4.5: Proposed architecture

We classified devices into two categories:

- Devices which are indirectly managed (indirectly managed devices) are very constrained (belong to Class 0 in Table 3.1) , limited power budget (Normally-off or Low-power device in Table 3.2) and belonged to Device-to-Device or Device-to-Gateway communication model.
- Devices which are directly managed (directly managed devices) are less constrained (belong to Class 1 or higher in Table 3.1), non limited power budget (Always-on or attached to a power source devices) and belonged to Device-to-Cloud or Back-End-Data-Sharing communication model.

Each devices have each own Manged Agent. The managed agent on the device executes configuring and gathering device information. Indirectly managed devices are connected to the HGW and provide information or execute instruction to and from the Resource information collector in the HGW.

The HGW has Intelligent Applications (IAs) which process data from the Resource information collector to provide management functions. The IAs at the HGW connect to the IAs at the MP to exchange data. After that data is stored into a database as virtual devices. Virtual devices data is provided to the applications or services to be treated as web resource.

Directly managed devices are not managed by the HGW but directly by the MP. Each devices has each own Managed agent to collect resource. The Intelligent Application of devices handle the collected data for local management and communicate with the MP. The deployment diagram of this proposed architecture is as in Figure 4.6.

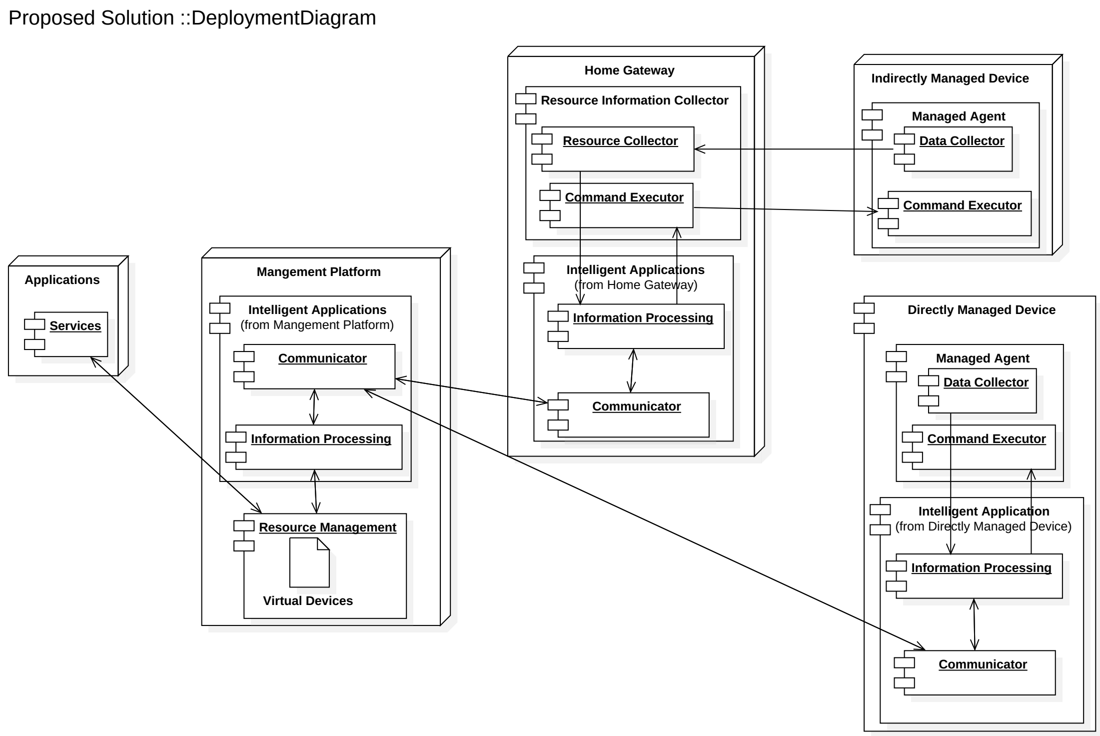

Figure 4.6: Proposed architecture: Deployment diagram

IAs are agents to process data, provide management functions and exchange data. There are three kinds of IAs:

- Intelligent Applications from Device: Due to devices are still resource constrained devices, they can not support applications that required large memory or processing power. IAs at this category can support functions related to self management such as: self management, error detection or fault recovery (e.g. very simple fault that can be recovered by rebooting device), etc.

- Intelligent Applications from the HGW: The HGW is more powerful than devices. IAs can process large amount of data to provide management functions. However, the HGW acts as a bridge between the HN and the internet, it can not allocate all resource for the IAs to mitigate the bottle effect. IAs can support function to local HN management such as: Automatic management, fault detection and recovery (e.g. faults that can be recovered after simple investigation), etc.

- Intelligent Applications from the MP: Due to IAs are deployed in cloud server, there is no limit on power consumption or processing power. they can cooperate with other services provider to process data and provide management functions. IAs are able to support smart management fuctions that need a lot of resource such as: Exception prediction, automatic management, etc.

Some examples of IAs that can be implemented are depicted in Figure 4.7.

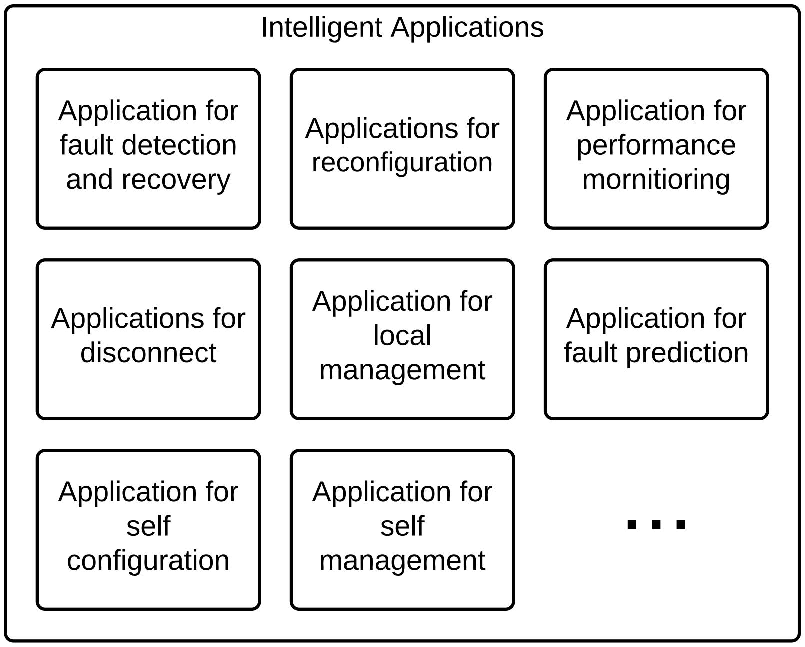

Figure 4.7: Example of intelligent applications

4.3.1 Protocols for indirect management

Devices which are indirectly managed are very constrained devices, they need to be managed using lightweight protocols or standards. In smart home environment, many standardization activities have been done for managing and controlling HN and devices. Such activities include UPnP([19]), SNMP ([6]), NETCONF ([7]), ECHONET ([20]) for home device management; LLDP ([21]), HTIP ([22]), WMI([23]) for home network management; IEC62608 for multimedia home network configuration; etc. These protocols are applicable and have each own advantages and disadvantages, we can select one depend on real situations.

4.3.2 Protocols for direct management

Recently, many standards related to remote device management are being introduced. It leads to a future that direct management is also applicable for managing HN devices. CWMP ([24]) and SNMP ([6]) are well known as management protocols that can be used for remote device management. In mobile device management, OMA DM ([25])is widely applied for managing mobile devices and OMA LWM2M ([17])is developed based on OMA DM customized for constrained devices. Another option is application layer protocol that intent to use in resource constrained devices such as: CoAP , XMPP ([26]), MQTT ([27]), etc. All of these protocols/ standards can be considered as a solution for direct management.

4.4 Summary

The proposed management architecture is based on HN service architecture recommended by I-TUT in [16]. I also considered about the HGW architecture because HGW is an essential part of the system and the implementation of the HGW affects to the whole system. There are two common approaches in designing a HGW: simple gateway and intelligent gateway. I made a qualitative analysis about instances of two above approaches such as: tunnelling approach, translation approach and using intelligent application approach. A brief summary is as below:

- Tunnelling approach: Due to lack of local management, a large number of packet will be exchanged between the HGW and the MP. It also lacks of interoperability with other protocols.

- Translation approach: This approach improves the interoperability of the HN by translating data into a common format. However, due to lack of local management, meaningless data still be exchanged and not all of information can be translated (common information of two protocols only).

- Intelligent Application approach: By having application to process data, this approach can dramatically reduce the exchanging of meaningless messages. However, the cost for the HGW is high (power and processing cost) and higher delay time between devices and the management platform.

I proposed the management architecture using intelligent gateway. We can overcome the delay time by having time-sensitive devices directly managed. The HGW is an always-on devices connected to a power source so the cost related problems can be ignored.

Chapter 5

Implementation

5.1 System overview

To validate the feasibility of the proposed architecture, a prototype was implemented. This prototype combines both direct and indirect management using intelligent application approach for management. The overview of this prototype is addressed in Figure 5.1.

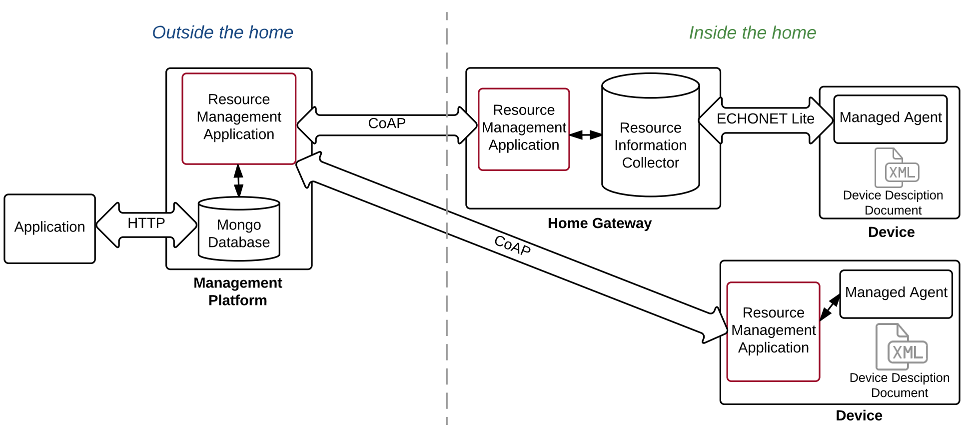

Figure 5.1: Implemented architecture

This prototype is one instance of the proposed architecture with some specified information as bellow:

- ECHONET Lite is the protocol for indirect management.
- CoAP is the protocol for direct management.
- Resource Management Application is an example of intelligent application.

The overview of functional architecture is as in Figure 5.2.

