석 사 학 위 논 문

An Ontology Model and Reasoner to Build an Autonomic System for U-Health Smart Home

김　　진 (金　鎭)

정보통신학과 (네트워크 전공)

포항공과대학교 정보통신대학원

2010

An Ontology Model and Reasoner to Build an Autonomic System for U-Health Smart Home

An Ontology Model and Reasoner to Build an Autonomic System for U-Health Smart Home

By
Jin Kim

Department of Computer and Communications Engineering
POSTECH Graduate School for Information Technology

A thesis submitted to the faculty of POSTECH Graduate School of Information Technology in partial fulfillment of the requirements for the degree of Master of Engineering in the Department of Computer and Communications Engineering.

Pohang, Korea
December 18, 2009

Approved by

____

Major Advisor: James Won-Ki Hong

An Ontology Model and Reasoner to Build an Autonomic System for U-Health Smart Home

김 진

위 논문은 포항공과대학교 정보통신대학원 석사 학위논문으로 학위논문 심사위원회를 통과하였음을 인정합니다.

2009년 12월 18일

학위논문심사 위원회 위원장 홍 원 기 (인)

위 원 Agoulmine Nazim (인)

위 원 Strassner John Charles (인)

MCC 20080675 김 진, Jin Kim, "An Ontology Model and Reasoner to Build an Autonomic System for U-Health Smart Home", Department of Computer Communications Engineering, 2009, 58P, Advisor: James Won-Ki Hong, Text in English.

ABSTRACT

Two primary concerns for our aging society are preserving good health and avoiding disease. A Smart Home system is one possible way to solve this complex problem. However, it is very complex to build a smart home system for several reasons. First, it is hard to track the behavior of elderly people at home, because each person has unique needs and behavior. In addition, it is very difficult for a system to know whether a given behavior is normal or not. Second, medical symptoms are very complex. Current medical treatment uses various disease symptoms to choose one or more appropriate medical specialists. Third, there can be many heterogeneous technologies at home that use different transport protocols, management data, command languages and specifications. These technologies are a mix of standard and proprietary solutions, making it difficult to build a universal system.

To solve these problems, this research which is a part of POSTECH U-Health smart home project, was started. This thesis, is the result of the research, makes two significant contributions to the area. The first contribution is general architecture for the U-Health smart home project at POSTECH. The requirements are introduced and proposed in this thesis. The second contribution is a semantic model that can be used by an autonomic system to make intelligent decisions in a smart home environment. To create the appropriate model, various semantic technologies, which use OWL as the programming language, Protégé 3.4.2 as a tool for ontology creation and SWRL as a language for reasoning, were used. Furthermore, for reasonable ontology, open standard CIM version 2.22, which is defined and published by DMTF, was used as a framework.

ACKNOWLEDGEMENTS

This research is a part of the U-Health smart home project at POSTECH. The project has been an initiative by Prof. Nazim Agoulmine and Prof. M. Jamal Deen. I participated in this project in August 2009.

Prof. James Won-Ki Hong who is the head and professor of ITCE at POSTECH is my major supervisor. For two years, he gave me his time and guidance. Thank you for everything.

Prof. Nazim Agoulmine who is an ITCE professor in autonomics at POSTECH and a professor at University Evry Val d`Essonne spent a lot of time giving me valuable advice. He has a great knowledge of autonomic computing and teaches me everything. I would like to thank him sincerely.

Prof. M. Jamal Deen, who is also an ITCE professor in nano-sensors & systems at POSTECH and a professor at McMaster University, always give valuable feedback and advice and encouraged me every time when I was in difficulty. I also want to give great thanks to him.

Prof. Strassner John Charles, who is an ITCE professor and expert in the autonomic computing area, taught me ontology and autonomic computing systems. Thank you very much.

Hyeuk-Soo Choi and Seong-Yeol Shin, who are also members of this project, always helped me. Without their support and concern, I could not complete this research. Thank you very much.

Also, I would like to special thanks to my family and the DPNM members and Ph. D student Hui wang.

Contents

List of Figures

List of Tables

1 Introduction

The smart home concept was first introduced in the early 1980s as an intelligent building. At that time, the smart home was only designed for user convenience. This meant,that the smart home contained a lot of different devices to analyze the current status of the smart home environment. It became impossible to manage these devices efficiently because each device had different transport protocols, languages, management data and specifications. A separate, but related, concern is developing the role of the computer to help to improve quality of life. Recently, the population of elderly people has increased. Most elder people are concerned with their health. To maintain their good health, elderly people must spend a lot of money; this unfortunate trend is increasing over the years. Therefore, in this thesis we are looking at one possible solution: a U-Health smart home system.

1.1 Motivation

How can semantic web technologies help build a U-Health smart home?

The Semantic Web is an evolving development of the World Wide Web in which the meaning of information and services on the web is defined, making it possible for the web to understand and satisfy the requests of people and machines when using web content. According to Tim Berners-Lee's vision of the semantic web [1], the semantic web is a universal medium for data, information and knowledge exchange. To build a smart home, the most important consideration is the exchange the information between heterogeneous devices and how use this information. So, using semantic web technology is an appropriate approach for us. However, this gives rise to two more questions concerning how the semantic web technology will be used:

1.	**Is it possible to create a model to capture all the details of the smart home and the status of elderly residents (including their health) at home, so that semantic web technologies may be effectively used?**
2.	**Is it possible to build some kind of autonomic decision-making system that can use this semantic model to manage any situations for the elderly at home with as little human intervention as possible?**

These questions were the motivation for building an autonomic decision system that uses an ontology model. Various semantic technologies can be used to simplify the implementation of this system. With this motivation, this research will provide a good starting point for the U-Health Smart Home Project at POSTECH.

1.2 Contribution

This thesis provides two contributions for the U-Health smart home system area at POSTECH, which is described in section 3. The first contribution is general architecture for managing a smart home. Detailed information and requirements for this architecture are contained in section 3. The second contribution is the semantic model to capture the behaviour of the elderly in a smart home. For this model, we use the various semantic technologies that are introduced in section 2.

1.3 Thesis Outline

The structure of this thesis is as follows. Section 2 introduces other state-of-the-art smart home projects, autonomic systems, semantic web technologies and related work. The U-Health Smart Home project at POSTECH is introduced in section 3. For the purpose of this thesis, the autonomic system portion of the U-Health smart home is important, and it is described in detail in this section. In section 4, the ontology model, SHOM, is introduced along with the creation principle and usage of CIM features and the reasoning behind SHOM. The development environment and use cases of the autonomic decision-making system are described in section 5. The thesis conclusions are presented in section 6, together with suggested further work and contributions.

2 Related work

This section outlines state of the art research on various elements of relevant technologies. This section is divided into five parts. In the first part, various smart home projects are introduced. For smart homes, employing an autonomic system is a very useful approach. So, the concept of an autonomic system is described in the second part. The third part explains the semantic web technologies that are used to make an autonomic system in this thesis. After that, a comparison between other approaches and my approach is given, and last part summaries and explains the motivation for the thesis.

2.1 State of the Art of Smart Home Projects

AILISA Project [2][3]: The AILISA project is promoting experimental platforms in France to evaluate technologies for remote monitoring and provide assistance to elderly people at home. It aims to set up interdisciplinary platforms for the evaluation of these technologies at three levels; technical, medical and ethical. The limitation of AILISA is that the AILISA system does not provide any intelligent system functionality, such as intelligent decision-making or context-awareness.

Telemedicine Information Monitoring System [4]: The object of this project is to develop a telemedicine information monitoring system. This project is more focused on the integration of a variety of wireless vital sign monitoring technologies using an OSGi healthcare gateway. The project suggests an efficient way to merge and manage this variable information. However, it does not provide any autonomic decision flow.

UbiSense Project [5]: This project aims to develop an unobtrusive health monitoring system for elderly people. It differs from existing approaches by using embedded smart vision techniques to detect changes in posture, gait and activities, and aims to capture signs of deterioration of the patients by analyzing subtle changes in posture and gait. To analyze the activities of a user or subject, it uses subject-specific information, called personal metrics. This project also concentrates on monitoring, and does not provide any autonomic functionality.

There are a lot of projects that are not mentioned in this section. Many projects use autonomic systems for specific reasons, such as monitoring and/or controlling user behavior, home care devices or smart home systems. The CodeBlue project [39], which is an initiative by Harvard University, has focused on developing wireless sensors for medical care. In [40], the smart home program is a live-in laboratory at Duke University. There are several other active projects that are in progress.

Note that the above examples all have one thing in common: each system has only one primary purpose. To truly support a "smart home environment", a system needs to analyze heterogeneous data that can be used for many different purposes. Otherwise, it is not helping the user live a better life; Instead, it is only making one specific aspect of the user's life easier.

Architecture comparison

In this section, architectural approaches for smart home are introduced and compared with my approach. The most similar approach to the one used in this thesis is described in [28]. However, the aim of [28] is focussed on one of the main tasks in the self-* paradigm [6][17]: self-configuration. The main difference between that example and this thesis is the ontology model used. In the architectural approach used in this thesis, the principle of the ontology design model is to capture the context and status of the smart home, but [28]'s ontology model is focused on the self-configuration of the OSGi-framework. Although the purpose of the ontology model is different, the global architecture for the autonomic system is quite similar with my architecture.

For SOCAM architecture [30], the main difference is the approach. SOCAM architecture is focused on providing context-aware functionality using a context model based on an ontology. It is not autonomic system architecture. Both the approach in this thesis and the SOCAM approach are designed for use with pervasive computing environments, so the architecture used to capture the context employing physical sensors is similar.

There are two types of potential architectural approach available: automated system architecture using a home gateway [31] or autonomic system architecture. The significant difference between them is the way they treat the information they use, and how they choose appropriate decisions. An automated system uses the information directly, whereas an autonomic system processes the information through a control loop. Almost all smart home systems [32][33] focus on how to merge heterogeneous information. On the other hand, the autonomic system in this thesis focuses on how to process the information to establish the context and choose the correct decision.

2.2 State of the Art on Autonomic System

Autonomic Computing [6][17][18]: Autonomic Computing is an initiative started by IBM in 2001 [6]. Its ultimate aim is to develop computer systems capable of self-management, to overcome the rapidly growing complexity of computing systems management, and to reduce the barrier that complexity poses to further growth. In other words, autonomic computing refers to the self-management characteristics of distributed computing resources, adapting to unpredictable changes while hiding its intrinsic complexity to operators and users. An autonomic system makes decisions on its own, using high-level policies; it will constantly check and optimize its status and automatically adapt itself to changing conditions by comparing a set of concepts between current computing and self-management autonomic computing. In autonomic computing, there are two main central concepts; Autonomic Managers and Control loops. In a selfmanaging Autonomic System, the human operator takes on a new role: the operator does not necessarily control the system directly. Instead, the operator defines general policies and rules that serve as the input for the self-management process. For this process, IBM has defined the following four functional areas:

- Self-Configuration: Automatic configuration of components.
- Self-Healing: Automatic discovery and correction of faults.
- Self-Optimization: Automatic monitoring and control of resources to ensure optimal functioning with respect to the defined requirements.
- Self-Protection: Proactive identification and protection from arbitrary attacks.

Table. 1 Four aspects of self-management autonomic computing [17]

Concept	*Current computing*	*Autonomic computing*
Self-Configuration	Corporate data centers have multiple vendors and platforms. Installing, configuring, and integrating systems are time consuming and error prone.	Automated configuration of components and systems follows high-level policies. Rest of system adjusts automatically and seamlessly.
Self-Healing	Problem determination in large, complex systems can take a team of programmers weeks.	System automatically detects, diagnoses, and repairs localized software and hardware problems.
Self-Optimization	Systems have hundreds of manually set, nonlinear tuning parameters, and their number increases with each release.	Components and systems continually seek opportunities to improve their own performance and efficiency.
Self-Protection	Detection of and recovery from attacks and cascading failures is manual.	System automatically defends against malicious attacks or cascading failures. It uses early warning to anticipate and prevent system wide failures.

The basic concept that is applied in Autonomic Systems is a closed control loop –Fig.1 –. This loop consists of four common functions:

- Monitor function: This function provides the mechanisms that collect, aggregate, filter and report details (such as metrics and topologies) collected from a managed resource.

- Analyze function: This function provides the mechanisms that correlate and model complex situations (such as time-series forecasting and queuing models). These mechanisms allow the autonomic manager to learn about the IT environment and help predict future situations.
- Plan function: This function provides the mechanisms that construct the actions needed to achieve goals and objectives. The planning mechanism uses policy information to guide its work.
- Execute function: This function provides the mechanisms that control the execution of a plan with considerations for dynamic updates.

