A Dynamic Service Composition Environment on Integrated Home Network System

by

Junsoo KIM

submitted to
Japan Advanced Institute of Science and Technology
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Supervisor: Professor Dr.Yasuo TAN

School of Information Science
Japan Advanced Institute of Science and Technology

September 24, 2013

Abstract

The goal of this thesis is a dynamic service composition environment on integrated home network system. To achieve this goal we investigated the following fourth research target: Firstly, our system provides a flexible service composition in various home environments by making the system to semantically understand an objective of a service. Moreover, it is possible to realize extendable services and priorities, since these semantic descriptions can be built separately in their own area by a name space. Secondly, the dynamic service priority makes the service manager determines various priority of a service element according to a required policy because various external policy definitions are providing a valuation of elements. Thirdly, we proposed the distributed service developing environment. the system is able to include an external ontology model by importing its name space. Such distributed development of a service model will improve scalability of the service model. In the near future, a distributed environment for developing home services is going to be needed by service provider Finally, the system provides a methodology on how to select the best link to integrated service manager in various home environment since the system can determines a transmitting route of contents suitably among networked home devices including legacy appliances. Our system provides to realize a dynamic service composition environment on integrated home network system.

Acknowledgments

The author wishes to express his sincere gratitude to his supervisor Professor Yasuo Tan of Japan Advenced Institute of Science and Techology for his constant encouragement and kind guidance during this work. The author would like to thank Professor Yoichi Shinoda of Japan Advanced Institute of Science and Technology for his helpful discussions and suggestions.

The author also wishes to express his thanks to Associate Professor Azman Osman Lim of Japan Advanced Institute of Science and Technology for his suggestions and continuous encouragements.

The author is grateful to Professor Mitsuru Ikeda of Japan Advanced Institute of Science and Technology for their helpful suggestions and discussions.

The author offers his special thanks to Ph.D Junya Nakata for his valuable and constructive suggestions during the planning and development of this research work.

The author is grateful to all who have affected or suggested his areas of research. Mr. Marios Sioutis, Ph.D Takashi Okada and Mr. Yoshiki Makino inspired the author through his constant activities on information theory, and gave the author valuable suggestions and kind encouragements.

The author's grateful thanks are also extended to Professor Kiho Hyun of Youngsan university for his valuable suggestions and to Professor Jungyeop Kim of Youngsan university for his words of encouragements.

Finally, the author wishes to thank his families for their support and encouragement. He wishes to thank his wife Jinju Shin for her constant encouragement. He also is grateful to Dayun Kim for her lovely smiling which makes him to stand up again. He would like to express his very great appreciation to Geunseok Kim and Geumja Jo for their warm care and prayer. He would like to offer his special thanks to his another parents Jeonga Shin and Jeonghee Jang for their trust for his success.

Contents

List of Figures

Chapter 1

Introduction

1.1 Services in networked home environment

For more than a decade, ubiquitous/pervasive technologies allow general household appliances to be connected within the network at home[1]. Since a number of researcher have tried to provide these environment as their own field such as Universal Plug and Play (UPnP) [3][2], Digital Living Network Alliance (DLNA) [4], Open Services Gateway Initiative (OSGi) [5] and Energy Conservation and Home care Network (ECHONET) [6], the home network system (hereby referred as HNS) is comprised of such networked appliances to provide various services and applications for home users. Therefore, HNS has been realized with an integration of features of multiple appliances. Especially, providing integrated services to end-user with briefly establishment of the appliances has been considered for actualizing ubiquitous home network environment.

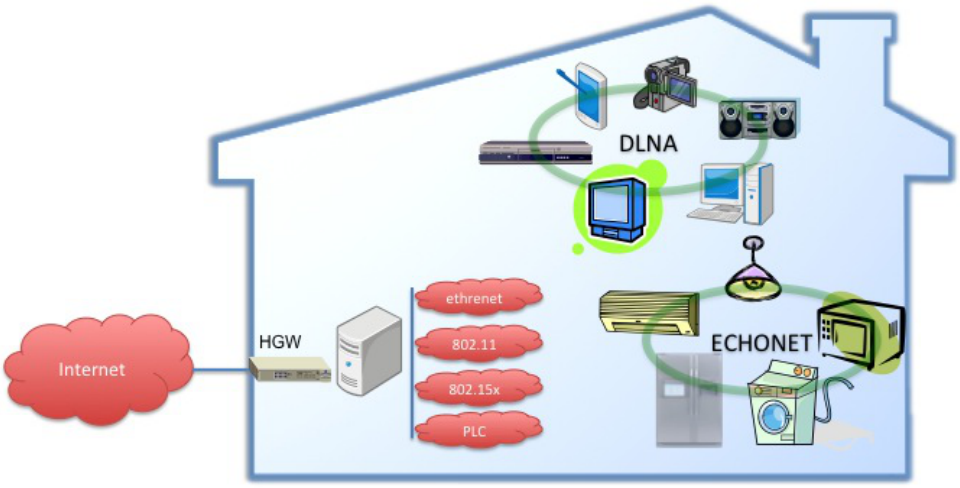


Figure 1.1: Networked home environment

To provide an integrated service, there are several challenges: 1) an agent which is a centralized management system, need semantical understanding of the service objective for composing these services automatically, 2) the composed service should be able to flexibly change it's elements according to service priority and 3) since the Service Intermediary (SI) model was conceived (see Section 2.3 for more discussion) [19], a service

developing environment needs to be distributed. These challenges will be explained in detail in Section 2.3.

This research is focusing on a dynamic service composition environment in the HNS.

1.2 Contributions

1.2.1 Semantic service composition and distributed developing environment

An appliance can provide multifunctional service according to an objective of a service. However, to provide services automatically, the agent needs to understand the meaning of these services, since the home appliance may have different meaning depending on service objective. For example, the curtain could be included in illumination service, temperature control service and privacy protection service.

To solve the above challenge, we need a knowledge based machine-readable service representation because the agent needs to understand the semantics of services and the relation between these services. The system makes it possible to increase the selection of an element by semantically representing a relation between services and elements. To define services semantically using ontology modeling, we used a web ontology language (OWL) that is designed for use by applications that need to process the content of information instead of just presenting information to humans.

Since there are many semantic representations according to services, to represent all meaning in a single representation is actually impossible. The name space of ontology solves such problems by combining the different ontology model. The system is able to include an external ontology model by importing its name space. Such distributed development of a service model will improve scalability of the service model. In the near future, a distributed environment for developing home services is going to be needed by service provider. We expect that our system can provide a platform to deploy a service that was made by an external service provider into the home network environment smoothly.

1.2.2 Policy based priority of a service element

Another important issue on the service composition is that the priority of a service changes depending on the policy. Providing service considering the policy influences the composition of service since the elements of a composed service are dynamically changed by priority of the policy. As we consider air conditioning service with energy consumption, an electric fan or curtain consumes less energy than an air conditioner. However, considering the cooling capability, the air conditioner may have higher priority than other elements. Hence, the agent needs to understand various policies as many as possible and decide a service composition according to the policy. The system provides priority based service composition because during service composition the system also takes into account a policy ontology model. It is also possible to provide a distributed developing environment for policies using a name space.

1.2.3 Adaptive link topology of an appliance

After composing a service, the system needs to consider how to determine a link topology among networked home devices including legacy appliances for transmitting service content. Since networked home devices are going to be increasingly complicated on a network infrastructure, possible link topology between devices is also increased. Moreover, when a device is occupied by a service, the integrated service manager can provide an alternative link path by a network routing.

In the appliance composition, the system makes three novel contributions: 1) a device model provides a representation of the relation of interfaces such as DLNA, HDMI and S-video on the device, 2) the composition algorithm searches the possible composition on the device model and 3) determining the link topology is done according to the quality and efficiency for providing a service in searched composition list. The system provides a method to select the best composition in various home environments since the system can determine a link topology suitable among networked home devices including legacy appliances. Moreover, the system provides assistance with the set up of new devices, discovering the most suitable network link topology for these devices.

1.3 Structure of the thesis

In chapter 2 we introduce several issues that are the background regarding home network system and services. We will proceed to describe ,the semantic web services technology, a priority resolution of service elements and the service intermediary model.

In chapter 3 since the goal of this thesis is to provide integrated service on HNS, we introduce several challenges for dynamic service composition and give examples of a service scenario on HNS. Subsequently, we identify and briefly overview our research target, a semantical understanding of the service objective, a dynamic service priority, a distributed service developing environment and an adaptive link topology of an appliance. Finally, we describe our proposed system architecture briefly.

In chapter 4 we discuss a service composition for integrated home service, and we clearly identify the problems of current service composition technique. we will proceed to propose our system for solve these problem. Finally we present simple prototype system as experimental result.

In chapter 5 we discuss a policy based a dynamic service priority. we proceed to describe our proposed model for dynamic service priority. Finally we present simple prototype system as experimental result.

In chapter 6 we introduce an adaptive link topology of an appliance. we will proceed to propose our system and give an experimental result.

Chapter 7 illustrates implementation of our research, then gives experimental result. Moreover, an evaluation is conducted with the result.

Finally, in chapter 8 and 9 we discuss our conclusions and contributions and point out futurework.

Chapter 2

Related work

2.1 Semantic service representation techniques

In this section we provide a brief introduction to semantic web services and the supporting semantic web and web services technology. We give examples of the use of these technologies in a concrete example scenario showing how they enhance information access and integration.

2.1.1 Semantic web services

Finding the right information on the world wide web, becomes increasingly difficult even with support from search engines such as Google. Current search technology relies on keyword based search. These techniques provide a relatively high recall (as all web sites that mention a given keyword are retrieved) but a low semantic recall (as pages about the desired topic but not containing the keywords are ignored). Their precision is low because only few of the retrieved pages contain the information that the user needs.

As a response to these limitations, the web encountered two revolutionary changes. First, the semantic web community aims at a semantic extension of the current syntactic web by augmenting existing web information with logics based formal descriptions of their meaning. This semantic markup would be machine processable and therefore allow easier access and integration of the vast amount of the available information than what can be achieved with keyword based search. Second, the web is changing from a collection of static web pages to a dynamic environment with the advent of the web services technology that makes software components accessible via web protocols. The semantic web services technology developed at the cross road of these two technologies by applying semantic web techniques to web services.

2.1.2 Ontology engineering

The term ontology, originating from philosophy as detailed in [9] and [10], was adopted by AI researchers to describe formal domain models. Several ontology definitions were provided in the last decades. The most frequently cited definition is that given by Gruber in 1993 according to which an ontology is“ an explicit specification of a conceptualization ”. In other words, an ontology is a domain model (conceptualization) which is explicitly described (specified).

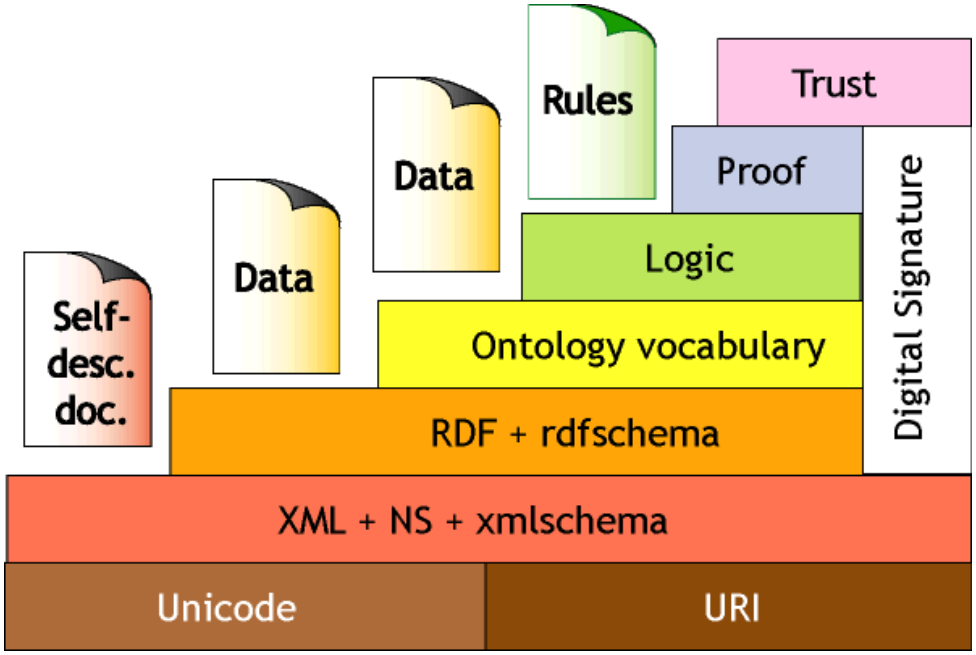


Figure 2.1: Semantic web leyaer

Later, in 1997 Borst defines an ontology as a " formal specification of a shared conceptualization "[11]. This definition requires, in addition to Gruber 's definition, that the conceptualization should express a shared view between several parties, a consensus rather than an individual view. Futuremore, this conceptualization should be expressed in a machine readable format (formal). In 1998, Studer et al. [12] merge these two definitions stating that: " An ontology is a formal, explicit specification of a shared conceptualization. "As consensual domain models, the primary role of ontologies is to enhance communication between humans (e.g., establishing a shared vocabulary, explaining the meaning of the shared terms to reach consensus). As formal models, ontologies represent knowledge in a computer processable format thus enhancing communication between humans and computer programs or two computer programs. Therefore, ontologies are investigated by several research fields in the context of diverse application areas. Already in 1998, Guarino offers an extensive list of references to research fields that have recognized by that time the importance of ontologies [13], ranging from knowledge engineering to information retrieval and integration. Four years later, McGuinness reports on the use of ontologies in several Web related tasks such as Web site organization, navigational support, browsing and searching [14].

2.1.3 Ontology languages for the semantic web

Ontologies are explicitly specified in a formal language. The work performed in this thesis relied on the RDF(S) and OWL languages.

- **RDF(S):** RDF and RDF Schema represented the first step towards a web based ontology language. RDF is a data model allowing to describe resources on the web. RDF Schema is based on RDF and allows the definition of basic ontology elements such as classes and their hierarchy, properties with their domain, range and

hierarchy [15]. As such RDF(S) is well suited for expressing lightweight ontologies. Several tools were developed for RDF(S) and a considerable number of semantic web applications used this language before a more expressive ontology language was developed.

- **OWL:** The Web Ontology Language(OWL) is a more expressive ontology language than RDF(S). The basis for developing OWL was the DAML+OIL language which originated by merging two language proposals that aimed at overcoming the expressivity limitations of RDF(S): DAML-ONT and OIL [16]. OWL enhances the expressivity of RDF(S) providing means to describe relations between classes (e.g., disjointness, union, intersection), cardinality and value restrictions on properties (e.g., cardinality, universal and existential quantifiers), property characteristics (e.g., transitivity, symmetry), equality etc.. OWL provides the constructs to encode ontological knowledge (such as in the previous subsection) using the XML based syntax.

2.2 Priority resolution of service elements

2.2.1 The Feature Interaction in the HNS

A major HNS application is the integrated service of networked home appliances (we simply call HNS integrated service in the following). The HNS integrated service orchestrates different home appliances to provide more comfortable and convenient living for users, which is considered one of the next-generation value-added services in the ubiquitous computing environment (see Section 4.1 for more discussion). Typical HNS integrated services include:

- **DVD Theater Service:** When a user switches on a DVD player, a TV is turned on in DVD mode, a blind is closed, the brightness of the lights is minimized, 5.1ch speakers are selected, and the sound volume of the speaker is automatically adjusted.

- **Coming Home Light Service:** When a door sensor notices that the user comes home, lights are automatically turned on. Then, the brightness of the lights are adjusted to an optimal value based on the current degree obtained from an illuminometer.

Feature interactions (from now on referenced as FI) may occur in HNS integrated services as well, since multiple services may be activated simultaneously. For example, the above two integrated services interact with each other.

- **Interactions between DVD Theater & Coming Home Light:** Suppose that a user A activates the DVD Theater service, and simultaneously that a user B comes home. Then, the following two interactions occur:

- **FI-(a):** Although the DVD Theater service minimizes a brightness of the lights, the Coming Home Light service sets the brightness comfortable for B. This may ruin A 's desire to watch the DVD in a comfortable atmosphere.

- **FI-(b):** If the blind is closed (by the DVD Theater) immediately after the lights read the degree from the illuminometer (by the Coming Home Light), the lights may fail to set the optimal illumination because the blind makes the room darker.

The feature interaction problem in the HNS integrated services was first addressed by Kolberg et al. [17]. These authors regard each HNS component (an appliance or an environmental variable) as a resource. In their model, each integrated service accesses some resources in a shared or not-shared mode. An interaction is detected when different services try to access a common resource with an incompatible access mode. Thus, each appliance is simply modeled by the two-valued access attributes, and each integrated service is characterized only by how the service sets values of the attributes. This simple modeling enables a light weight and realistic implementation framework for feature interaction avoidance

2.2.2 FI resolution techniques

Stemming heavily from traditional operating system concepts, in [18], six techniques of conflict resolution between services are proposed. We will briefly describe each one of them.

- **Inquiry to the User:** When a conflict occurs, an inquiry is made to the user regarding the method of the resolution. The following are the possible methods which the user can choose from: 1) execute one service and terminate the other or 2) if possible, reset and change the parameters of the conflicting services so that FI does not occur.

- **Priority of Service:** According to a predetermined service priority, if FI occurs, only the service with the highest priority is executed. This method is considered to be more effective in cases were multiple services exist, and possibly many users. To implement such a resolution scheme, a database for managing the order of priority of services is considered to be necessary.

- **Priority of User:** When FI occurs, only the service activated by the user with the highest priority is executed based on the predetermined order of priority among users. This method is considered to be more effective in scenarios were multiple users exist.

- **Priority of Interface:** When FI occurs, only the service activated from the interface of highest priority is activated. A predetermined priority order of interfaces is assumed. This resolution scheme works best when there are multiple possible user interfaces for a given service.

- **Priority of Timing:** There are two methods in determining the service that is to be given priority based on the order of service execution, namely, to give priority to the service that is executed first or to the one that is executed last. This resolution method can be used in any classification of FI. In order to implement this method of resolution, a database for managing the order of priority of timing is required.

- **Meta Priority:** Meta priority is the prioritization of all the above-mentioned priority levels. When a conflict occurs, it is possible to utilize more than one type of priority level in resolving the conflict. At that point, it is also likely that interactions could occur between different types of priority levels. In order to avoid such situation, it is necessary to prioritize the different types of priority levels.

2.3 The Service Intermediary model

2.3.1 SI model structure

After realizing the limitations of the traditional model to provide services in the HNS, the Service Intermediary (SI) model was conceived. The core idea of this model is presented in [19]. The SI platform is designed to solve the problems of the traditional service model. In the SI model, there are three main entities.

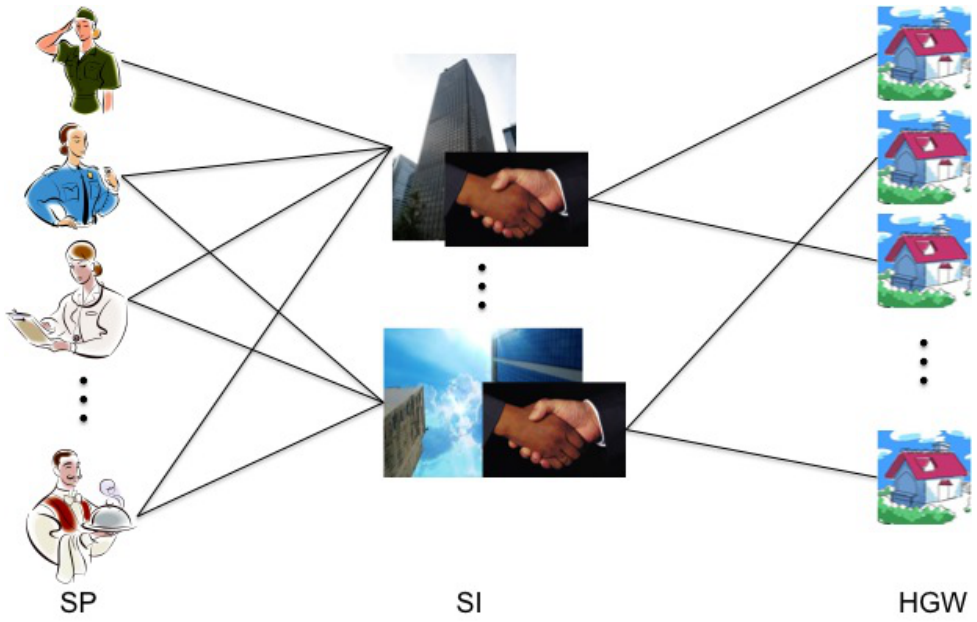


Figure 2.2: The SI service provision platform

- The back-end Service Provider (SP). This is the actual provider of a specific service. The SP designs services that will be provided to the HNS, the final deployment target. This role is the only one that remains unchanged compared to the traditional model.
- The Home Gateway (HGW). The HGW acts as the gateway of the HNS to the outside world. Services can be executed directly on the HGW hardware. Moreover, the HGW acts as a software bridge, capable of controlling or passing command to other devices in the HNS.
- The Service Intermediary (SI) ,from which this model for service providing takes its name. The SI acts as an intermediary between the SP and the HGW. In this model, the HGW does not contact the SP to have access to a service. Instead, the SI aggregates services from various SPs and presents them to the user. The user makes a contract with the SI to have access to the services of the various SPs.

A representation of the relationships between the three parties can be seen in figure 2.2. Usually, SIs aggregate services from the SPs and then a house can enter a contract with an SI.

2.3.2 Advantages of the SI model

The presence of the SI is the major difference between the SI model and the traditional model of services. By having a service intermediary between the various service providers and the HNS, the problem of a fragmented deployment environment is resolved. This is because, the SI can enforce a specification of the hardware platform on which the services will be executed. The SI will be responsible to verify that all services deployed on the controlled platform do not misbehave and are consistent.

The existence of an SI saves the user of the hassle of having to make contracts with a substantial number of different SPs. Moreover, the problem of the configuration of a multitude of different devices is resolved, since the SI can take care of it. The configuration of the services will also be taken care by the SI, and we expect that the SI will be able to offer a comprehensive and consistent user interface, not possible otherwise. Furthermore, the duplication of hardware is minimized and the same hardware may now be used by many different services, increasing its value. Moreover, the SI will be able to provide the user with all the equipment that is necessary (depending on the services the user is subscribing to), or designate compatible equipment that can be used effectively as part of the platform. Finally, according to [20], it is possible to have computationally intensive services execute on computational nodes of the SI transparently, as in the case of cloud computing. The benefits for the service providers are also quite substantial. Firstly, the SI is now able to provide the SP with a set of application programming interfaces (APIs) for the design of services. By adhering to the APIs, now the SP does not have to worry about inconsistencies and differences in the deployment HNS. The SP now must target the service platform provided by the SI. It is the responsibility of the SI to guarantee the provision of a consistent service platform on which services can be deployed. Thus, the SP is freed from the mundane task of having to test the service with possibly hundreds of different configuration environments, simplifying the design of services targeted for the HNS.

Chapter 3

An overview of the research

3.1 Goal of this research

In this section, we identify and briefly overview our research. Because the goal of this thesis is to provide integrated service on HNS, we introduce several challenges for dynamic service composition and give examples of a service scenario on HNS.

3.1.1 Semantical understanding of the service objective

The characteristic of the HNS service is that home devices could be multifunctional features. For example, a curtain has three functionality. Firstly, it is usually worked for excluding the outside light of the sun. As same time, the curtain conceals a home life of an user from outside. Finally, it is used to control a temperature of an indoor. If these devices are categorized for the integrated home network service, such ambiguous functionality should be identified exactly according to the service objective.

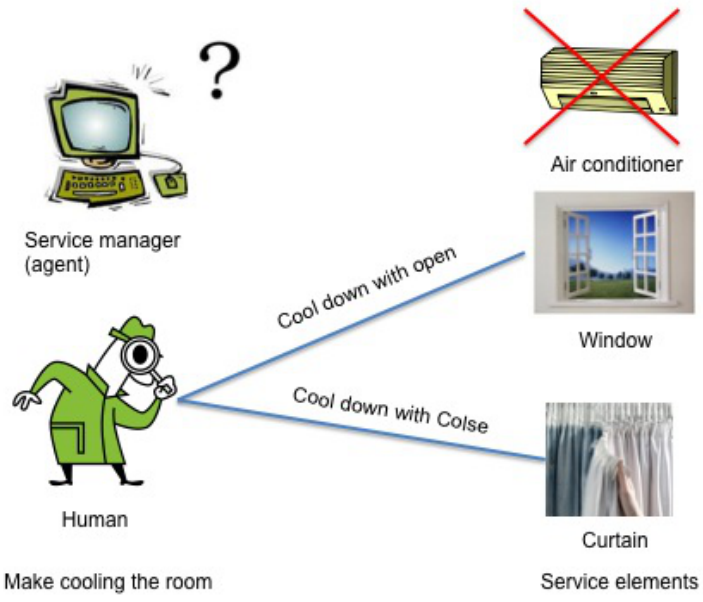


Figure 3.1: Human vs Service manager (agent)

Another characteristic of the HNS service is that a threshold of a service quality is flexible. We assume that a temperature control service is executed on HNS as Figure 3.1. Then the service manager tries to occupy an air conditioner. However, the air conditioner is unusable with a breaking down or occupied by other service. In this case, the service manager can not find any device for the temperature control service. But a human may controls the temperature by opening (or closing) a window or the curtain because these features can getting close to a target temperature even if it is not enough.