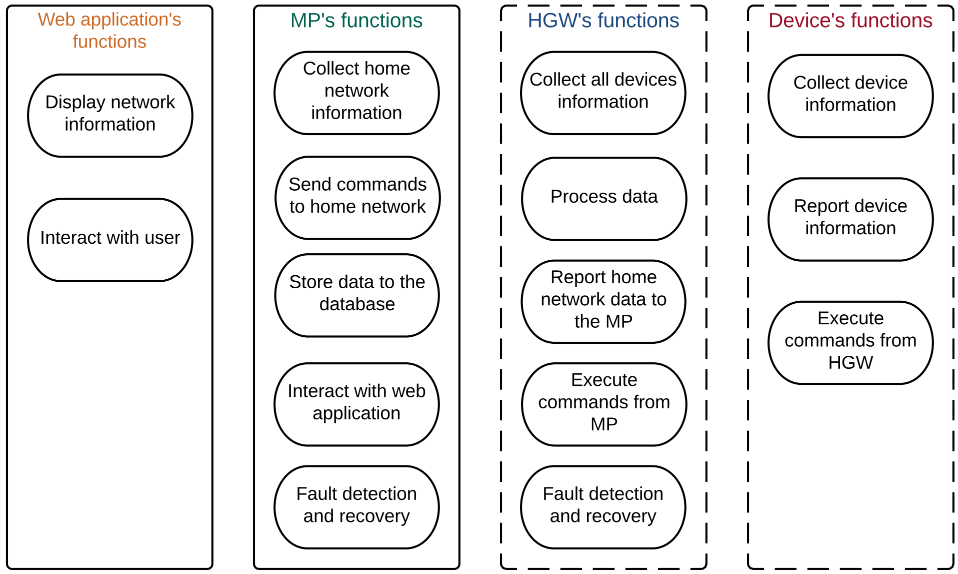

Figure 5.2: List of functions

5.1.1 Devices

We have two kinds of device: ECHONET Lite device and CoAP enabled device as in Figure 5.3.

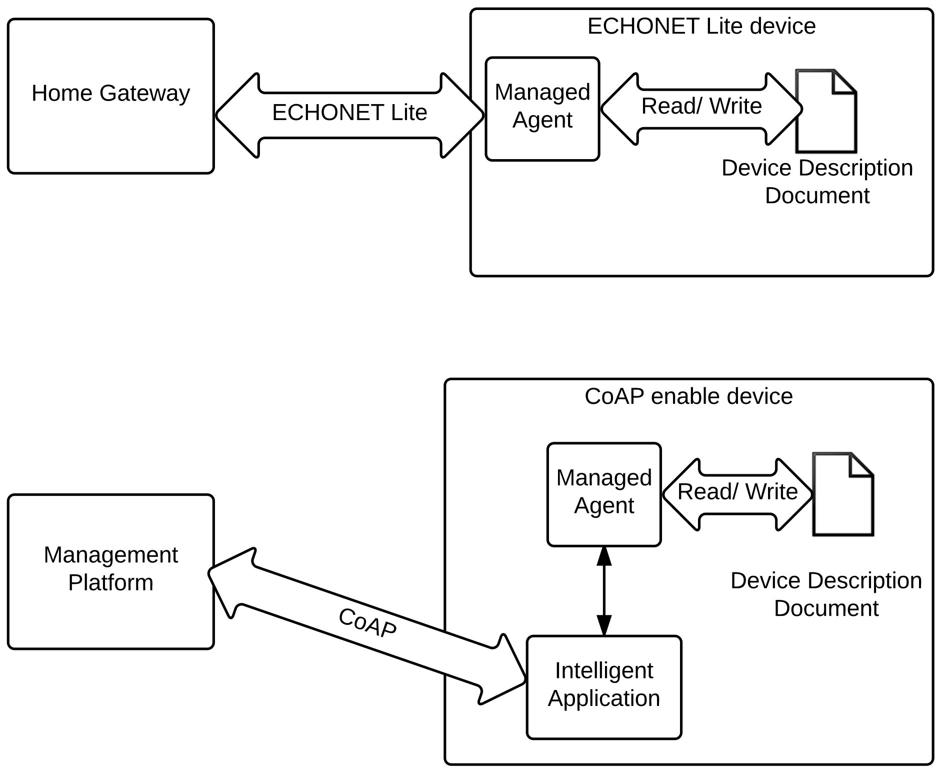

Figure 5.3: Device overview

Each devices has a Device Description Document (DDD) which is an XML document contains device resource. Managed Agent interacts with device through this DDD. The managed agent on the device executes configuring and gathering the home environment information by the instruction from the resource information collector function on the HGW. The DDD are constructed as in Table 5.1.

Table 5.1: Device management attributes

Property name	Data type	Access Rule	Mandatory	Observable
Device name	unsigned char	GET	YES	NO
Device IP	unsigned char	GET	YES	NO
Operation status	unsigned char	GET/ SET	YES	YES
Installation location	unsigned char	GET/ SET	YES	YES
Standard version information	unsigned char	GET	YES	NO
Identification number	unsigned char	GET	NO	NO
Instantaneous power consumption	unsigned short	GET	NO	NO
Cumulative power consumption	unsigned long	GET	NO	NO
Manufacturers fault code	unsigned char	GET	NO	NO
Current limit setting	unsigned char	GET/ SET	NO	NO
Fault status	unsigned char	GET	YES	YES
Fault description	unsigned short	GET	NO	NO
Manufacturer code	unsigned char	GET	YES	NO
Business facility code	unsigned char	GET	NO	NO
Product code	unsigned char	GET	NO	NO
Production number	unsigned char	GET	NO	NO
Production date	unsigned char	GET	NO	NO
Power-saving operation setting	unsigned char	GET/ SET	NO	NO
Remote control setting	unsigned char	GET/ SET	NO	NO
Current time setting	unsigned char	GET/ SET	NO	NO
Current date setting	unsigned char	GET/ SET	NO	NO
Power limit setting	unsigned short	GET/ SET	NO	NO
Cumulative operating time	unsigned char	GET	NO	NO

As this table, it is mandatory to implement "Operation status" in all device object as a gettable and settable property. And the device can notify the MP when there is any change in the "Operation status". ECHONET Lite devices and CoAP-enabled devices are different in the way these attributes are stored in the DDD.

ECHONET Lite devices

Device resources are followed by ECHONET Consortium specification [20]. Property names are described as ECHONET Property (EPC) and the corresponding values are ECHONET Property Value Data (EDT). Each attribute and corresponding data is a **EPC**, **EDT** tuple.

Listing 5.1: Sample Device Description Document of ECHONET Lite device

```
<?xml version="1.0" encoding="UTF-8"?>
<device>
  <profile>
    <!-- Operation Status -->
      <property epc="80" notify="enabled" set="enabled">
```

```
      <data type="variable">30</data>
    </property>
    <!-- Installation Location -->
    <property epc="81" notify="enabled" set="enabled">
      <data type="variable">00001000</data>
    </property>
    <!-- Version Information -->
    <property epc="82">
      <data type="variable">00001000</data>
    </property>
    <!-- Fault Status -->
    <property epc="88" notify="enabled" set="enabled">
      <data type="variable">41</data>
    </property>
    <!-- Fault description -->
    <property epc="89" notify="enabled" set="enabled">
      <data type="variable">00</data>
    </property>
    <!-- Manufacturer fault code -->
    <property epc="8A" notify="enabled" set="enabled">
      <data type="variable">00</data>
    </property>
 </profile>
</device>
```

In above example, **epc = ”80”** indicates the operation status of device and textbf30 means that this device is ON as the ECHONET Lite specification.

CoAP enabled devices

Due to devices are less constrained devices, attributes and data do not need to be encoded. The DDD for CoAP-enabled devices is as below:

Listing 5.2: Sample Device Description Document of CoAP enabled device

```
<?xml version="1.0" encoding="UTF-8"?>
<device>
  <object Name="TemperatureSensor">
   <property>
      <IP_Adress>"150.65.230.114"</IP_Adress>
      <type>"variable"</type>
   </property>
      <property>
         <OperationStatus>"ON"</OperationStatus>
         <set>"enabled"</set>
         <notify>"enabled"</notify>
         <type>"variable"</type>
      </property>
      <property>
         <InstallationLocation>"Living Room"</InstallationLocation>
         <set>"enabled"</set>
         <notify>"enabled"</notify>
```

```
            <type>"variable"</type>
        </property>
        <property>
            <VersionInfor>"1.1"</VersionInfor>
            <type>"variable"</type>
        </property>
        <property>
            <Fault_Status>"No Fault"</Fault_Status>
            <notify>"enabled"</notify>
            <type>"variable"</type>
        </property>
    <!-- Optional Property -->
        <property>
            <Fault_Description>"0000"</Fault_Description>
            <type>"variable"</type>
        </property>
        <property>
            <Manufacturer_Code>"TOSHIBA"</Manufacturer_Code>
            <type>"variable"</type>
        </property>
    </object>
</device>
```

In above example, we have a temperature sensor installed in the living room. The sensor is on and work properly. Device IP address can be obtained at runtime.

5.1.2 Home Gateway

The overview of HGW is depicted in Figure 5.4.

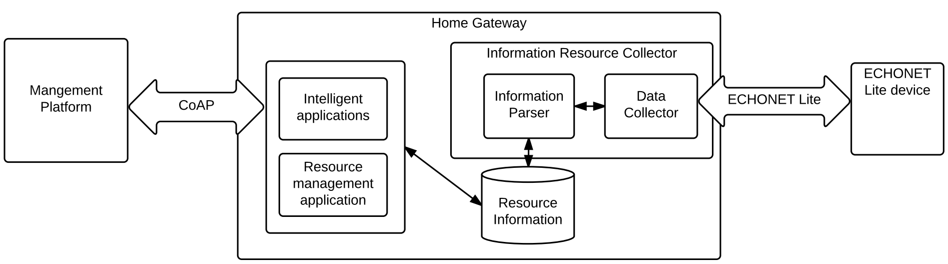

Figure 5.4: Home Gateway overview

The HGW communicates with devices in the HN using ECHONET Lite protocol. The Information Resource Collector has functions to collect and convert ECHONET Lite data into readable data and also convert commands, configuration information then apply to devices.

Resource information will be process by Management Application to provide management functions such as: monitoring devices, configuring devices, observing devices, etc.

Application for fault detection and recovery is an application for local fault detection and recovery by analysing device resource information.

The HGW communicate with the MP to exchange data for management and interact with users using CoAP protocol.

5.1.3 Management Platform

The overview of the MP is depicted in Figure 5.5.

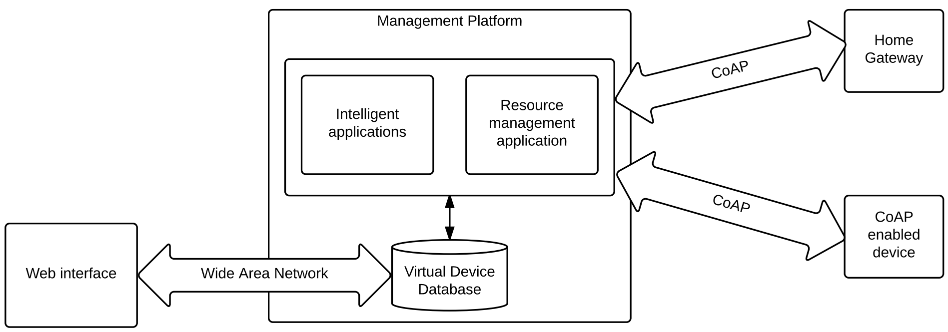

Figure 5.5: Management Platform overview

The resource management on the MP provides the function to gather information of the HN resources which the managed agent directly sent to it or passed to the HGW . It also manages the internal status of the device, the network device and the network capacity for each of the HGW. The information is stored into the database as virtual devices.