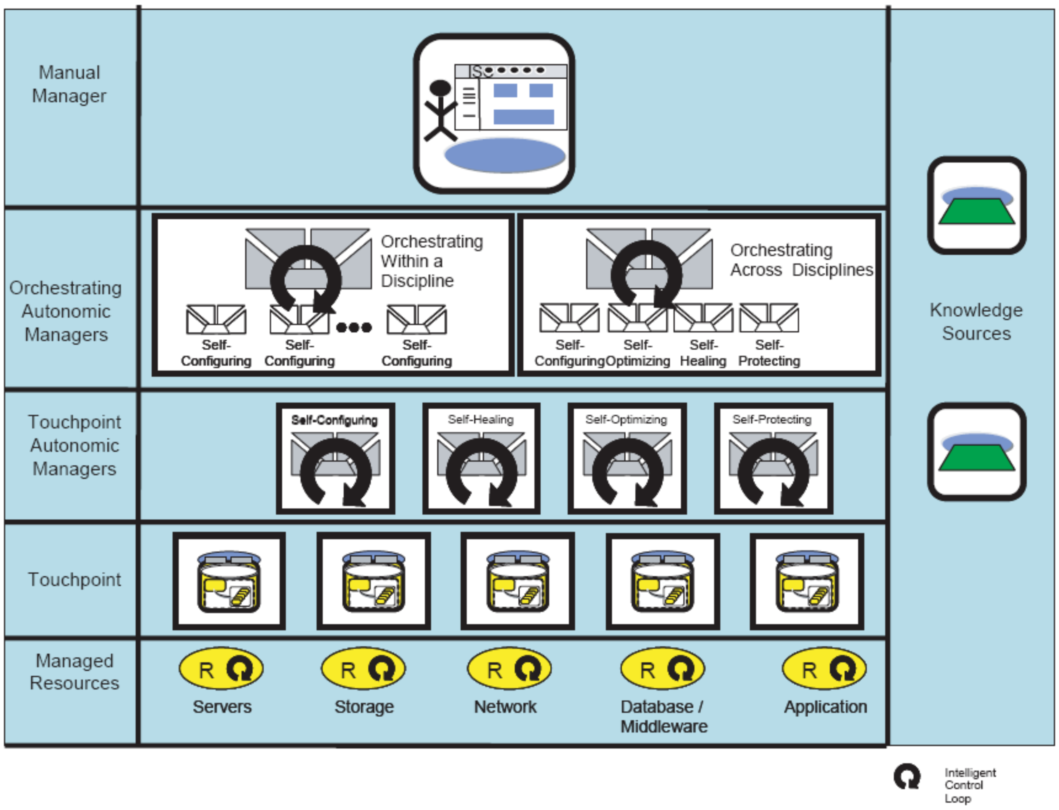

Fig. 1 Autonomic computing reference architecture by IBM [17]

These functional compositions are sometimes referred to as a MAPE (for Monitor-Analyze-Plan-Execute) control loop. However, Fig. 2 also shows that, in addition to the MAPE functions, an autonomic manager also contains a knowledge block that is connected to each functional component. If this type control loop is used, this is one place where the work of this thesis could fit in,.

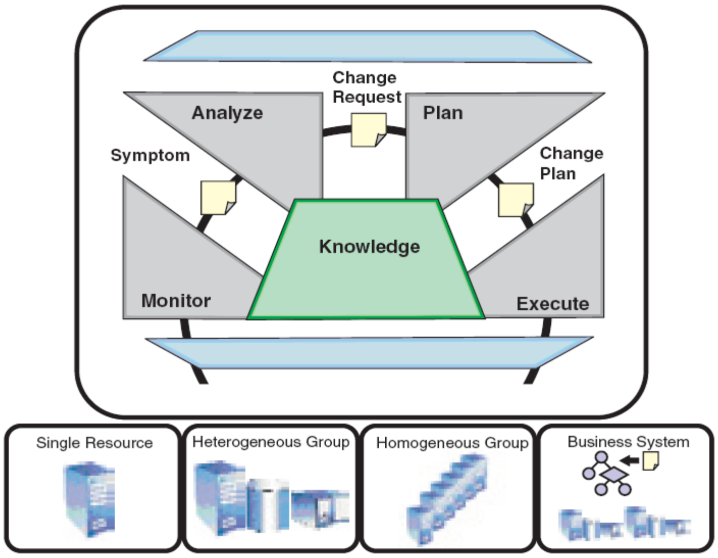

Fig. 2 Four common managed resource arrangements by IBM [17]

2.3 State of the Art on Semantic Web Technologies

In this section, semantic web technologies and concepts are introduced for a model-based decision-making system. In this type of system, a model is used to express and interpret the context of a specific environment like a smart home.

Ontology [7][8][9][14]: Ontology is a widely accepted tool for the modeling of information. The term *ontology* comes from the field of philosophy, and is concerned with the study of being or existence. Although there are a lot of definitions of ontology in the context of computer and information sciences [7][14], ontology is generally regarded a formal representation of a set of concepts within a domain and the relationships between those concepts. It is used as the reasoning behind the properties of the domain, and may be used to define the domain. The foundations that represent ontology are typically classes, attributes and relationships. The definitions of these foundations include information about their meaning and constraints on their logically consistent application.

Ontology comparison

In this section, various ontology models are compared with my model. GUMO [35] is the general user model ontology. The main purpose of GUMO is to exchange user model data between different user-adaptive systems. The model focuses on the user`s dimensions which include Emotional States, Characteristics and Personality. The GUMO model would be useful as a topic futher research, because the system in this thesis cannot obtain detailed information about users (like emotions).

DogOnt [36] is larger than GUMO. DogOnt is an ontology modeling system for an intelligent domotic environment, and it was proposed as part of an ontology for new house modeling designed to fit real-world domotic system capabilities. The target of DogOnt is a building environment, and it contributes to the EHS [37] taxonomy, which is a home appliance classification system designed by the EHS consortium. Also, the model is based on real-world environments [38]. It is a well made ontology model for a building environment, but it lacks a user modeling ability. In SHOM, the relationships between the user and the environment are defined.

A U-Health smart home system requires the convergence of various technologies and concepts. The SHOM semantic model can provide a good foundation for the management of users or equipment in a smart home along with health care services and other services. SHOM model will be introduced in section 4.

CIM [15]: The Common Information Model (CIM) is a standard that defines how managed elements in an IT environment are represented as a common set of objects along with the relationships between them. CIM provides a common definition for management information between systems, networks, applications and services, and allows for vendor extensions.

RDF [16]: The Resource Description Framework (RDF) is a framework for representing information on the Web, and has an abstract syntax that reflects a simple graph-based data model. RDF uses various concepts such as an URI-based vocabulary, datatypes, literals, XML serialization syntax, expression of simple facts and entailment. These and other properties enable RDF to be used to describe data in a way that facilitates logic reasoning for the data.

OWL [8]: The Web Ontology Language (OWL) uses RDF and XML to represent information in a manner suitable for implementing in an ontology. An OWL ontology may include descriptions of classes, properties and their instances. With this type of ontology, the OWL formal semantics specifies how to derive its logical consequences, i.e., facts not literally present in the ontology, but determined by the semantics. OWL provides three increasingly expressive sub-languages designed for use by specific communities of implementers and users.

- *OWL Lite* supports those users that primarily need a classification hierarchy and simple constraints. For example, while it supports cardinality constraints, it only permits cardinality values of 0 or 1. It is simpler to provide tool support for OWL Lite than for its more expressive relatives, and OWL Lite provides a quick migration path for thesauri and other taxonomies that are not formal ontologies in their own right.

- *OWL DL* supports users who want more expressiveness. In general, OWL DL is intractable unless constraints are used (e.g., only using small ontologies and not using all of its features). It is possible to build decidable fragments of OWL-DL, which is what was done for this project. OWL DL includes all OWL language constructs, but they can be only used under certain restrictions (for example, while a class may be a subclass of many classes, a class cannot be an instance of another class). OWL DL is named due to its usage of *description logics* [42], a field of research based on the study of formal logic.
- *OWL Full* is meant for users who want maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. For example, in OWL Full, a class can be treated simultaneously as a collection of individuals and as an individual in its own right. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary.

2.4 Literature Review Summary and Motivation

In the previous sections, a brief overview of related approaches and some current smart home projects and their limits were given. The major problem for these examples is to process and correlate a huge amount of information on human behavior. Although the initial motivation of autonomic systems is to handle the complexity of managed devices or programs, if the target changes from devices to users, then the autonomic system approach fits our purpose. To help improve decision-making abilities, semantic web technologies are useful in this regard.

This section has provided a brief overview of three important technologies and related work to our U-Health smart home project. Furthermore, it has provided the fundamental hypothesis of this thesis: are autonomic decision-making systems using semantic web technologies a good solution for our U-Health smart home system?

The next section will describe the general architecture and autonomic decision-making system using semantic web technologies for smart home applications. In this thesis, OWL-DL will be used to create an ontology model to represent and interpret facts, because it represents a good compromise between expressivity and computational complexity.

3 Autonomic System Internal Architecture

3.1 Autonomic System for U-Health Smart Home at POSTECH

Fig. 3 represents the requirements for our U-Health smart home project. At the bottom of the stack, there are two components; sensors and actuators. An actuator is a component that can control devices including windows, doors and other devices. A sensor generates information about the smart home user and/or the smart home. The information that is generated in the sensing component goes to the home networking component. In this component, the information is gathered, and a service or appliance discovery will be done. The gathered information will be sent to the autonomic computing component. In this component, the information is filtered and aggregated to build a model of the current context of the smart home and smart home user. The autonomic system interprets the information using one or more decision-making and semantic reasoning algorithms. The top of the stack represents health care services and personnel. For a U-Health smart home system, various health services must be provided to the smart home user based on decisions generated by the autonomic computing part.

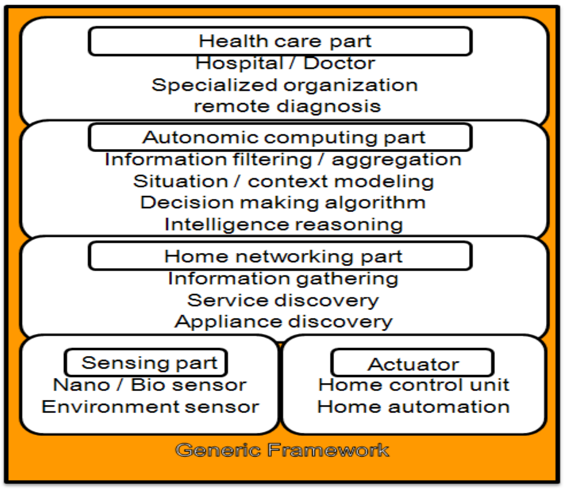

Fig. 3 High level stack view for U-Health smart home

3.2 General Architecture

The general system architecture for the U-Health smart home project at POSTECH is introduced in this section.

The general system architectureas shown in Fig. 4 consists of three parts. The first part is the smart home environment. To know the context of the smart home, one or more networks are constructed. The main three types of networks are wireless ad-hoc networks along with wired and wireless networks. The wireless ad-hoc network consists of many different types of heterogeneous sensors, possibly organized as separate networks. To compute the user health context, body sensors are organized into a Wireless Body Area Network (WBAN), and for computing the environmental context, different types of wireless sensors are organized into one or more Wireless Area Networks (WANs). For a seamless environment, a wireless network is more convenient than a wired network, especially in the case of body area networks.

In an autonomic smart home, each set of sensors send their information to their home gateway. The role of the home gateway is to communicate between sensors, appliances and the autonomic management system. The sensors provide the autonomic management system with different types of information, which are used by the autonomic management system to construct one or more of the contexts that are present in the smart home. If an emergency situation develops, then the autonomic management system uses this information to generate an alarm to a doctor, user, or both, alerting them of the situation.