To provide the integrated service, a service manager (agent) needs to semantically understand a service objective. In other words, usable services that are contained on each appliance have to be defined on a semantics which is a machine-readable.

3.1.2 Dynamic service priority

To provide an integrated service in home network, various policies have exert an influence on a service priority. We assume that there are many devices that provide a service on the same objective, these devices are different each other on a capability, an efficiency or a responsibility. In this case, a service priority is changed according to what a policy needs to precede. For example, as mentioned previous section, the air conditioner and the curtain could provide same service with cooling functionality. However, in considering for energy saving policy, the curtain is more efficient than the air conditioner. The service manager need to determine a priority between the curtain and the air conditioner for the service composing. We illustrated an example of a priority based on policies as Figure 5.1. Each device has capability on considering several policies. When service manager determine a sound device for the service, priority is considered according to policy.

Policy	5.1ch speaker	Table speaker	Headphone
Sound quality	High	Middle	Low
Privacy	Low	Middle	High
Mobility	Low	Low	High

Figure 3.2: A priority based on policies

For this issue, when the policy were changed in same service objective, an element which is used for service composing is also changed. Therefor, the service manager have to understand these policies and determine the priority of services base on the policy. In this research, we defined these policies semantically and provided a service composing with a suitable element according to the policy.

3.1.3 Distributed service developing environment

We discussed about SI model in section 2.3. In SI model structure, important duty of the service intermediary is to manage and integrate the services from the exterior (service provider). Currently, Since a service manager (or agent) is not only simply provide a service to a user but also make a new service by integrating these services, the services from each exterior need a combinability. For example as Figure 3.3, we assume that the SP A provides temperature control service and the SP B provides an illumination control service. When SP C wants to make a service which provides a home theater service, the SP C could include these two services that is from SP A and SP B.

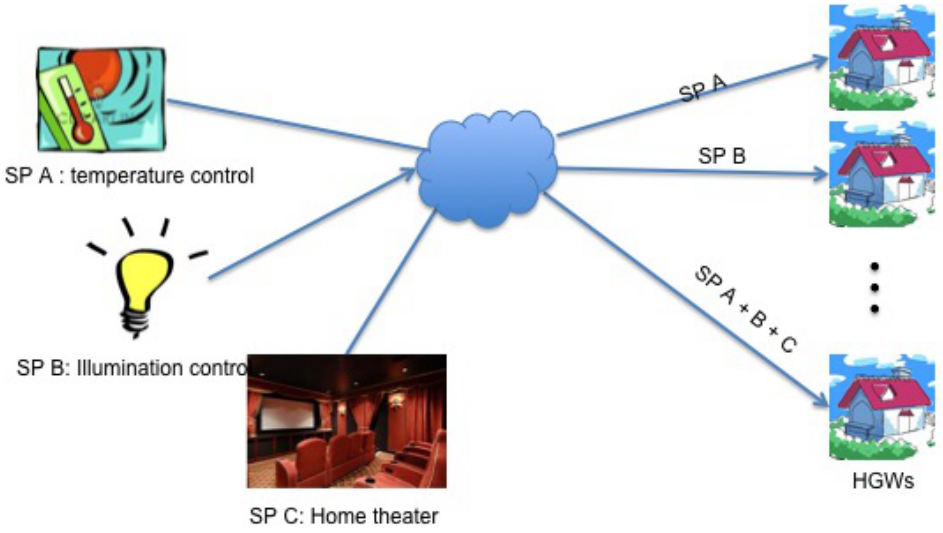


Figure 3.3: Example of distributed service developing environment

These service combinable structure reaches that the SP may develop they own service efficiently by using an existing services that are already developed from other SPs. Moreover, the service manager can provide an integrated service to user by composing services that is developed separately from each SP.

To build the distributed developing environment, the service manager needs to consider several challenges as follow items.

- **Service Identify -** Each services are clearly identified who made it and what the service do.
- **Combinable Structure -** Services need to be combined each other on the semantics

In mention the Service identify, the service manager has to recognize "what the service do". For example When the service manager searching a service element for composing a home theater service, the SI should semantically know which service could be the AV controller, the temperature controller and the illuminate controller. Moreover, these services should be identified "who made the service". There are many services that are provided as same functionality. the SI needs to recognize where such services are made from.

Moreover, since the service manager needs to map every services for understanding a relation between each service and providing integrated service to the user, services from the SP should be combinable with other services by the service manager.

3.1.4 Adaptive link topology of an appliance

Generally, an infrastructure of appliances in the home network environment is composed by diversification of an agreement from different kinds of industry. For example, there exist Audio and Video electronics, not only a brand new DVD player which can provide a functionality of DLNA and capability of HDMI, but also the past generation of TV that only have a terminal of S-video. Moreover, these devices are complexly connected each other.

In above environment, after the service manager determined a service, a devices (or appliances) that are used by the service need to be linked from a transmit device to a receive device. Especially, a home devices frequently occupied by other service. In this case, the service manager should find the other link topology except the occupied device.

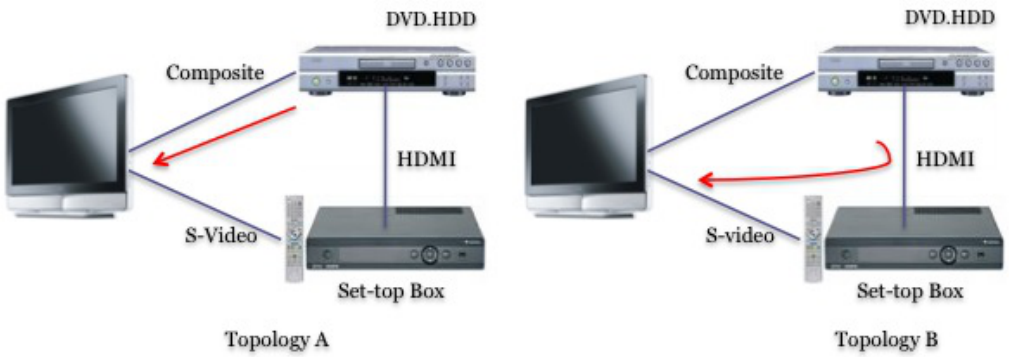


Figure 3.4: Example of the link topology of appliances

To look up the link topology between devices, the service manager has to consider several challenges as follow.

Flexible link topology

Home network environment has been more complex and an infrastructure of the home appliance is composited by various industry. the service manager need to look up the link topology among the appliances when service is provided.

Quality and Efficiency

The service manager has to consider capability of the appliance and efficiency of linked topology. In Figure 3.4, the topology B has high quality as a resolution then the topology A but, topology A is more efficiency then the topology B because A used one more device then B.

Observation of a state of device

The devices have to be watched on the state carefully since they could provide other service on a concurrent operations. For example, if the home theater service already occupying a TV, the other service which wishes to use the TV has to find another appliance as a substitute.

3.2 Proposed system architecture

In section 3.1, we discussed about the goal of this research with several challenges for the integrated service environment. This research fundamentally considers the following components.

- The semantic service representation
- The policy based dynamic service priority
- The distributed developing environment
- The Adaptive link topology of an appliance

In this section we explain an outline of the proposed system architecture for a building the integrated service environment. We assume that appliances in the home network environment provide their informations by own APIs. Moreover, these informations are contained devices's status, service description and commend description as XML (Extended Markup Language)/RDF (Resource Description Framework). When the service manager controls appliances, such informations are used.

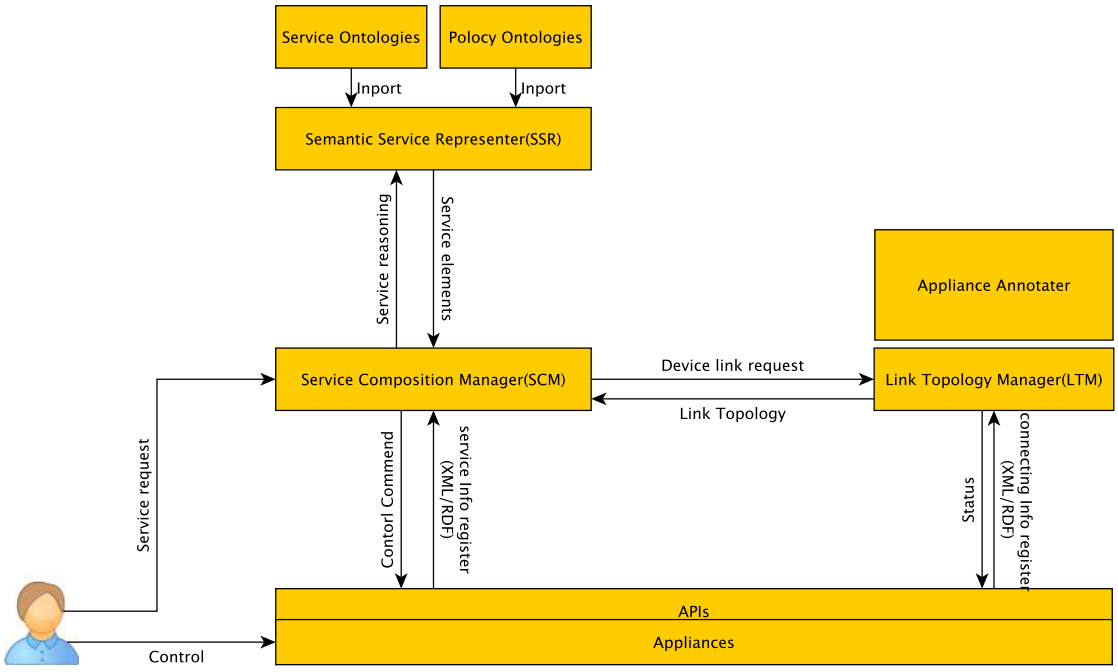


Figure 3.5: The system architecture

As we can see Figure 3.5, the system is divided into three parts, Service Composition Manager, Semantic Service Representer and Link Topology Manger. the following subsection will explain the role of each component on brief.

3.2.1 Service Composition Manager

The Service Composition Manger (hereby referred as SCM) basically registers the service information from appliances. When a user requests a service, the SCM composes registered services according to a service elements that are the result from Semantic Service Representer (hereby referred as SSR). For example, When user want to watching a movie, the SCM requests what elements are needed for the movie to the SSR, the SSR giving back a TV, DVD, speaker as elements of a theater service. If these service elements are needed to connect between elements, the SCM requests a link topology to the Link Topology Manager (hereby referred as LTM).

Since already occupied element is also exist in the SCM, an information of services from an appliance has to be noted whether registered element is usable or not. If the selected service element is unable to be used, the SCM reasons other service element which is a substitute to compose services except the occupied service element.

3.2.2 Semantic Service Representer

As mentioned in section 3.1.1, a service manager (agent) needs to semantically understand a service objective. The SSR provides the SCM to compose a service on the semantics. The SSR provides representation of a relation between services using the ontology. The represented services are used for determining suitable service element according to the service objective.

An external ontology model might be included into the SSR using mapping. Moreover a mapped ontology models are classified each other using their own namespace even if they has same functionality on a semantics. For example, there are two ontology models that are both a service to provide a temperature control but the SSR recognizes that both ontology models are different as a service.

In the SSR component, two domain ontologies are considered as follow.

- **The service ontology -** "For understanding service objective"
- **The policy ontology -** "For determining service priority"

The service ontology represents a relation between services according to the service objective. When the SCM requests a service composition with the service objective, then the SSR semantically searches service elements. For example, the SCM provides a temperature control service, then the SSR will find elements such as an Air conditioner, a window and a curtain because the SSR recognizes that not only the Air conditioner but also the window and curtain could be an element of temperature control service.

The policy ontology is for determining of service priority. As mentioned in section 3.1.2, a service priority is changed according to what a policy needs to precede. Therefor, the SSR contains many kind of policies such as the QoS policy, efficiency policy and economy policy. Moreover, these policies are based to determine a priority of a service elements.

3.2.3 Link Topology Manager

The LTM provides a adaptive link topology between appliances. After the SCM determines a service composition, It's needs to decide a path from a transmit device to a receive

device. Since a device can connects to a target device as many links, the LTM considers these links according to a resolution quality and device efficiency.

The LTM consider the following three components.

- **Possibility of the composition -** Service manager has to define all of the possibility of composition on existing appliance in the home.
- **Capability of service -** The composition has to be considered the QoS of an appliances which is composited by service manager.
- **Observation of a state of device -** The devices have to be watched on the state carefully since their could provide other service on a concurrent operations.

If a device which is needed for current service is occupied by another service, the LTM provides a possible link to target device. Moreover, the LTM provides suitable link topology to user, when new device set up into the home.

Chapter 4

Semantic service composition and distributed developing environment

4.1 Service composition for integrated home service

4.1.1 The integrated home network service

The HNS provides many applications and services. The applications typically take advantage of wide-range control and monitoring of appliances inside and outside the home. Moreover, integrating different appliances via network yields more value-added and powerful services [21], which we call HNS integrated services.

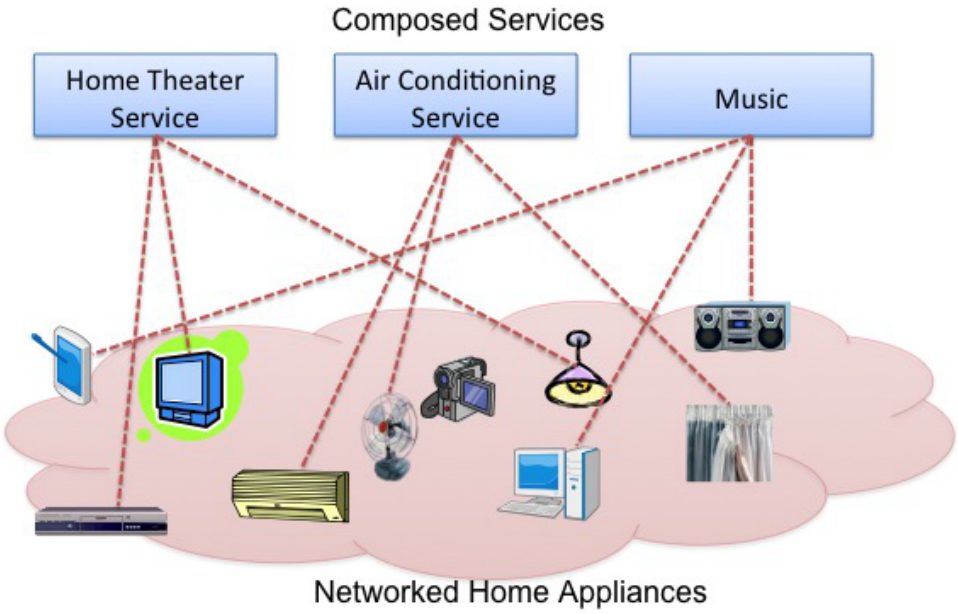


Figure 4.1: Integrated service composition

For instance, integrating a TV, a DVD player, speakers, lights and a curtain would implement a HNS integrated service, say, DVD theater service. When a user requests the service, the lights become dark, the curtain is closed, the 5.1ch speakers are selected, the

sound volume is adjusted, and the contents are played with the DVD player. Thus, the user can watch movies in a theater-like atmosphere within a single operation.

In general, each networked appliance is equipped with smart embedded devices, including a network interface, a processor and storage, in order to provide and execute the appliance features required for various HNS applications and services. As the embedded devices become more down-sized, cheaper, and more energy-saving, it is expected in the near future that every object will be networked [22].

4.1.2 The service composition

Recently, functionality of an appliance has been composed by one or more process. Moreover, these processes are possible to used as one by one for other service by defining an interface of each process as showing Figure 4.2. It means that an agent, which is centralized management system, can use these independent processes to compose (or decompose) virtual service. For example, in the figure, there are two appliances, appliance A and B. These appliances are composed by several processes. So, we can imagine that processes can be composed for other objective such as the home theater service Even appliance A already perform with process No 1, process No2 can be used composing unless it 's independent with process No1.

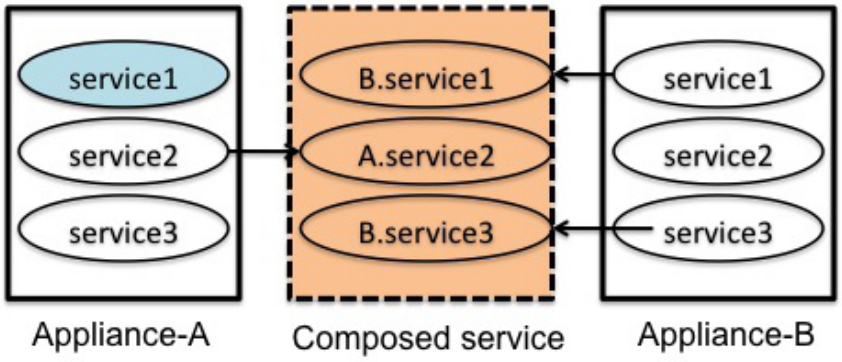


Figure 4.2: Example of the Service Composition

As we mentioned above, the appliance can provide multifunctional service according to an objective of a service. However, to provide service automatically, the agent need to understand a meaning of these service, since the home appliance may have different meaning depending on service objective. For example, the curtain could be included in illumination service, temperature control service and privacy protect service.

To solve above challenge, we need a knowledge based machine-readable service representation because the agent need to understand a semantics of services and a relation between services. Also, we need to consider a decentralized development setting for defining many of a service model.

4.2 Dynamic service composition

A service (or application) composition method for integrated service in HNS can be divided into two approaches: Proactive (Static) service composition, and Reactive (Dy-

namic) service composition[23].

In the web service, the static service composition is an approach in which application designers implement a new application manually by designing a workflow or a state chart describing the interaction pattern among components. BPEL4WS [24] or WSCI [25], for example, are primarily designed for supporting this approach. The static service composition supports applications involving complex interaction patterns, such as branch or iteration, but requires those applications to be manually designed before being deployed. Therefore, the static service composition is suitable for B2B (Business to Business) type applications where interactions among components are often complex but static and easy to provision.

Dynamic service composition, on the other hand, composes an application autonomously when a user queries for an application. eFlow [26] and SWORD [27] are the examples of dynamic service composition systems.

In this section, we discuss about a dynamic service composition.

4.2.1 Template based service composition

For example, ICARIS [28] and eFlow [26] proposed a template based dynamic service composition system that supporting applications involving complex interaction patterns such as conditional branch or iteration which is problem with static service composition.

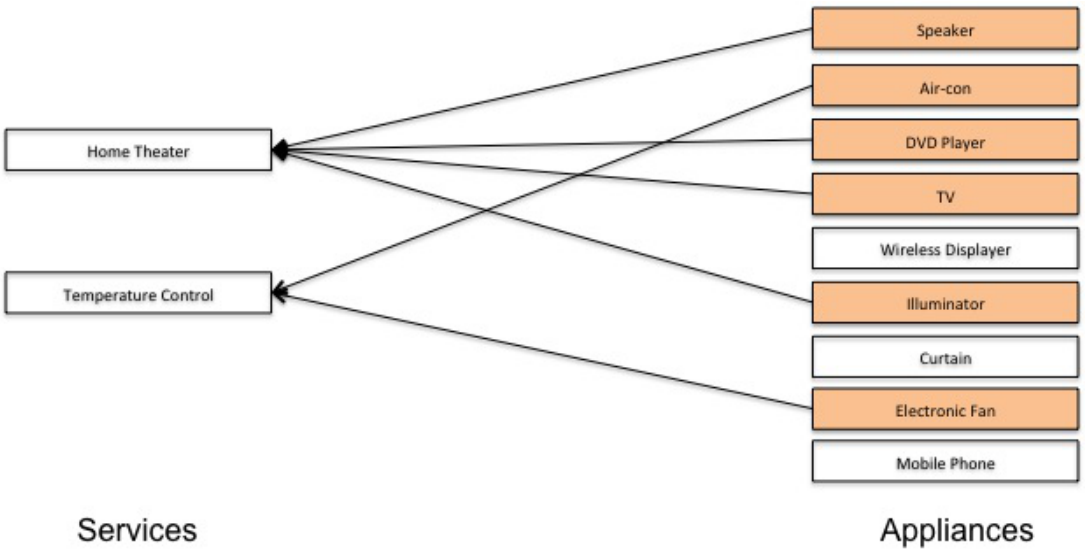


Figure 4.3: Example of a template based service composition

Figure 4.3 shows an example of a template based service composition, the left side illustrates a composed service, Right one is service elements. For home theater service, elements of service are fixed with speaker, DVD player, TV, and illuminator. When a user wants to change an element of composed service with policy like privacy and efficiency, the user has to re-range elements by himself. Of course, a centralized computer can ’t understand the relation between service and elements. For instance, if the Home Theater service can not use the illuminator which is for brightness adjustment of the room because it’s breakdown, the service will be stop or keep going without the illuminator.

4.2.2 Semantic based service composition

Since the dynamic service composition does not depend on a human to compose a service, it may have difficulty in composing services. To compose a service dynamically without a manipulation of the human, an agent has to understand a meaning of an objective of the service that the agent would provide. In addition, the agent need to recognize what possible service does it has semantically for composing the service.

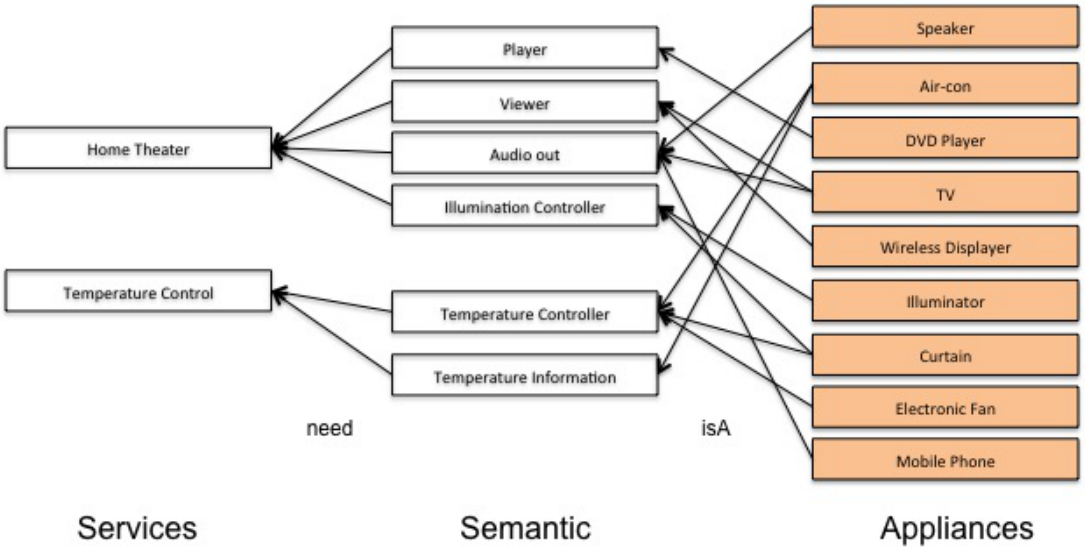


Figure 4.4: Example of a semantic based service composition

To solve these problems, Ninja [29] and SeGSeC [30] created an execution path from a user request by performing interface matching based on semantics. These solutions are considered how to matching the interface between services. However, fundamentally, the dynamic service composition in the home network system is needed to perform flexibly depend on home environment.

These dynamic service compositions using the semantics are need to resolve number of general issues that the ontology engineering has, suitable definition every notions, representation of extremely diverse relationship between components and optimization of a reasoning engine. Nevertheless, the dynamic service composition has the potential to provide flexible and adaptable services by properly selecting and combining components based on the user request and context.

We used semantic representation of services. This solution makes it possible to increase a selection of an element by a semantically representing a relation between services and elements.

As mention in the Figure 4.4, the theater service needs the viewer, and the viewer could be a TV and wireless displayer. Therefor, we assume that possible element for the viewer is increased compared to the template based service composition.

4.2.3 Advantage of the semantic based dynamic service composition

In previous section, we discussed the template based service composition and the semantic based service composition. In this section, we explain an advantage of the semantic based dynamic service composition by giving specific examples.

	Semantic based (proposed)	**Template based**
Extensionality of a service	Including a service by a namespace	Need to replace all
Flexibility on a home environment	Variable service composition according to the environment	Limited service composition capability
Selection of service element	Extensive	Fixed
Cope with a trouble	Intelligently suggests a substitution of a service element	has limited service element substitution capability
Priority of a service element	Variable Priority based on a policy	Fixed

Figure 4.5: Template based composition VS Semantic based composition

The home network service is considered on several challenges. The semantic based service composition shows a robustness with these challenges as follow.

Extensionality of a service

As mentioned in secetion 3.1.3, the home network services are provided from various external service provider. Moreover, these services are frequently updated by the provider. Especially, when a new service is provided into the home, the new service has to be included on already existing service. In case of the template based service composition, all existing services need to be replaced by a provider for including the new service. However, the semantic based service composition includes the new service by using a namespace of a service definition.

A namespace is an abstract container or environment created to hold a logical grouping of unique identifiers or symbols (i.e., names). An identifier defined in a namespace is associated only with that namespace. The same identifier can be independently defined in multiple namespaces. That is, the meaning associated with an identifier defined in one namespace may or may not have the same meaning as the same identifier defined in another namespace. Languages that support namespaces specify the rules that determine to which namespace an identifier belongs

Flexibility on a home environment

Home network environment has been more complex and an infrastructure of the home appliance is composited by various industry. Therefor, an integrated service needs adaptability for various home environment.