Application for fault detection and recovery process fault data passed from the HGW. It has functions to learn from history information and cooperate with other services.

5.1.4 Web interface

Web interface interacts with the database at the MP to provide the graphic user interface (GUI) for users. It also provides functions to allow users interact with the HN devices.

5.2 Management Interfaces

5.2.1 Device registration

To be managed, devices need to register to the MP directly or via the HGW. The device registration flow is depicted in Figure 5.6.

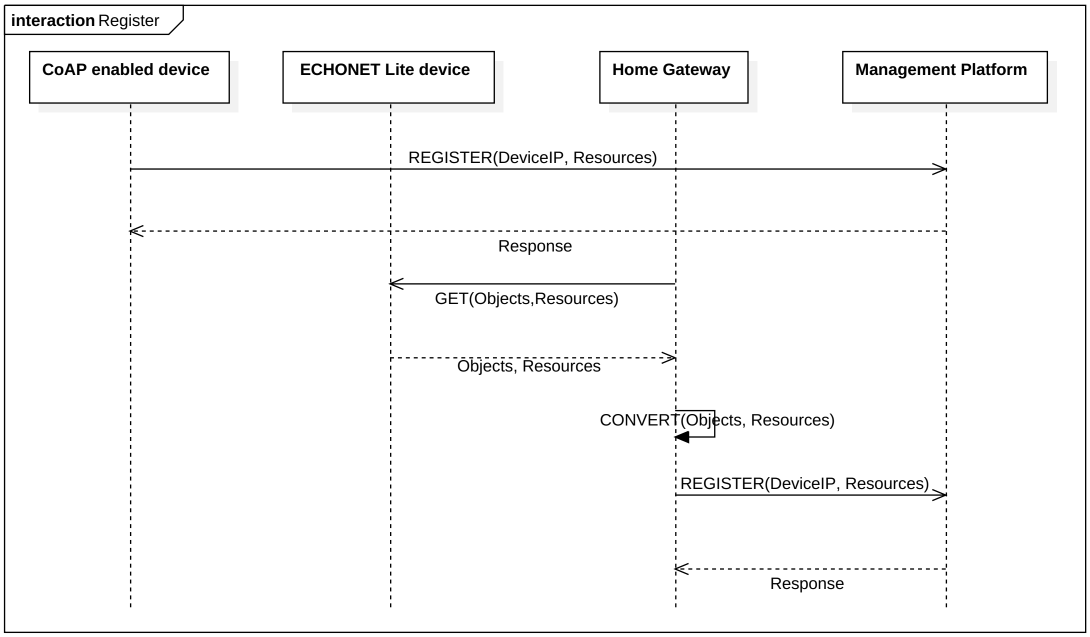

Figure 5.6: Device registration flow

5.2.2 Update resources

The update resources interface is almost the same to get resources interface but the information conversion process is opposite. The resources must be translated into EPC and EDT before applying to the ECHONET Lite devices.

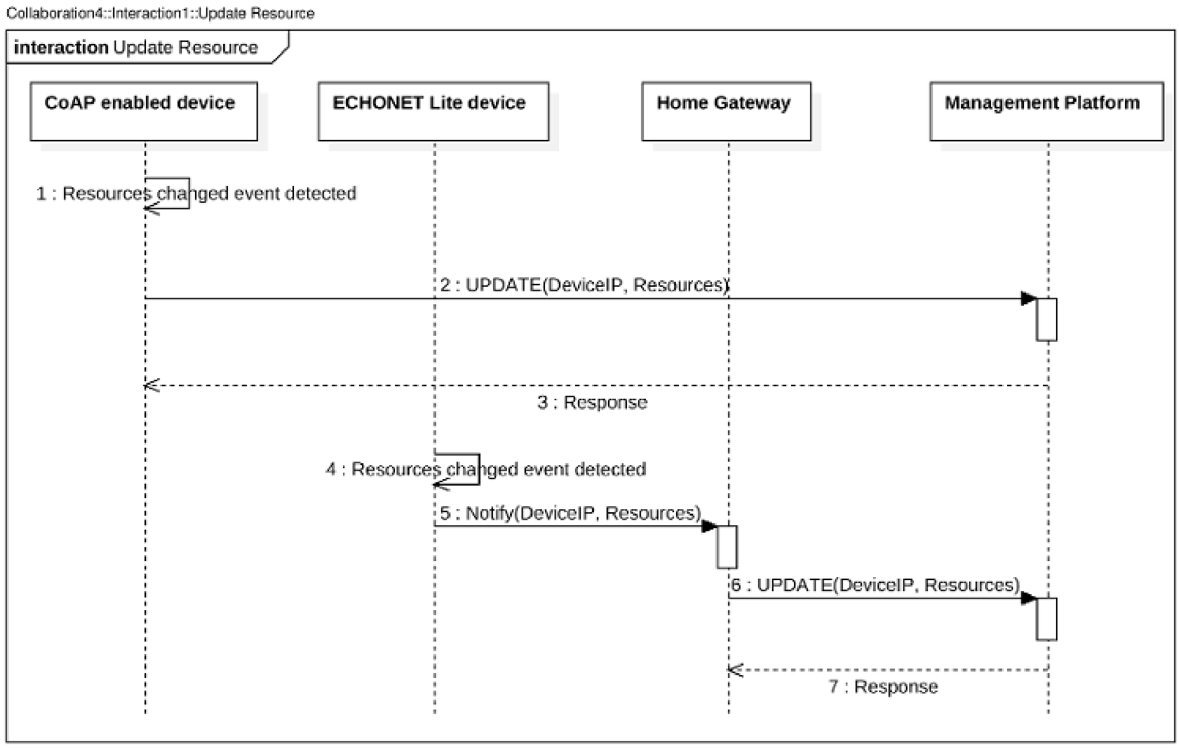

Figure 5.7: Update device resources flow

5.3 Management Flows

5.3.1 CoAP enabled devices

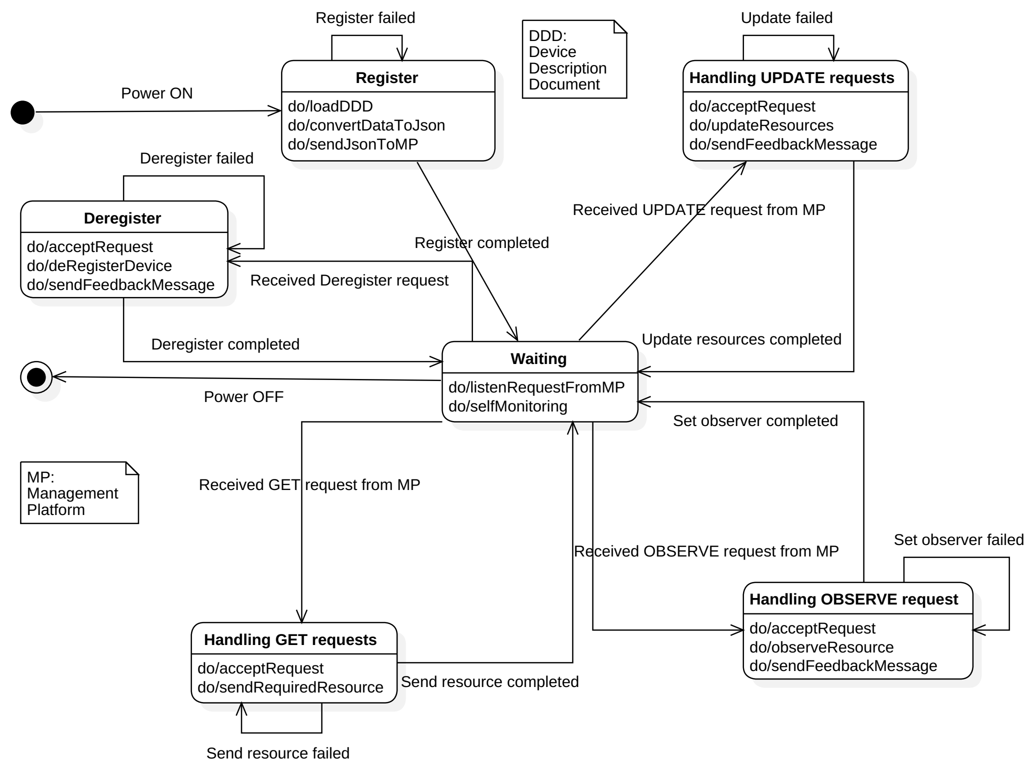

Figure 5.8: CoAP enabled device management scenario

5.3.2 Home Gateway main loop

The HGW main loop is as follow :

- Bootstrapping HN resources to the MP at the start up time.
- Monitoring HN devices resources
  - New added device: Get this device resource and register to the MP.
  - Resources changed devices : Update changed resources to the MP.
- Subscribing to the MP for message exchanges
  - GET request: Get specified device resource and post to the MP.
  - SET request: Update resources to specified devices.

- Observing devices resource
    - Data changed:Update device resource to the MP.

5.3.3 Management Platform main loop

The MP's main loop is as below:

- Initiating CoAP server.
- Add server endpoints.
- Add server resources.
    - Registration handler resources.
    - Updating handler resources.
    - Observing handler resources. /
    - Monitoring handler resources.
- Initiating mongo database
- Start server

Listing 5.3: Management Platform main loop

```
// Initialize CoAP server
CoapServerEx server = new CoapServerEx();
// Add available endpoints
server.addEndpoints();
// Add handler resources
server.addResources();
// Initialize mongo database
MongoClient mongoClient = MongoUtils.getMongoClient();
// Start the MP
server.start();
```

5.4 Experiment

5.4.1 Overview

The logical network diagram is as in Figure 5.9. The HGW can be divided into 2 parts: Access Gateway and Service Gateway. Access Gateway is a broadband router and Service Gateway is a controller to manage the HN. CoAP enabled devices are directly connected and managed by the MP via the Access Gateway. ECHONET Lite devices are managed by the Service Gateway, information after processing will be managed by the MP.