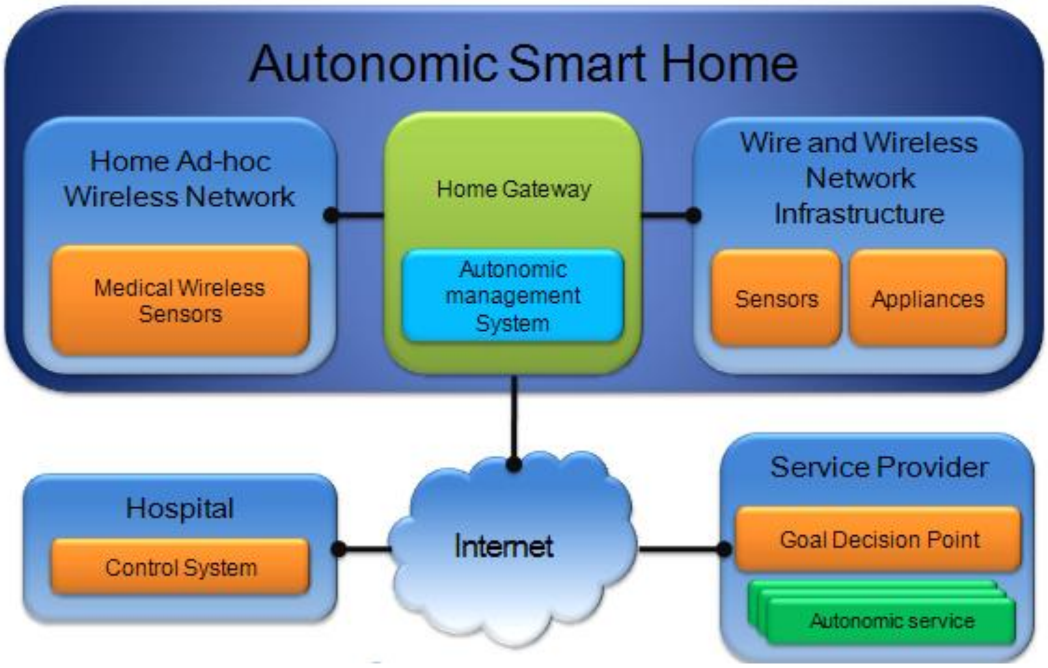

Fig. 4 General system architecture

3.3 Decision Making System

The decision-making system for the U-Health smart home at POSTECH is based on an ontology model. An ontology model is used in this thesis to build a formal representation of smart home users and a smart home environment, so that a context can be defined to support intelligent reasoning and decision-making. Fig. 5 shows a diagram containing detailed architecture of the autonomic system. The sensors that are in the smart home can generate a lot of information. The control flow that decides on which action to take depending on the context uses the same sequence as the MAPE loop. Each symptom is monitored and analyzed, and if a change is required, then the change is planned for and sent to the execution part. In this part, the core system uses an ontology and rules. To get context information in a smart home, the decision making core part, which is the top part of Fig. 5, interprets the request and chooses the appropriate rules. Then, the answer is generated and sent to the plan and execute bundles. If some appropriate action is required to control the context, then the execution bundle sends the appropriate action to the devices in a smart home like sensors or actuators.

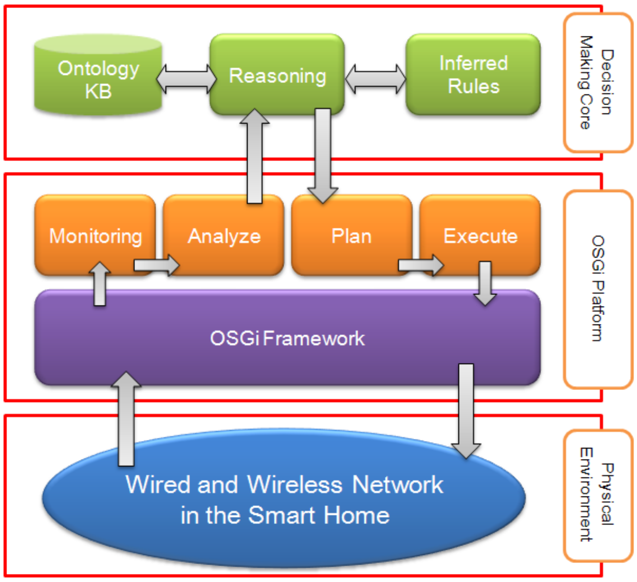

Fig. 5 Detailed architecture for the autonomic system

4 Ontology Modeling

As the first step for the system is to create the semantic model, the DMTF Common Information Model (CIM) was used in this research. There are several reasons why this decision was taken. The first reason is that CIM is a free technology that defines how managed elements in an IT environment are represented as a common set of objects and determines the relationships between them; this provided a framework for representing the various elements in a smart home system. The second reason is that there are a lot of useful classes and relations for the model presented in this thesis that are present in the CIM model. However, because CIM classes and relations do not have any formal defined semantics, the CIM classes and relations need to be augmented to represent the semantics for smart home users and smart homes. To do this, we tried to understand the semantics of the classes and their relations and select those CIM classes and relations which are useful to build our model.

4.1 Use the CIM Classes and Relations

The CIM schema is defined by a “Core” model and one or more “Common” models. The Core model captures notions that are applicable to all areas of management, and represents a starting point for determining how to extend the common schema. An attempt was made in this research to maintain the structure of the Core model in order to be compliant with CIM. Common models are information models that capture notions that are common to particular management areas. There are a lot of common models, such as CIM_Device, CIM_System, CIM_User, CIM_Service, CIM_Application, and others, which cover specific functions and applications.

How can one choose which components to use from the CIM?

To choose useful components that are defined in the CIM, a question lists were made which the decision making system must solve. Table. 2 and Table.3 show the list of questions for the ontology and the list of questions for the decision system, respectively. Based on these questions, the useful type of classes and relations were decided on in the

CIM along with what kind of classes and relations must be created to define the semantics that the CIM is lacking The list of questions for the ontology defines the type of information that will be represented in the ontology model, while the list of questions for the decision system defines the type of rules that can be implemented using the ontology model. In Table. 2, for example, the first six questions are used to specify the device environment in a smart home. So, this means that the CIM classes can be directly used because of their generic nature, but some new classes will need to be created, because the CIM did not model smart homes. For the first question shown in Table. 3, the information about sensors will be represented as CIM classes and relations, but if the smart home class is not created, then users and devices cannot be associated with a smart home. Although in CIM, the Location Class is used to indicate the position of managed elements, it cannot represent a smart home, because a smart home has more semantics than just a position. For semantic meaning, the POSTECHSmartHome class was created in the model in this research. This class conceptually defines a smart home, so its position in this model is under the Organization class, which is defined in the CIM. As another example, in Table. 3, in order to solve the first question (i.e., “where is the user?”), a number of additional questions must be resolved. Fig. 6 describes the decision flow to solve the example question.

Table. 2 List of questions for the ontology

Questions for the Ontology	
1	How many and what kinds of sensors are in the smart home?
2	How many and what kinds of sensors are in the specific room?
3	How many and what kinds of sensors make up this WSN?
4	How many and what kinds of sensors make up this BAN?
5	How many and what kinds of appliances are in this smart home?
6	How many and what kinds of appliances are in this specific room?
7	What is the specification of each sensor?
8	What is the user type?
9	What kinds of services are provided to the user?
10	What time is it now?
11	What is the data value of each sensor?
12	How can an emergency situation be communicated?

Table. 3 List of questions for the decision system

Questions for the decision system	
1	Where is the user?
2	What is the user state?
3	Does the sensor mode need to change?
4	Is the smart home in a state of emergency?

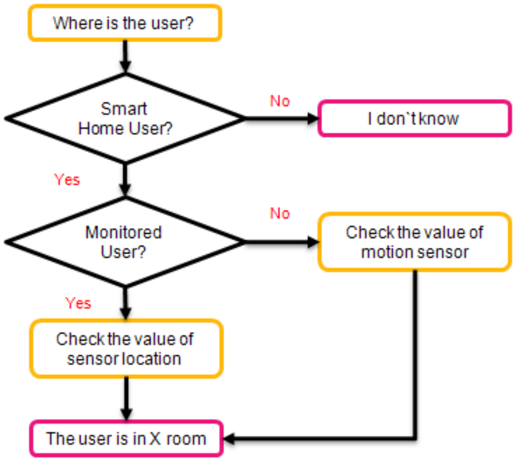

Fig. 6 Example flow to establish where the user is

What kinds of CIM models are used?

CIM_Core: The Core model establishes a basic classification of the elements and associations of the managed environment; the class hierarchy begins with the abstract ManagedElement class. In this research model, Fig. 7 shows the extended ManagedElement class. From the classes in the Core model, the model expands in many directions, addressing many problem domains and the relationships between managed entities. Note, however, that not all of the classes and relationships are used here that are defined in the Core model. For example, the eight recursive defined associations are not used from this class. This is because these classes and relationships are either not relevant for this model, or because they introduce semantic ambiguities. Following the CIM_Core document, several classes were considered; ManagedSystemElement, EnabledLogicalElement, Statistic and the relationships between PhysicalElements and LogicalDevices, SettingData and Profiles and Capabilites, Settings and Configurations and Method Parameters.

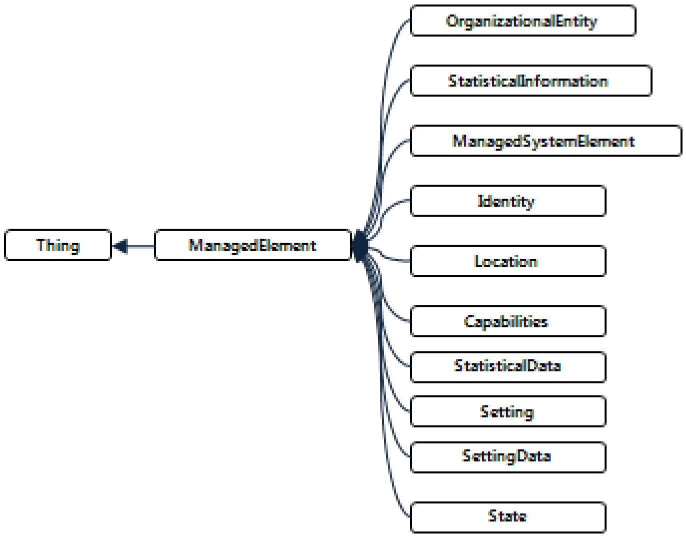

Fig. 7 Extended ManagedElement Class

CIM_Device: The Cim_Device model describes the functionality provided by hardware, as well as providing configuration and state data. The components which are represented in the CIM_Device typically describe a hardware specification. However, there are several concepts related to the CIM_LogicalDevice, which is an abstraction of a hardware entity that is logical in nature, such as the paper sizes supported by a Printer. This is captured in the CIM model by a relationship that enables the Physical Element to “realize” the LogicalDevice. Specifically, the CIM hierarchy of the Sensor and Alarm was used in this model. To use the Sensor class, the hierarchy is: Thing – ManagedElement – ManagedSystemElement – LogicalElement – EnabledLogicalElement – LogicalDevice –

Sensor. The Sensor class is under the LogicalDevice class, which means the Sensor class will be realized by some physical sensor because of the realization relationship described above. The sensors are divided into two types; BodySensor and EnvironmentSensor. BodySensor generates user health information, and EnvironmentSensor generates information about the smart home context including fire, light, motion and temperature. So, in this model, several classes, which are pairs of PhysicalElement and LogicalDevice, are created – see Fig. 8 and Fig. 9