The semantic based service composition has variable service composition according to the environment. For example, there are two different home environment, one of a home has a TV as display and another home has a smartphone only. In this case, the semantic service still be executed using both two devices on same objective.

Selection of service element

A semantic service definition makes the service manager to searches more various composing elements since service elements are defined according to varied service objective. As we already gave example, a meaning of the curtain could be changed according to a service objective, illumination service, temperature control service and privacy protect service.

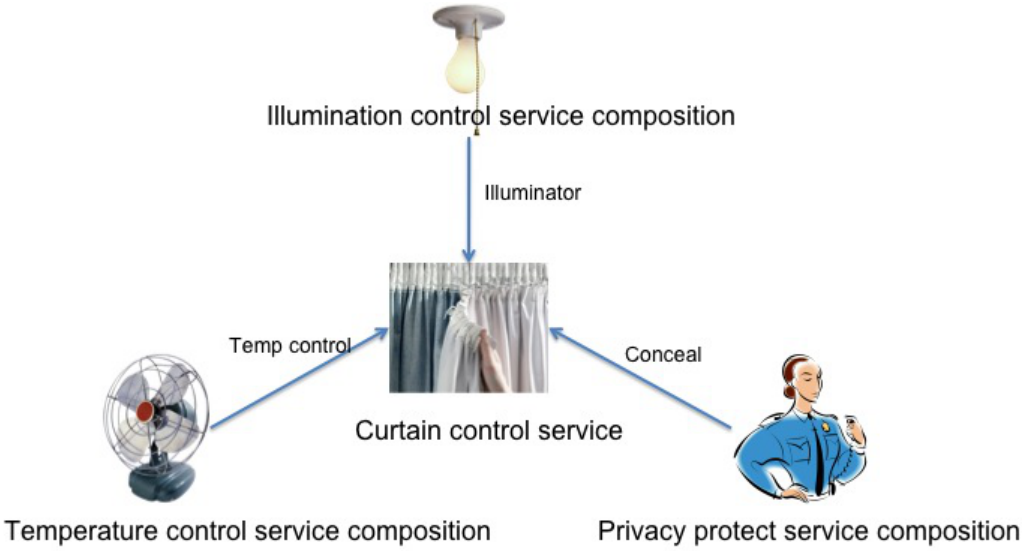


Figure 4.6: A semantic definition of services

Cope with a trouble

An extensive environment of a service element makes the service manager to intelligently suggests a substitution of an element when a selected service element is unable to use cause of a trouble. The template based composition also provides a substitution. However it's extremely limited by a categorization.

Priority of a service element

Providing service considering the policy influences the composition of service since the elements of a composed service are dynamically changed by priority of the policy. The se-

mantic service composition provides dynamic priority according to the policy by including a policy ontology model.

4.3 Distributed service developing environment

4.3.1 Distributed service design

Since we can imagine that there are many semantic representations according to services and policies, to represent all meaning in a single representation is actually impossible. The ontology 's name space solves such problem by matching each other. As you can see the Figure 4.7, The service composition agent can include an external model by importing its name space.

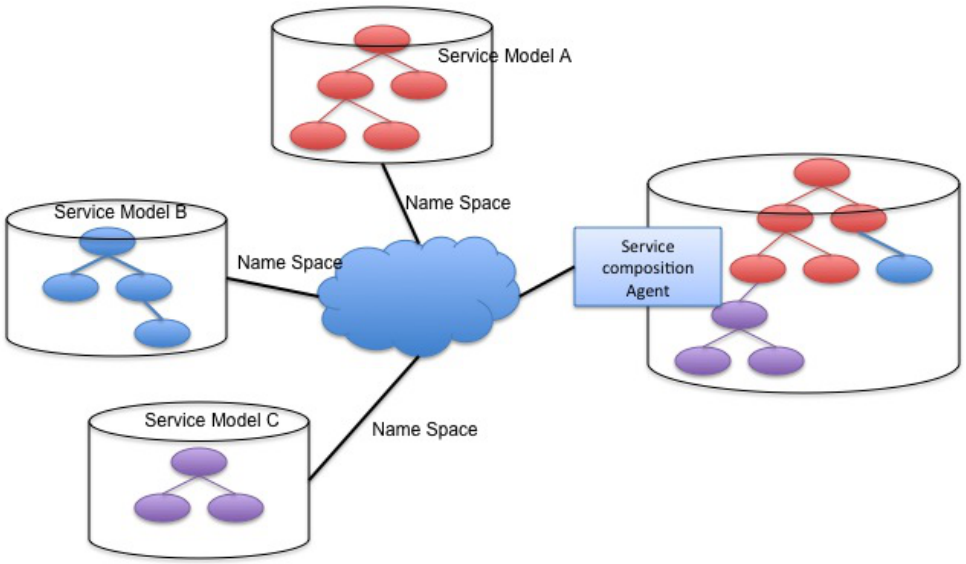


Figure 4.7: A distributed service composing environment

4.3.2 An ontology matching technique

Ontology matching, or ontology alignment, is the process of determining correspondences between concepts. A set of correspondences is also called an alignment. The phrase takes on a slightly different meaning, in computer science, cognitive science or philosophy.

The problem of Ontology Alignment has been tackled recently by trying to compute matching first and mapping (based on the matching) in an automatic fashion. Systems like DSSim, X-SOM[32] or COMA++ obtained at the moment very high precision and recall [31]. The Ontology Alignment Evaluation Initiative aims to evaluate, compare and improve the different approaches.

More recently, a technique useful to minimize the effort in mapping validation and visualization has been presented which is based on Minimal Mappings. Minimal mappings are high quality mappings such that i) all the other mappings can be computed from them in time linear in the size of the input graphs, and ii) none of them can be dropped without losing property i).

Given two ontologies $i = (C_i, R_i, I_i, A_i)$ and $j = (C_j, R_j, I_j, A_j)$, we can define different type of (inter-ontology) relationships among their terms. Such relationships will be called, all together, alignments and can be categorized among different dimensions:

- **similarity vs logic -** this is the difference between matchings (predicating about the similarity of ontology terms), and mappings (logical axioms, typically expressing logical equivalence or inclusion among ontology terms)
- **atomic vs complex -** whether the alignments we considered are one-to-one, or can involve more terms in a query-like formulation
- **homogeneous vs heterogeneous -** do the alignments predicate on terms of the same type (e.g., classes are related only to classes, individuals to individuals, etc.) or we allow heterogeneity in the relationship?
- **type of alignment -** the semantics associated to an alignment. It can be subsumption, equivalence, disjointness, part-of or any user-specify relationship.

Subsumption, atomic, homogeneous alignments are the building blocks to obtain richer alignments, and have a well defined semantics in every Description Logic. Let's now introduce more formally ontology matching and mapping.

An atomic homogeneous matching is an alignment that carries a similarity degree $s \in [0, 1]$, describing the similarity of two terms of the input ontologies i and j. Matching can be both computed, by means heuristic algorithms, or inferred from other matchings.

Formally we can say that, a matching is a quadruple $m = \langle id, t_i, t_j, s \rangle$, where ti and tj are homogeneous ontology terms, s is the similarity degree of m. A (subsumption, homogeneous, atomic) mapping is defined as a pair $\mu = \langle t_i, t_j \rangle$, where ti and tj are homogeneous ontology terms.

He and Chang [33] propose a generative statistical model for performing schema matching. In contrast, our model is discriminatively trained in a fully supervised setting. Furthermore, changes in the structure of our model are not needed to accommodate additional dependencies, making it easier for users who are not experts in statistics to adapt it to new domains.

GLUE [34] is a system that produces mappings across two taxonomy trees. The system contains features for modeling both concepts and structures, but combines them with post processing rather than modeling the uncertainty jointly. Additionally, GLUE is a less supervised approach in the sense that only part of the model is learned from labeled data. The meta classier is hand-tuned and relaxation labeling, the approach that combines additional structural constraints from neighbors, is an unsupervised method. In contrast our model is entirely supervised and is capable of learning all parameters from labeled mappings (including those tied to structural constraints).

RiMOM [35] is another system that discovers mappings across ontologies. Mapping is performed by finding a configuration that minimizes Bayesian risk. However, weights for various pieces of evidence are hand tuned in advance rather than learned from the data. An extension, iRiMOM[36] learns to select a strategy from multiple sub-strategies, but treats the possibly overlapping sub-strategies as independent of each-other. In contrast, our discriminatively trained method makes it easy to model dependencies between overlapping pieces of evidence without any independence assumptions.

APFEL [37] is a system that learns parameters through user interaction with the alignment process. Although supervised machine learning is explored, the problem of ontology mapping itself is not formalized as a statistical model.

4.4 System model

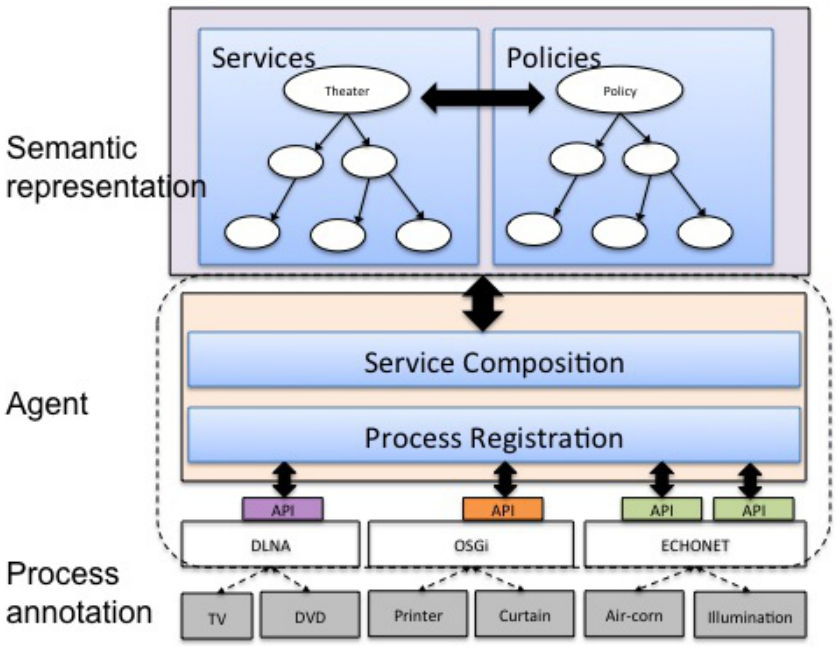


Figure 4.8: System Architecture

We proposed the dynamic service composition using ratiocination on integrated home network system. Our research provides a flexible service composition in various home environments by making the system to semantically understand an objective of a service. Moreover, it is possible to realize extendable services and priorities, since these semantic descriptions can be built separately in their own area by a name space.

Basically, our system considers that providing a flexible service composition in various home environments as follow.

- **Semantical service composition -** understanding an objective of service
- **Distributed developing environment -** efficiency development including an external service model
- **Robust to an environment variation -** when an element in composed service is stopped, substitution could be found

4.4.1 The system architecture

Architecture of our proposed system is shown in Figure 4.8. First of all, the agent registers the information of atomic process that can't be divided any more from appliances to

the process registration. These atomic processes have an information that is a process description and resource type of input/output. Secondly, when the agent gets request with composing service, the service composition decides what elements will be used based on the semantic representation. Finally, the semantic representation defines services semantically using ontology modeling. The service model provides to consider with various policies as importing the policy model. In this section, we discuss specifies of our system. Especially, it'll be focused on representing relations between services and policies.

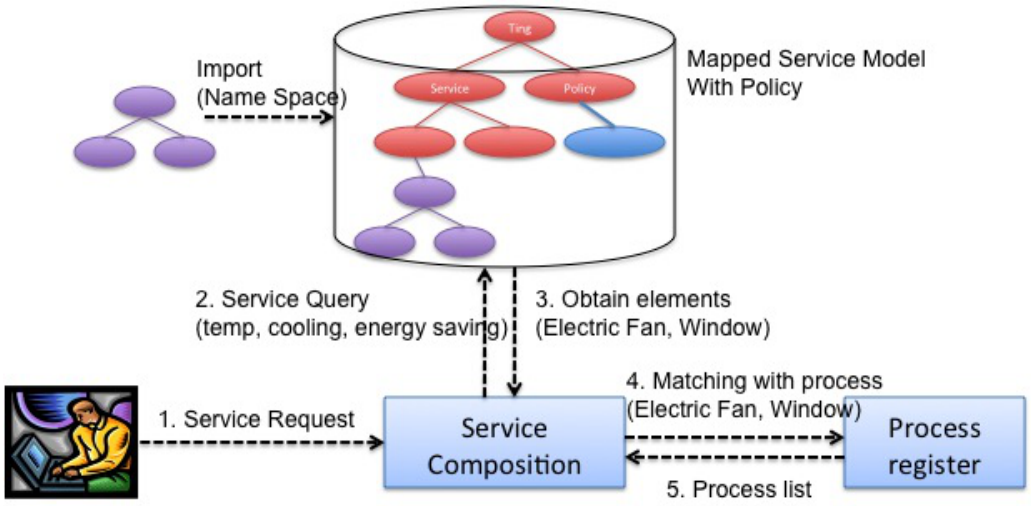


Figure 4.9: System flow of the Service Composition Module

Figure 4.9 is that describes the system flow of the service composition module. The service composition module maps a service model with policy by including external models. When user requests service composition, the service composition module queries with several conditions such as temp, cooling and energy saving. Obtained elements from the mapped service model will be matched with process information from the process register. Finally, process register provides process list to service composition.

4.4.2 Semantic Service Representation

To define services semantically using ontology modeling, we used a web ontology language (OWL) that is designed for use by applications that need to process the content of information instead of just presenting information to humans[7]. We discuss how to build the service model within a temperature control service as a representative service in this section.

A scenario of the temperature control service

In this section, we give a simple scenario of the temperature control service which provides to maintain a temperature of a room as user setting. In the methodology of this service, 1) the service sets a target temperature as an input from user, 2)measures current temperature of the room by using some sensor, 3)if the current temperature is different with the target temperature, it makes the current temperature to reach the target temperature by using some device which is able to heat or cool.

```
Step1: set target temperature (TargetTemp)
Step2: get current temperature (CurTemp)
Step3: If TargetTemp > CurTemp
       Then Cool down (with cooler).
Step4: If TargetTemp < CurTemp
       Then Heat up (with heater)
Step5: If TargetTemp = CurTemp
       Then go to Step2
```

Figure 4.10: A procedure of the temperature control service

When the Service Provider (SP) builds above service, they do not consider "which sensor will be used" and "which device will be used for heat up or cool down", because SP does not know what devices (or precess) are exist each home. The SP only consider a capability of these devices. In other world, the service manager determines "what devices are used for the service" within a considering of a capability.

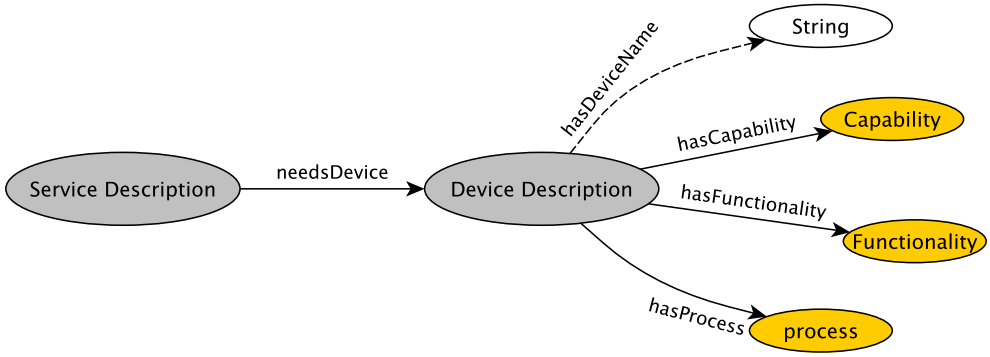


Figure 4.11: A part of a service description

In above service, we define device components as follow.

- Definition of the Temperature Sensor: a device (or process) which measures a temperature
- Definition of the Cooler: a device (or process) which makes to cool down
- Definition of the Heater: a device (or process) which makes to heat up

These definitions are used for the service manager determines suitable devices to provides the service. For example, according to above definition, a air conditioner, window and curtain could be the Cooler since these devices are able to cool down on semantics. However, each device has different capability on a service objective, because we imagine that the air conditioner is more powerful than the window and curtain as a credibility. Therefore, a capability of required device needs to be defined by the SP.

- A required device: Cooler
- A required capability of the device: Hight

The capability of the device might be changed according to the home environment. When the service can not use hight credibility device, the service manager provides other device which can be substitute.

Modeling of the service structure

In order to look at devices from a service objective, a device has various functionality. For example, the curtain could be an illuminator, a temperature controller and a concealer. In this case, the device description also needs to be defined a functionality of the curtain as three cases. Moreover, the capability of the curtain on each service objective should be defined independently.

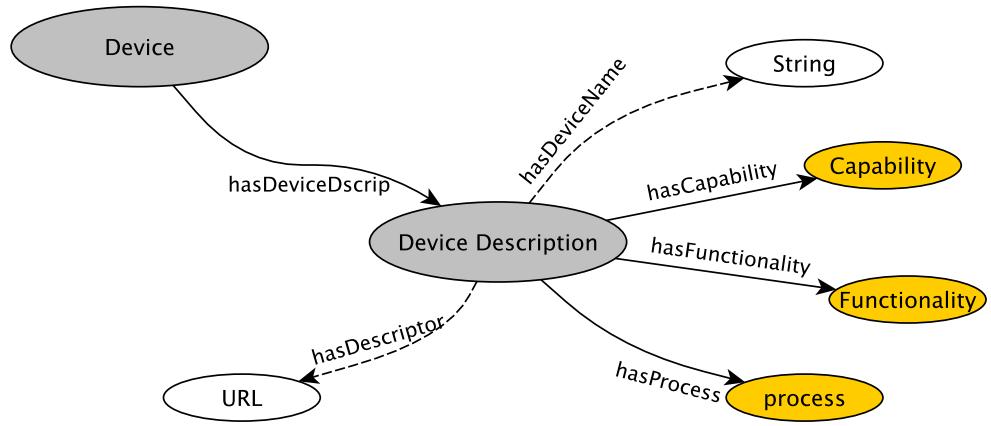


Figure 4.12: Device description

In Figure 4.12, the functionality is defined according to each service objective, and identified by the device descriptor which is URL namespace.

```
DeviceName : Cutain
    DeviceDescription1:
        Name: Cutain
        Descriptor: URL#concealer
        Functionality: Concealer
        Capability: High
    DeviceDesciption2:
        Name: Cutain
        Descriptor: URL#illuminator
        Functionality: Illuminator
        Capability: Variable
    DeviceDescription3:
        Name: Cutain
        Descriptor: URL#temp_controller
        Functionality: Temperature Controller
        Capability: Variable
```

Figure 4.13: An example of the curtain description

Figure 4.13 shows to describe each functionality and a capability of each function, Concealer, Illuminator, Temperature Controller. These three descriptions are identified by the namespace which describes the descriptor.

The Functionality class has function description as figure 4.14. The function description explains a functionality of device, function's name and descriptor.

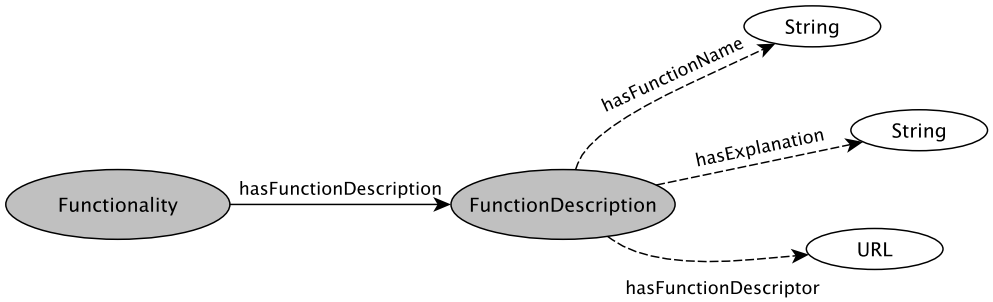


Figure 4.14: A Description of the Functionality

The Capability class has capability description class figure 4.15. The capability description provides a valuation of the capability of a device.

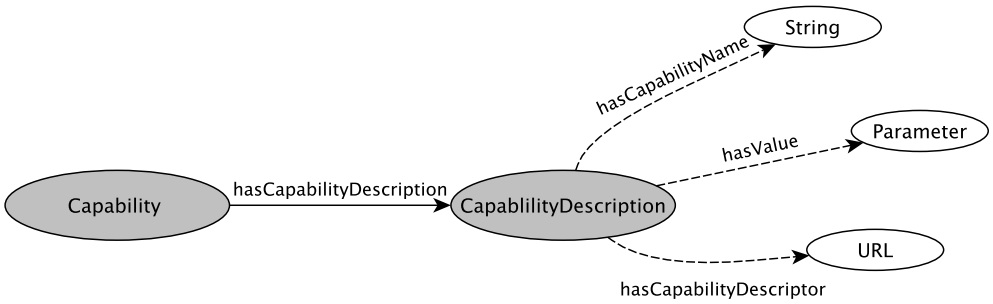


Figure 4.15: A Description of the capability

As a service structure modeling, it's needed to describe property that service has. For example, temperature control service has a temperature control descriptor as a property. The temperature control descriptor has subclass of functionality of temperature as follow.

- Heating : High, Low, Variable
- Cooling : High, Low, Variable

In addition, the service structure model has information of relation between services. For example, the possible services to use for temperature control are an air conditioner, electric fan and window. We illustrated the ontology modeling of a service as figure 4.16.

A device description using the service ontology model

In the service modeling, the services are represented by their features. the service has property within the service descriptor that is described by the service structure model. For example, each service has property of heating and cooling capability as follow.

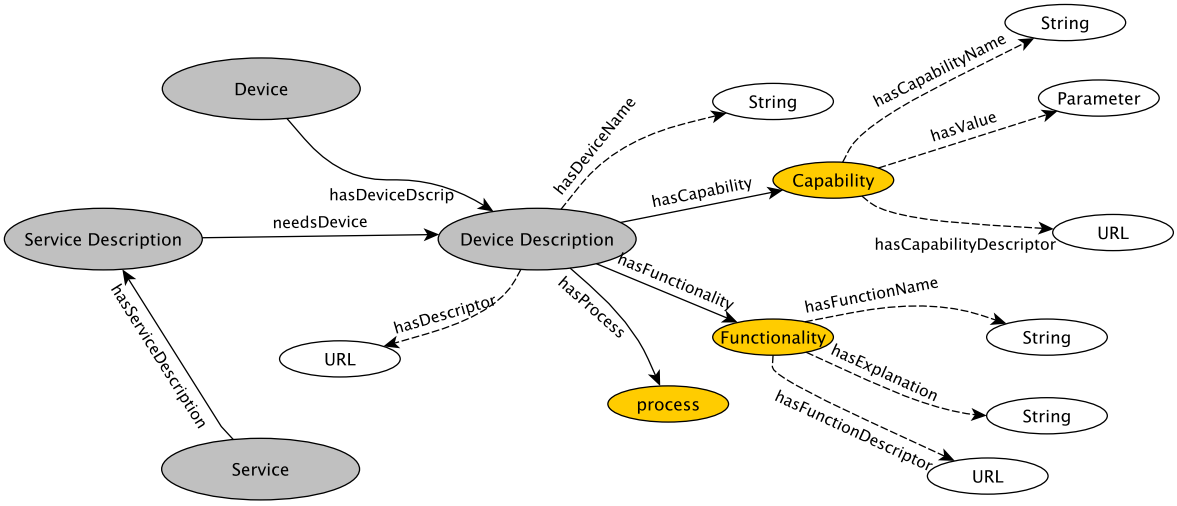


Figure 4.16: The ontology modeling of a service

- AirConditioner
  - Heating Capability High
  - Cooling Capability High
- Electric fan – Cooling Capability Low
- Window – Cooling Capability Variable

4.4.3 Services annotation

To annotate a process of appliances, we used an OWL-S (semantic markup for web service) that is an ontology of services that makes these functionalities possible[8]. The OWL-S provides to discover, invoke and composite a web service. However, there is a difference of a characteristic between the web service and home network service as shows follow.

- Appliance has an area of effect or not
- Service has occupancy or not
- Service has depended on capability of a device

When the agent composites a service, service's area of effect has to be defined, since some appliance provides only limited area such as display service or cooling service. These services have another characteristic of the occupancy. For example, if agent already used display service, other composited service cannot use it. Finally, the appliances have their own capability of process. When the agent decides element among atomic processes that performs as same functionality, the decision will be capability of an appliance.