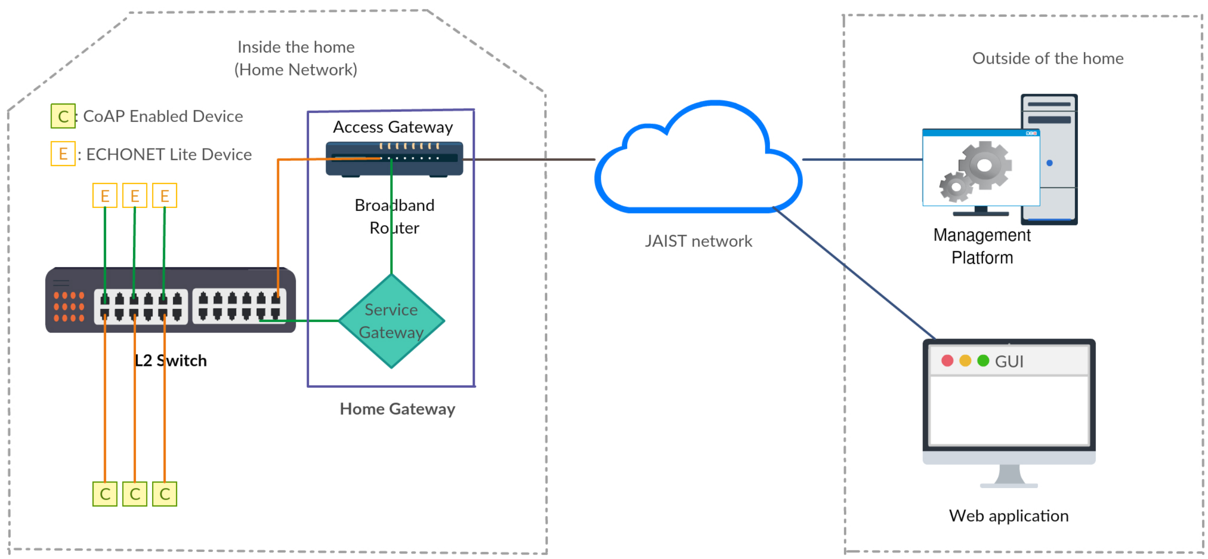

Figure 5.9: Logical network diagram

Detailed hardware and run-time environments are addressed in Figure 5.10.

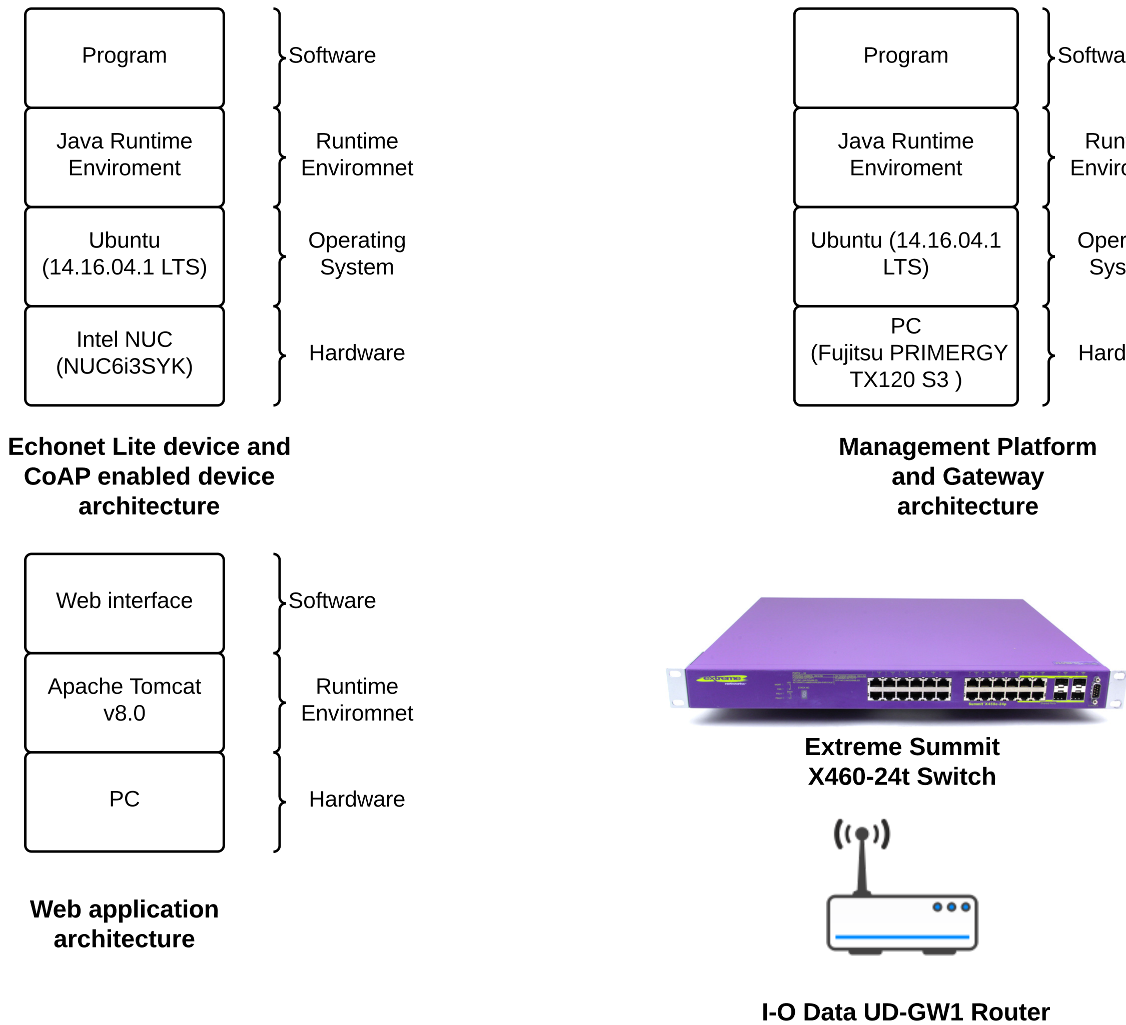

Figure 5.10: Hardware and runtime environments

Note: All programs are written in Java so they need Java Runtime Environment to be executed.

5.4.2 Experiment scenarios

To made a quantitative analysis about simple gateway and intelligent gateway I captured the packet that exchanged from the HGW and the MP using two kinds of HGW. A table that shows number of information that can be translated from ECHONET Lite protocol to OMA LWM2M standard using translation approach.

To prove the feasibility of the intelligent gateway, I measured the round trip time to finish one transaction of each devices, number and size of packets in a network using 3 scenario as below:

- 1. Network contains only ECHONET Lite devices.

- 2. Network contains only CoAP enabled devices.
- 3. Home Network contains both CoAP devices and ECHONET Lite devices

Note: All of devices are using the same attributes as described in Device Description Document and using the same hardware platforms.

5.5 Summary

To prove the feasibility of the proposed architecture, a prototype that manages ECHONET Lite devices and CoAP enabled devices are implemented. To make the qualitative analysis, I captured the packets that exchanged from the HGW and the MP by using intelligent HGW and simple HGW. The intelligent HGW has application to process data before exchanging while the simple gateway does not have this functions. I also made an analysis about number of attributes that can be translated from ECHONET Lite standard to OMA LWM2M standard to prove that all attributes can not be translated if using translation approach.

I also made a comparison between (i) the home network that contains only ECHONET devices, (ii) the home network that contains only CoAP enabled devices and (iii) the home network that combines both of them.

Chapter 6

Result and Evaluation

6.1 Experiment environment

Network diagram of the experiment is described in Figure 6.1.

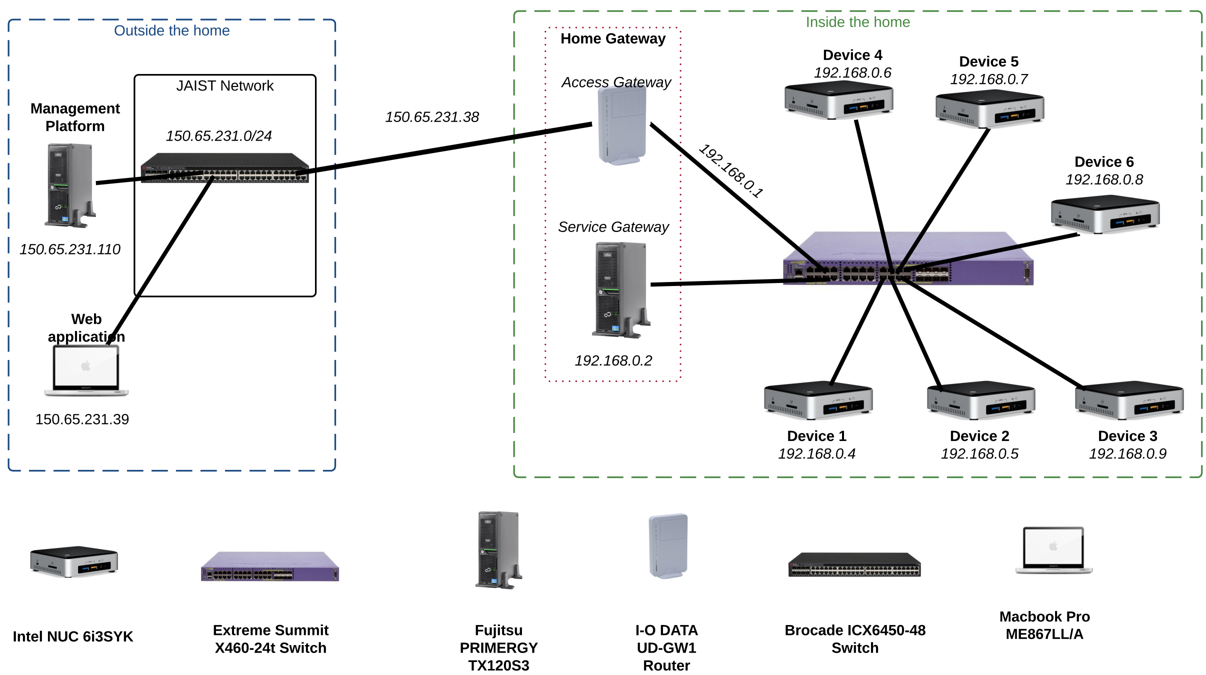

Figure 6.1: Experiment network diagram

Devices, HGW and MP's application are written in Java.

- CoAP enabled devices were developed using Californium [28] framework.
- ECHONET Lite devices were developed using Humming library [30].

- Home Gateway (Service Gateway) are developed with two interfaces. One interface is used for managed ECHONET Lite devices on port **3610**. Another interface is used for communicating with the MP on port **5683**.
- MP were developed using Californium framework.