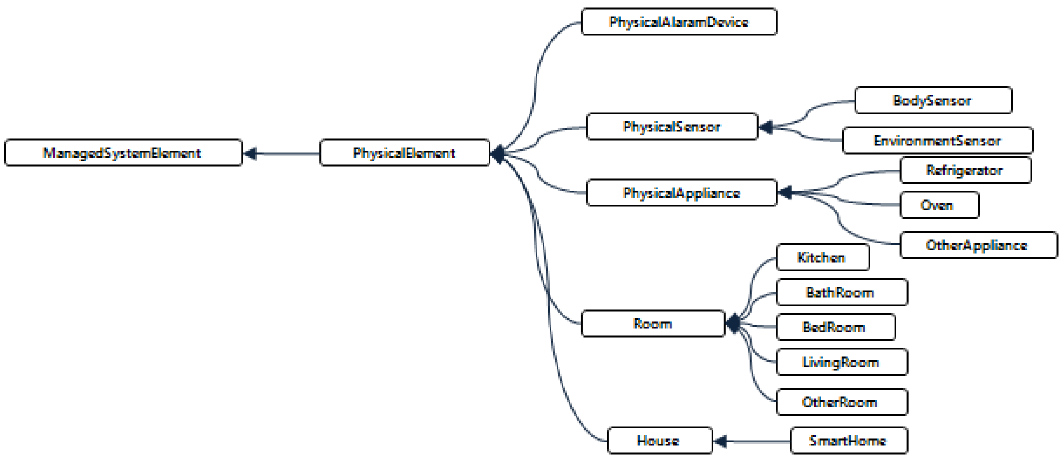

Fig. 8 Extended PhysicalElement class

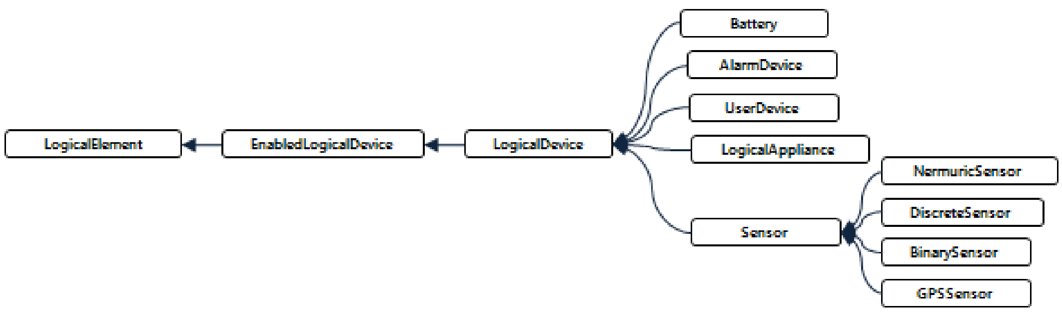

Fig. 9 Extended LogicalDevice class

CIM_System: the CIM_System defines computer system related abstractions. Many of the concepts related to computer systems derive from the CIM_System abstraction in the Core model. It describes the aggregation of parts (or components) into a single manageable system. In this schema, Time (which is a subclass of Setting, which can be related to a CIM_System) and Log (which is subclassed from EnabledLogicalElement) are useful classes for the model in this thesis. The main hierarchy for time is ManagedElement – Setting which has "ElementSetting" relation with ManagedSystemElement. LogRecord class is the subclass of ManagedElement class has various message formats about system elements.

CIM_User: The focus of the CIM_User/Security model includes (1) general contact and white pages information for organizations, organizational units and people, (2) "users" of services, and (3) the related security information to authenticate and authorize a "user". The main hierarchy of CIM_User is: Thing – ManagedElement – OrganizationalEntity – UserEntity – UserContact – Person. A Person is related to the concept of security in CIM. So, I create a SmartHomeUser class which is under the UserEntity class. UserEntity class is an abstract class that represents users – their names, contact data and similar information. A lot of information about User in CIM is related to contacts or security; the Person class in CIM is not suitable for the model shown in this research. The desired model was not to change any information about existing classes in CIM, so that more meaningful classes could be create about the user. The added class hierarchy, as shown inFig. 10, is UserEntity – SmartHomeUser – MonitoredUser and UnMonitoredUser. MonitoredUser means a user that uses a body sensor for monitoring to prevent accidents or to give some convenient service. For the security problem, this model was not a desirable option because the special component in a distributed framework of a smart home can provide a faster and stronger security level.

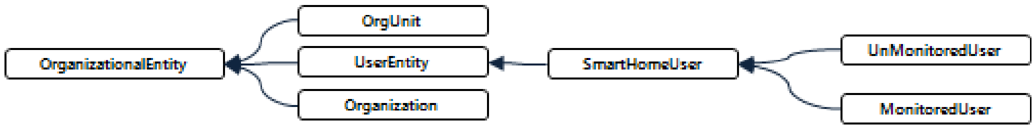

Fig. 10 Extended UserEntity class

Essential components for adding semantics?

PhysicalElement class: The subclasses of CIM_PhysicalElemenet define any component of a System that has a distinct physical identity. Under the PhysicalElement class, PhysicalComponent, PhysicalConnector, PhysicalLink, PhysicalPackage classes are defined in CIM, enabling the system to represent whole physical elements in a smart home. For this reason, Room and Apartment classes were chosen in this model to answer questions like, where is the smart home user? The positions of the Apartment class and SmartHome class (which is subclass of Apartment class) are under the PhysicalElment class and the isBelongTo relation is created between the SmartHome class and the POSTECHSmartHome class, which is a subclass of Organization. That means the SmartHome belongs to POSTECH. The POSTECHSmartHome class does not indicate a real home as it is a conceptual class. It could be possible to build many smart homes in POSTECH. To answer the previous sample question, the locality of devices is defined in this model. PhysicalSensor and PhysicalAppliance were created under the PhysicalElement, and LogicalAppliance was ceated under the LogicalDevice to make a pair between PhysicalElement and LogicalDevice. (For Sensor, the Sensor class under the LogicalDevice is already defined in the CIM model.) PhysicalSensor has two subclasses; BodySensor and EnvironmentSensor.

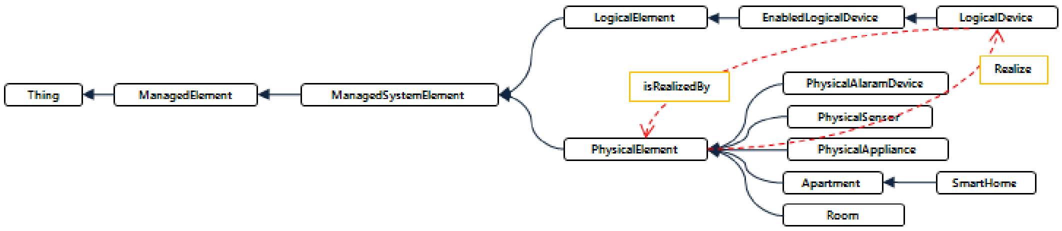

Fig. 11 Extended ManagedSystemElement class

The one of major considerations when designing taxonomy of PhysicalElement class was the position of the Apartment and Room classes. In fact, in the CIM, the Location class can indicate the position and address of a PhysicalElement, but a smart home application needs additional semantics. A subclass was created in this research to not disturb the CIM and also to better classify conceptual things. For example, the specific location of a sensor is used to infer information and make decisions in the POSTECH smart home. In this case, it is easy to know the role of that sensor without checking the specification of the sensor. This example shows that we don`t need to use all of the CIM classes and relations.

Service class: In CIM, a service is a LogicalElement that represents the availability of functionality that can be managed. The modeled Service typically provides only the functionality required for its own management or the elements it affects. In the model for this research, the Service class is used to indicate specific services that are provided to a user or for smart home management. For example, the MedicalCareService class (which is a subclass of the Service class) is set to provide medical home care services, such as a pulse monitoring service. A SmartHomeManagementService is for an administrator that manages a smart home.

UserActivityState class (Fig. 12): The state of the smart home user can be associated with a main activity in the smart home, such as watching TV or sleeping in the bedroom.

The smart home management system needs to define the semantics of each user activity to make appropriate autonomic decisions. In the model shown in this research, there are several classes that can be used to define user activity and behavior. These classes are used in conjunction with reasoning rules to determine one or more appropriate actions to take in certain situations. For example, if the smart home user is in a sleeping state in the bedroom, then the lights in the living room and the bath room do not need to be turned on except in special situations. In addition, OtherActivityState can be used to extend the UserActivityState to express other user activities in the smart home.

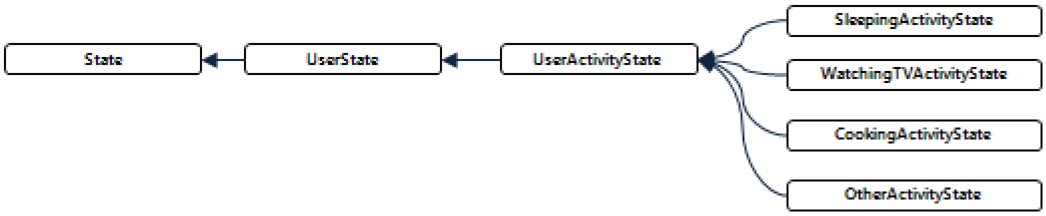

Fig. 12 New created class: State class

Attributes: Every new class has its own attributes to describe details about the class and/or to manage the objects created from the class. For example, a smart home user has two different types of states; activity and health states. The activity state is defined as a class, but the health state is defined as a set of attributes, because the health state of the user is classified as either Critical, Warning, and Normal, and every user is defined to be in one of these three states. Thus, it should be defined as an enumerated attribute, and not as a separate class, since every smart home user has this state.

Relationships: For new classes, new relationships between new classes and new or existing classes, classes may need to be created. The relationships are important to properly represent how different entities in the smart home work together to model a user or a situation that occurred in a smart home.

4.2 SHOM: Smart Home Ontology Model

The Smart Home Ontology Model (SHOM) is this proposed model for the U-Health smart home project at POSTECH. As mentioned in the previous section, several parts of SHOM are introduced as an example of a generic approach to build a semantic model. The SHOM model has 103 classes and 73 relationships.

Two of the main strengths of SHOM are scalability and flexibility. If a required concept is related to a CIM concept that is associated with a component in an IT environment, then it is possible to create unused classes and relations that are defined in the CIM model. Alternatively, if a required concept is a new concept, then there are many different classes in the SHOM model that can serve as the superclass of the new class. For example, the UserActivityState class has a class called OtherActivityState for scalability.

Converting required CIM classes and relationships to the SHOM model

Several efforts have been made to use the CIM model as an ontology, as well as to translate CIM to OWL [22][23][24][25]. In [25], the various difficulties in translating CIM to OWL are characterized. While there are ways to use or translate the CIM, there is no standard solution. The authors of [25] made a table for the mapping the definition from CIM to OWL, which is also useful for this reseach. Although their results are used here, it is only part of a solution, because all of the converting principles or solutions which are in [25] or [23][24][26] are not needed. The general ideas are the same including class, generalization and properties. So, a means to convert the whole CIM to OWL does not need to be considered here. However, in the future, the SHOM can be changed for a specific purpose if needed. Therefore it is better to choose the criteria as shown in [25].

OWL Properties characteristics in Protégé

To create the SHOM model, Protégé 3.4.1 and 4.0 are used. In Protégé, OWL properties represent the relationships between two individuals. There are two main types of properties; *Object properties* and *Datatype properties*. Object properties link an individual to an individual. Datatype properties link an individual to an XML Schema Datatype value or an rdf literal. OWL also has a third type of property – *Annotation properties*. Annotation properties can be used to add information to classes, individuals and object/datatype properties with useful information such as labels, comments, creation date, author or references. For object properties, there are four characteristics; Inverse, Functional, Transitive and Symmetric properties.

Table. 4 The mapping definition from CIM to OWL [25]

CIM artifact	OWL Construct
Class	<owl:Class>
Generalization	<rdfs:subClassOf>
Association	<owl:Class rdf:ID: "..“> <rdfs:subClassOf Rdf:resource="cim-meta:CIM_Association/> </owl:Class>
Property	<owl:DatatypeProperty>
REF Property	<owl:ObjectProperty>
Method	<cim-meta:hasMethod>
Default Value	<cim-meta:defaultValue>
Override	<rdfs:subPropertyOf>
Key	<owl:InverseFunctionalProperty>
Min, Max	<owl:minCardinality>, <owl:maxCardinality>
ValueMap, Values	<cim-meta:CIM_Value> composed of <cim-meta:value> and <cim-meta:valueMap>
Deprecated	<owl:deprecatedClass> or <owl:deprecatedProperty>
Required	<owl:minCardinality rdf:datatype="&xsd;int">1 </owl:minCardinality>
Experimental	<cim-meta:Experimental>
Alias	<owl:equivalentClass>, <owl:equivalentProperty> or <owl:sameAs>
ModelCorrespondence	<rdfs:seeAlso>

Read, Write	<cim-meta:readable>, <cim-meta:writeable>
Version	<cim-meta:cimVersion>
Abstract	<cim-meta:Abstract>
Units	<rdfs:comment>
Vectors	<rdfs:comment>

Table. 5 Four characteristics of object property

● **Inverse Properties:** Each object property may have a corresponding inverse property. If some property links individual A to individual B, then its inverse property will link individual B to individual A.
● **Functional Properties:** If a property is functional, for a given individual, there can be at most one individual that is related to the individual via the property.
● **Transitive Properties:** If a property is transitive, and the property relates individual A to individual B, and also individual B to individual C, then we can infer that individual A is related to individual C via property P.
● **Symmetric Properties:** If a property P is symmetric, and the property relates individual A to individual B, then individual B is also related to individual A via property P.

Fig. 13 and Fig. 14 show the whole taxonomy and object properties, respectively, of SHOM. The arrow represents an “is-a” relation. The red-dotted arrow in Fig. 14 relates to object properties. The root class is Thing, which is a basic root class in OWL. The subclass of Thing is ManagedElement, which is the root class in CIM. The SHOM maintains the structure of the CIM model and adds new classes and relations for semantics.