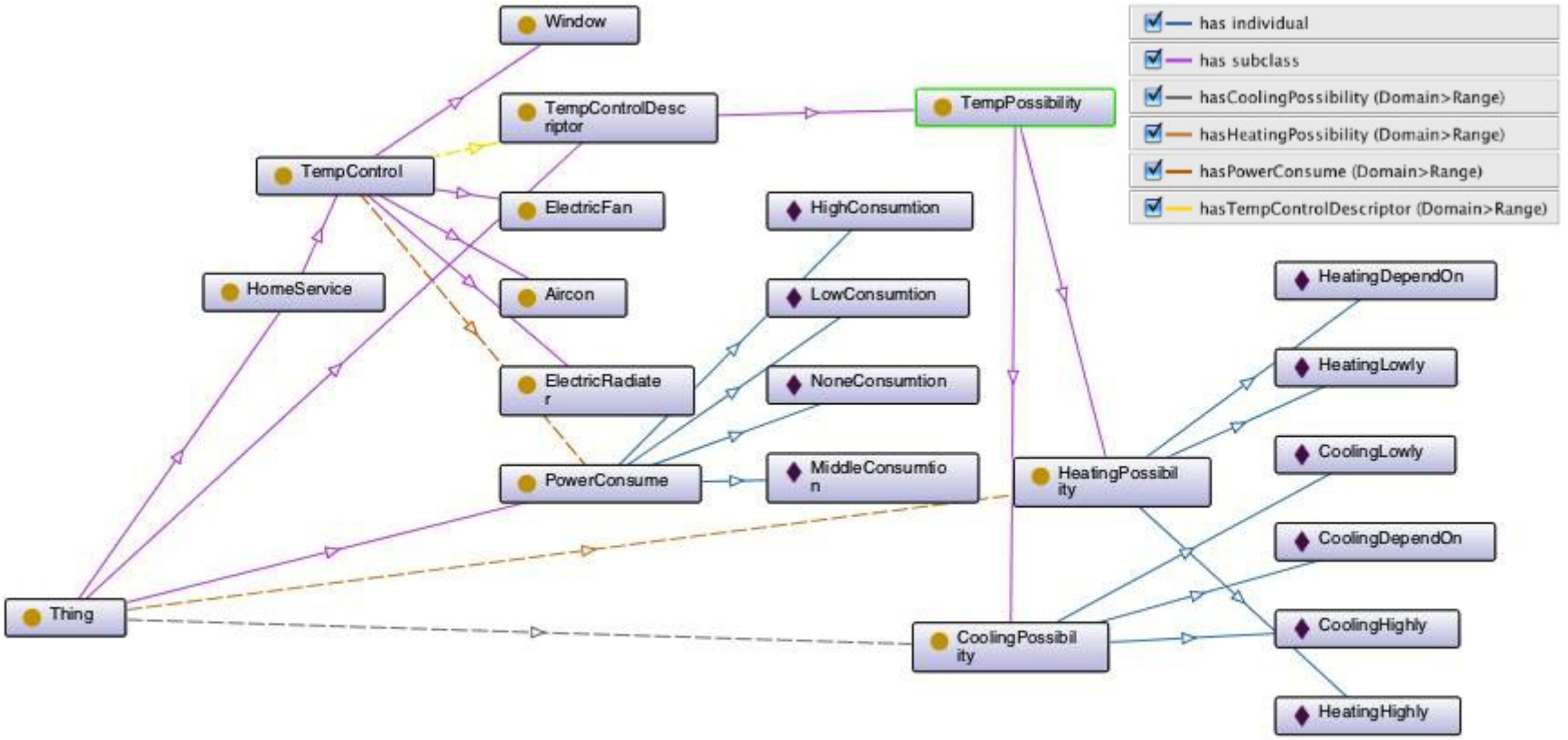


Figure 4.17: Description of the Serivce Model

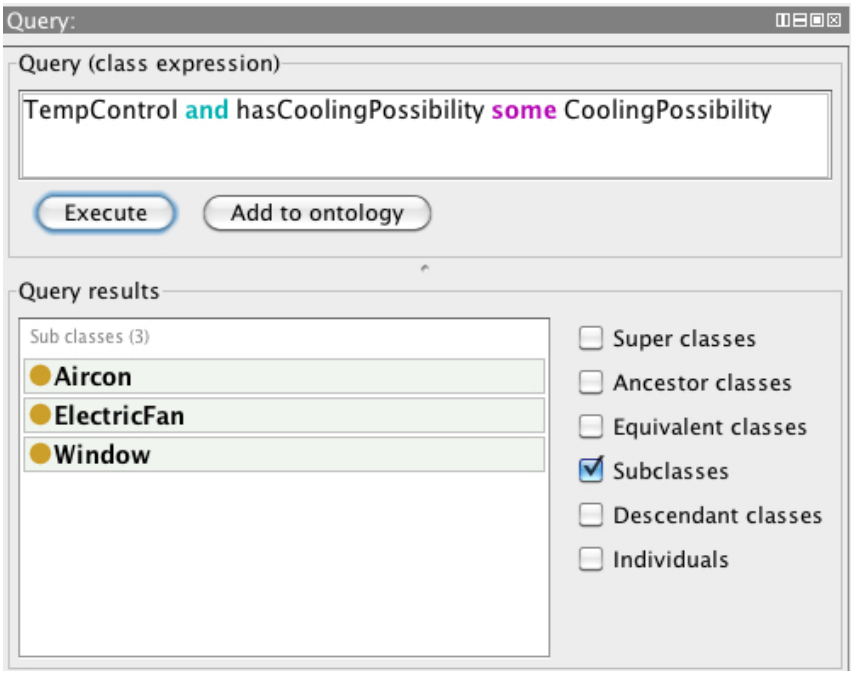
4.5 Experimental result

We have designed and built a simple prototype to demonstrate the reasoning of services elements using semantic service representation. To build an ontology modeling of service and policy, we used the protege that is open-source platform to construct domain models and knowledge-based applications with ontologies. In addition, for reasoning, we used the FaCT++ that is the new generation of the well-known FaCT OWL-DL reasoner. FaCT++ uses the established FaCT algorithms, but with a different internal architecture.

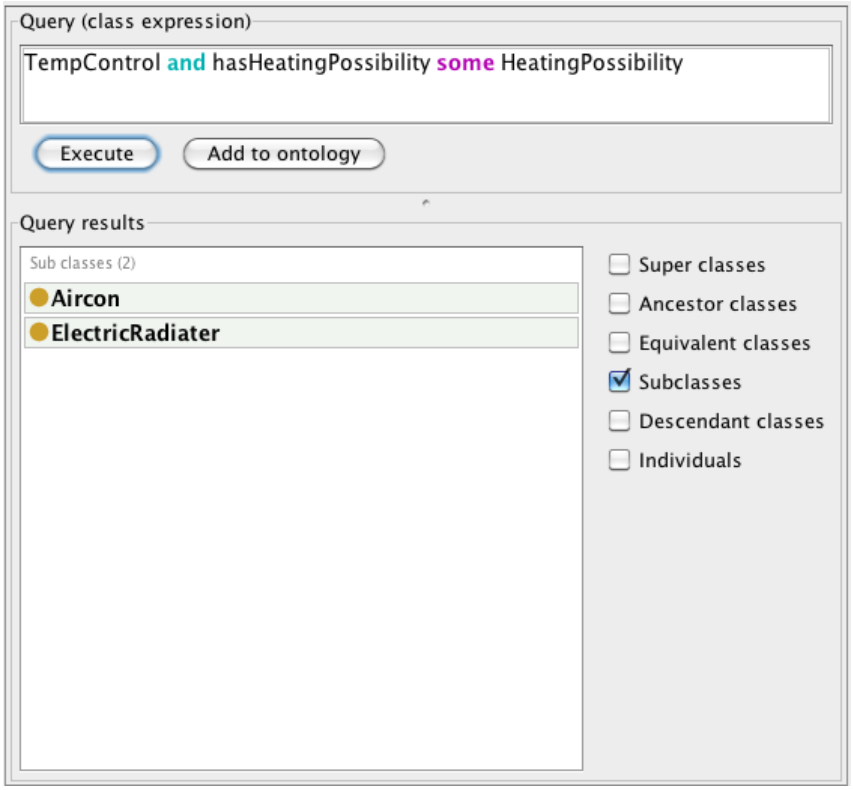
As an experiment, we gave two different queries to find a service element as a scenario. 1) the agent semantically finds service elements that are can provide cooling the room as illustrated in figure 4.5. 2) the agent searches service elements that can provide heating functionality as illustrated in figure 4.5.

In the case of 1), the reasoner searched possible elements such as Air Conditioner, Electric Fan and Window. Since the window represented its functionality on the semantic, the agent could infer that the window has functionality of the cooling. It's meaning that agent can understands diversified functionality of services according to the service object. Moreover, In 2), the agent could infer that the Air conditioner and the electric radiator can provide heating functionality.

Since there are many semantic representations according to services, to represent all meaning in a single representation is actually impossible. The name space of ontology solves such problems by combining the different ontology model. Our system is able to include an external ontology model by importing its name space. Such distributed development of a service model will improve scalability of the service model. In the near future, a distributed environment for developing home service is going to be needed by service provider. Our system provides a platform to deploy a service that was made by an external service provider into the home network environment smoothly.



(a) A query with a cooling functionality



(b) A query with a heating functionality

Figure 4.18: A result of the temperature control serivce

4.6 Summary

As we discussed, service composition needs to dynamically change according to service objective and various policies in the home network environment. In this paper, we proposed the dynamic service composition using Ratiocination on integrated home network system. Our system provides machine-readable service representation to understand the semantics of services and the relation between these services. It is also possible to provide a distributed developing environment using a name space.

We believe that our research provides the solution to decide a service semantically according to service objective considering the priority on the ground of a policy. Furthermore, since there are many semantic representations according to services and policies, to represent all meaning in a single representation is actually impossible. The system that used an ontology model can solves the challenge by importing the policy model from external service providers.

However, we still have several challenges such as suitable ontology mapping between policy models and service models, automated collecting of a policy which is needed when agent decide a service.

Chapter 5

Policy based a dynamic service priority

5.1 Background of service priority

Recently, a research of the service conflict resolution has considered to determine a possessory service to a device and an area using the service priority. As mention the solution of this Issue, at least of one service suspends or stops for other service which has priority to the device. In otherwise, plural services are executed with in a compromise on an inviolable rule of each service[38]. However, as mentioned in section 4.1.2, there are exist several devices that provide service which is semantically equivalent, since the devices in HNS are going to have multifunctional unit. Therefor, the service may find a device which has same functionality.

In this thesis, we defined, a service priority can be divided into two different approaches, a service priority for conflict resolution of elements and a service priority for required condition of policy.

approach 1. "which service will occupy the device or the area"
approach 2. "which device (or service) will be used for the service"

In this section, we discuss a service priority according to these two approaches. Moreover, our research is considered on a service priority for required condition of policy according to the approach 2.

5.1.1 A service priority for conflict resolution of elements

Typical HNS integrated services include:

- **DVD Theater Service:** When a user switches on a DVD player, a TV is turned on in DVD mode, a blind is closed, the brightness of the lights is minimized, 5.1ch speakers are selected, and the sound volume of the speaker is automatically adjusted.
- **Coming Home Light Service:** When a door sensor notices that the user comes home, lights are automatically turned on. Then, the brightness of the lights are adjusted to an optimal value based on the current degree obtained from an illuminometer.

Feature interactions (from now on referenced as FI) may occur in HNS integrated services as well, since multiple services may be activated simultaneously. For example, the above two integrated services interact with each other.

- **Interactions between DVD Theater & Coming Home Light:** Suppose that a user A activates the DVD Theater service, and simultaneously that a user B comes home. Then, the following two interactions occur:
- **FI-(a):** Although the DVD Theater service minimizes a brightness of the lights, the Coming Home Light service sets the brightness comfortable for B. This may ruin A 's desire to watch the DVD in a comfortable atmosphere.
- **FI-(b):** If the blind is closed (by the DVD Theater) immediately after the lights read the degree from the illuminometer (by the Coming Home Light), the lights may fail to set the optimal illumination because the blind makes the room darker.

The feature interaction problem in the HNS integrated services was first addressed by Kolberg et al. [17]. These authors regard each HNS component (an appliance or an environmental variable) as a resource. In their model, each integrated service accesses some resources in a shared or not-shared mode. An interaction is detected when different services try to access a common resource with an incompatible access mode. Thus, each appliance is simply modeled by the two-valued access attributes, and each integrated service is characterized only by how the service sets values of the attributes. This simple modeling enables a light weight and realistic implementation framework for feature interaction avoidance. However, a conflict resolution methods have one distinct disadvantage: the suspension of at least one service. This is necessary to allow the remaining services to continue without any conflicts. Regarding device conflicts, it may be inevitable to completely suspend/stop a service to at least allow other services run successfully.

5.1.2 A service priority for required condition of policy

Policy	5.1ch speaker	Table speaker	Headphone
Sound quality	High	Middle	Low
Privacy	Low	Middle	High
Mobility	Low	Low	High

Figure 5.1: A priority based on policies

To provide an integrated service in home network, various policies have exert an influence on a service priority. We assume that there are many devices that provide a service

on the same objective, these devices are different each other on a capability, an efficiency or a responsibility. In this case, a service priority is changed according to what a policy needs to precede. For example, as mentioned previous section, the air conditioner and the curtain could provide same service with cooling functionality. However, in considering for energy saving policy, the curtain is more efficient than the air conditioner.

There the service manager need to determine a priority between the curtain and the air conditioner for the service composing. We illustrated an example of a priority based on policies as Figure 5.1. Each device has capability on considering several policies. When service manager determine a sound device for the service, priority is considered according to policy.

For this issue, when the policy were changed in same service objective, an element which is used for service composing is also changed. Therefor, the service manager have to understand these policies and determine the priority of services base on the policy. In this research, we defined these policies semantically and provided a service composing with a suitable element according to the policy.

5.2 Priority of element using policy

5.2.1 Dynamic service priority with a policy

As mentioned in section 5.1.2, a priority of the service is dynamically changed by a policy. We can imagine that the air condition service, the electric fan service and the curtain control service have same effect on the temperature control as shown finugre 5.2.

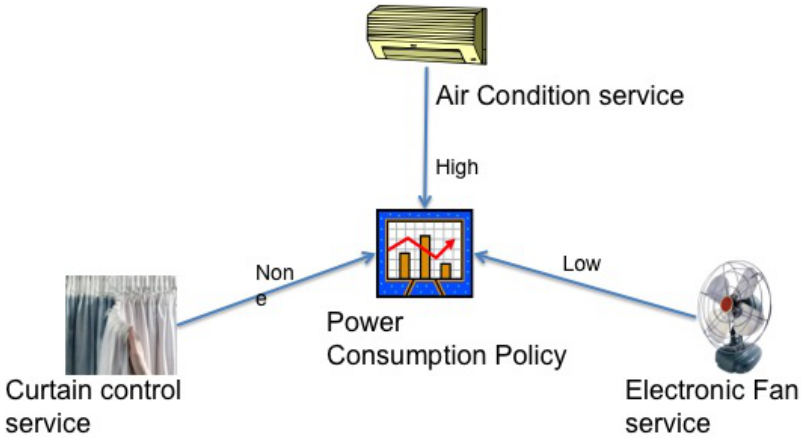


Figure 5.2: A priority based on policies

As considering the air condition service with energy consumption, an electric fan or curtain consumes less energy than an air conditioner in figure 5.2. The figure illustrates that only consider the power consumption of services. For example, when the user want to control a temperature within the energy saving, the service manager determine a suitable service which has low energy consumption such as the curtain control service or the electric fan service.

In otherwise, as considering of effectively response, the priority is changed that the air condition service is fastest, the electronic fan service has slow and the curtain control

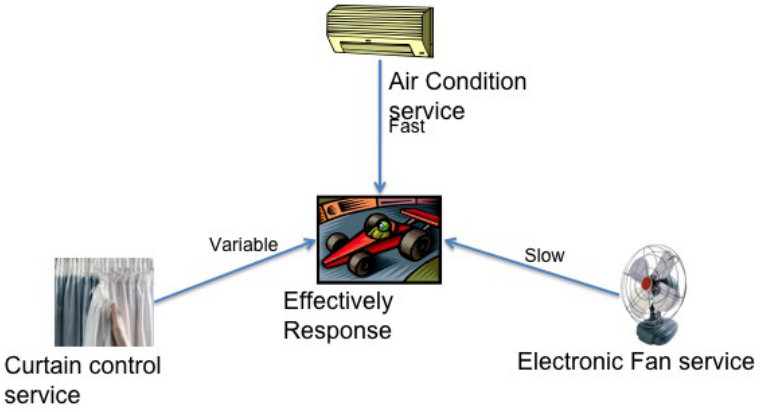


Figure 5.3: A priority based on policies

service is variable with an envrionment. if the user considers a capability of services, the air conditioner may have higher priority than other elements as shown figure 5.3. Hence, the service manager needs to understand various policies as many as possible and decide a service according to the policy.

5.2.2 Distributed policy defining for the dynamic service priority

To provide a service with considering the policy is influence to composite services because elements of composed service are dynamically changed by priority of the policy. However, there are several challenge for defining of various policies. First of all, the service has to be defined with it's priority according to every single policy. In this case, the SP which describes their own service carries the burden inefficiently. Secondly, even if the SP defines every single policy, these definition are valid only specified service.

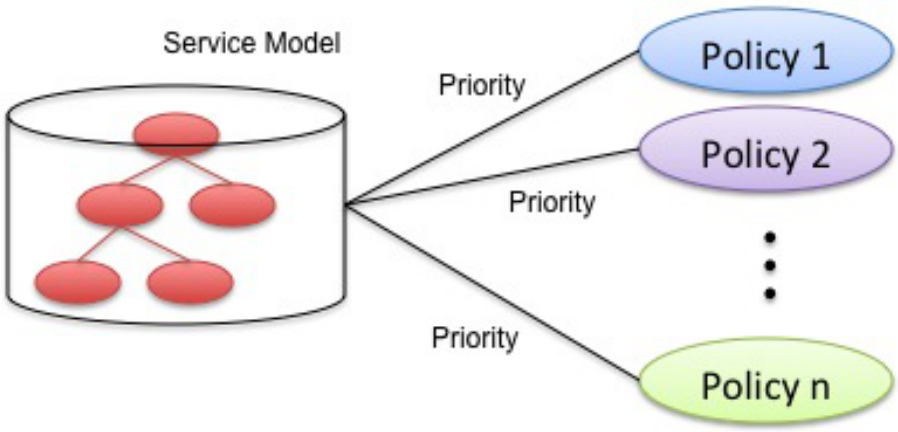


Figure 5.4: A distributed policy defining

To solve above challenges, we proposed the distributed policy defining environment for the dynamic service priority using semantic policy modeling. As we can see figure 5.4,

the service has various priority about each external policy. the external policy defines a priority of a specified service.

5.2.3 Advantages of the distributed policy definition

The following components are describes advantages of the distributed policy definition.

- **A various priority of a service**
- **A specialization between a policy and service definition**
- **Reusable of existing policy definition**

In oder to consider the component, the distributed policy definition provides three advantages on the service definition. In first component, the service manager determines various priority of a service element according to a required policy because various external policy definitions are providing a valuation of elements . As second component, view point of a developer (or descriptor), the distributed policy definition provides a specialized develop environment between a service and priority. Finally, these defined external policies are reused for describing other service.

5.3 Policy ontology model

In this section, we describe proposed system, the policy based a dynamic service priority using a semantic policy model. Moreover, we provide a methodology "how to define an ontology model of the policy".

5.3.1 A structure of the system

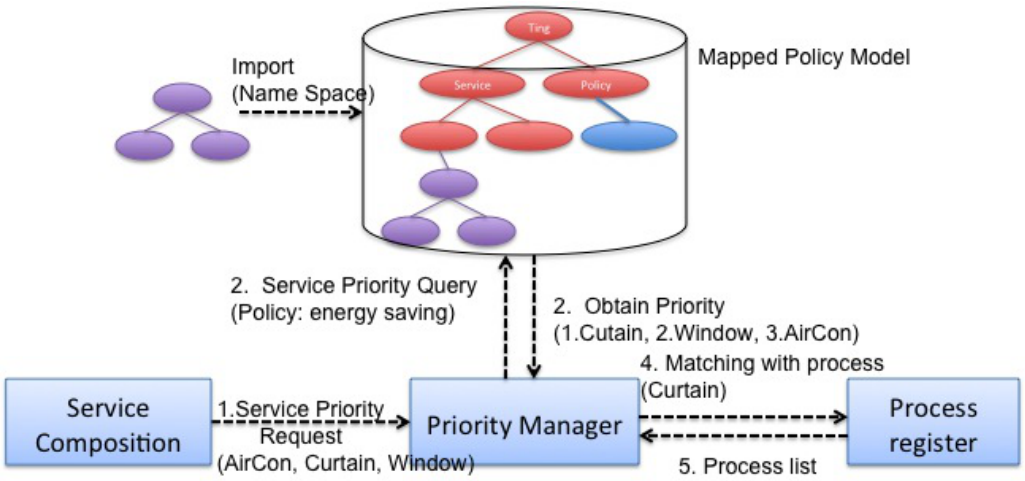


Figure 5.5: A structure of the distributed policy definition

As we can see the figure 5.5, there are three components, Priority Manager, Policy Model, Process register. Policy Model defines a notion between services and imported

policy model which is identified as a namespace. Priority Manager queries policy based service priority with service elements, and obtained priority is matched with registered process list. Process register provides process list which is exist in the home.

For instance, we can imagine that Service Composition needs a priority of service elements, air conditioner, curtain and window, and the user want energy saving on this service. Firstly, Priority Manger queries to Policy Model with such condition. Secondly, Policy Model provides their priority as a result based on the energy saving policy (the curtain lowest then other). Finally, Priority Manger provides obtained priority after matching with a process list.

5.3.2 Modeling the Policy

In this section, we give a simple scenario of a policy based a service priority. In case of the earphone, it has many priority according to policies on a sound output device.

```
Devcie Name : earphone
    Policy : Privacy
        Priority : High
    Policy : Sound Quality
        Priority : Low
    .
    .
    .
    Policy : Mobility
        Priority : High
```

Figure 5.6: A priority of the earphone according to policies

In the figure 5.6, the earphone is evaluated by several policies, privacy, sound quality and mobility. According to privacy policy, the earphone is highest then other devices. However, the sound quality policy evaluates the earphone as less priority.

The ontology modeling of the policy

For such a reason above, a device needs to be evaluated by each different policy. We defined the policy model as follow.

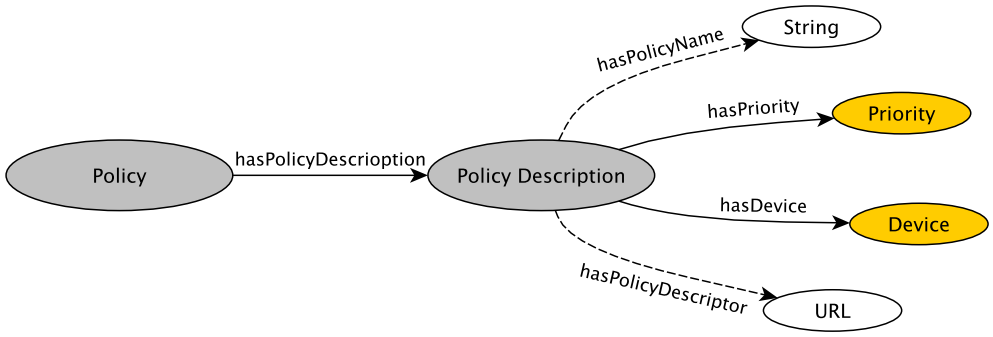


Figure 5.7: A policy ontology model

As mention in figure 5.7, the class Policy has a object property of *hasPolicyDescription* to Policy Description. The class Policy Description has a data property of *hasPolicyName* and *hasPolicyDescriptor*. Moreover, Policy Description class has object properties, *hasPriority* to Priority class and *hasDevice* to Device class.

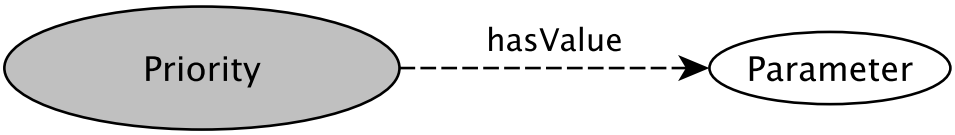


Figure 5.8: The priority class

Figure 5.8 illustrates the priority class which has the data property of *hasValue*. The parameter is determined by a valuation of priority.

The Device class is described in section 4.4.2. Figure 5.9 illustrates all model of the service and policy.