6.2 Message structure

6.2.1 ECHONET Lite message

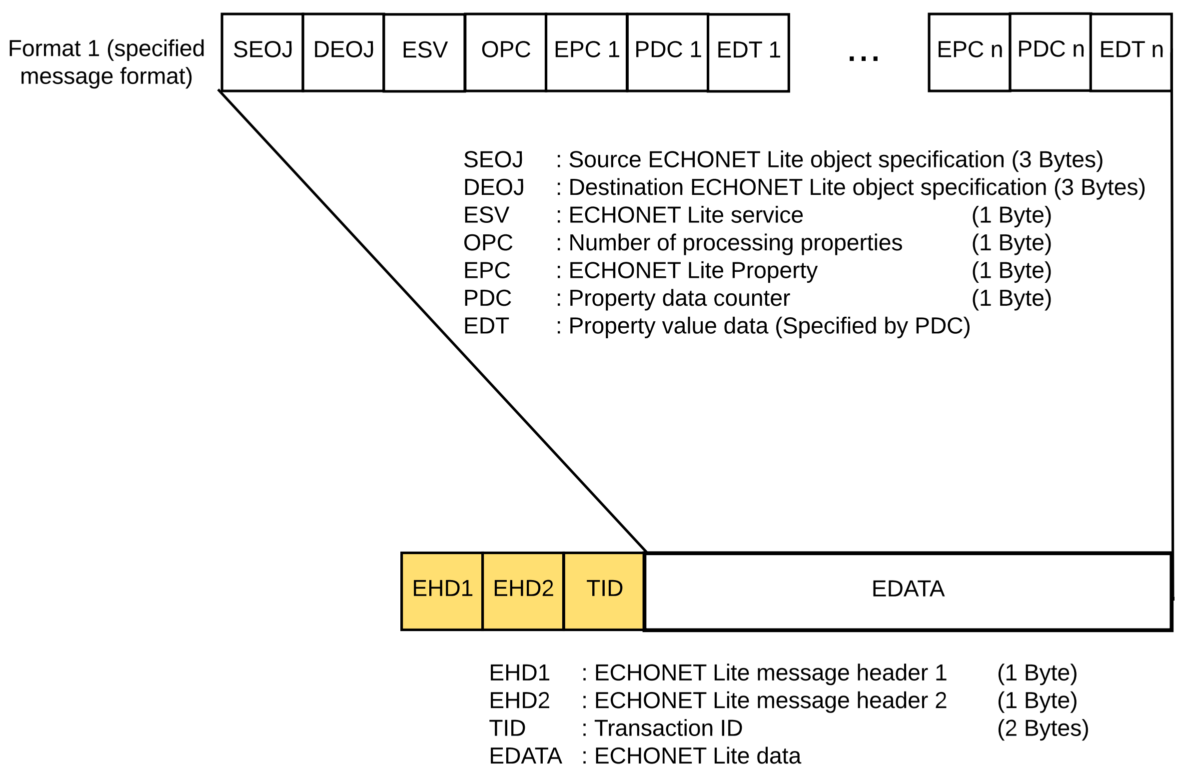

Figure 6.2: ECHONET Lite message formart

ECHONET Lite message struture is described in Figure 6.2.

ECHONET Lite Header 1 is a 1-byte value specifies ECHONET protocol type. EHD1 with value **00010000** indicates ECHONET Lite protocol.

ECHONET Lite Header 2 is a 1-byte value indicates format of EDATA filed. There are two options : **10000010** (EDATA is in arbitrary message format) and **10000001** (EDATA is in Format 1 as describing in Figure 6.2).

TID is a 2-byte transaction ID parameter that matches the request and response.

EDATA is variable-length ECHONET Lite data field of message exchanged between ECHONET Lite devices. The format of this field in this implementation is as in Figure 6.2

- ECHONET Lite Object (EOJ) is a 3-byte field describes the class group code, class code and instance code of devices. E.g. EOJ with value **0x05FD01** indicates the switch 01 belonging to "Management/control-related device class group".
- ESV is a 1-byte value indicates an operation (e.g. GET, SET, NOTIFY) for the specified ECHONET Lite properties.
- OPC is a 1-byte value indicates number of processing properties.
- EPC is a 1-byte field indicates ECHONET Lite property.
- PDC is a 1-byte value indicates the number of bytes in EDT (ECHONET Property Value Data).
- EDT indicates value of corresponding EPC.

6.2.2 CoAP message

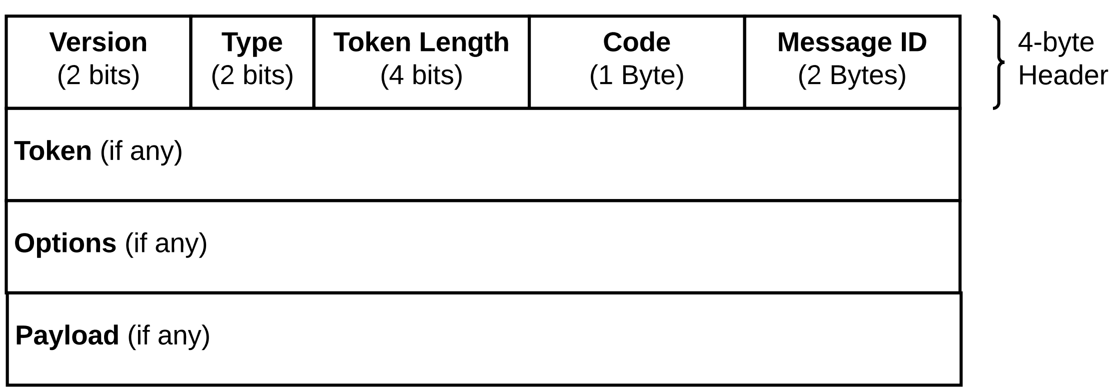

Figure 6.3: CoAP message formart

CoAP message starts with 4-byte header followed by a 0 to 8 bytes Token. Following the Token is Options in Type-Length-Value format followed by a payload.

The header is as follows:

- Version: This is a 2-bit integer that indicates the CoAP version number. In this implementation, Version is set to **01**.
- Type: This is a 2-bit integer indicates message is one of types (i) Confirmable (**00**), (ii) Non-confirmable (**01**), (iii) Acknowledgement(**10**) or (iv) Reset (**11**).

- Token Length: This is an 4-bit integer indicates the length of Token field.
- Code : This is a 8-bit integer (3-bit class and 5-bit detail) documented as **"c.dd"**. **"c"** can indicate (i) a request (**000**) , (ii) a success response (**010**), (iii) a client error response (**100**) or (iv) a server error response (**101**). In case of a request, the Code field indicates the Request Method(**(000.00001, GET)**, **(000.00010, POST)**, **(000.00011, PUT)**, **(000.00100, DELETE)**; in case of a response, a Response Code (**(010.00001, CREATED)**, **(010.00010, DELETED)**, **(010.00011, VALID)**, **(010.00100, CHANGED)**, etc.).
- Message ID: This is a 16-bit integer used to detect message duplication.

Token is a 0 to 8 bytes value used to match a response with a request.

Options are used to specified the target resource of a request to a server, make a request to a forward-proxy, indicate representation format of message payload, etc.

Payload indicates a representation of a resource or the result of the requested action.

6.3 Assumptions

- The HGW collects device resources in the HN each 5 minutes.
- Intelligent Gateway has functions to process device resources before sending resources to the MP (HGW will send HN resources to the MP only if resources were changed) but Simple Gateway does not provide this function.
- ECHONET Lite devices and CoAP enabled devices are using the same Device Description Document (21 management attributes).
- ECHONET Lite devices are able to exchange 1 attribute each packet (to exchange total 21 attributes, ECHONET Lite devices must send 21 packets) due to device's resource constraint.

6.4 Intelligent Gateway versus Simple Gateway

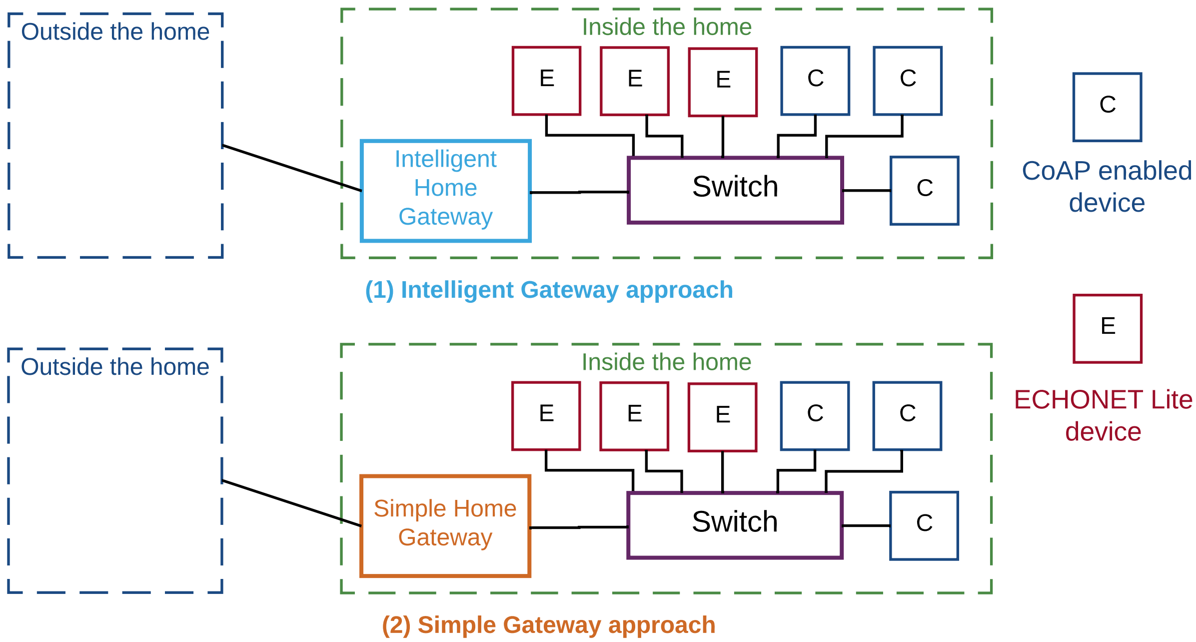

Figure 6.4: Scenario to evaluate simple gateway and intelligent gateway

We measured sent and received packets using 2 kind of Gateway in the experiment environment shown in Figure 6.4. The result was captured by WireShark ([29]) version 2.0.4 installed in the HGW and the MP.

6.4.1 Inside the home

Transmitted/ received bytes and packets in the HN by using Intelligent Gateway and Simple Gateway are depicted in Figure 6.5 and 6.6.