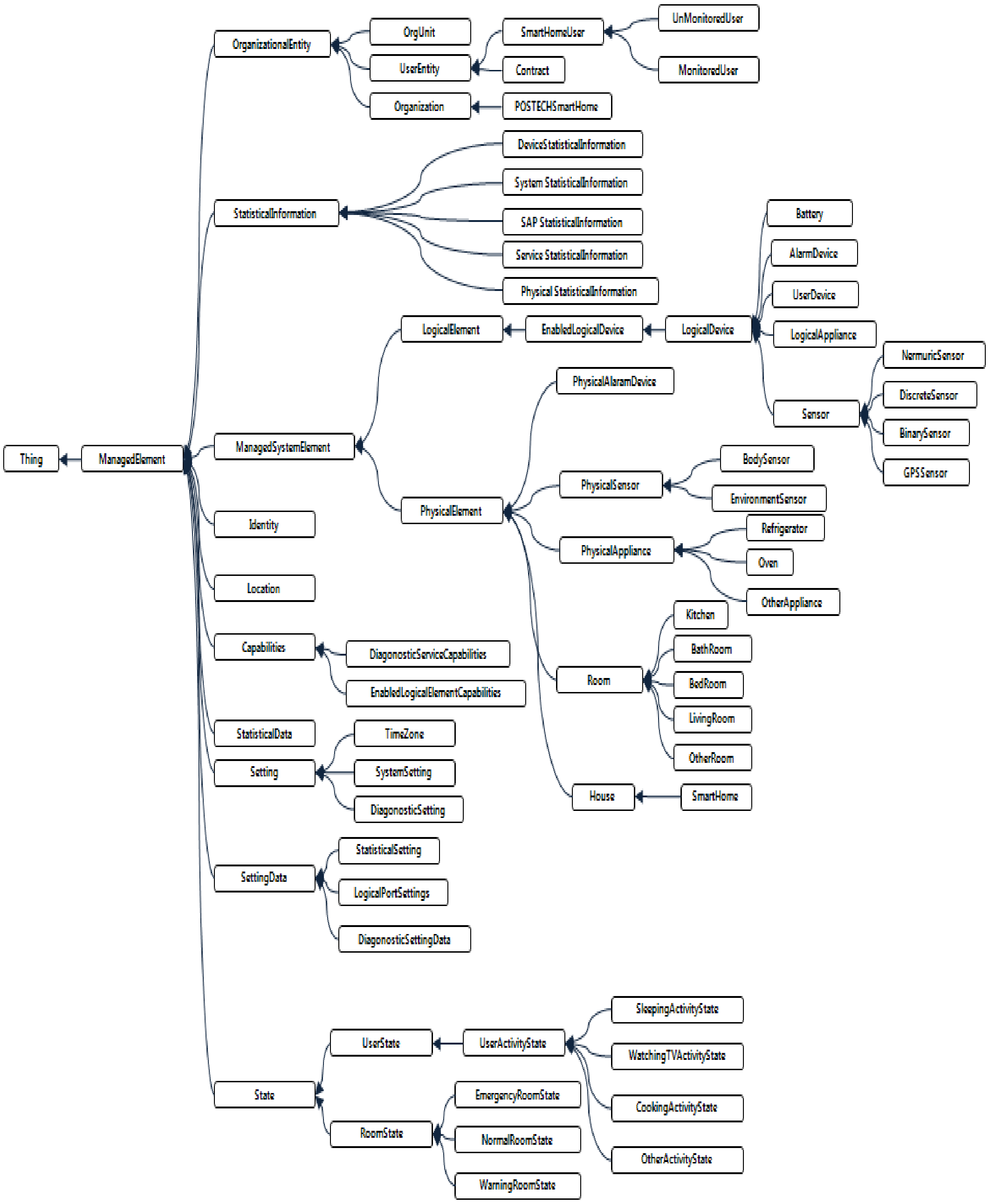

Fig. 13 Whole Classes Taxonomy of SHOM

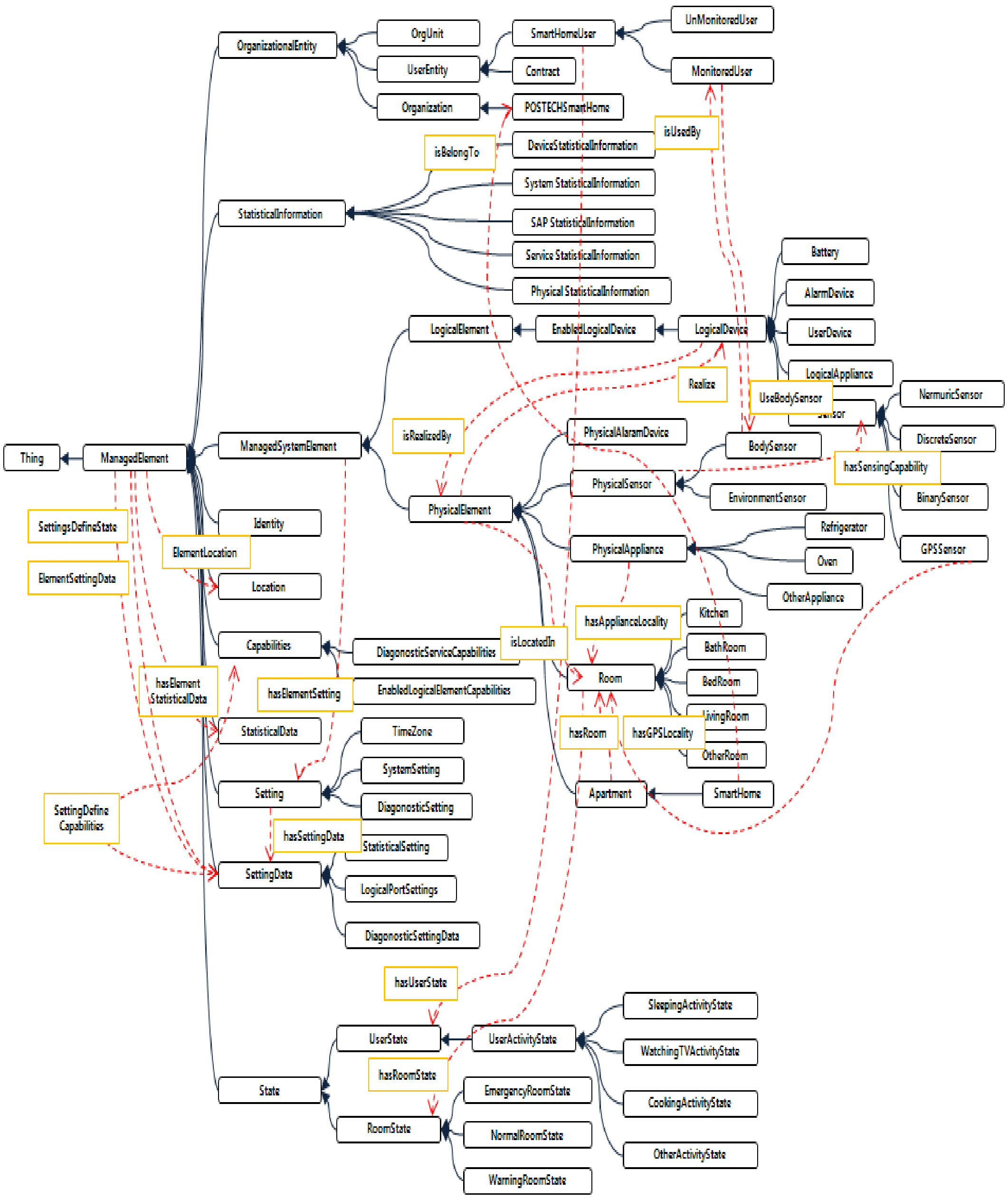

Fig. 14 Taxonomy with object properties of SHOM

4.3 Reasoning

The reasoning is responsible for checking ontology inconsistency and the implied relationships, along with testing and checking the rules for understanding the context. There are two kinds of approach; ontology reasoning and user-defined role-based reasoning. Ontology reasoning is used to check the ontology itself, such as looking for inconsistencies between classes and relations. User-defined role-based reasoning is based on first order logic, and computes the inferred reason using user-defined roles. Both approaches were used in this research. For ontology reasoning, we used the functionality of the Protégé tool. In Protégé, we can easily check class inconsistencies by selecting an appropriate reasoner. User-defined roles using SWRL were used to take advantage of inferencing.

4.3.1 Ontology Reasoning

Ontology reasoning uses transitive, inverse, symmetric, subclassof, subPropertyOf, disjointwith, and other properties or axioms to reason about the consistency of the ontology (see Table. 6). For example, the Realize relation in the SHOM model is an inverse property between PhysicalElements and LogicalDevice. If there is a physical element, such as a physical sensor, then the physical sensor must realize a logical sensor. A logical sensor is realized by a physical sensor. In subclassof usage, because the physical sensor and physical appliance are subclasses of PhysicalElements, those also have a Realize relation.

Table. 6 OWL ontology reasoning rules

Transitive	(?P rdf:type owl:TransitiveProperty) ∧ (?A ?P ?B) ∧ (?B ?P ?C) => (?A ?P ?C)
subClassOf	(?A rdfs:subClassOf ?B) ∧ (?B rdfs:subClassOf ?C) => (?A rdfs:subClassOf ?C)
subPropertyOf	(?A rdfs:subPropertyOf ?B) ∧ (?B rdfs:subPropertyOf ?C) =>

	(?A rdfs:subPropertyOf ?C)
disjointWith	(?C owl:disjointWith ?D) ∧ (?X rdf:type ?C) ∧ (?Y rdf:type ?D) => (?X owl:differentFrom ?Y)
inverseOf	(?P owl:inverseOf ?Q) ∧ (?X ?P ?Y) => (?Y ?Q ?X)
Symmetric	(?P rdf:type owl:SymmetricPropery) ∧ (?A ?P ?B) => (?B ?P ?A)

4.3.2 User-defined Role-based Reasoning

User-defined role-based reasoning, which is defined using first order logic, means user-defined roles are used to infer reasons. To do reasoning using this method, the Pellet reasoner was chosen here [20] and a rule set was established using the Semantic Web Rule Language (SWRL)[10], which is a proposal for a semantic web rule-language, combining sub-languages of the OWL Web Ontology Language (OWL-DL and Lite) with those of the Rule Markup Language. SWRL uses logic operators to define rules, and SWRL rules are stored in the ontology as concepts.

For example, consider the following two sentences:

If the elderly person is in their bed and the TV is off and the light is off, then the elderly person is sleeping. If the elderly person is sleeping, then set the temperature in the room to 27 degrees.

A corresponding SWRL formal specification is:

SmartHomeUser(?x) ∧ isInRoom(?x, bedroom) ∧ hasApplianceState(tv_bedroom, false) ∧ hasApplianceState(light_bedroom, false) → hasUserActivityState(?x, sleeping)
hasUserActivityState(?x, sleeping) ∧ TemperatureSensor(?y) → changeTheValue(?y, 27)

The first part of this SWRL specification defines the rules for reasoning, while the second part defines the appropriate actions to take. For example, hasUserActivityState(?x, sleeping) means the user X is presumed to be sleeping. In this project, information about user behavior is obtained from the system.

One area for future enhancement is adding feedback to the system. For example, if the decision was wrong, then appropriate feedback must be presented to the system in order to appropriately adapt the system and/or ontology model.