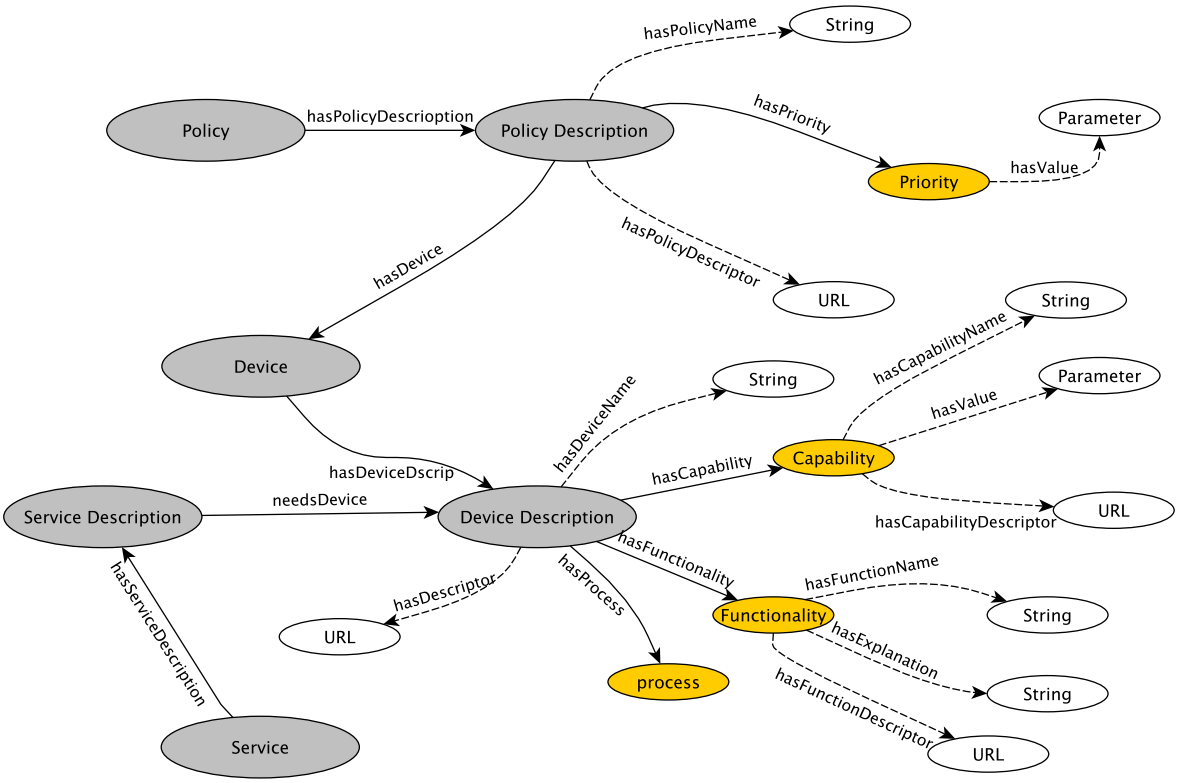


Figure 5.9: A service model and policy model

Building policy instance

As we mentioned before, providing service with considering the policy is influence to composite services because elements of composed service are dynamically changed by

priority of the policy as defined follow.

Policy based composition – composed element could be changed depend on policies

In this section, we describe to define a policy with example of power consumption policy. The policy model provide various point of view to understand services such as following example of power consume policy.

- AirConditioner – High Consumption of Power
- Electric fan – Low Consumption of Power
- Window – None Consumption of Power

```
        <Class IRI="#Aircon"/>
        <ObjectHasValue>
            <ObjectProperty IRI="http://www.semanticweb.org/ontologies/2010/11/response↩
                .owl#hasResponse"/>
            <NamedIndividual IRI="http://www.semanticweb.org/ontologies/2010/11/↩
                response.owl#Fast"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasCoolingPossibility"/>
            <NamedIndividual IRI="#CoolingHighly"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasHeatingPossibility"/>
            <NamedIndividual IRI="#HeatingHighly"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasPowerConsume"/>
            <NamedIndividual IRI="#HighConsumtion"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#CoolingPossibility"/>
        <Class IRI="#TempPossibility"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ElectricFan"/>
        <Class IRI="#TempControl"/>
    </SubClassOf>
```

Figure 5.10: A example of air conditioner description using OWL

Figure 5.10 shows a example of air conditioner description using OWL. In this description, the air conditioner has a functionality of Cooling and Heating as highest capability, and the power consumes is High Consumption.

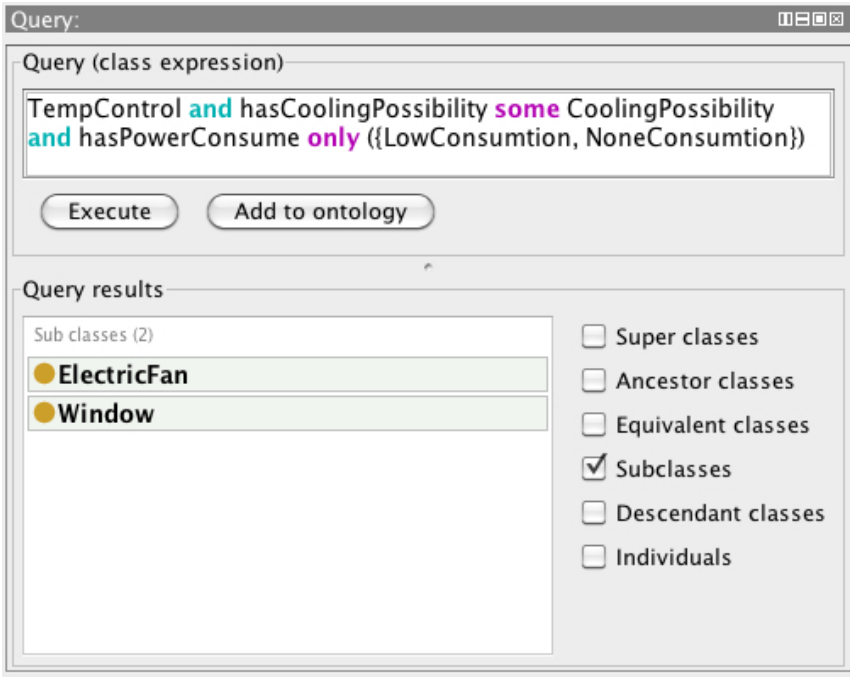
5.4 Experimental results

We have designed and built a simple prototype to demonstrate the reasoning of services elements using semantic service representation. To build an ontology modeling of service and policy, we used the protege that is open-source platform to construct domain models and knowledge-based applications with ontologies. In addition, for reasoning, we used the FaCT++ that is the new generation of the well-known FaCT OWL-DL reasoner. FaCT++ uses the established FaCT algorithms, but with a different internal architecture.

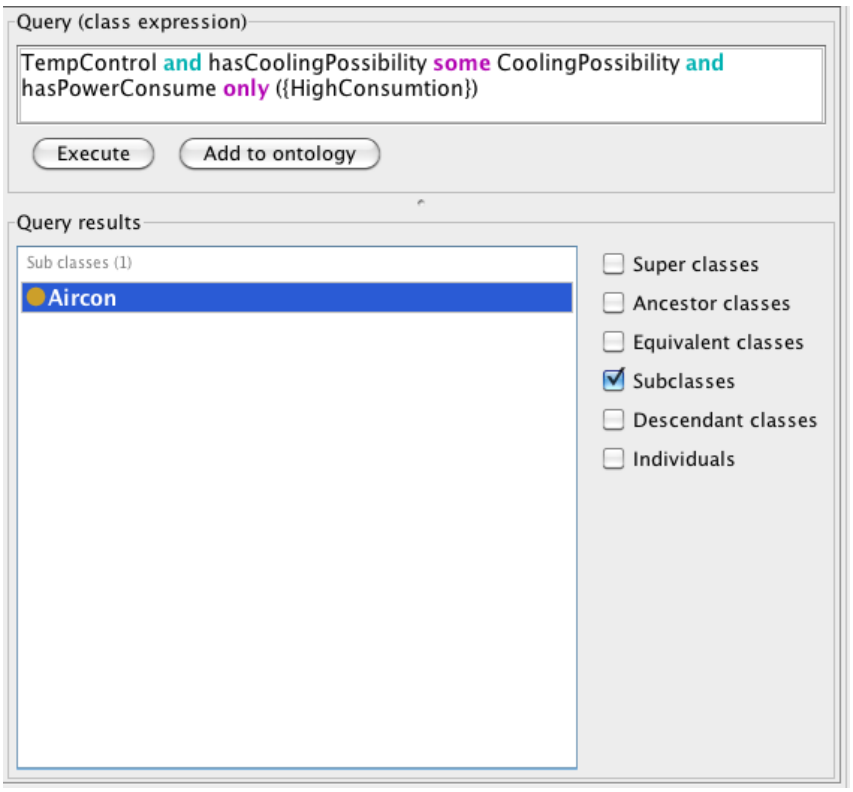
As an experiment, Basically, the agent semantically finds service elements that are can provide cooling the room as illustrated in section 4.5. Additionally, the agent considers an energy saving policy within searching cooling service as illustrated in figure 5.11.

For the experiment, we defined the policy of power consumption about appliances. Moreover, We defined number of services such as window, electric Fan, Air Conditioner and Electric Heater as the service model.

As a result, we can see that Air Conditioner is excepted in query results by considering of power consumption as shown figure 5.11.(a). Thus, a priority of service elements could be changed by considering the policy. We believe that our system provides to infer a relation between services and policy.



(a) A query with low consumption



(b) A query with high consumption

Figure 5.11: A result of the temperature control serivce with the consumption policy

5.5 Summary

As we mentioned previous, when the policy were changed in same service objective, an element which is used for service composing is also changed. Therefor, the service manager have to understand these policies and determine the priority of services base on the policy. In this chapter, we defined these policies semantically and provided a service composing with a suitable element according to the policy.

As we can see the experimental, our research gives three novels, 1)the service manager determines various priority of a service element according to a required policy because various external policy definitions are providing a valuation of elements . 2)In view point of a developer (or descriptor), the distributed policy definition provides a specialized develop environment between a service and priority. 3)these defined external policies are reused for describing other service. We expect that our system provides priority based service composition because during service composition the system also takes into account a policy ontology model. It is also possible to provide a distributed developing environment for policies using a name space.

In the future work, we need to consider such as suitable ontology mapping between policy models and service models, automated collecting of a policy which is needed when agent decide a service.

Chapter 6

Adaptive link topology of an appliance

6.1 Appliance composition for a service

6.1.1 Background of the appliance composition

Generally, an infrastructure of appliances in the home network environment is composed by diversification of an agreement from different kinds of industry. For example, there exist Audio and Video electronics, not only a brand new DVD player which can provide a functionality of DLNA and capability of HDMI, but also the past generation of TV that only have a terminal of S-video.

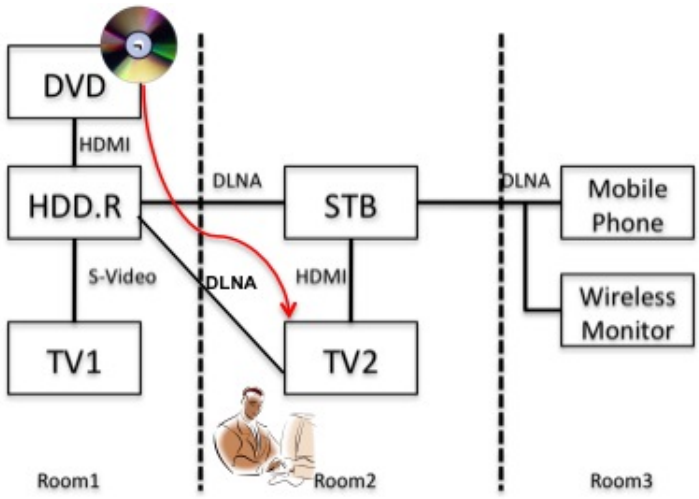


Figure 6.1: Example of networked home environment

Figure 6.1 shows an example of networked home environment. The appliances can provide a multimedia service with DLNA between each room, are composed by legacy AV such as TV1, DVD player and TV2. In this case, a centralized service manager has difficulty in providing an integrated service since the service manager could not be looked up the composition of these appliances. For example, when user located on front

of TV2 in the room2, want to watch a movie using DVD player located in Room1, service manager has to establish the composition of the appliance such as DVD, HDD.R, STB and TV2 also DVD, HDD.R and TV2. Moreover, the capabilities of the appliances need to be considered for Quality of Service(QoS). It seems similar to network routing [**?**][**?**]. However, there are several differences that home appliance have a number of connecting instrument for transmitting the content, generally, ad-hoc networks have used only one communication protocol meaning that input/output are fixed. Also the home devices are represented with network topology by service manager. For these reasons, we need to approach variable according to our aim.

6.1.2 Adaptive appliance composition

In previous section, we discussed background of the appliance composition in the various home environment. In below, we have defined a number of requirements for flexible composition between the variety of appliances.

- **Possibility of the composition -** Service manager has to define all of the possibility of composition on existing appliance in the home.
- **Capability of service -**The composition has to be considered the QoS of an appliances which is composited by service manager.
- **Observation of a state of device -** The devices have to be watched on the state carefully since their could provide other service on a concurrent operations.

These challenges form the basis of our approach and in the following sections we discuss how they have been addressed within our system.

6.2 Proposed system

The system we have designed ensures the integration of appliances composition, which means that the system covers a vast range of agreements from various industries, including legacy appliances. In our system, we preponderantly consider a methodology how to discover suitable composition in complicate networked appliances by resolving the following two challengers: 1)finding out all of possible compositions among thus appliances, 2)the system determine the best path according to QoS and an efficiency on the discovered list of the composition.

6.2.1 Structure

A structure of our composition system is shown in Figure 6.2. The Service Manager handles the appliances through the API within providing the Integrated service according to a service scenarios.

The information of the appliances such as model name, number and functions are registered to the service manager using the XML or RDF(Resource Description Framework). When the service is required with the compositing of the appliances, the Service Manager request it to the Composition Module with providing the information of the Device and the status of the appliances. The Composition Manager discover all of the

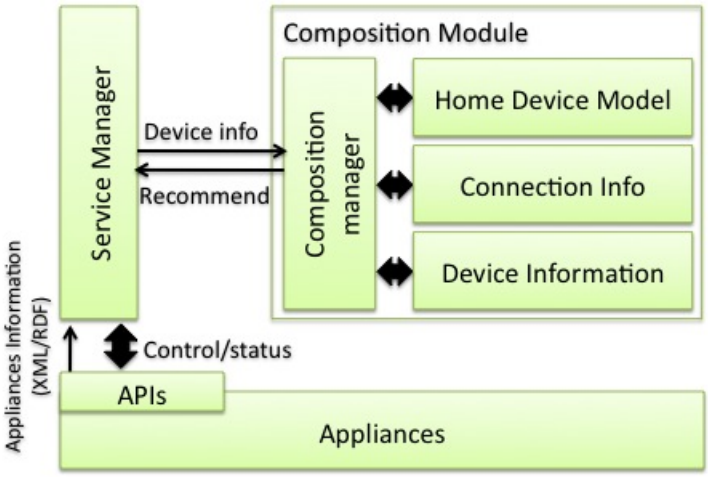


Figure 6.2: Structure of the Composition system

composition list and transmit the recommendation which is determined on these lists as more appropriate composition.

6.2.2 Device Information

The Composition Manager stores the Device Information received from the Service Manger, such as a device type, model name and a connector type. It is used to discover the possible compositions of appliances by the Home Device Model. In the follow, we have define the contents of data in the Device Information.

- Index: index of the Model Name
- Device Type: TV, DVD, STB
- Model Name: a proper name from a manufactory
- Connecter Type: HDMI, S-video,DLNA
- Direction: input/output

6.2.3 Connection Information

The possible composition list obtained from the Home Device Model are needed to compare with the existing connection of the appliances which is in the real home environment. Also Connection Information module provides the state informations of the device since an appliance might be in the working as concurrent operation. In this case, the composition manager has to discover other possible composition.

- Connection: BOOL(True/False)
- Operation: BOOL(True/False)

If the appliance is working on another service, the service manager announces the state to the composition manager and these information are renewed to the 'False' on the connection information module.

6.2.4 Home Device Model

The Home Device Model provides a brief of a structure for the composition modeling of devices; the existing devices in the home environment are referred on this model, which is contained an information of the capability of the appliances. Figure 6.3 shows the description of the home device model, which is covered three different agreement of device.

The model is considered as a relation between the device and efficiency of the connection instrument their own. In the describing the composition model, the classification might be three divisions as follows:

- Devices
- Connection instrument
- Interrelation

Each device has several connection instrument and the direction of input/output, is expressed by the lines between class of the devices and class of the connection instrument. A same device type which has different connection instrument may exist same agreement area by indexing the each device from the model name.

The interrelation between the device and connection instrument is represented by a weight of the lines, it is determined according to the quality of capability.

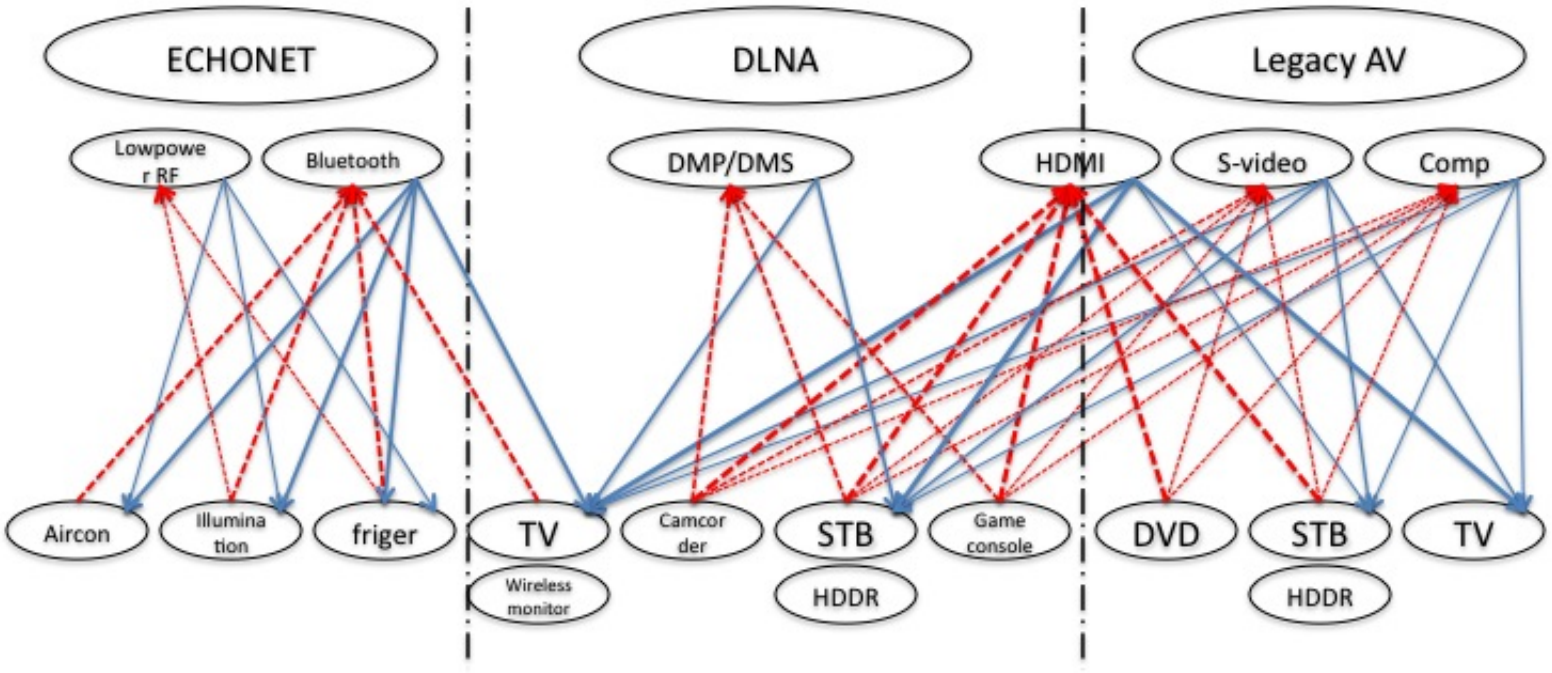


Figure 6.3: Description of the Home Device Model

6.3 Algorithm

The algorithm of device composition is divided into a number of procedures. First of all, searching the whole composition route on the device model. Second, determining the best recommendation on the composition list as the QoS and efficiency of operation.

6.3.1 Route searching

To search the whole route on the device model, we need to consider the possibility of an infinity loop between devices. In this case, we need to define several precondition such as follows:

- The searching must not visit the same device within the process.
- A number of hope has to be smaller then the number of devices in the home.

In this algorithm, we referenced the DFS(Depth first Search) that is an uninformed search that progresses by expanding the first child node of the search tree that appears and thus going deeper and deeper until a goal node is found, or until it hits a node that has no children.

```
def fun_stack
            Stack_in[count]=dev
            Steck_out[count]=cand
.
:
def fun_Composition(start_index,reach_index):
  count=Num_Recurrence
  DeviceOut=fun_capable_output(start_index)

  for dev in DeviceOut:

            Possible_dev=fun_capable_input(Dev)

            for cand in Possible_dev:

              fun_stack(dev)

              if DeviceInfo[cand][DeviceType] !=
DeviceInfo[reach_index][DeviceType]:

                fun_stack(cand)

                resch=DeviceInfo[cand][Index]
                fun_Composition(resch,reach_index)
              else:

                fun_stack(cand)
```

Figure 6.4: Composition Algorithm

The figure 6.4 shows a summary of the composition algorithm. In depending on this figure, `fun_composition` receive an devices index of start and object as a factor. The function of `fun_capable_output` is searching the list that is possible output instrument on the start device. `fun_capable_input` receive the factor of each index searched by `fun_capable_output`, looking up the possible input as same variety of instrument. If a result of the index which is input instrument searched by `fun_capable_input` is matched with the index of the object, the output instrument is being next list. In negative case, `fun_composition` is called out recursively by renewing the index of start device.

6.3.2 Determination of the link

For determining the best link topology, need to preponderantly considered how to define the relation between two items as follows:.

- Quality of Service
- Efficiency of operation

Tho QoS is determined by the capability of the connection instrument. Composition manager stores the worst weight of the connection instrument on the result which as each possible composition since the quality of the connection is depended on a consistency. For example as figure 6.5, the quality of third line just has the value of S-video.

The service manager prevents ignoring the accidental increased the scale of the composition considering number of the hop between the appliance for the efficiency of the operation.

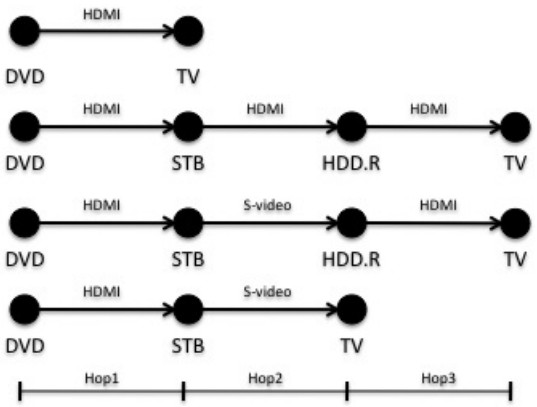


Figure 6.5: Relation between the Quality and Efficiency

$$Q = Cw/Ct * 100 \tag{6.1}$$

The Cw is the worst weight, Ct is number of instrument.

$$E = Hn/Hm * 100 \tag{6.2}$$

The Hn is the number of hop, Hm is maximum number of hop.

The above quality Q and efficiency E are used to determine the flexible composition depending on the user preference by the service manager.

As you can see in figure 6.5, the best composition might be the first line on the list Hence, in case of ambiguity such as second and fourth line, service manager needs a standard of valuation.

6.4 Experimentation

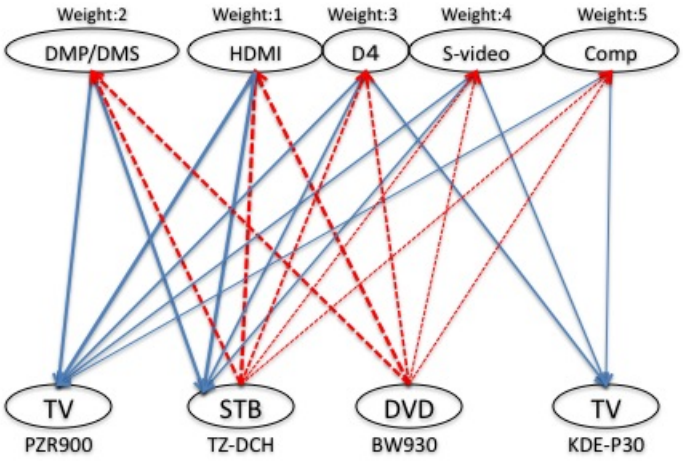


Figure 6.6: Modeling the Device

In order to implement the system, we constructed using four home appliance informations of which has virtually distributed in the world, Panasonic TV: PZR900, DVD: BW930, Set-top-box: TZ-DCH and Sony TV: KDE-P3 as the Table 6.1, is that described the connection instrument of each device.

We described the relations between a devices and a connection instruments based on the information which is from Table 6.1 as shown in Figure 6.6, according to the Home Device Model. In this implementation, each connection has the weight of capability, we chose the precedence of the minimum and, all of the connection instruments were established in the enable of a state which mean that the system might search whole possibility to composite the device.

A value of the weight is decided depend on quality of the connection instrument, HDMI provided the resolution of maximum 1080i(Interlace scan), D4 is 1080i, S-video is maximum 720p(Progressive scan) and Comp is maximum 480i. In fact, we need to determine a standardization of a quality of connection instruments for automatically configuring the home appliance. in this experiment, we discretionary define the quality of connection instruments.

The Figure **??** illustrates the result of the composition list obtained from the device model, the searching condition is that find the all possible composition list from DVD BW930 to TV PZR900. The possible compositions are indicated as Quality 'Q' and Efficiency 'E'. In the result we can see first composition list and second list have same quality. However, second list is inferior as efficiency since it used more device. If state of

Table 6.1: Device Information

Index	Device Type	Mode name	Manufacturer	HDMI		DLNA		D4		S-video		Comp	
1	TV	PZR900	Panasonic	In	X	In	X	In	X	In	X	in	X
2	DVD	BW930	Panasonic	X	Out	X	Out	X	Out	In	Out	X	Out
3	STB	TZ-DCH	Panasonic	In	Out	In	Out	In	Out	In	Out	in	Out
4	TV	KDE-P32	SONY	X	X	X	X	In	X	In	X	In	X

device were zero, composition manager do not count the device during search the route in the device model.