Tx : Transmitted, Rx: Recevied.

bytes

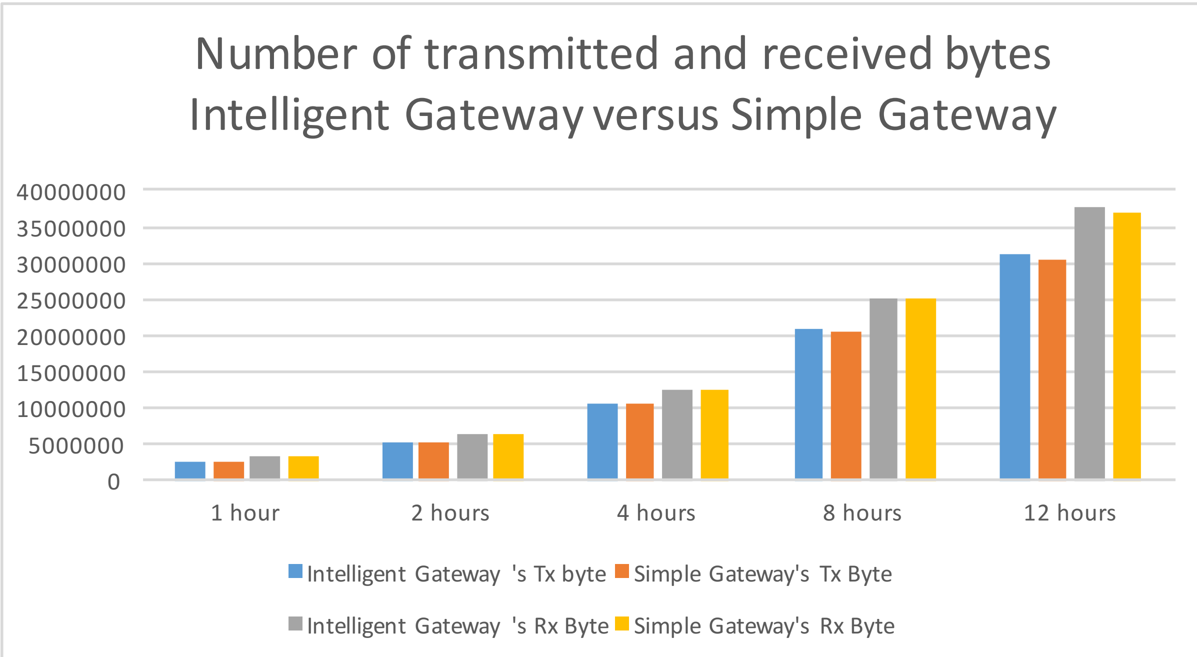

Figure 6.5: Exchanged bytes between devices and home gateway using simple gateway and intelligent gateway

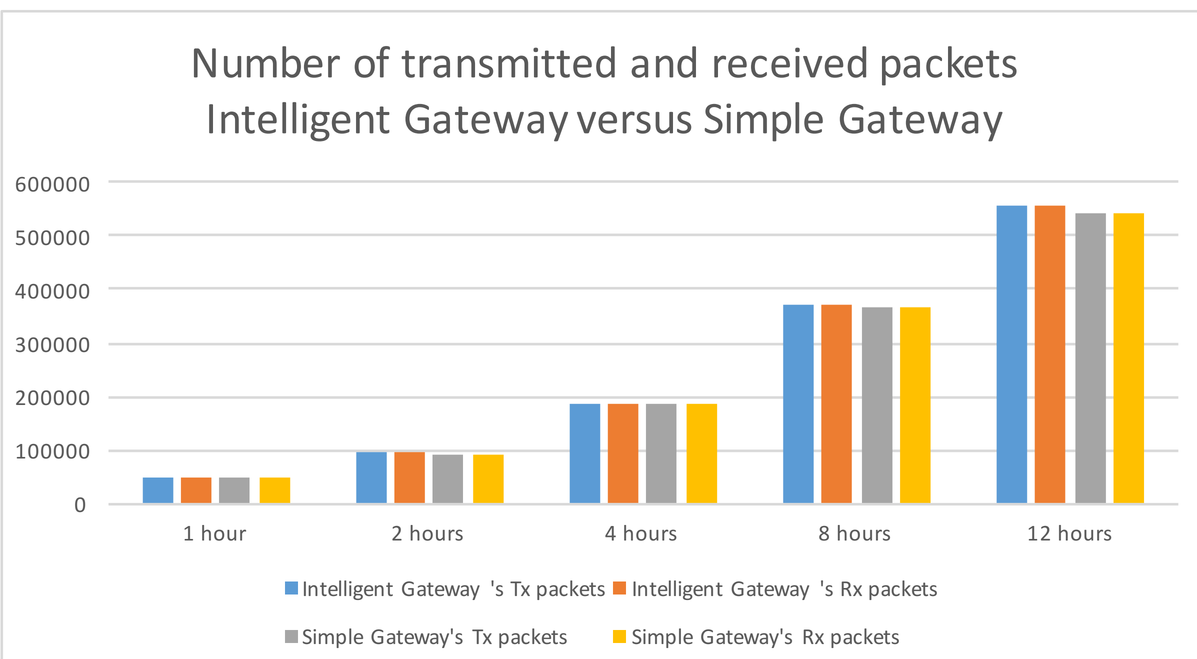

Figure 6.6: Exchanged packets between devices and home gateway using simple gateway and intelligent gateway

We can see that transmitted/ received bytes and packets of two approach are almost the same.

6.4.2 Outside of the home

Transmitted/ received bytes and packets in the HN by using Intelligent Gateway and Simple Gateway are depicted in Figure 6.7 and 6.8.

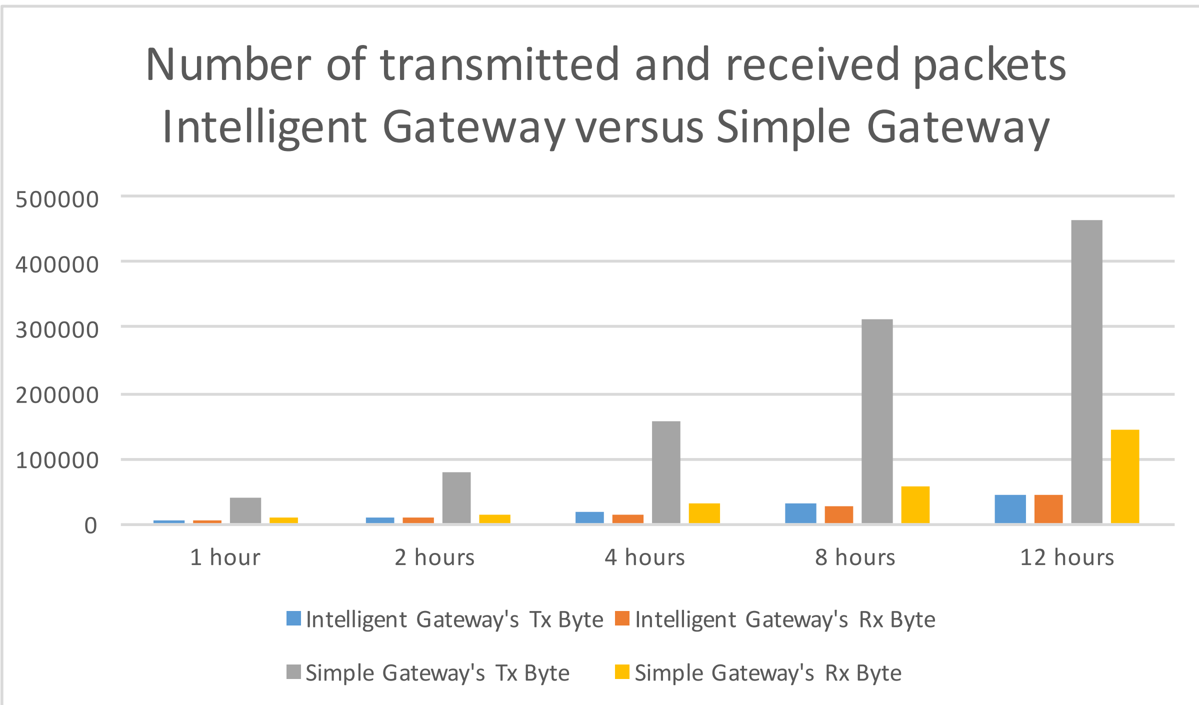

Figure 6.7: Exchanged packets between home gateway and management platform using simple gateway and intelligent gateway

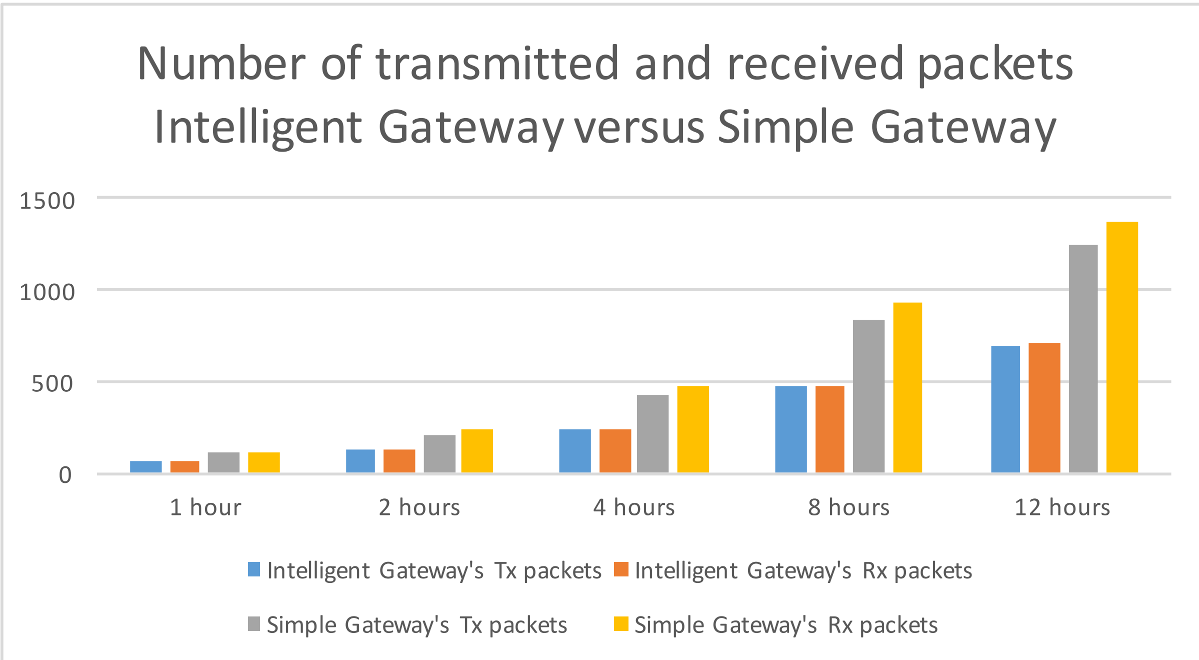

Figure 6.8: Exchanged bytes between home gateway and management platform using simple gateway and intelligent gateway

We can easily see that transmitted/ received bytes and packets by using simple gateway are much higher than the intelligent gateway. The reason is the simple gateway exchanged meaningless information due to lack of local data processing.

6.4.3 Conclusion

By capturing the exchanged information using Intelligent Gateway and Simple Gateway, we can claim due to lack of data processing, Simple Gateway causes dramatically high cost for data transmission and processing at the MP. Therefore, application for local management reduce the transmission cost from the HGW to the MP.

Note: Translation approach also lacks of data processing.

Translation approach is another instance of Simple Gateway. We did not analysis this approach by capturing the exchanged packets but we can claim that translation also faces the same problem in transmitting meaningless information. By using translation approach we can inherit built-in management functions, security mechanisms, easy to implement but not all of information can be translated. Table 6.1 shows the number of ECHONET Lite protocol's properties can be translated into OMA LWM2M standard (11/21 properties can be translated).