Table. 7 Comparison between reasoners [27]

	Bossam	Pellet	KAON2	RacerPro	Jena	FaCT++
OWL-DL entailment		Yes	Yes	Yes		Yes
Expressivity		SROIQ(D)	SHIQ(D)	SHIQ(D-)		SROIQ(D)
Reasoning algorithm	Rule-based	Tableau	Resolution &Datalog	Tableau	Rule-based	Tableau
Consistency checking		Yes		Yes	Incomplete for OWL - DL	Yes
DIG support	No	Yes	Yes	Yes	Yes	Yes
Rule support	SWRL&own rule format	SWRL	SWRL	SWRL-not fully support & own rule format	Own rule format	No
Licence	Free/clased-cource	Free/open-source&nonFree /closed-source	Free/closed source	nonFree /closed-source	Free /open-source	Free/open-source

Fig. 15 is an owl representation of the part of the ontology for SWRL rule that knows the user state (e.g., whether the user is sleeping or not). The rules were made in this project using SWRL and were tested using the Jess rule engine [13]. To use Jess in Protégé 3.4.1, the first step is downloading the jess jar file, which is contained in the standard Jess distribution. This jar file must be copied to the Protégé-OWL plug-in`s sub-directory in the Protégé installation directory. When the SWRLTab is activated, the

SWRL Editor will display a list of icons for plug-in that have registered themselves with the SWRLTab. The Jess tab is activated by pressing the “J” icon. To test the SWRL rule, the“OWL+SWRL->Jess” button should be pushed. Then, the Jess engine should be executed by selecting the “Run Jess” button to validate the rules. The last step is to push the “Jess->OWL” button, which places the result in the owl file.

```
<swrl:head>
        <swrl:AtomList>
          <rdf:first>
            <swrl:IndividualPropertyAtom>
              <swrl:propertyPredicate rdf:resource="#hasUserActivityState"/>
              <swrl:argument2>
                <SleepingActivityState rdf:ID="sleeping"/>
              </swrl:argument2>
              <swrl:argument1 rdf:resource="#x"/>
            </swrl:IndividualPropertyAtom>
          </rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </swrl:AtomList>
      </swrl:head>
    </rdf:Description>
    <swrl:Imp rdf:ID="UserSleepState">
      <swrl:body>
        <swrl:AtomList>
          <rdf:first>
            <swrl:ClassAtom>
              <swrl:classPredicate rdf:resource="#SmartHomeUser"/>
              <swrl:argument1 rdf:resource="#x"/>
            </swrl:ClassAtom>
          </rdf:first>
          <rdf:rest>
            <swrl:AtomList>
              <rdf:first>
                <swrl:IndividualPropertyAtom>
                  <swrl:propertyPredicate rdf:resource="#isInRoom"/>
                  <swrl:argument2 rdf:resource="#bedroom"/>
                  <swrl:argument1 rdf:resource="#x"/>
                </swrl:IndividualPropertyAtom>
              </rdf:first>
              <rdf:rest>
                <swrl:AtomList>
                  <rdf:first>
                    <swrl:DatavaluedPropertyAtom>
                      <swrl:propertyPredicate rdf:resource="#hasPhysicalApplianceState"/>
                      <swrl:argument1 rdf:resource="#tv_bedroom"/>
                      <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
                      >false</swrl:argument2>
                    </swrl:DatavaluedPropertyAtom>
                  </rdf:first>
                  <rdf:rest>
                    <swrl:AtomList>
```

```
                              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                              <rdf:first>
                                <swrl:DatavaluedPropertyAtom>
                                  <swrl:propertyPredicate rdf:resource="#hasPhysicalApplianceState"/>
                                  <swrl:argument2 rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean"
                                  >false</swrl:argument2>
                                  <swrl:argument1 rdf:resource="#light_bedroom"/>
                                </swrl:DatavaluedPropertyAtom>
                              </rdf:first>
                            </swrl:AtomList>
                          </rdf:rest>
                        </swrl:AtomList>
                      </rdf:rest>
                    </swrl:AtomList>
                  </rdf:rest>
                </swrl:AtomList>
              </swrl:body>
              <swrl:head>
                <swrl:AtomList>
                  <rdf:first>
                    <swrl:IndividualPropertyAtom>
                      <swrl:propertyPredicate rdf:resource="#hasUserActivityState"/>
                      <swrl:argument1 rdf:resource="#x"/>
                      <swrl:argument2 rdf:resource="#sleeping"/>
                    </swrl:IndividualPropertyAtom>
                  </rdf:first>
                  <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
                </swrl:AtomList>
              </swrl:head>
            </swrl:Imp>
```

Fig. 15 Part of ontology for SWRL rule

For example, the previous SWRL rule was;

SmartHomeUser(?x)∧

isInRoom(?x, bedroom)∧

hasPhysicalApplianceState(tv_bedroom, false)∧

hasPhysicalApplianceState(light_bedroom, false)

→ hasUserActivityState(?x, sleeping)

Fig 16 and 17 show the result of this rule test using protégé 3.4.2. For the example, the jess rule engine is used. Mr.Kim is an instance of SmartHomeUser. The instance has three data properties and tree object properties. For this example, note the red box (hasUserActivityState). In Fig. 16, the initial UserActivityState is cooking and the position of the user is in the bedroom. The reason why the relationship between each property is not reasonable is that values in the SHOM can be updated automatically. In the UserActivity case, the value of that class changes according to the user-define rule. However, Fig. 16 and Fig. 17 are the result of the jess rule engine, so Fig. 17 shows two UserActivitys. The reason, why the result of rule is two, is because this is a test of the rule using the jess engine. In the real system, the result will be one.

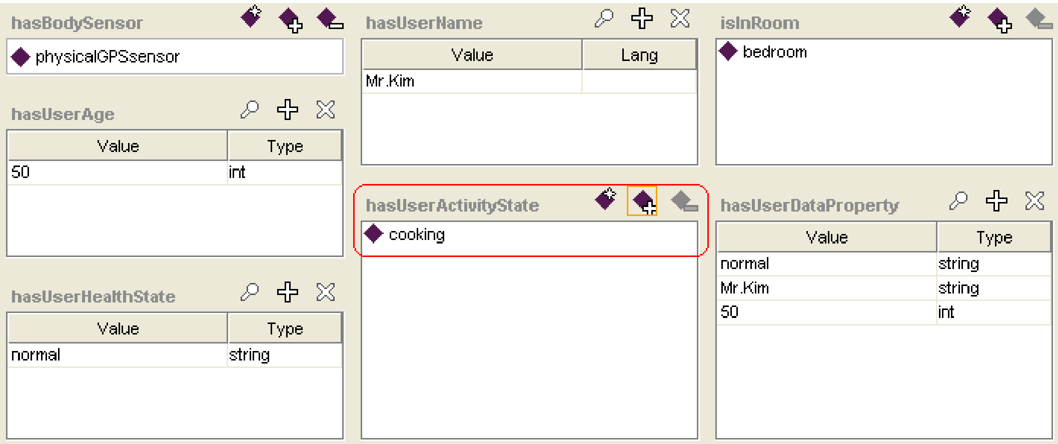

Fig. 16 Before the rule is applied

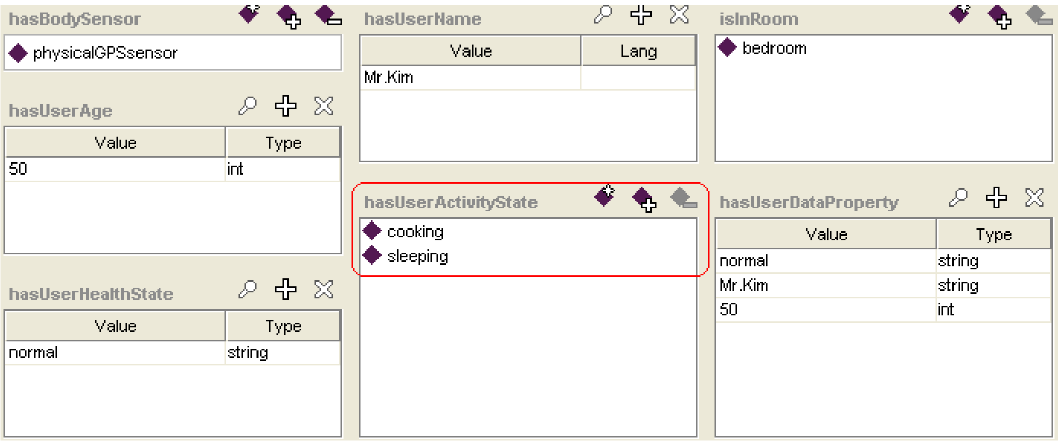

Fig. 17 After the rule is applied

5 Validation

This section introduces the development environment of the system and the use cases using the SHOM model.

5.1 Development Environment

Version cim_schema_2.23.0_final of the CIM model is used as the foundation of the ontology model for this research. However, only parts of the CIM are useful for this ontology. Those parts are the CIM_Core, CIM_Device and CIM_System. To make the ontology model, OWL language and Protégé [11] were used in this research, which is an open source tool to create and edit ontology. The OWL API [19] is a java interface and implementation for the World Wide Web Consortium OWL. Version 2.2.0 of this API,was used to communicate between the ontology and the decision making system, and manipulate the ontology according to information from the distributed framework (see Fig. 5 in section 3). For reasoning, Version 2.0.0 of Pellet API [20] was used. Pellet is an OWL 2 reasoner that provides standard and cutting-edge reasoning services for OWL ontologies. To use these APIs, version 3.5.1 of the Eclipse SDK was used, and JAVA version was JAVA SE Runtime Environment 1.6.0_17.

At the first run, protégé 4.0 was used as a tool to create the ontology for OWL 2, but to test SWRL rule, protégé 3.4.2 was also used (protégé 4.0 doesn't provide a way to use SWRL). The current version of SHOM is made by protégé 3.4.2. (Using OWL 1).

5.2 Usecases

To explain the usage of the SHOM model on the autonomic decision system, several usecases scenarios are presented here. There are two kinds of use cases. The first one is to provide convenience for elderly people, and the second use case is to detect health problems and take proactive actions for elderly people.

Usecase 1. When Elderly People Leave the Smart Home

Most elderly people spend a lot of time at home alone. Sometimes, they leave their home. At that time, they frequently forget something, such as to turn off the light or gas, or lock the door. In this case, the decision-making system in the project knows that the elderly person goes out and the smart home is empty. According to defined rules, the decision-making system will turn off the TV or light for energy reduction, turn off the gas for safety, and lock the door for security automatically. In addition, environment and body sensors don't need to send information to the system, so they change their mode to sleep mode to reduce their energy consumption, except for the motion detect sensors for security reasons.

```
SmartHomeUser(?x)
        If no, then error message print using alarm device.
MonitoredUser(?x)
        If yes, useBodySensor(?x, ?y)  ∧  Realize(?y, ?z)  ∧  hasGPSLocality (?z, ?a)
        Else,
                ¬  isInRoom(?x, ALL_ROOMS)
                // y is all physicall appliances.
                ∧  (PhysicalAppliance(?y)  ∧  hasApplianceState(?y, off))
                // b is all motion sensors.
                ∧  EnvironmentSensor(?b)  ∧  hasSensorState(?b, off)
If the smart home is not empty, then return.
Else, then
Result: Decide to execute SmartHomeEmpty mode. In this mode, the autonomic
system will turn off sensors and appliance except special sensors and appliance, such as
motion capture sensors and air sensors and temperature sensor in kitchen and door lock,
etc.
```

Fig. 18 Pseudo SWRL code to know the situation SMART_HOME_EMPTY

Usecase 2. ECG Signals Something is Wrong

This use case represents one of the most important situations for elderly people. In a smart home, elderly people must use body sensors to monitor their health status. Specifically, an ECG sensor can detect serious problems, such as the danger of an imminent heart attack. This use case is for this and other emergency situations. If the system detects dangerous signals generated by the ECG sensor, then it checks whether the elderly person is currently using that sensor or not. If yes, then it changes the status to the emergency status and makes an emergency call. If not (e.g., the elderly person takes the sensor off), then it changes the status to warning until the system receives enough information from other sensors to verify the status of the elderly person.

```
SmartHomeUser(?x) ∧ MonitoredUser(?x)
BodySensor(?y) ∧ isUsedBy(?y, ?x) // y is ECG sensor
hasECGSensorValue(?y, strange)
        If no, then return.
        Else, then
            Change the x`s UserHealthState to warning.
            AnalyzeECG_Signal( ) // this function is in the system.
            If the result of AnalyzeECG_Signal( ) indicates the serious state of x, then
                Change the x`s UserHealthState to Emergency.
                Call the doctor, and
                PhysicalAlarmDevice(?z), change the z`s state to activate.
            Else, then
                Keep the x`s UserHealthState to warning.
                PhysicalAlarmDevice(?z), change the z`s state to activate
                for given time period.
                   If x do appropriate action such as turn off the alarm, then
                          Change the x`s UserHealthState to normal.
                   Else then, call the doctor.

Result : In this case, otherwise previous usecase, there are two steps to make an emergency call. First step is decided by the result of AnalyzeECG_Signal( ) function. Second step is decided by decision making system according to the situation.
```

Fig. 19 Pseudo SWRL code to make an emergency call

6 Conclusions

6.1 Thesis Summary

Section 1 presented an introduction to the thesis and the goals for the research. This section states the open problems for the research community.