As we can see the result, there are many possible way to composite devices in networked home environment. It's meaning that the integrated service manager needs to determine suitable composition of devices according to a property of service. For example: 1)a home theater service which is needed a hight quality of resolution has to consider Q more, 2)a media converting service that has time delay for processing has to consider E more. We believe that the integrated service manager can adjust an importance between Q and E base on our system.

```
2   'DVD'   'HDMI'      'Output'    1   'BW930'     1
1   'TV'    'HDMI'      'Input'     1   'PZR900'    1
Q=1,    E=50.0%
OK----------------------
2   'DVD'   'HDMI'      'Output'    1   'BW930'     1
3   'STB'   'HDMI'      'Input'     1   'TZ-DCH'    1
3   'STB'   'HDMI'      'Output'    1   'TZ-DCH'    1
1   'TV'    'HDMI'      'Input'     1   'PZR900'    1
Q=1,    E=100.0%
OK----------------------
2   'DVD'   'HDMI'      'Output'    1   'BW930'     1
3   'STB'   'HDMI'      'Input'     1   'TZ-DCH'    1
3   'STB'   'Svdo'      'Output'    4   'TZ-DCH'    1
1   'TV'    'Svdo'      'Input'     4   'PZR900'    1
Q=4,    E=100.0%
OK----------------------
2   'DVD'   'HDMI'      'Output'    1   'BW930'     1
3   'STB'   'HDMI'      'Input'     1   'TZ-DCH'    1
3   'STB'   'COMP'      'Output'    5   'TZ-DCH'    1
1   'TV'    'COMP'      'Input'     5   'PZR900'    1
Q=5,    E=100.0%
OK----------------------
2   'DVD'   'DLNA'      'Output'    2   'BW930'     1
1   'TV'    'DLNA'      'Input'     2   'PZR900'    1
Q=2,    E=50.0%
OK----------------------
2   'DVD'   'DLNA'      'Output'    2   'BW930'     1
3   'STB'   'DLNA'      'Input'     2   'TZ-DCH'    1
3   'STB'   'HDMI'      'Output'    1   'TZ-DCH'    1
1   'TV'    'HDMI'      'Input'     1   'PZR900'    1
Q=2,    E=100.0%
OK----------------------
:
:
2   'DVD'   'COMP'      'Output'    5   'BW930'     1
3   'STB'   'COMP'      'Input'     5   'TZ-DCH'    1
3   'STB'   'HDMI'      'Output'    1   'TZ-DCH'    1
1   'TV'    'HDMI'      'Input'     1   'PZR900'    1
Q=5,    E=100.0%
OK----------------------
2   'DVD'   'COMP'      'Output'    5   'BW930'     1
3   'STB'   'COMP'      'Input'     5   'TZ-DCH'    1
3   'STB'   'Svdo'      'Output'    4   'TZ-DCH'    1
1   'TV'    'Svdo'      'Input'     4   'PZR900'    1
Q=5,    E=100.0%
OK----------------------
2   'DVD'   'COMP'      'Output'    5   'BW930'     1
3   'STB'   'COMP'      'Input'     5   'TZ-DCH'    1
3   'STB'   'COMP'      'Output'    5   'TZ-DCH'    1
1   'TV'    'COMP'      'Input'     5   'PZR900'    1
Q=5,    E=100.0%
```

6.5 Summary

In this paper, we proposed a framework to flexibly composite the appliances on the networked home environment for providing integrate service. Our system makes tree novel contributions: 1) The device model provides a representation of the relation of the connection instruments on the device, 2) the composition algorithm search the possible composition in the device model and 3)determining the route for providing a service in searched composition list is done according to the quality and efficiency.

We strongly expect that our system will provides a methodology on how to select the best composition to integrated service manager in various home environment since the system can determines a transmitting route of contents suitably among networked home devices including legacy appliances. Also, we believe it could be worked on an assistance to build an appliance into the home by predicting of best composition before device is built in. However, we still have several challenge as an embodiment of the methodology such as considering the QoS, efficient and context aware service for the understandable centralization computer [39].

In the future work, we will carefully develop the system according to the integrate service application over considering what we mentioned above. Also, a standardization for defining a quality among connection instruments of appliances has to be considered for best composing of appliances.

Chapter 7

Experimentation and Evaluation

Up to now we have looked at the methodology of semantic service composition using the service ontology model and the link topology of appliances. This chapter will show how to realize the service composition system and demonstrate the relation between the model and the instance devices. After then the system will experiment with composed services on networked appliances in experimental house (iHouse). Finally, these composed service will be evaluated on their feasibility.

7.1 Semantic Service Composition System

7.1.1 Service structure ontology

To build an ontology service model, we use Hozo ontology editor[40]. Hozo is an environment for building and using ontologies based on a fundamental consideration of " Role " and " Relationship ". Superclass and subclass relations are also utilized in our ontology for the definition of environment context in the smart home. For example, the Electric appliance class and Measurement device are subclasses of the Object class. Thus, all properties in the superclass (Object class) will be inherited to the subclass (Electric appliance, Measurement device classes).

Figure 7.1 shows the service structure description ontology. This model include the following classes.

- *Object* class: to describe a type of various home devices.
- *AtomicProcess* class: to describe a functionality which independently used.
- *ServiceElements* class: to describe service elements on smart home network.
- *Location* class and *LocationInstance* class: to describe location type and location's name in each difference smart home environment.
- *objectInstance* class: to identify an object with a location and an universally unique identifier (in this case, IP will be UUID) .
- *Status* class: to describe an activity of devices.
- *Object-AP* class: to describe a relationship between *Object* class and *AtomicProcess* class.

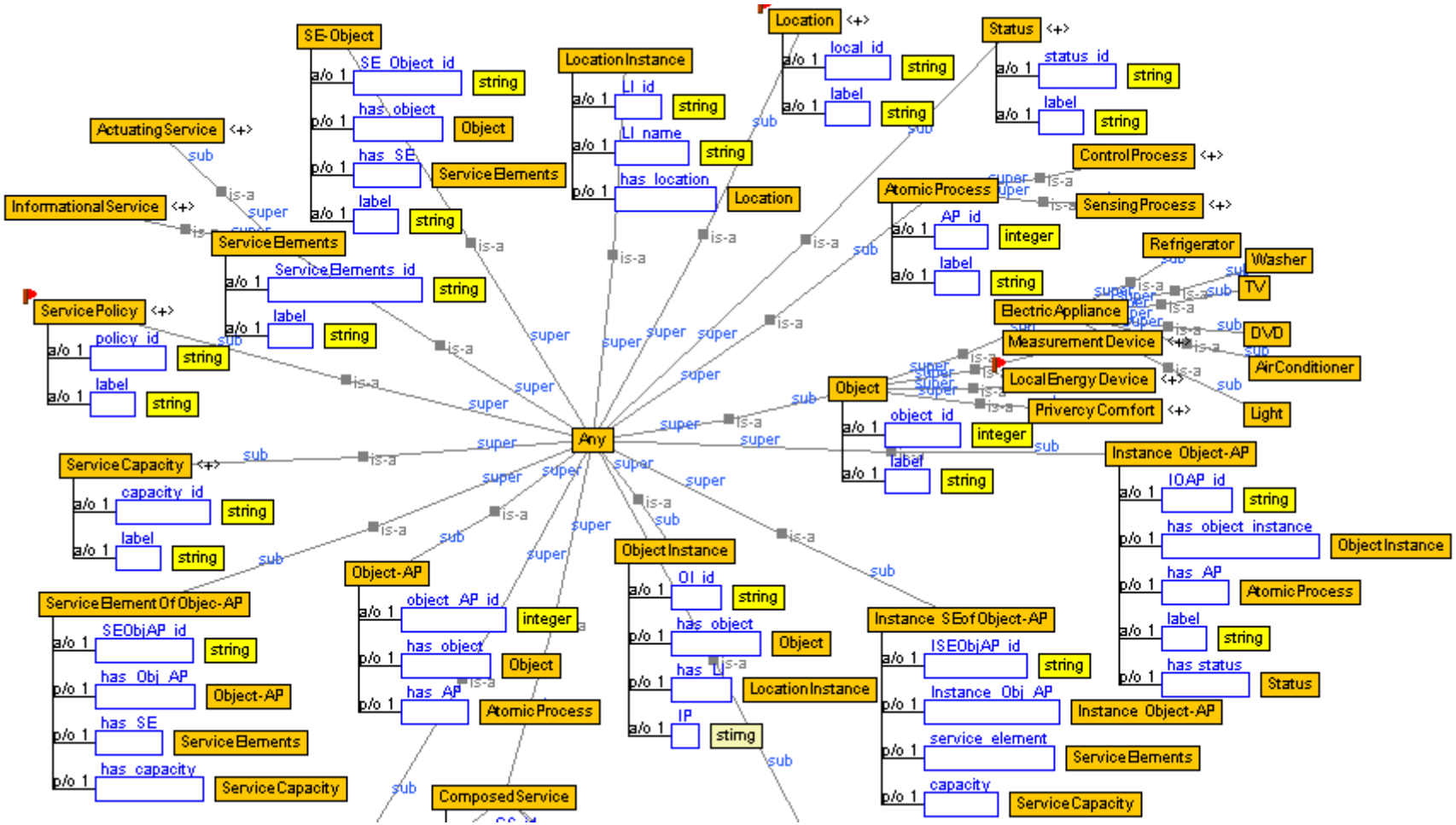


Figure 7.1: A service structure description ontology

- *ServiceElementofObject-AP* class: to describe a relationship between *Object-AP* class and *ServiceElement* class with a service capacity.
- *Instance Object-AP* and *Instance SEofObject-AP* class: to identify each instance device bases on *Object-AP* and *ServiceElementOfObject-AP* class as illustrated in figure 7.2.

Considering the relationship between classes, properties in our ontology can be divided into two principal properties: Object Property and Data Property. The Object Property describes a " part of " relationship between two classes, whereas the Data Property identifies an " attribute of " in each class. These properties are used for describing a relationship between classes. For example, the *ServiceElementOfObject-AP* class contains the *Object-AP* class as object property, and the *Object-AP* class contains the *Object* class as illustrated in Figure 7.2. Relationship among classes such as those explained above make a semantic connection between services and objects. For example, a "curtain" object has a functionality of "open-close" and such a pair provide "Privacy service", "Air control service" and "Illumination control service" as service element. To describe their relationship, (1) "curtain" has "open-close" function that is "Privacy service", (2) "curtain" has "open-close" function that is "Air control service", (3) "curtain" has "open-close" function that is "illumination control service". Likewise, the service model can describe multiple service elements for each functionality of the object by connecting between classes.

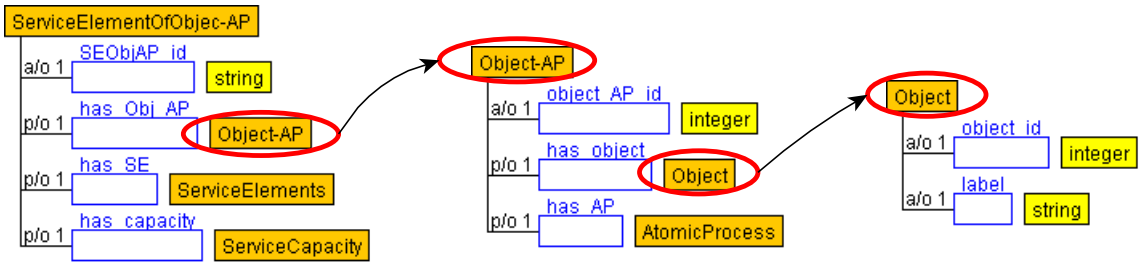


Figure 7.2: Defined relation between classes using OP

7.1.2 The data structure of a service instance

Since the service structure ontology can only support the concept of a relationship on defined classes, a service composition system needs a repository of a service description data.

A data structure modeling

To build the repository structure, we use enhanced entity-relationship (EER) model, a high-level of conceptual data model incorporating extensions to the original entity-relationship (ER) model, used in the design of databases.

According to Figure 7.3, a data base structure can be divide into two domain : service model and the Instances.

- **Service model domain** mainly contains *Object*, *AtomicProcess*, *ServiceElement* and their relations. The tables are connected by a primary key. These tables contain general relation data. For example, DVD generally has "media playback" and "media record" as *AtomicProcess* and it is related to "play movie", "record TVprogram" and "play music" *ServiceElement*s.

- **Instances domain** is for identified device instances with UUID, location, status and so on. Contained tables are connected with the service model domain as "one-to-one" relationship. When an instance is added into a repository, the Instance domain refers to the Service model domain which alreardy describes the base type of this instance.

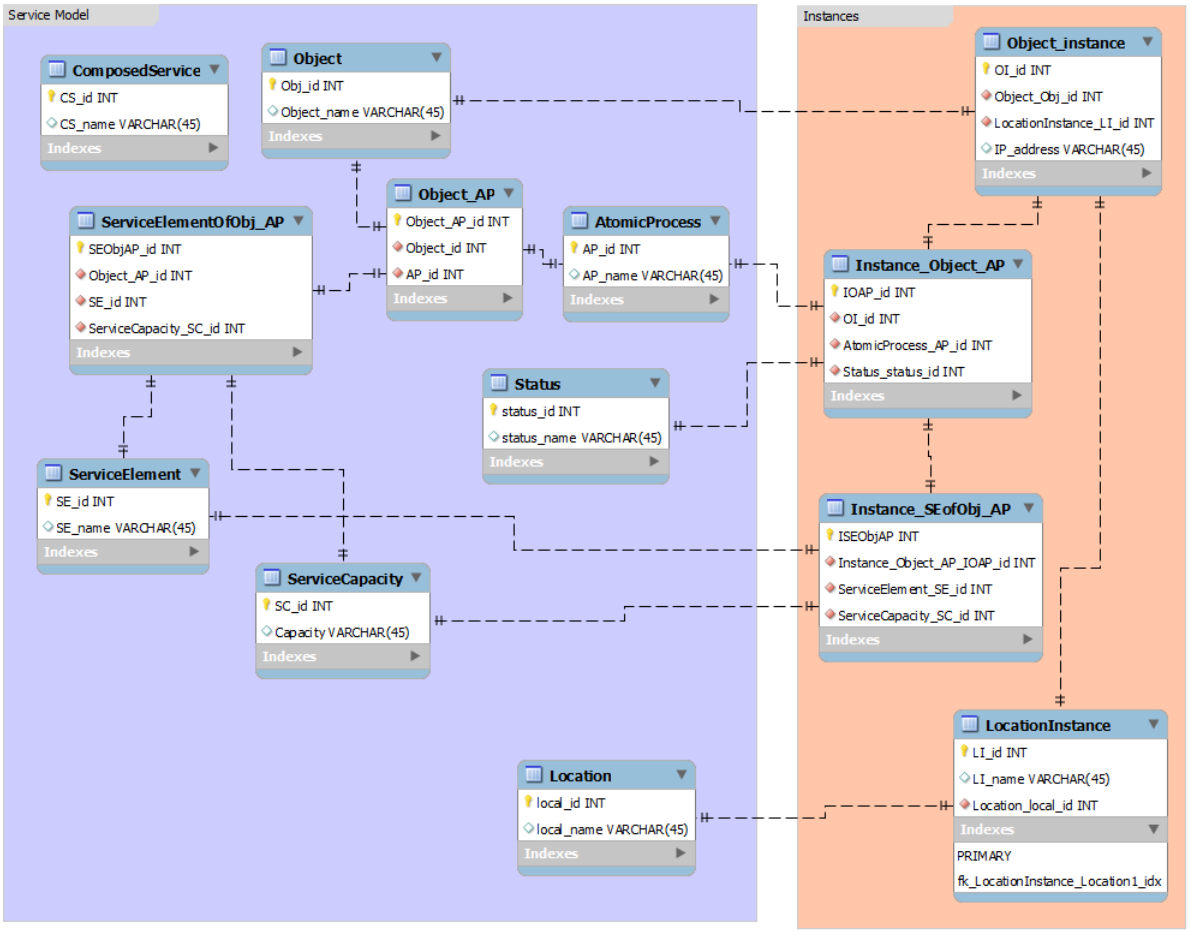


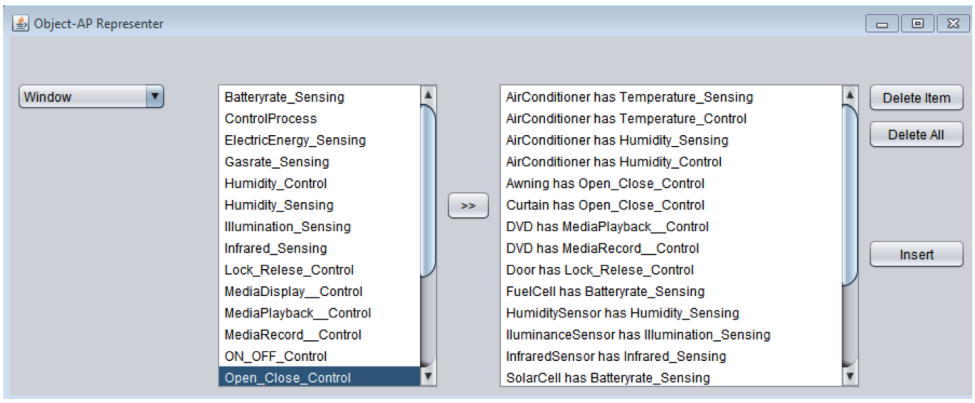
Figure 7.3: EER model of an instance data

A complexed relational data

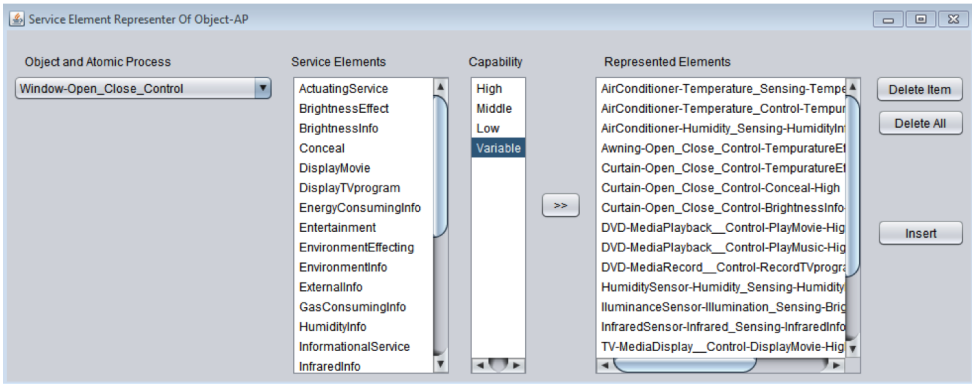
In Figure 7.3, a simple database table such as *Object*, *AtomicProcess* and *ServiceElement* is easy to store data by people directly. However, the relational tables, which are stored just primary key numbers such as *Object-AP* and *ServiceElementOfObj-AP*, are in a form inappropriate for direct use by humans. For example, to explain relation between an object

"DVD" and an atomic process "media playback", the table *Objtct-AP* stores primary key number "1" from a DVD's ID in *Object* table and primary key number "4" from the media playback's ID in *AtomicProcess* table. Moreover, in case of *ServiceElementofObject-AP*, the table get a primary key over in two step. Therefor, GUI based relation descriptor is needed.

As illustrated in Figure 7.4, the relational data between classes can be stored into the *Object-AP* table and *ServiceElmentofObj-AP* table by an Object-AP Representer (OAR) and an ServiceElement of Object-AP Representer (SOAR). OAR gets and shows data list from *Object* table and AtomicProcess table. A developer maps between them and stores mapped data into the Object-AP table. SOAR refers to the previously defined relation data from *Object-AP* table on a mapping with *ServieElement* table.



(a) An Object-AP Representer

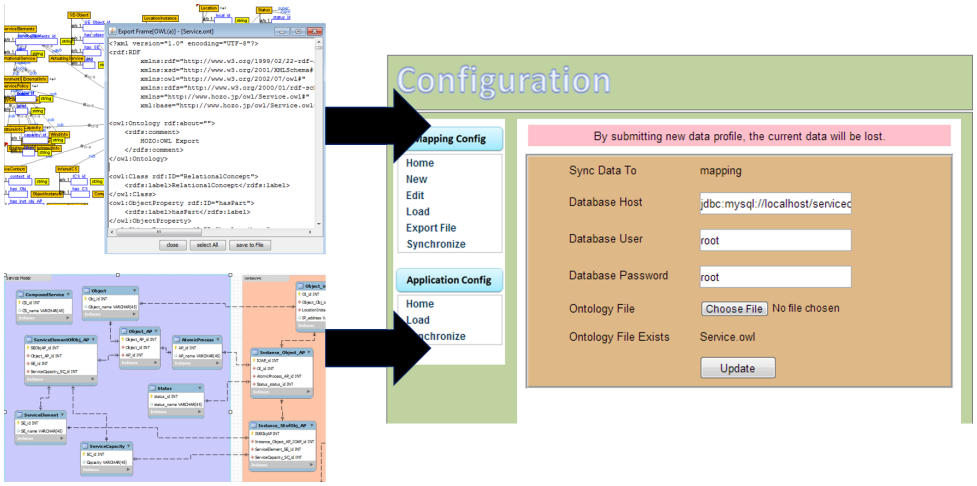


(b) Service Element of Object-AP Representer

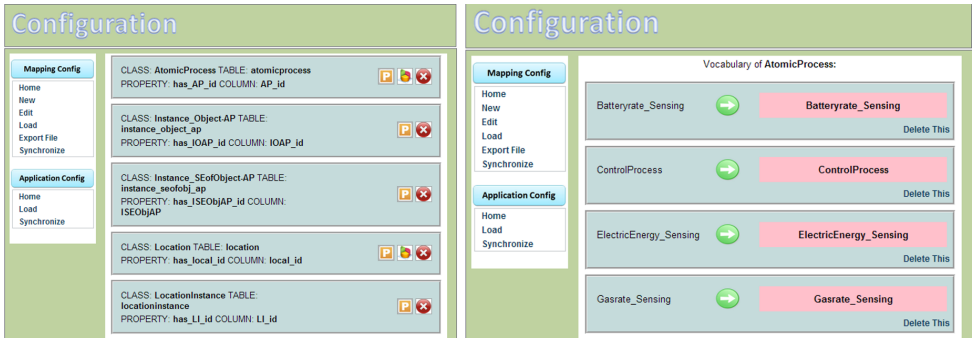
Figure 7.4: GUI based relation descriptor

7.1.3 A mapping between the service structure ontology and the Instance data structure

Due to their different nature, the ontology model and the relational database cannot be mapped directly. There are some steps for mapping the data. Firstly, the ontology model has to be exported to OWL, and then map the OWL with the database through the OAM framework[41]. According to the OAM framework, it provides the interface for configuration between the database and the OWL as shown in Figure 7.5(a). After that,



(a) OAM framework GUI base mapping configuration



(b) Mapping Property and Data (left), Mapping Vocabulary (right)

Figure 7.5: An Example of Mapping between a data and an ontology model

the relationship between the class in ontology and the table in database (Class-Table Mapping) is created as shown in left of Figure 7.5(b)

Next step is Property-Column Mapping. The properties of each class and columns of the database table are linked through the OAM interface. In case of the data property, it is linked to a data column such as an object name or an atomic process name. The object property is linked to a foreign key column which is specified to the connected table.

After mapping finish between properties and columns, then the OAM interface needs to map a vocabulary between subclasses of the class and the data of matched column. The right of Figure 7.5(b) shows an example of mapping the vocabularies. If the mapping vocabulary process is not conducted, the ontology searcher can not find the data even if it exists on the database.

After the Mapping Data process is finished, the system will generate the results as resource description framework (RDF), a standard model for data interchange on the Web, for easy use when applying to semantic web technology. The generated RDF file contains data from database as instances which means that the original database is no longer necessary for knowledge based service searching. In other words, the knowledge information can be used independently.

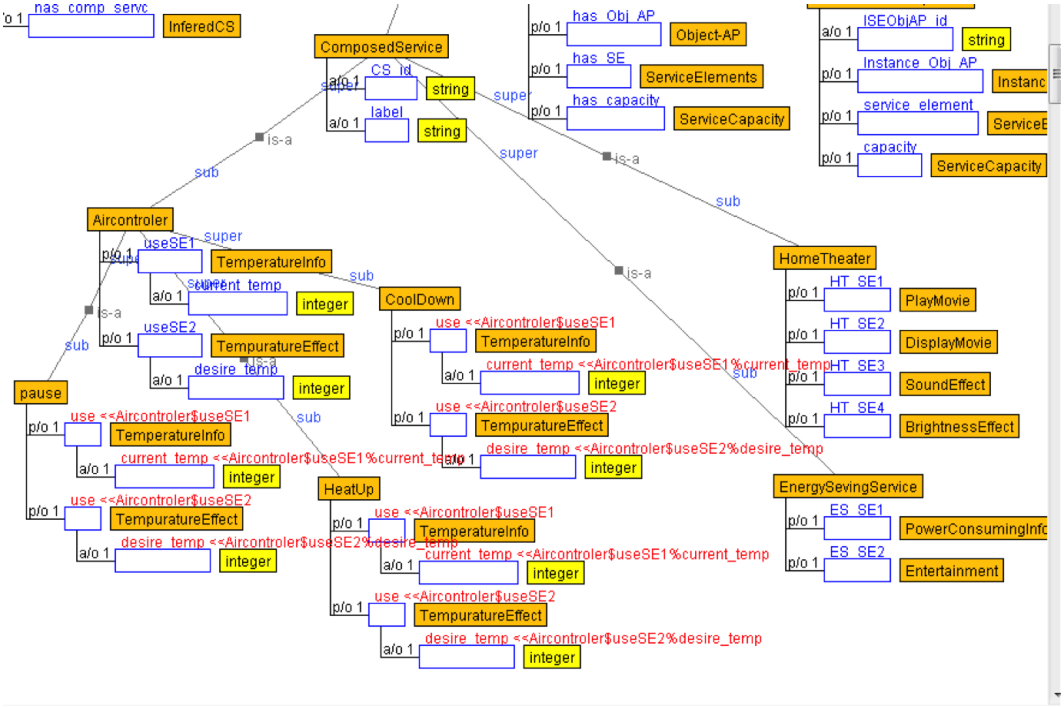


Figure 7.6: Prototype of composed service

7.2 A knowledge-base service search

In this section, the semantic service composition system (SSCS) will be verified on a validation of the integrated service composition.