Table 6.1: Corresponding attributes of ECHONET Lite and OMA LWM2M

No	ECHONET Lite(v1.12)	OMA LWM2M (v1.0)
1	Operation status	
2	Installation socation	
3	Standard version information	Firmware Version
4	Identification number	Available Network Bearer
5	Instantaneous power consumption	
6	Cumulative power consumption	
7	Manufacturers fault code	
8	Current limit setting	Power Source Current
9	Manufacturers fault code	Error Code
10	Fault description	Reset Error Code
11	Manufacturer code	Manufacturer
12	Business facility code	
13	Production number	Serial Number
14	Production date	
15	Power-saving operation setting	
16	Product code	Model Number
17	Remote control setting	
18	Current time setting	Current Time
19	Current date setting	Current Time
20	Power limit setting	
21	Cumulative operating time	Uptime

6.5 Proposed architecture versus direct and indirect management

To prove the time efficiency of the combination, experiments to measure time and packet have been made. The experiment scenario is as in Figure 6.9 and the experiment's network topologies are described in the Appendix B

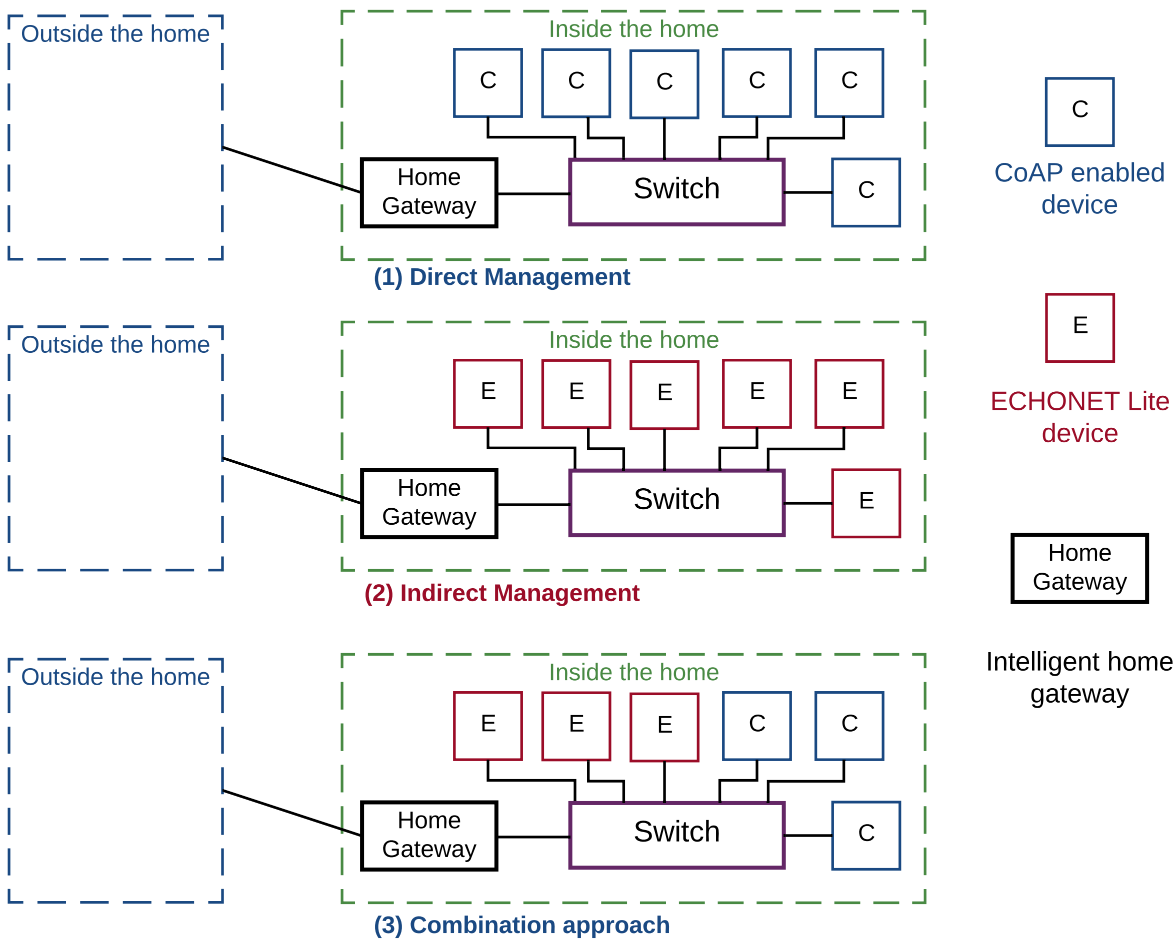

Figure 6.9: Scenario to evaluate direct management, indirect management and the proposed architecture

6.5.1 Time measurement

Average time to register or update resources of CoAP enabled devices and ECHONET Lite devices are describe in Table 6.10

	Direct Management	Indirect Management	Proposed Architecture	
			Direct management	*Indirect management*
RTT to register resources	136 ms	5372 ms	133 ms	4172 ms
RTT to update resources	28 ms	4440 ms	26 ms	3010 ms

Figure 6.10: Average round-trip time (RTT) to register and update device resources

Detailed sequence diagram and time consuming phase are described in Figure 6.11 and Figure 6.12

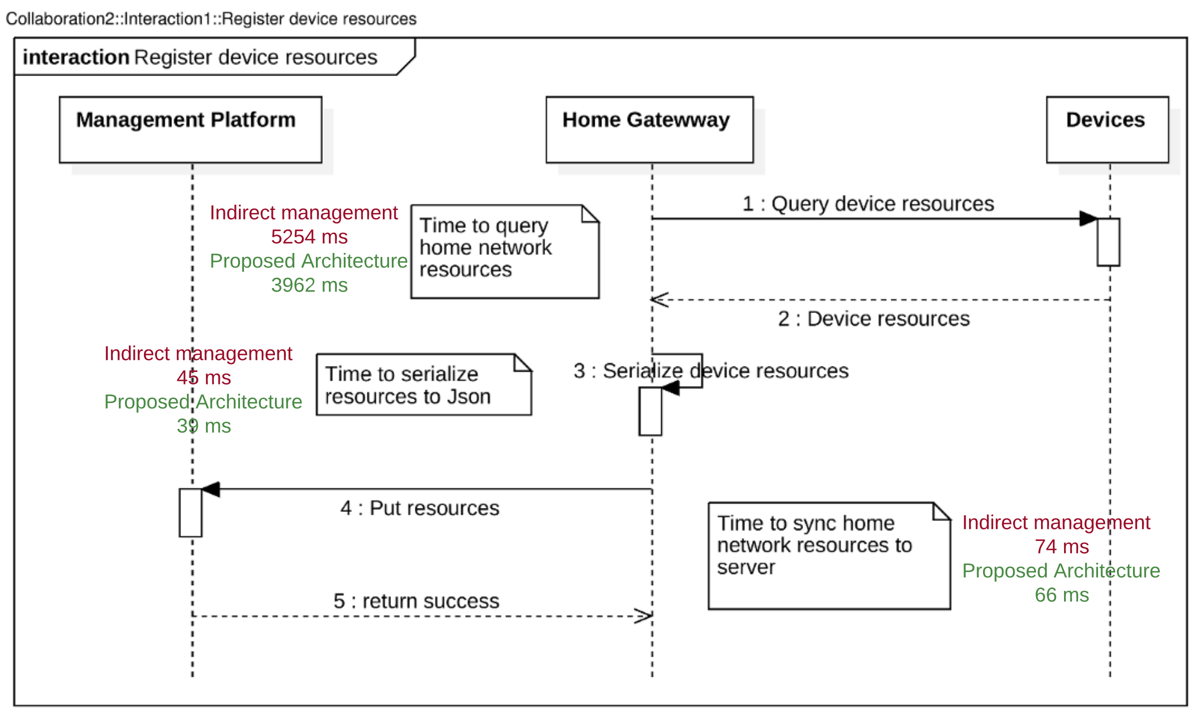

Figure 6.11: Detailed time to register device resources: **indirect management** versus **proposed architecture**

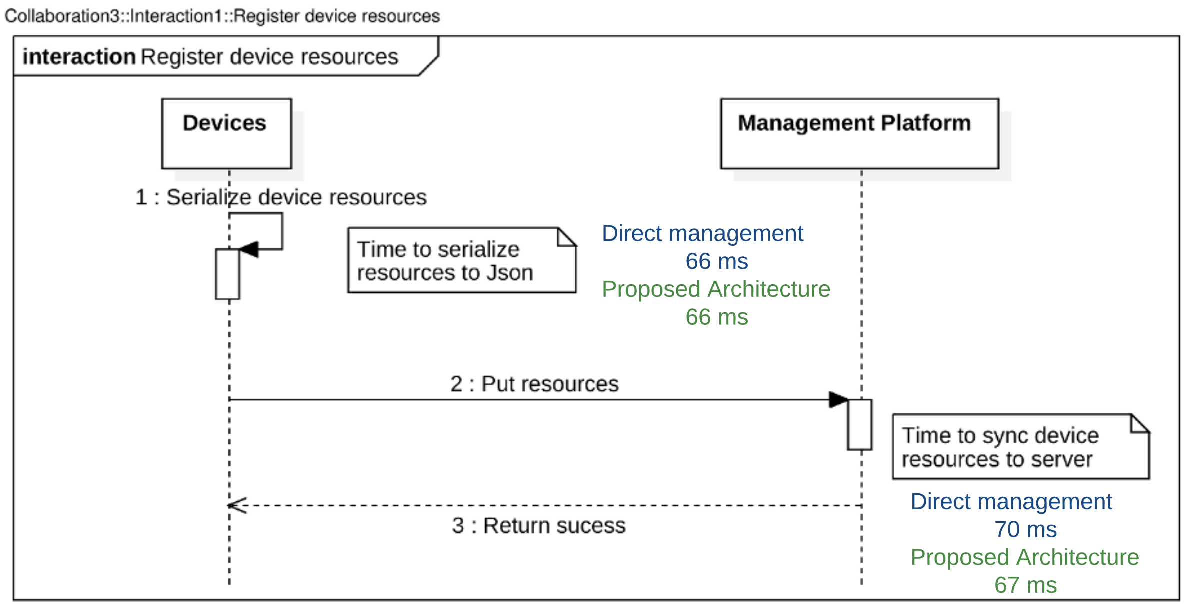

Figure 6.12: Detailed time to register device resources: **direct management** versus **proposed architecture**

Indirect management approach spends much time on collecting device resources in the home network at the home gateway. Therefore, latency from devices and the MP is very high. In contrast, direct management approach supports extremely low latency because there is no intermediate processing at the home gateway.

The proposed architecture that combines both management approaches so it can support both high latency and low latency devices. High latency management is suitable for devices which are constrained, low-power or incompatible communication protocols. While, low latency management is critical for dangerous or time-sensitive devices.

The proposed architecture is applicable for low latency devices and also enables management services for constrained, low-power and incompatible communication protocol devices.

6.5.2 Packet measurement

The packet measurement was conducted based on the scenario in Figure 6.9 to measure number of exchanged packets inside the home and outside the home by applying three management approach: direct management, indirect management and the proposed architecture. The results are shown in Figure 6.13.