Section 2 described related work on smart home projects, autonomic systems and semantic web technologies. The introduction included why and how these technologies and concepts can be used in this work. Also, they were compared with the approach of this thesis. Section 3 introduced the proposed general architecture and autonomic decision making system for the U-Health smart home. Also, the requirements for the U-health smart home system were investigated and presented. Section 4 introduced the SHOM model for the smart home at POSTECH. The principles used to create the SHOM and the reasoning of SHOM and the usage of CIM features were shown. Section 5 showed the development environment and explained about the usercases and implementation.

Section 6 summarizes the contributions of this thesis and outlines future work options.

6.2 Appraisal of the Thesis

The major contributions of this thesis are summarized as follows.

A proposal for the architecture for an U-Health smart home system was presented for the U-Health smart home at POSTECH. The SHOM ontology model was defined. SHOM uses some of the CIM model components (classes and relations). SHOM defines a good introductory context ontology model compared to other models. In addition, it can be used to track different situations in a smart home as well as manage different kinds of elements in a smart home. The autonomic system for the U-Health smart home project at POSTECH can use the SHOM semantic model, which provides a good starting foundation for making decisions using an autonomic system.

6.3 Future Work

Although SHOM has been defined, it can be enhanced in different ways. For example, the model and reasoning components can be made more complete, and the reasoning part can support more intelligent decisions and actions. In addition, other components, which are defined in CIM, can be used. It is possible because the structure of CIM is still maintained. Also, there are several considerations to be made relating to SHOM. For example, a decision should be made on how to use the *state*. In SHOM, the user activity state is defined as a class, but the health state is defined as a dataproperty.

In this thesis a first approach was defined for constructing an ontology model and reasoning rules for the U-Health Smart Home project, but these must be used in a real test-bed. To do this, the autonomic decision-making system can communicate with other distributed framework components, such as OSGi bundles. Fig 20 shows a sample sequence depicting the flow of events in this type of program. The information that is generated in a smart home is sent to the OSGi framework (1). In the OSGi framework, a monitor bundle can gather information and send it to the Ontology knowledge base to update it (2, 3). Also, if the value of the information reaches a specific threshold, the symptom can then be sent for further analysis (4). To analyze the symptom, the bundle attaches the reasoning part in an autonomic decision system (5). In this step, pellet API is used for reasoning. If the symptom is simple and doesn't need to use an ontology, then the result will be directly available (6). Otherwise, inferred rules and ontology knowledge in an autonomic decision system will be used (7, 8, 9, and 10). The result is sent to the plan bundle to compare it to the planning rule and the result (11, 12, 13, and 14). As the final decision goes through the execute bundle (15), the corresponding actions will be executed in the smart home (e.g., change some mode or send an alarm) (16, 17).

To enhance the model and system, testing will be performed in the POSTECH Smart Home to understand and predict user behavior at home and test the possible medical care services. Using these tests, the autonomic system will learn about user behavior, and the SHOM model will be made more suitable for the POSTECH U-Health smart home.

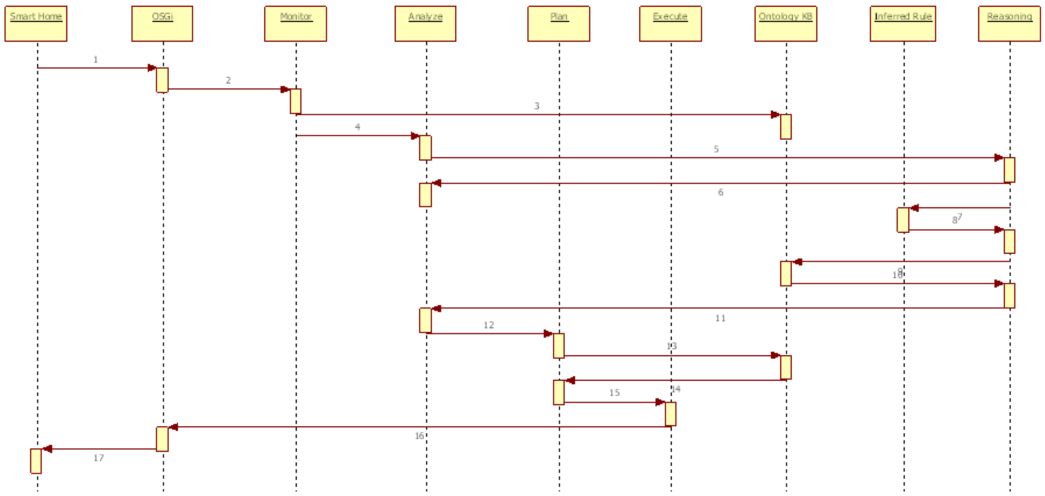

Fig. 20 Sequence diagram for autonomic decision

References

[1] Berners-Lee, “The Semantic Web Vision”, Keynote session at XML 2000, 6 Dec 2000

[2] Norbert Noury, “AILISA: Experimental platforms to evaluate remote care and assistive technologies in gerontology”, HEALTHCOM 2005, 23-25 June 2005, pp. 67-72

[3] Norbert Noury, Pierre Rumeau, “Ambient Multi-Perceptive System for Residential Health Monitoring Based on Electromic Mailings”, HEALTHCOM 2006, 17-19 August 2006, pp. 95-100

[4] Chih-Jen Hsu, Chiou Yii-Tay, Chang Cheng-Lii, Hwang Jlng-Huei, Jone-Chiung, “Telemedicine Information Monitoring System”, HEALTHCOM 2008, 2008

[5] UbiSense project, http://www.doc.ic.ac.uk/vip/ubisense/home/home.html

[6] IBM Corporation, “An architectural blueprint for autonomic computing”, April 2005, http://www.ginkgo-networks.com/IMG/pdf/AC_Blueprint_White_Paper_V7.pdf

[7] Tom Gruber, “Ontology”, Encyclopedia of Database Systems, Lin Ling Liu and M. Tamer Ö zsu (Eds.), Springer-Verlag, 2008

[8] OWL Web Ontology Language Overview, http://www.w3.org/TR/owl-features/

[9] Berners-Lee, T., Hendler, J. and Lassila, O., “The Semantic Web”, Scientific American, May 2001

[10] SWRL: A Semantic Web Rule Language Combining OWL and RuleML, http://www.w3.org/Submission/SWRL/

[11] Protégé, http://protege.stanford.edu/.

[12] Jena framework, http://jena.sourceforge.net/

[13] Jess rule engine, http://www.jessrules.com/

[14] John Strassner, "Knowledge engineering using ontologies", Handbook of Network and System Administration, chapter 3, section 4, pages 425-457

[15] Common Information Model Standard, http://www.dmtf.org/standards/cim/

[16] Resource Description Framework(RDF): Concepts and Abstract Syntax, http://www.w3.org/TR/rdf-concepts/

[17] IBM research, "The Vision of Autonomic Computing", Jun 2003

[18] P.Horn, "Autonomic Computing: IBM`s Perspective on the State of Information Technology", IBM Corp., October 2001, http://www.research.ibm.com/autonomic/research/papers/AC_Vision_Computer_Jan_2003.pdf

[19] OWL API, http://owlapi.sourceforge.net/

[20] Pellet: OWL 2 Reasoner for Java, http://clarkparsia.com/pellet

[21] IRIS Smart Home project, http://www.dom-iris.si/en/

[22] S.Neumann, A Devos, S Widergren, J Britton, "User of the CIM ontology", Distridutech 2006

[23] Daniela Berardi, Diego Calvanese, Giuseppe De Giacomo, Giuseppe Santucci, Antonio Perrone, Guido Vetere, "Using the CIM Conceptualization in Autonomic System Management the System Management Ontology Project", DSOM 2003, Heidelberg, Germany, October 20-22

[24] Dennis Heimbigner, "DMTF – CIM to OWL: A Case Study in Ontology Conversion", SEKE2004, Canada, 2004, pp.470-474

[25] Marta Magewska, Bartosz Kryza, Jacek Kitowski, “Translation of Common Information Model to Web Ontology Language”, Lecture Notes in Computer Science, Volum 4487, 2007, pp.414-417

[26] CIMTool Project, http://cimtool.org/

[27] Semantic reasoner, http://en.wikipedia.org/wiki/Semantic_reasoner

[28] Jorge E. Lopes de Vergara, Victor A. Villagra, Carlos Fadon, Juan M. Gonzales, Jose A. Lozano and Manuel Alvarez-Campana, “An Autonomic Approach to Offer Services in OSGi-Based Home Gateways”, Computer and Communications, Volume 31, Issue 13, 15 august 2008, pp. 3049-3058

[29] James E. Hanson, Ian Whalley, David M. Chess, Jeffrey O. Jephart, “An Architectural Approach to Autonomic Computing”, International Conference on Autonomic Computing, 2004, pp.2-9

[30] Tao Gu, Hung Keng Pung, Da Qing Zhang, “A Service-Oriented Middleware for Building Context-Aware Services”, Journal of Network and Computer Applications, Volumes 28, Issue 1, January 2005, pp. 1-18

[31] Valtchev, D., Frankov, I., “Service Gateway Architecture for a Smart Home”, IEEE Communications Magazine, Volume 40, Issue 4, April 2002, pp. 126-132

[32] Vincent Rialle, Florence Duchene, Nerbert Noury, Lionel Bajolle, Jacques Demongeot, “Health “Smart” Home: Information Technology for Patients at Home”, Telemedicine Journal and e-Health, Volume 8, Issue 4, July 9, 2004, pp. 395-409

[33] Haitao Zhang, Fei-Yue Wang, Yunfeng Ai, “An OSGi and Agent Based Control System Architecture for Smart Home”, IEEE Networking, Sensing and Control, 19-22 March 2005, pp. 13-18

[34] Wolfgang Trumler, Faruk Bafci, Jan Petzold, Theo Ungerer, “AMUN-Autonomic

Middleware for Ubiquitous Environments Applied to the Smart Doorplate Project", Advanced Engineering Informatics, Volume 19, Issue 3, July 2005, pp.243-252

[35] Dominik Heckmann, Tim Schwartz, Bris Brandherm, Micheal Schmitz, Margeritta von Wilamowitz-Moellendorff, "GUMO – The General User Model Ontology", Lecture Notes in Computer Science, Volume 3538, pp. 428-432

[36] Dario Bonino, Fulvio Corno, "DogOnt – Ontolgoy Modeling for Intelligent Domotic Environments", Lecture Notes in Computer Science, Volume 5318, October 26-30, 2008, pp. 790-803

[37] European Home System, http://www.ehsa.com

[38] Dario Bonino, Emiliano Castellina, Fulvio Corno, "DOG: An Ontology-Powered OSGi Domotic Gateway", IEEE International Conference on Tools with Artificial Intelligence, 3-5 November, 2008

[39] CodeBlue project, http://fiji.eecs.harvard.edu/CodeBlue

[40] Duke Smart Home Program, http://www.smarthome.duke.edu/index.php

[41] Ontology editor survey result, http://www.xml.com/2002/11/06/Ontology_Editor_Survey.html.

[42] Franz Baader, "Description Logics", Lecture Notes in Computer Science, Volume 5689, 2009, pp. 1-39

Summary

과학 기술이 발전하고 컴퓨터가 사람의 생활에 필수적인 요소로 들어오게 된 이후, 그 활용에 대한 연구가 끊임없이 진행되어왔다. 그 중, 건강한 삶을 영위하고자 하는 사람들을 도와주기 위한 연구들이 진행되어왔다. 왜냐하면 나이가 들어가면서 건강을 지키고 질병에 걸리지 않는 데 많은 노력을 아끼지 않기 때문이다. 이런 건강에 대한 우려에 대해 스마트 홈 시스템이 그 대한으로 제시될 수 있다. 하지만 효율적이고 지능적인 스마트 홈 시스템을 구축하기 위해서는 많은 고려할 점이 존재한다. 나이든 사람들이 지닐 수 있는 질병의 다양성과, 질병에 의한 진단의 복잡함에 따른 어려움, 스마트 홈 내부에 존재하는 다양한 종류의 기기들에 대한 관리와 네트워킹 등, 많은 문제점들이 풀어야 할 문제로 제기되고 있다.