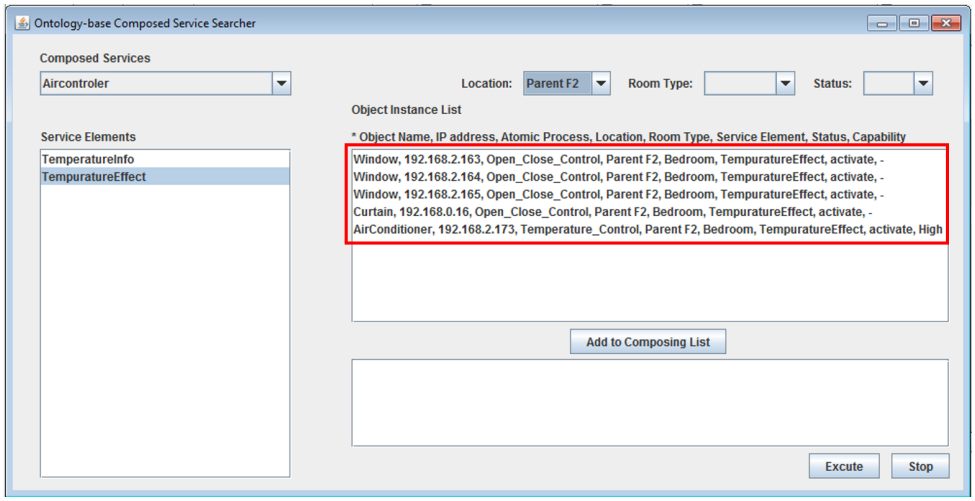
7.2.1 A composed service description

To evaluate our system, we created a simple service composition model as shown in Figure 7.6. A developer can make their own service which is composed by various service elements. The composed service will be located at the bottom of *ComposedService* class. In the same figure, the *ComposedService* class has three services, *Aircontroer*, *HomeTheater* and *EnergySavingService*. Each composed service contains necessary service elements and the described relationship of properties. As an instance, the *Aircontroler* class has *TemperatureEffect* class as the object property. This class is located at the bottom of *ServiceElement* class as subclass. Therefore, SSCS can infer the right device that should be used for a service composition.

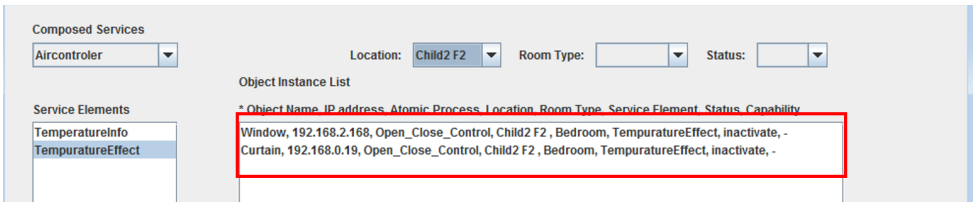
The composed service could be more specifically described by an ontology rule or a relation definition between properties. However, these description methods are executed from the experimental evaluation.

7.2.2 Service searching results

The Ontology-base Composed Service Searcher (OCSS) provides searching a device function (Atomic process) base on the SSCS. In Figure 7.7(a), the OCSS gets the composed service list from *ComposedService* class. On the Service Elements box, the system illus-



(a) A result of Aircontroler service (parent room)



(b) A result of Aircontroler service (child room)

Figure 7.7: The Ontology-base Composed Service Searcher

trates the object properties of selected service. When the Service Element is selected, the OCSS searches a device that contains a selected service element from an instance device. The searched devices are illustrated on the Object Instance List box as Object Name, IP address, Atomic Process, Location Instance, Room Type, Service Elements, Status and Capability. These devices are composed by adding to the composing list selectively.

The experiment uses existing devices in the iHouse (experimental smart house, explained in section 7.3.2) which uses the ECHONET Lite protocol. Although, some devices such as TV and DVD do not support the EHONET Lite protocol, for searching purposes, we assume that these devices do take part in the network.

Flexibility in the home environment

To realize the integrated service composition, the system has to provide possible devices in different home environments. In this experiment, we search the Temperature Effect element as the Air control service on two different home environments: the parent room which has window, curtain and air conditioner and the child room which has window and curtain. As illustrated in Figure 7.7, the OCSS obtained different device list as a result. In this result, it is highly probable that our system could flexibly provide a service composing for each different home environment.

Of course, the window and curtain could not match the desired temperature as well as the air conditioner. However, the service can be executed with them toward the desire temperature. This is concept resemble a human being that adapts itself to new circumstances.

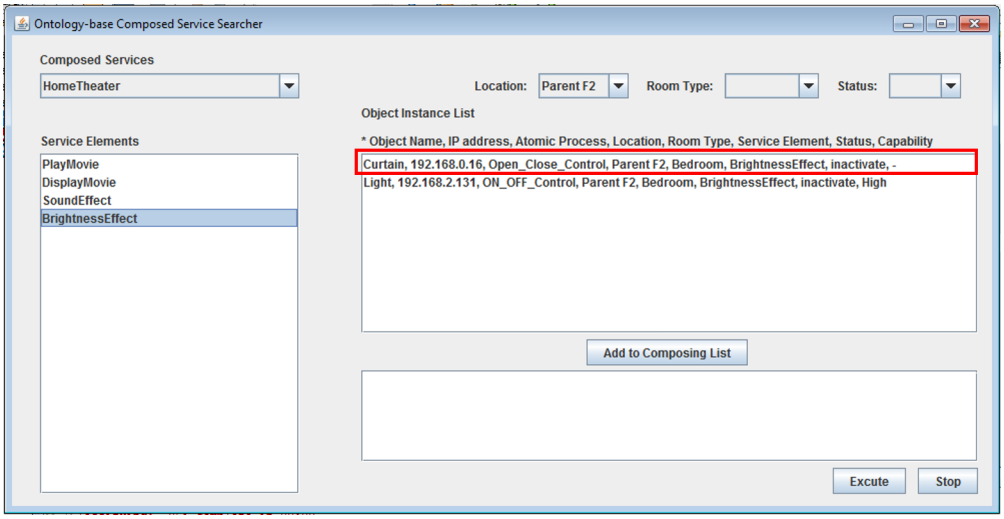


Figure 7.8: A result of Home Theater service

Selectivity of the service element

Deffernte possible selections of the service element give many options to the service composing system as to satisfy various priority.

In Figure 7.8, Object Instance list contains the Light and the curtain as the Brightness Effect element for Home Theater service. A noteworthy feature is that the curtain already has been chosen by Temperature Effect element as shown in Figure 7.7. the reason for this behavior is the curtain will be change a meaning according to the objective of a service.

In this regard, the OCSS fulfills the condition that the system has to provide variable service element according to the service objective.

Understanding of the service element

Let us consider an extreme scenario: a local blackout has occurred and the home system run on local battery and tries to conserve energy. The system will want to stop unnecessary elements for the living. At that moment, the legacy system has to regulate the devices one by one.

In case of our system, the OCSS provides semantic grouping of service elements. It is not just grouping devices. Figure 7.9 shows a good example of the semantic grouping. Energy Saving Service has Entertainment as a service element. In the service ontology, the Entertainment class has subclasses of DisplayMovie, RecordTVprogram, DsiplayTVprogram, PlayMovie and PlayMusic. These subclasses (service elements) will find matched device's function and control on the function level. If "TV" has the "emergency information" function, OCSS won't stop the function. In other words, a user can watch the emergency information but can't watch the movie.

Proceeding from what has been said above, it should be concluded that our system understands the semantics of a service element and it can adapt to the semantic context by controlling devices minutely.

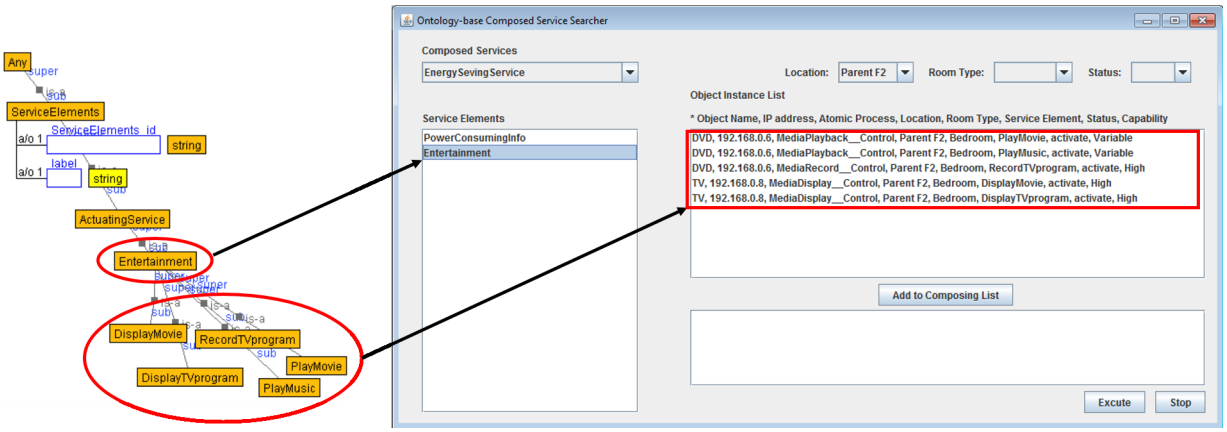


Figure 7.9: A result of Energy Saving service

7.3 Application

In this section, the validity of the system is demonstrated by executing three different service composition: which are obtained from the OCCS. The service compositions take into consideration the priority of a service element according to various policies. The obtained data will be analyzed according to the service objective.

7.3.1 Service composing scenario

The service scenario is divided into two different composition of a service element according to different priority, defined by a give policy. The first scenario is that an air control service will be required that can achieve the desired temperature quickly. On this policy, the system does not consider the efficiency of devices. Second scenario is to consider power consumption of devices on the air control service. In this case, the system tries to curb the power consumption of devices.

These two different scenarios towards achieving the desired temperature. However, in a semantic approach as mentioned above, the two services are going to use different devices as shown below.

- **Fast achievement policy** (Scenario 1)- uses an air conditioner as service element.
- **Energy saving policy** (Scenario 2) -uses a window as service element.

According to the fast achievement policy, the service composing system will find a air conditioner as the service element, since the air conditioner has best performance on cooling capability. However, considering the energy saving policy, the window will be selected despite the cooling performance being considerably worse.

Including the scenario 0, a situation of nothing executed for the service, assuming the legacy integrated service is ended in failure because of the trouble of fixed device, the experiment menu has three status as shown in table 7.1.

In order to verify of our research, these tree state was conducted on the experimental house. In addition, an environment change was observed during on each state.

State	Service element	
	Air Conditioner	Window
Scenario 0	inactive	inactive
Scenario 1	acitie	inactive
Scenario 2	inactive	active

Table 7.1: States based on scenarios

To stabilize a temperature in the iHouse, the experiment was stared in the afternoon. Since the weather might change extremely, the duration of each status was less than an hour.

Figure 7.10: A picture of the iHouse, an experimental smart house

7.3.2 An experimental environment

For this experiment, iHouse [42] was selected as an experimental smart home environment. The iHouse is designed for development of the next-generation home network system. Two floors with 107.76 m^2, more than 250 sensors and home appliance are connected through Energy Conservation and Home care Network (ECHONET), Universal Plug and Play (UPnP), and Zigbee. A photograph of iHouse is presented in Figure 7.10.

A structure of the experimental environment

The iHouse is established with various actuators and sensor nodes on ECHONET Lite protocol. As shown in Figure 7.11, each device in the iHouse connect with the ECHONET Lite middleware adapter through the their own protocol. The middleware adapter converts their lower-communication layer to a ECHONET lite object [43].

As mentioned in section 7.2.2, the SSCS uses existing devices in the iHouse. To map between them, *Object* is mapped with the ECHONET Object (EOJ), the *AtomicProcess* is mapped with the ECHONET Property (EPC) and the instance device ID is mapped with an IP address of ECHONET object.

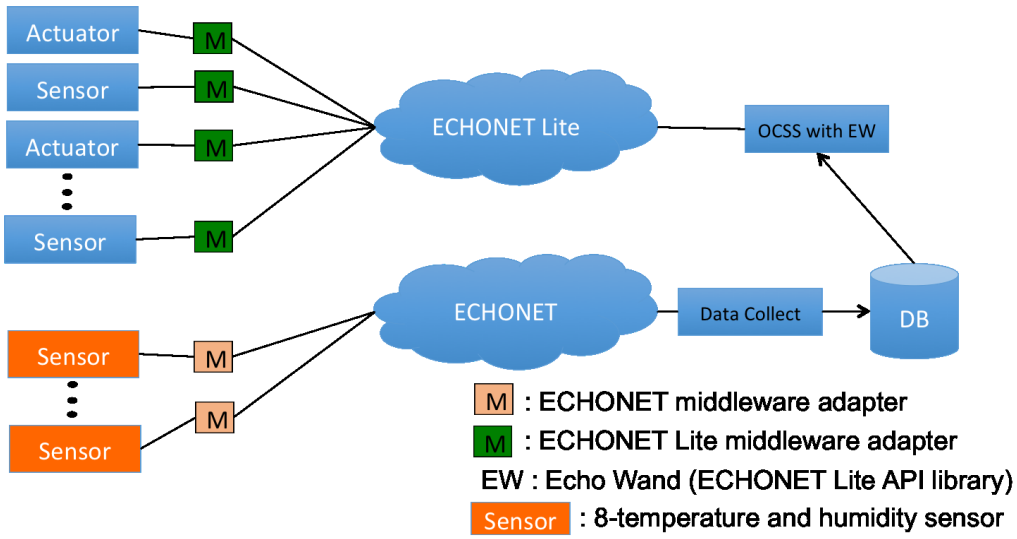


Figure 7.11: A network structure of iHouse

The OCSS obtains device information from nodes then control the devices with an ECHONET Lite library (Echowand)[44]. The Echowand library provides an API that supports getting object properties from devices and to set value into the EPC of the object.

To analyze the experiment, an accurate observation system is used. This system consists of 8 temperature and humidity sensors in each room. The system could measure not only overall room temperature and humidity but also the specified location temperature and humidity on the room.

Target location in the iHouse

The target location is the parent's room and children's room on the second floor (hereafter simply referred to as "parent room" and "chid room"). An area of the parent room is 20,495,475mm^2, child room is 13,249,600mm^2. As illustrated in Figure 7.12, each room has a window controller and an air conditioner as an actuator. The sensors used are temperature sensors, humidity sensors, a power consuming measurement sensor, an outdoor wind speed sensor and an outdoor temperature sensor. Each of the 8 temperature and humidity sensors is located in different corner of the room. The power consuming measurement sensor is attached to a circuit breaker of the windows and the air conditioner. The there different scenarios were conducted one by one in both the parent room and child room at the same time.

7.3.3 Experimental results

To consider the experimental process, firstly the scenario 1 was conducted for one hour from 12:00PM which is stabilized on a temperature. Secondly, the scenario 0 was executed for one hour continually from the scenario 0, since to observe a temperature change on the room. Finally, the scenario 2 was conducted after one hour from scenario 0 which means that the temperature got stable again.

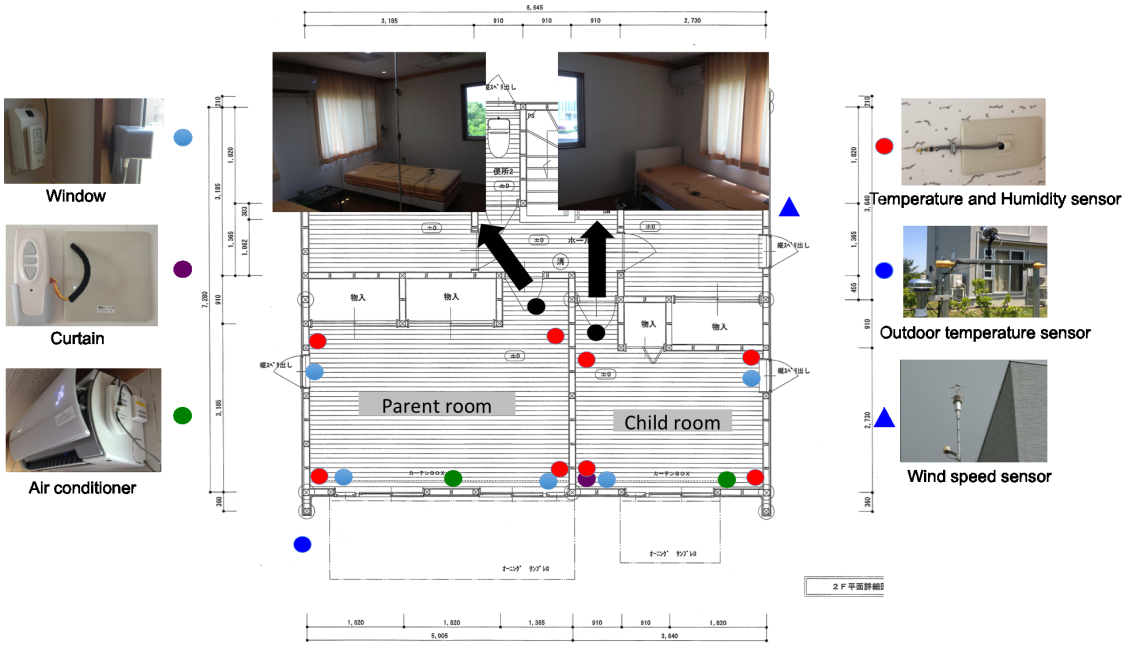


Figure 7.12: A target location for the experimental

In this experiment, the desired temperature of an air conditioner is set to 26 °C. During the experiment, environmental data from outdoor temperature, wind speed, power consumption, temperature and humidity from 8 points inside the room was collected for analisys.

The weather condition was a sunny day in early summer season. The average outdoor temperature was 27.6 °C. The outdoor wind speed was 1.55 m/s, wind speed of each room was 0.7 m/s for the parent room and 0.4 m/s for the child room. The outdoor humidity was 55 %.

Achievement based on the scenario

Figure 7.13 shows the result of each scenario. On the graph, left Y axis illustrates the power consumption of devices, right Y axis illustrates temperature, the X axis shows experimental time.

According to scenario 0, the inside temperature (room temperature) toward 31 °C with the time, even though the outdoor temperature was stable between 27 °C and 28 °C. This result will be used to compare the achievement of the remaining two scenarios.

Regarding scenario 1 in Figure 7.13(b), the inside temperature slowly decreases to the outdoor temperature. The air control service could not achieve the desire temperature. However, to compare with scenario 0, the service could approach to the desire temperature, Moreover, the service kept the priority which considers the energy saving policy, sine the electric power consumption is extremely low. In this perspective, the service is executed with suitable process on the policy.

Finally, regarding scenario 2, Figure 7.13(c) shows that the service quickly achieves the desire temperature based on the Fast achievement policy. In fact, there exist, a difference of temperature between the air conditioner's temperature sensor and measured

temperature, since the measured data is changed according to the location in the room. Meanwhile, this service composition is invalid considering the scenario 1 because a consumption of electrical power is over $250kWh$.

Evaluate on the thermal comport of the result

In the experiment of the scenario 1, the service seems to end in failure as get toward desire temperature. However, we need to consider a thermal comfort of a human body, Because the desire temperature is severely fixed, not considering other elements such as wind speed or humidity. According to ANSI/ASHRAE Standard 55, Thermal comfort is the condition of mind that expresses satisfaction with the thermal environment and is assessed by subjective evaluation[45][46].

To calculate the thermal comfort, there are six primary factors that directly affect thermal comfort that can be grouped in two categories: personal factors, because they are characteristics of the occupants, and environmental factors which are conditions of the thermal environment. The former are metabolic rate (MET) and clothing level, the latter are air temperature, radiant temperature, air speed and humidity.

In this evaluate, we assume that user reads a book (MET=1), puts on knee−length skit and long−sleeve shirt (Clothing level=0.67).

The Mean Radiant Temperature of an environment is defined as that uniform temperature of an imaginary black enclosure which would result in the same heat loss by radiation from the person as the actual enclosure. The equation for the calculation of Mean Radiant Temperature is:

$$\bar{t_r} = \sqrt[4]{\sum_n F_{p-i}(t_i + 273)^4} - 273 \tag{7.1}$$

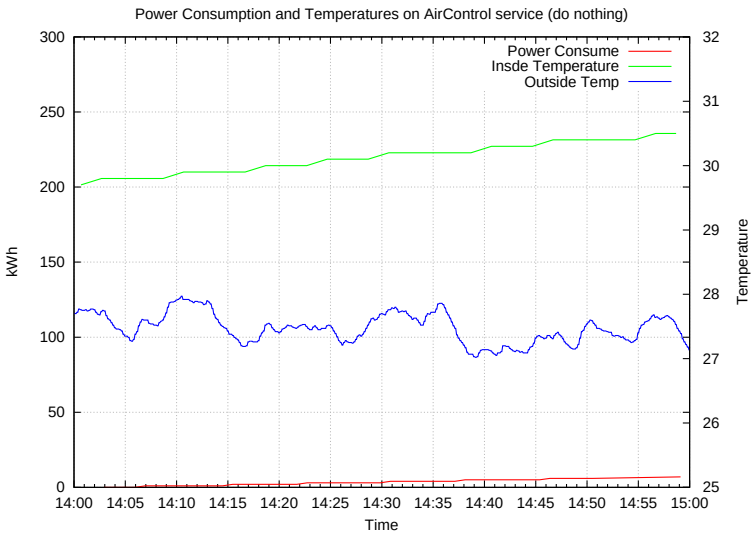
In the equation above, t_i is surface temperature of surface i (°C). F_{p-i} is an angle factor between person and surface i ($\sum_n F_{p-i}$=1)[47].

To obtain the surface temperature, we used the 8-temperature sensors which are located on each wall. these temperature sensors are evaluated by comparing the temperature data between the sensors and the thermo-graphic camera as illustrated in Figure 7.14. An observational error between the sensors and the thermo-graphic camera was ±0.3 °C.

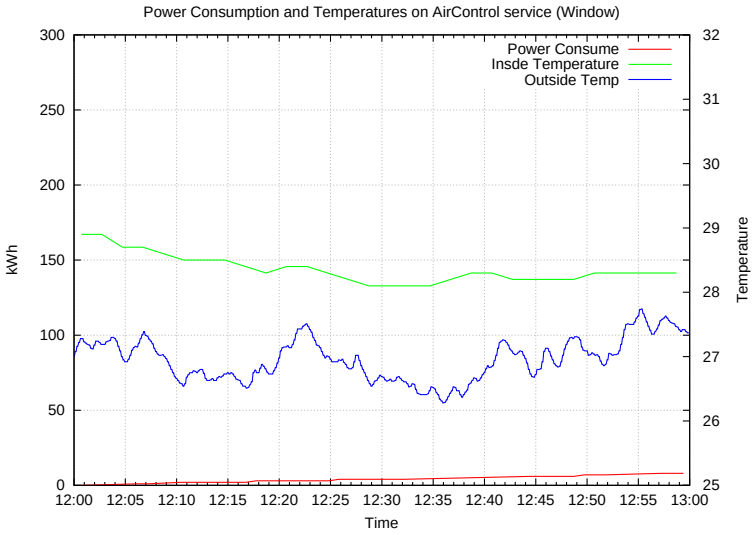
Factor	Scenario		
	Scenario 0	Scenario 1	Scenario 2
Air temperature [°C]	29.2	27.5	26.8
MRT [°C]	28.3	27.3	27
Air speed $[m/s]$	0.4	0.4	0.4
Humidity [%]	55	55	55
Metabolic rate $[met]$	1	1	1
Clothing level $[clo]$	0.67	0.67	0.67

Table 7.2: The measured six factors

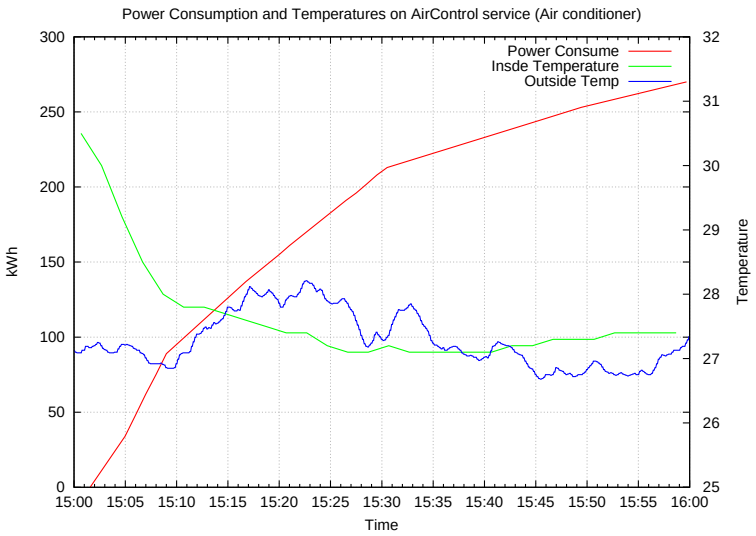
The measured factors are shown in table 7.2. The Air temperature, Wind speed and Humidity are mean value of measured data in each conducting time.