	Outside the home		Inside the home	
	Exchanged packets (to and from MP)	**Exchanged bytes (to and from MP)**	**Exchanged packets (Devices and HGW)**	**Exchanged bytes (Devices and HGW)**
Direct Management	796 packets	50KB	-	-
Indirect Management	227 packets	47 KB	187,518 packets	11352 KB
Proposed architecture	619 packets	45 KB	95,253 packets	5766 KB

Figure 6.13: Exchanged packets and bytes using three approach within 1 hour

Based on the results, within 1 hour, amount of exchanged data (bytes) to the management server is almost the same by using three management approaches (50KB, 47KB and 45KB). However, the proposed architecture reduces a large number of exchanged message between devices and home gateway comparing to traditional indirect management approach. Therefore, the proposed architecture can mitigate the bottleneck issues for the home gateway and also reduce processing cost for the home gateway.

Note: We can not only use direct management approach even though it completely reduces the traffic to and from the home gateway due to direct management is not applicable for constrained devices and the home network is heterogeneous with both constrained and unconstrained devices

6.5.3 Conclusion

The proposed architecture supports multiple device classes from constrained devices to less constrained devices, enables management services for incompatible communication protocol devices, is applicable for dangerous or time-sensitive devices and mitigates the bottleneck issues for the home gateway.

6.6 Summary

The experiment shows that:

- A quantitative analysis of translation approach was made. In this analysis, properties that can be translated from ECHONET Lite protocol and OMA LWM2M standard are described (only 11/21 properties can be translated). We can claim that translation approach can not translated all properties between 2 protocols.
- Because intelligent gateway provides functions to process Home Network resources before transmitting, the cost for data exchanging from/to the HGW and the MP

was dramatically reduced comparing to the simple gateway (tunnelling approach is one instance of simple gateway).

- The intelligent gateway design approach enables local management functions to be able to support intelligent management.
- The proposed approach is able to support multiple device classes, manage dangerous and time-sensitive devices and mitigate bottleneck issues for the home gateway.

Chapter 7

Conclusion and Future Works

In accordance with the development of the internet of things (IoT), connected devices in smart home system are heterogeneous. Management a large number of heterogeneous devices are quite challenging and current management approach may not adapt to this challenging. This study proposed a management architecture that combines current management approaches (direct and indirect management) to benefit from these advantages.

In a management architecture, the HGW plays an important role. Therefore, clarifying designs of HGW is also important. The qualitative and quantitative analysis about three kinds of HGW are made. By supporting applications for local management, the intelligent gateway has advantages in minimizing exchanged data from the HGW to the internet. The proposed architecture is a combination of direct and indirect management approach using intelligent HGW.

To prove the feasibility of the proposed architecture, a prototyped based on this architecture was developed. The prototype combines ECHONET Lite protocol for indirect management and CoAP protocol for direct management. By capturing the exchanged packet using WireShark, we verified that the intelligent application gateway dramatically reduces the exchanged data to and from the HGW.

The proposed architecture also clarify with devices should be directly managed and indirectly managed. With this architecture, we can manage multiple device classes from very constrained devices to less constrained devices, provide options for managing dangerous and time-sensitive devices, enable local management to support local fault detection and recovery and reduce the traffic for the home gateway to mitigate home gateway bottleneck issues.

The future research can be carried out by applying artificial intelligence for management to provide auto configuration, fault detection and self management functions.

Appendices

Appendix A

Source Code

The source code of this project can be founded at **https://github.com/Cupham/HNM**

Appendix B

Home Network Topology

Listing B.1: Home Network Topology (Scenario: All devices are ECHONET Lite devices)

```xml
<?xml version="1.0" encoding="utf-8"?>
<HomeNetwork>
    <ECHONET_Lite>
        <Device_IP>192.168.0.6</Device_IP>
        <Device_Status>OFF</Device_Status>
    </ECHONET_Lite>
    <ECHONET_Lite>
        <Device_IP>192.168.0.7</Device_IP>
        <Device_Status>OFF</Device_Status>
    </ECHONET_Lite>
    <ECHONET_Lite>
        <Device_IP>192.168.0.8</Device_IP>
        <Device_Status>OFF</Device_Status>
    </ECHONET_Lite>
    <ECHONET_Lite>
        <Device_IP>192.168.0.9</Device_IP>
        <Device_Status>OFF</Device_Status>
    </ECHONET_Lite>
    <Home_Gateway>
        <Device_IP>192.168.0.2</Device_IP>
        <Device_Status>ON</Device_Status>
    </Home_Gateway>
    <ECHONET_Lite>
        <Device_IP>192.168.0.4</Device_IP>
        <Device_Status>ON</Device_Status>
    </ECHONET_Lite>
    <ECHONET_Lite>
        <Device_IP>192.168.0.5</Device_IP>
        <Device_Status>OFF</Device_Status>
    </ECHONET_Lite>
</HomeNetwork>
```

Listing B.2: Home Network Topology (Scenario: All devices are CoAP enabled devices)

```xml
<?xml version="1.0" encoding="utf-8"?>
```

```xml
<HomeNetwork>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.5</Device_IP>
        <Device_Status>ON</Device_Status>
    </CoAP_enabled_device>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.9</Device_IP>
        <Device_Status>ON</Device_Status>
    </CoAP_enabled_device>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.8</Device_IP>
        <Device_Status>ON</Device_Status>
    </CoAP_enabled_device>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.7</Device_IP>
        <Device_Status>ON</Device_Status>
    </CoAP_enabled_device>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.6</Device_IP>
        <Device_Status>ON</Device_Status>
    </CoAP_enabled_device>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.4</Device_IP>
        <Device_Status>ON</Device_Status>
    </CoAP_enabled_device>
</HomeNetwork>
```

Listing B.3: Home Network Topology: Combination approach

```xml
<?xml version="1.0" encoding="utf-8"?>
<HomeNetwork>
    <ECHONET_Lite>
        <Device_IP>192.168.0.8</Device_IP>
        <Device_Status>OFF</Device_Status>
    </ECHONET_Lite>
    <Home_Gateway>
        <Device_IP>192.168.0.2</Device_IP>
        <Device_Status>ON</Device_Status>
    </Home_Gateway>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.4</Device_IP>
        <Device_Status>ON</Device_Status>
    </CoAP_enabled_device>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.5</Device_IP>
        <Device_Status>ON</Device_Status>
    </CoAP_enabled_device>
    <CoAP_enabled_device>
        <Device_IP>192.168.0.9</Device_IP>
        <Device_Status>OFF</Device_Status>
    </CoAP_enabled_device>
    <ECHONET_Lite>
        <Device_IP>192.168.0.6</Device_IP>
```

```
        <Device_Status>ON</Device_Status>
    </ECHONET_Lite>
    <ECHONET_Lite>
        <Device_IP>192.168.0.7</Device_IP>
        <Device_Status>OFF</Device_Status>
    </ECHONET_Lite>
</HomeNetwork>
```

Bibliography

[1] ITU-T, *Overview of the Internet of things*, Y.2060, March 2012.

[2] [RFC7547] M. Ersue, D. Romascanu and J. Schoenwaelder, *Management of Networks with Constrained Devices: Problem Statement and Requirements*, RFC 7547, May 2015.

[3] 802.1CF Network Reference Model and Functional Description of IEEE 802 Access Network *http://www.ieee802.org/OmniRANsg/*

[4] S. Rao, D. Chendanda, C. Deshpande and V. Lakkundi, *Implementing LWM2M in Constrained IoT Devices*, ICWiSe, p.52-57, 2015.

[5] A. Sehgal, V. Perelman, S. Kuryla and J.Schonwalder, *Management of Resource Constrained Devices in the Internet of things*, IEEE Communitcations Magazine,, Volume 50, p.144-149, 2012

[6] [RFC3411] D. Harrington, R. Preshun and B. Wijnen, *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*, RFC 3411, December 2002.

[7] [RFC6241] R. Enns, M. Bjorklund and A. Bierman, *Network Configuration Protocol (NETCONF)*, RFC 6241, June 2011.

[8] S. Pandey, M. Choi, M. Kim and J.W. Hong, *Towards Management of Machine to Machine Networks*, APNOMS, 2011.

[9] [RFC7452] H. Tschofenig, J. Arkko, D. Thaler and D. McPherson, *Architectural Considerations in Smart Object Networking*, RFC 7452, March 2015.

[10] Nest Thermostat, *https://nest.com/thermostat/meet-nest-thermostat/*

[11] [RFC7228] C. Bormann, M. Ersue, and A. Keranen, *Terminology for Constrained-Node Networks*, RFC 7228, May 2014.

[12] [RFC768] J. Postel, *User Datagram Protocol*, RFC 768, August 1980.

[13] [RFC7252] Z. Shelby, K. Hartke and C. Bormann, *The Constrained Application Protocol (CoAP)*, RFC 7252, June 2014.

[14] [RFC793] Information Sciences Institute University of Southern California, *TRANSMISSION CONTROL PROTOCOL*, RFC 793, September 1981.

[15] [RFC2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach and T. Berners-Lee*Hypertext Transfer Protocol – HTTP/1.1*, RFC 2616, June 1999.

[16] ITU-T, *Requirements and architecture of home energy management system and home network services*, Y.2070, January 2015.

[17] Open Mobile Alliance Lightweight M2M, *http://technical.openmobilealliance.org/Technical/technical-information/release-program/current-releases/oma-lightweightm2m-v1-0*.

[18] Joe Folkens, *Building a gateway to the Internet of Things*, December 2014.

[19] UPnP, *https://openconnectivity.org/upnp*.

[20] ECHONET CONSORTIUM, *http://www.echonet.jp/english/*.

[21] Paul Congdon, *Link Layer Discovery Protocol and MIB* , March 2002.

[22] TTC, *JJ-300.00 Home-network Topology Identifying Protocol (HTIP)*, February 2011.

[23] Window Management Instrumentation, *https://msdn.microsoft.com/en-us/library/aa384642(v=vs.85).aspx* .

[24] CPE WAN Management Protocol *https://www.broadband-forum.org/cwmp.php*.

[25] Open Mobile Alliance Device Management, *http://openmobilealliance.org/about-oma/work-program/device-management/*.

[26] Extensible Messaging and Presence Protocol, *https://xmpp.org/*.

[27] MQ Telemetry Transport, *http://mqtt.org/*.

[28] Californium (Cf) CoAP framework, *http://www.eclipse.org/californium/*.

[29] Wireshark, *https://www.wireshark.org/*.

[30] Marios Sioutis, Yoshiki Makino and Yasuo Tan, *Developing ECHONET Lite libraries for rapid prototyping*, IEICE 2013.

List of Publications

[1] Cu Pham, Yuto Lim and Yasuo Tan, "Management Architecture for Heterogeneous IoT Devices in Home Network, *Joint Conference of Hokuriku Chapters of Electrical Societies 2016*, Fukui , Japan.

[2] Cu Pham, Yuto Lim and Yasuo Tan, "Management Architecture for Heterogeneous IoT Devices in Home Network, *IEEE-GCCE 2016*, Kyoto, Japan (Accepted, to be presented).