본 연구는 포항공대 U-health 스마트 홈 프로젝트의 일부분으로써, 앞서 언급한 문제들을 해결할 수 있는 시스템 구축의 초기 단계의 수행결과이다. 연구의 목적은 시멘틱 웹 기술들을 이용하며 구축한 온톨로지와 리즈너를 만드는 것이며, 이는 U-health 스마트 홈을 위한 자율 컴퓨팅 환경 구축의 기본이 되는 것이다. 스마트 홈에 관련한 많은 온톨로지들이 이미 구현이 되어있지만, 해당 온톨로지에 대한 정당성과 타당성을 언급하기는 어렵다. 온톨로지 구현의 목적은 해당 시스템에 최적화되어 만들어지기 때문이다. 이러한 이유로 본 연구에서 제안하는 모델은 이와 같은 단점을 극복하고 모델에 합리성을 부가하기 위해 국제 표준인 CIM 모델을 변형하여 구축함으로써 스마트 홈 내부에 있는 다양한 기기들에 대한 관리를 효율적이고 합리적으로 수행할 수 있게 하였고, 추가로 다양한 클래스와 프로퍼티들를 생성함으로써 의사 결정 시스템이 상황을 인지하여 정확한 결정을 내릴 수 있게 하는 기반을 제공한다. 또한 다양한 SWRL로 작성된 규칙을 제공함으로써 시스템의 영리한 판단을 내릴 수 있고, 정확한 의사 결정을 가능하게 하는 논리적인 의사 결정 흐름을 생성하여 시스템의 동작에 대한 자율화를 가능하게 한다.

기존의 연구들과 구분이 되는 있는 점은 다양한 분야를 위한 온톨로지 모델 구축이라는 것이다. 단순히 상황을 인지하기 위함이 아닌, 인지 후 시스템에 의한 효율적이고 지능적인 의사결정, 스마트 홈 시스템 내부의 기기들에 대한 합리적인 관리가 가능하게 하기 위함이 본 연구에서 제시하는 모델의 목적이기 때문이다.

논문의 구성은 다음과 같다. 1장에서는 문제 상황에 대한 제기와 파악, 연구의 목적이 기술되어 있다. 2장에서는 U-health 스마트 홈에 대한 관련 연구들과 본 연구에서 사용되는 기술이나 개념에 대한 전반적인 정의들이 설명되었고, 연구에서 해당 기술들이 어떻게 사용되면 기존의 연구들과 어떠한 점이 다른지에 대한 설명이 되었다. 3장에서는 본 연구에서 제안하는 모델이 사용될 스마트 홈을 위한 자율 컴퓨팅 환경에 대한 설명이 기술되었다. 이 자율 컴퓨팅 환경은 분산된 모듈에 의한 프레임워크로 되어 있으며, 각 모듈의 역할과 의사결정에 의해 필요한 흐름에 대해서 설명한다. 4장에서는 제시된 모델의 구현 목표와 방법이 제시되었으며, 본 모델을 이용한 리즈너의 역할과 구현에 대한 방법이 예시와 함께 기술되었다. 또한, 국제 표준인 CIM에 대한 설명과 온톨로지로 사용하기 위해서 사용된 요소와 시멘틱 웹 기술을 사용하기 위한 CIM 요소의 변환방법과 활용에 대한 설명이 되었다. 추가로 의사결정 시스템이 효율적으로 의사결정을 하기 위해 추가적으로 생성된 요소에 대한 설명이 되어 있으며, 의사 결정의 논리적인 흐름에 대한 검증과 모델의 무결성 보장을 위한 리즈너의 설명이 되어있다. 5장에서는 온톨로지 모델과 리즈너의 구현환경과 사용가능한 시나리오가 기술되고, 해당 시나리오를 지지할 수 있는 의사결정을 위한 SWRL 규칙 생성을 위한 구현 방법이 기술되었다. 마지막 6장에서는 본 논문에 대한 요약과 공헌, 앞으로 개선해야 할 사항이 기술되었다. 본 모델은 U-health 스마트 홈 시스템에서 사용될 모델이기 때문에 해당 시스템이 어떻게 구성이 되어야 할지에 대한 언급과 데이터 흐름에 대한 시퀀스 다이어그램이 설명되었다.

감사의 글

저에게 포항공대에서의 2년이라는 시간은 지금까지 중에서 가장 빠르게 지나간 시간이었습니다. 대한민국 최고의 학교에서 공부를 할 수 있게 되었다는 자긍심과, 한편으로는 잘 해낼 수 있을까 하는 우려에 포항공대에서의 생활이 시작되었었습니다. 남들보다 늦게 시작한 공부였기에, 너무 의욕이 앞서 힘들었던 때도 있었습니다만, 홍원기 지도교수님의 넘치는 관심과 연구실 선/후배님들의 격려와 지원 속에 2년간의 공부를 잘 마무리 할 수 있었습니다. 이에 짧게 나마 감사의 글을 올립니다.

제가 우리나라 최고의 교육을 받을 수 있게 허락해 주시고, 끝까지 관심과 지도 아끼지 않으신 홍원기 교수님께 진심을 담아 감사의 말씀드립니다. 교수님의 훌륭하신 지도와 바른 생활모습은 많은 사람들에게 모범이 되십니다. 언제나 교수님의 지도학생이었다는 것을 생각하며, 교수님 명성에 누가 되지 않는 사람이 되도록 노력하겠습니다. 또한, 대학교때 저를 지도해주시고 먼 곳에서도 저의 공부와 생활에 대해서 관심과 격려를 아끼지 않으신 고려대학교 김명섭 교수님께도 감사의 말씀드립니다. 그간 연락 자주 드리지 못하고 찾아뵙지도 못해서 죄송합니다. 앞으로는 자주 연락드리도록 하겠습니다. 그리고 계명대학교 주홍택 교수님께도 감사드립니다. 연구실에 오실 때마다 들러서 격려의 말씀 잊지 않으셨고 연구활동에도 훌륭한 조언 해주신 점 잊지 않겠습니다. 강원대학교 최미정 교수님께도 감사의 말씀드립니다. 제가 가장 힘들 때, 옆에서 토닥여주신 기억 잊지 않고 있습니다. 행복한 가정꾸리시고 연구활동도 잘 되시기를 진심으로 바라겠습니다.

이번 학위 논문을 작성하면서 많은 조언과 도움을 주신 외국 교수님들께도 감사하다는 말씀드리고 싶습니다. 4개월 동안 새로운 분야에 대한 연구를 원활히 진행될 수 있도록 1주일에 3번, 2시간씩 시간을 내서 진행상황이나 문제점들에 대한 조언을 해주신 Nazim Agoulmine 교수님과 Jamal Dean 교수님, 진심으로 감사합니다. 저의 부족함을 채워주시고 매번 잘한다는 칭찬에 제가 무사히 논문을 완성할 수 있었습니다. 또한, 온톨로지와 자율 컴퓨팅에 무지한 저에게 웃으면서 훌륭한 지도해주시고, 바쁘신데도 불구하고 영어로 작성된

논문에 대해 세세한 부분까지 관심과 지도해 주신 John Charles Strassner 교수님 감사합니다.

2 년간 동고동락한 DPNM 연구실 선배님들과 하나뿐인 동기에게도 감사의 말씀 전합니다. 항상 듬직하시고 뛰어난 연구활동하시는 준명 형님. 올해 큰 계획 잘 이루시길 바라며 행복한 가정의 훌륭한 아버지 되시리라 믿습니다. 마지막까지도 격려와 관심 감사했습니다. 나이는 저보다 한살 어리지만 정말 뛰어난 인재인 박사 영준이. 졸업 후 새로운 환경에서도 지금까지 처럼 잘해낼 것입니다. 랩장으로써 형인 저를 챙기느라 고생 많이 했습니다. 그리고 서로의 고민에 대해서 가장 이야기를 많이 하고, 서로를 격려하는 진지한 시간을 가장 많이 가진 성철이. 지금 비록 연구하는 부분에 대해서 잘 풀리지 않고 있지만, 잘 할수 있을 것이라 믿습니다. 포항공대 내에서도 특유의 명석함을 뽐내는 병철이. 앞으로는 랩장으로써의 역할도 잘 해내고 연구활동도 잘 해내길 바랍니다. 랩의 분위기 메이커이자 미래의 인터넷을 선도할 김성수. 공부와 생활, 모든 면에서 모범을 보이는 모습에 동생이어서 배울 점 많다고 생각합니다. 그리고 저의 사랑스러운 동기이자 연구에서도 생활에서도 모범적이며 뛰어난 신석이. 고민이 있을 때 잘 들어주고 힘내라고 격려해준 점 감사하게 생각하고 있습니다. 앞으로 연구활동도 잘 해내길 바랍니다.

바쁘다고 많은 시간을 함께하지는 못했지만, 저를 항상 응원해준 후배님들께도 감사의 말을 전합니다. 가장 힘든 연구 중, 가장 큰 힘이 되어준 바른 생활 후배인 혁수. 덕분에 논문 진행을 잘할 수 있었습니다. 고맙습니다. 먼 곳에서 공부하러 와서 힘들겠지만 항상 웃으면서 반겨준 리건. 잊지 못할 추억 만들어준 점 역시 고맙습니다. 포항공대 최고 미모를 자랑하는 아름이. 학교에서 즐거운 생활하고, 연구도 생활에서도 최고가 되길 바랍니다. 연구실 막내로써 잡다한 일, 궂은 일 도맡아서 하면서도 연구도 열심히 하는 재윤이. 더 많은 시간을 같이 하지 못해서 아쉽지만, 앞으로 잘해낼 것이라고 믿습니다. 막내이지만 듬직한 영락이. 올해부터는 그만 아프고 연구활동과 학교생활 모두 잘해내길 바랍니다.

잠시동안이지만, 연구실에서 같이 생활한 모든 분들께도 감사의 말씀드립니다. 군인의 신분으로 돌아가서 열심히 나라를 지키는 이성주 대위, 이제는 다

른 환경에서 새로운 출발을 하는 성호, 한 아이의 어머니가 된 Suman Pandy, 그녀의 남편 Pawan Kumar Tiwari, 이탈리아에서 온 커피 친구 David Monichi, 훤칠한 키에 웃는 얼굴이 참 멋진 Maarten Aersten, 술과 노는 것을 좋아했던 프랑스 친구 Vivian, 많은 이야기를 나누었던 순진하고 착한 인도청년 Tara sasank, 마지막 연구에 많은 격려와 도움을 준 중국에서 오신 형님 Hui Wang. 연구실의 행정적인 일을 제일 마냥 신경써주고 수고해주었던 정임이와 혜정이에게도 고맙다는 말 전하고 싶습니다. 여러분과의 소중한 인연 잘 간직하도록 하겠습니다.

기숙사에서 같은 공간과 시간을 함께하며 많은 이야기와 서로에 대한 격려를 아끼지 않은 수학과 친구 최성훈 군. 마음맞는 친구 만나기 쉽지 않은데 덕분에 재미있는 생활이 가능했던 것 같습니다. 고맙습니다. 또한 마지막까지 행정지원 해주시면서 응원의 글 남겨주신 이태화 선생님 감사합니다. 마지막으로 동기로써 서로 격려하며 졸업을 하게되는 정보통신 대학원 동기들에게도 감사의 말씀드리고 싶습니다.

이상 제가 언급하지 못하였지만, 저의 2년간 연구생활을 무사히 마치게 해주신 모든 분들께 진심으로 감사의 말씀 전합니다. 또한, 세상에서 가장 존경하는 저의 부모님과 형님, 감사하고 사랑합니다.

여러 고마운 분들에게 이렇게 밖에 감사의 말씀 전할 수 밖에 없음을 죄송하게 생각합니다. 이런 감사한 인연 소중히 간직하겠습니다. 끝으로, 저를 믿고 이끌어 주시고 격려해주신 모든 분들께 이 논문을 바칩니다.

이 력 서

성　　명 : 김　진

생년월일 : 1979 년 11월 28일

출 생 지 : 서울

주　　소 : 대전광역시 중구 선화동 163-18

학 력

2004.3 ~ 2008.2 고려대학교 컴퓨터정보학과 (B.S)

2008.3 ~ 2010.2 포항공과대학교 정보통신학과 (M.S.)

학 술 활 동

International Conference Papers

1. Seong-Cheol Hong, Jin Kim, Byungchul Park, Young J. Won, and James W. Hong, "Traffic Growth Analysis over Three Years in Enterprise Networks", 15th Asia-Pacific Conference on Communications, Shanghai, China, Oct. 8-10, 2009.

Domestic Conference Papers

1. 김진, 박병철, 홍성철, 홍원기, "엔터프라이즈 네트워크의 트래픽 경향 변화에 대한 분석", 한국신학회 하계종합학술발표회, Jeju, Korea, Jun 4.

연 구 활 동

1. U-Health Smart Home at POSTECH, 포항공과 대학교, 2009.