(a) The result of scenario 0



(b) The result of scenario 1



(c) The result of scenario 2

Figure 7.13: The result of experiment (Parent room)

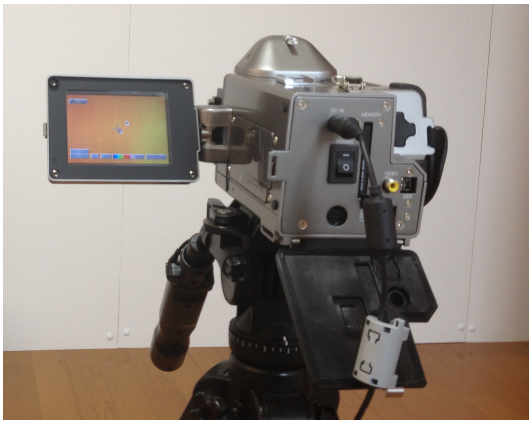
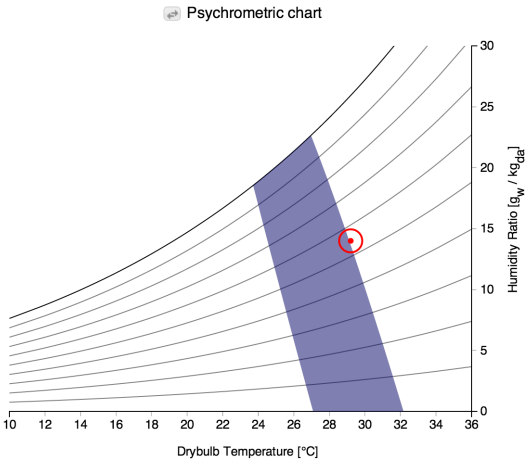


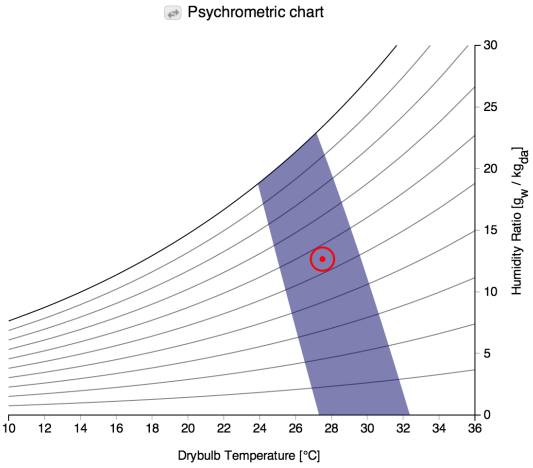
Figure 7.14: Measuring the wall temperature

Thermal Comfort Tool for ASHRAE-55 is used for calculate thermal comfort. In order to explain the graph as shown in Figure 7.15, X axis illustrates Dybulb Temperature and Y axis illustrates Humidity ratio. The blue area illustrates the thermal comfort zone.

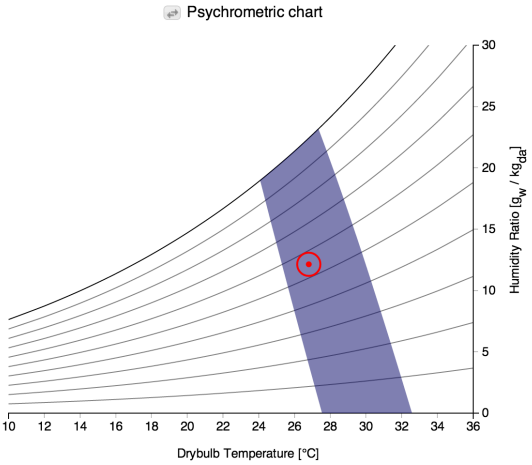
the scenario 1 is in the thermal comfort area different form the scenario 0. The result shows that the used device might be changed according to a macro−environment. On this issue, it is highly probable that the service element needs to have more options regarding the objective of policy.



(a) The scenario 0



(b) The scenario 1



(c) The scenario 2

Figure 7.15: Thermal comfort on scenarios (Parent room)

7.4 Summary

In this chapter, we presented three aspects of our system. Firstly, an implementation of Semantic Service Composition System was conducted that provides the ontology model and description tool of service element on complexed relational data.

Secondly, an evaluation of the knowledge-base service search performed with the prototype service composing model. The system were able to:

1. flexibly adapt to different home environments.
2. successfully select appropriate Service Elements given a policy
3. understand the semantic context and use the individual device properties appropriately.

Finally, the power of the semantic service composition system is demonstrated with several scenarios. These results bespoke a basis that our system contributes various service environment which can dynamically changed according to a given policy.

Chapter 8

Discussion

In this thesis, we provided a dynamic service composition environment on integrated home network system. The distributed development of a service and policy model will improve scalability of the service model. Moreover, it also improves a variability of a service priority according to a policy. In the near future, a distributed environment for developing home services is going to be needed by service provider. We believe that our system can provide a platform to apply a service that was made by an outside service provider into the home network environment smoothly. Especially, As we discussed about SI model in section 2.3, SI model needs to compose external services from SPs.

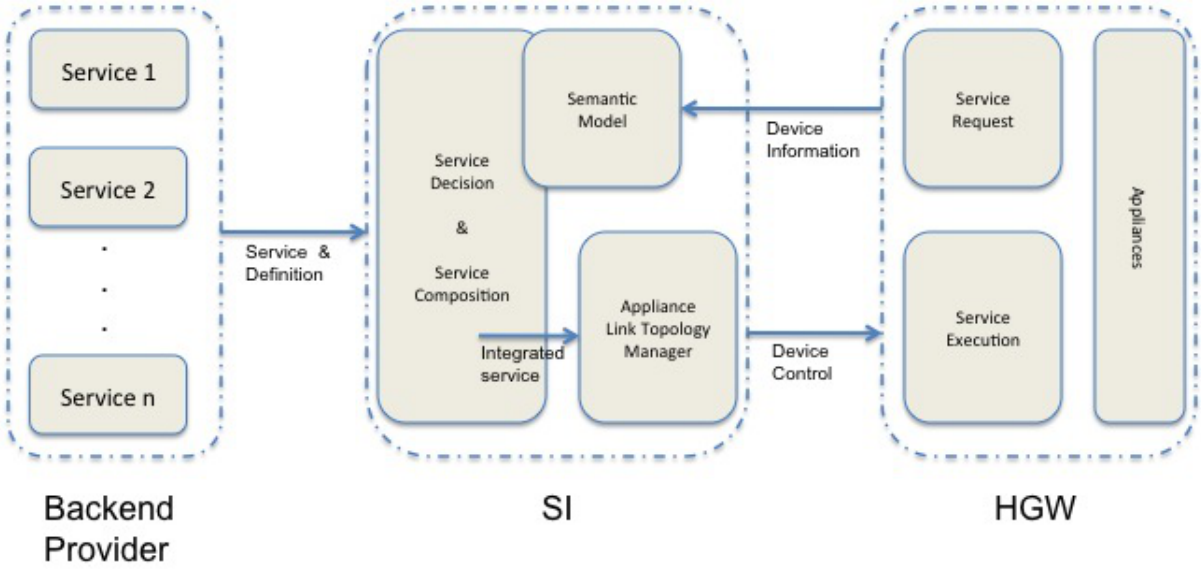


Figure 8.1: A service structure of SI

As we can see figure 8.1, the SI have to recognize ”what the service do ”. For example When the SI searching a service element for composing a home theater service, the SI should semantically know which service could be the AV controller, the temperature controller and the illuminate controller. Moreover, these services should be identified ” who made the service ”. There are many services that are provided as same functionality. the SI needs to recognize where such services are made from. Moreover, since the SI need to map every services for understanding a relation between each service and providing

integrated service to the HGW, services from the SP should be combinable with other services by the SI.

in chapter 6, we proposed the adaptive link topology to flexibly composite the appliances on the networked home environment for providing integrated service. However, there is problem to realize such system since industries do not consider about standardization of connection instrument quality. For that, manufacturers need Co-operating with the problem.

Chapter 9

Conclusion

The goal of this thesis is a dynamic service composition environment on integrated home network system. To achieve this goal we investigated the following fourth research target:

- The semantic service representation
- The policy based dynamic service priority
- The distributed developing environment
- The Adaptive link topology of an appliance

Firstly, our system provides a flexible service composition in various home environments by making the system to semantically understand an objective of a service. Moreover, it is possible to realize extendable services and priorities, since these semantic descriptions can be built separately in their own area by a name space.

Secondly, the dynamic service priority makes the service manager determines various priority of a service element according to a required policy because various external policy definitions are providing a valuation of elements.

Thirdly, we proposed the distributed service developing environment. the system is able to include an external ontology model by importing its name space. Such distributed development of a service model will improve scalability of the service model. In the near future, a distributed environment for developing home services is going to be needed by service provider

Finally, the system provides a methodology on how to select the best link to integrated service manager in various home environment since the system can determines a transmitting route of contents suitably among networked home devices including legacy appliances.

In addition, this thesis realizes the service composition system and demonstrate the relation between the model and the instance devices. After then the experimentation and evaluation, with composed services on networked appliances in the experimental house, were conducted. The experiment presented three aspects of our systemz: 1)an implementation of Semantic Service Composition System was conducted that provides the ontology model and description tool of service element on complexed relational data. 2)an evaluation of the knowledge-base service search performed with the prototype service composing model. The system were able to:

1. flexibly adapt to different home environments.

2. successfully select appropriate Service Elements given a policy

3. understand the semantic context and use the individual device properties appropriately.

3)the power of the semantic service composition system is demonstrated with several scenarios. These results bespoke a basis that our system contributes various service environment which can dynamically changed according to a given policy.

We expect that above features are provide to realize a dynamic service composition environment on integrated HNS.

In our future work we also need to address the requirements for future home appliances in terms of communications. As described in the previous paragraph a middleware solution is being developed that will enable multiple communication protocols, including different home networking platforms to be seamlessly interconnected. Furthermore we also need to quantify the cost of invoking a service and the time it takes to compose and invoke a composition of services including the overall scalability of our system.

Bibliography

[1] Dutta-Roy, A., (December) Networks for Homes IEEE spectrum 36(12):26-33.

[2] Miller, B., Nixon, T., Tai, C., and Wood, MD., Home networking with universal plug and play. IEEE Commun Mag 39(12):104-109.

[3] Microsoft. UPnP Forum Web site, Microsoft Corporation, http://www.upnp.org/ [accessed: Mar 22, 2011]

[4] DLNA. digital living network alliances web site, DLNA overview and vision whitepaper 2007, http://www.dlna.org/ [accessed: Mar 22, 2011]

[5] Marples, D. and Kriens, P., (2001,December) The open services gateway initiative: an introductory overview. IEEE Commun Mag 39(12):110-114.

[6] ECHONET. web site ECHONET Specification Ver2.11 (English), http://www.echonet.gr.jp/ [accessed: Mar 22, 2011].

[7] OWL. web ontology language web site, OWL Web Ontology Language Overview, http://www.w3.org/TR/owl-features/ [accessed: Mar 22, 2011]

[8] OWL-S. Semantic Markup for Web Services, a white paper describing the key elements of OWL-S, http://www.ai.sri.com/daml/services/owl-s/1.2/overview/ [accessed: Mar 22, 2011]

[9] Kivela, A. and Hyvonen, E.,Ontological Theories for the Semantic Web, (Hyvonen, 2002).

[10] Smith, B. and Welty, C., FOIS introduction: Ontology?towards a new synthesis, Proceedings of the International Conference on Formal Ontology in Information Systems, 3-9, Ogunquit, Maine, USA. ACM 2001.

[11] Borst, W.,Construction of Engineering Ontologies, PhD thesis, University of Tweente, Enschede, NL?Centre for Telematica and Information Technology, 1997.

[12] Studer, R., Benjamins, V., and Fensel, D., Knowledge Engineering: Principles and Methods, Data and Knowledge Engineering, 25(1-2):161-197, 1998.

[13] Guarino, N., Formal Ontology and Information Systems, In Guarino, N., editor, Formal Ontology in Information Systems, Proceedings of FOIS?f98, 3-15, IOS, Amsterdam, Trento, Italy, 1998.

[14] McGuinness, D., Ontologies Come of Age, In Fensel, D., Hendler, J., Lieberman, H., and Wahlster, W., editors, Spinning the Semantic Web: bringing the World Wide Web to Its Full Potential. MIT, 2002.

[15] McBride, B., The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS, In (Staab and Studer, 2004), 51-66, 2004.

[16] Horrocks, I., Patel-Schneider, P., and van Harmelen, F., From SHIQ and RDF to OWL: The Making of a Web Ontology Language, Journal of Web Semantics, 1(1):7-26, 2003.

[17] M. Kolberg, E. H. Magill, and M. Wilson, ?gCompatibility issues between services supporting networked appliances?h, IEEE Communications Magazine, vol. 41, no. 11, Nov 2003 pp. 136-147

[18] Pattara Leelaprute, ?hResolution of Feature Interactions in Integrated Services of Home Network System?h, Proceedings of Asia-Pacific Conference on Communications, 2007

[19] “次世代ホームネットワークが描く新たな価値進化時代へ向けた挑戦 ’’, pp.58-60, 次世代 IP ネットワーク推進フォーラムホームネットワーク WG.

[20] Y. Kiyoumi, ?hA task allocation method considering resource availability for providing services to Home Network Environment?h, Master thesis, 2010

[21] Kolberg, M., Magill, E. H., and Wilson, M. (2003). Compatibility issues between services supporting networked appliances. IEEE Communications Magazine, 41(11), 136-147.

[22] Geer, D. (2006). Nanotechnology: The growing impact of shrinking computers. IEEE Pervasive Computing, 5(1), 7-11.

[23] Chakraborty, D. and Joshi, A. Dynamic Service Composition: State-of-the-Art and Research Directions,Technical Report TR-CS-01-19, Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, Baltimore, USA, 2001.

[24] Business Process Execution Language for Web Services Version 1.1, [online] http://www.ibm.com/developerworks/library/specification/ws-bpel/ [accessed: May 22, 2011]

[25] Web Service Choreography Interface (WSCI) 1.0, [online] http://www.w3.org/TR/wsci/ [accessed: May 22, 2011]

[26] Casati, F., Ilnicki, s., Jin, L., Krishnamoorthy, V and Shan, M., Adaptive and dynamic service composition in eFlow, In Proc. of the Int. Conference on Advanced Information Systems Engineering(CAiSE), Stockholm, Sweden, 2000.

[27] Ponnekanti, S. R. and Fox, A. SWORD: A Developer Toolkit for Web Service Composition, to appear in The Eleventh World Wide Web Conference (Web Engineering Track), Honolulu, Hawaii, May 7-11, 2002.

[28] Mennie, D. and Pagurek, B., An Architecture to Support Dynamic Composition of Service Components, Proceedings of the 5th International Workshop on Component-Oriented Programming(WCOP 2000), Sophia Antipolis, France, 2000.

[29] Chandrasekaran, S., Madden, S. and Ionescu, M., Ninja Paths: An Architecture for Composing Services over Wide Area Networks, CS262 class project write up, UC Berkeley, 2000.

[30] Fujii, K. and Suda, T., Dynamic Service Composition Using Semantic Information, ICSOC'04, November 15-19, New York, USA, 2004.

[31] Aumueller, D., Do, H., Massmann, S., Rahm, E., Schema and ontology matching with COMA++, Proc. of the 2005 International Conference on Management of Data, pp. 906-908, 2005.

[32] Carlo A, C., Giorgio, O. and Letizia, T., (2007), X-SOM: A Flexible Ontology Mapper, International Workshop on Semantic Web Architectures for Enterprises (SWAE?f07) in conjunction with the 18th International Conference on Database and Expert Systems Applications (DEXA?f07).

[33] He, B. and Chang, K., Statistical schema matching across web query interfaces, In SIGMOD'03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data, pages 217-228, New York, USA, 2003, ACM.

[34] Doan, A., Madhavan, J., Dhanmankar, R., Domingos, P. and Halevy, A., Learning to match ontologies on the semantic web, The VLDB Journal, 12(4):303-319, 2003.

[35] J. Tang, J. Li, B. Liang, X. Huang, Y. Li, and K. Wang., Using bayesian decision for ontology mapping, Web Semant., 4(4):243-262, 2006.

[36] J. Tang, B.-Y. Liang, and J.-Z. Li., Toward detecting mapping strategies for ontology interoperability, In WWW 2008, 2005.

[37] Ehrig, M., Staab, S. and Sure, Y., Supervised learning of an ontology alignment process., In In KMTools workshop at Konferenz Professionelles Wissensmanagement, 2005.

[38] M. Sioutis., Area of effect and compromising techniques for the detection and resolution of environmental conflicts between services in the Home Network System, Master thesis, 2011.

[39] Fulvio, M., Antonio, S., and Renato, Z., Understanding Events Relationally and Temporally Related: Contest Assessment Strategies for a smart Home ISUC 2008 IEEE.

[40] Kozaki, K., Kitamura, Y., Ikeda, M., Mizoguchi, R.: Hozo: An environment for building/using ontologies based on a fundamental consideration of role and relationship. In: Proceeding of 13th International Conference Knowledge Engineering and Knowledge Management. pp. 213-218, Siguenza (2002).

[41] Buranarach, M., Thein, Y., Supnithi, T.: A community-driven approach to development of an ontology-based application management framework. In: The 2nd Joint International Semantic Technology Conference (JIST), Nara (Dec. 2012)

[42] Tan, Y.: Home Network Technologies for Smart Houses. Impress R&D (2011) (in Japanese).

[43] ECHONET Lite, ECHONET Lite Specification Version 1.01 (English), http://www.echonet.gr.jp/english/spec/spec_v101_lite_e.htm [accessed: June 12, 2013].

[44] ECHOWAND, ECHONET Lite Library for JAVA, https://github.com/ymakino/echowand [accessed: July 01, 2013]

[45] Richard J. de Dear Gail, S. Brager,Thermal comfort in naturally ventilated buildings: revisions to ASHRAE Standard 55, Energy and Buildings 34 (2002) 549??561.

[46] ASHRAE Standard 55 thermal environmental conditions for human occupancy, ASHRAE Inc., 1992, Atlanta.

[47] INNOVA, AirTech Instruments, HVAC Handbook - Thermal Comfort by INNOVA, December 20, 2002.

Publications

[1] –岡田 崇, 牧野 義樹, キム ジュンスー, 中田 潤也, 丹 康雄, “ 住宅におけるエネルギーマネジメントの効果を検証する実証的ホームシミュレータの提案と実装 ”, 情報処理学会論文誌 53(1), January 2012, pp. 365-378.

[2] –Marios Sioutis, Takashi Okada, Junsoo Kim, Azman Osman Lim, Yasuo Tan “ On the Use of Sensors for the Detection and Resolution of Conflicts Between Services in the Home Network Environment ”, IEICE technical report 111(134), pp. 37-42, 2011-07-07.

[3] –金　準修, 中田　潤也, 岡田　崇, Marios SIOUTIS, LIM Azman Osman, 丹　康雄, “ホームネットワークにおける意味的推論を利用した動的なサービス構成”, 信学技報, Vol.110, No.289, IN2010-83, pp. 13-17, 2010 年 11 月.

[4] –SIOUTIS Marios,　岡田　崇 , 中田　潤也, KIM Junsoo, LIM Azman Osman, 丹　康雄, “影響範囲と妥協手法によるホームネットワークサービスの可用性の向上に関する研究”, 信学技報, Vol.110, No.289, IN2010-82, pp. 7-11, 2010 年 11 月.

[5] –岡田 崇, 中田 潤也, 金 準修, 丹 康雄, “実世界指向ホームネットワーク開発環境 ”, 電気情報通信学会 2010 総合大会, 仙台, 2010.

[6] –岡田 崇, 中田 潤也, 牧野義樹, 金 準修, Sioutis Marios, 丹 康雄, “ホームネットワークにおける人間行動シミュレータ ”, 電気情報通信学会 2010 ソサイエティ大会, 大阪, 2010.

[7] –清海 佑太, 金 準修, 岡田 崇, 中田 潤也, 丹 康雄, “ホームネットワークにおける資源を考慮したタスク配置手法に関する研究”, 信学技報, vol. 109, no. 438, IA2009-102, pp. 101-106, 2010 年 3 月.

[8] –金 準修, 今井　智大, 中田　潤也, 丹　康雄, “ホームネットワークにおける柔軟な機器利用を考慮した機器接続構成アルゴリズム”, 学術刊行物　情処研報, Vol.2009, No.17, 2009-UBI-21, pp. 9-14, 2009.3.

[9] ◎ Junsoo Kim, Junya Nakata, Takashi Okada, Sioutis Marios, LIM Azman Osman, and Yasuo Tan, “A Composition Algorithm of Appliances for Flexible Integrated Service on Home Network Systems”, (UBICOMM 2009) IEEE, ISBN 978-1-4244-5083-1, 11-16 October 2009 Sliema, Malta.

[10] ◎ Takashi Okada, Marios Sioutis, Junsoo Kim, Junya Nakata, Yasuo Tan, Yoichi Shinoda: A Component-Based Simulation Environment for Large-Scale Simulation of Home Network Systems. TRIDENTCOM 2010: 626-628

[11] ◎ Junsoo Kim, Junya Nakata, Okada Takashi, and Yasuo Tan, “A Dynamic Service Composition using Semantic Ratiocination on Integrated Home Network System ”, ICAS2011, ISBN: 978-1-61208-006-2, pp.170-174, May 22-27, 2011, Venice/Mestre, Italy.

[12] ◎ Sioutis, M.; Junsoo Kim; Lim, A.-O.; Yasuo Tan, ”A home service deployment platform with support for detection and resolution of physical resource conflicts,” Consumer Electronics (GCCE), 2012 IEEE 1st Global Conference on , vol., no., pp.333,336, 2-5 Oct. 2012

[13] ◎ Konlakorn Wongpatikaseree, Junsoo Kim, Yoshiki Makino, Azman Osman Lim, Yasuo Tan: Architecture for Organizing Context-Aware Data in Smart Home for Activity Recognition System. HCI (25) 2013: 173-182

[14] -北陸先端科学技術大学院大学, “ホームネットワーク向けサービス提供基盤のスケーラビリティ検証”, シミュレーション評価分析, 技術サービス報告書.

Appendices

Appendix A

OWL modelings

A.1 An ontology model of service and policy

```
<?xml version="1.0"?>


<!DOCTYPE Ontology [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY xml "http://www.w3.org/XML/1998/namespace" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<Ontology xmlns="http://www.w3.org/2002/07/owl#"
     xml:base="http://www.semanticweb.org/ontologies/2010/9/HomeService101011.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     ontologyIRI="http://www.semanticweb.org/ontologies/2010/9/HomeService101011.owl">
    <Prefix name="xsd" IRI="http://www.w3.org/2001/XMLSchema#"/>
    <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="rdf" IRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
    <Prefix name="rdfs" IRI="http://www.w3.org/2000/01/rdf-schema#"/>
    <Import>http://www.semanticweb.org/ontologies/2010/11/response.owl</Import>
    <Declaration>
        <Class IRI="#Aircon"/>
    </Declaration>
    <Declaration>
        <Class IRI="#CoolingPossibility"/>
    </Declaration>
    <Declaration>
        <Class IRI="#ElectricFan"/>
    </Declaration>
    <Declaration>
```

```
35          <Class IRI="#ElectricRadiater"/>
        </Declaration>
37      <Declaration>
            <Class IRI="#HeatingPossibility"/>
39      </Declaration>
        <Declaration>
41          <Class IRI="#HomeService"/>
        </Declaration>
43      <Declaration>
            <Class IRI="#PowerConsume"/>
45      </Declaration>
        <Declaration>
47          <Class IRI="#TempControl"/>
        </Declaration>
49      <Declaration>
            <Class IRI="#TempControlDescriptor"/>
51      </Declaration>
        <Declaration>
53          <Class IRI="#TempPossibility"/>
        </Declaration>
55      <Declaration>
            <Class IRI="#Window"/>
57      </Declaration>
        <Declaration>
59          <ObjectProperty IRI="#hasCoolingPossibility"/>
        </Declaration>
61      <Declaration>
            <ObjectProperty IRI="#hasHeatingPossibility"/>
63      </Declaration>
        <Declaration>
65          <ObjectProperty IRI="#hasPowerConsume"/>
        </Declaration>
67      <Declaration>
            <ObjectProperty IRI="#hasTempControlDescriptor"/>
69      </Declaration>
        <Declaration>
71          <NamedIndividual IRI="#CoolingDependOn"/>
        </Declaration>
73      <Declaration>
            <NamedIndividual IRI="#CoolingHighly"/>
75      </Declaration>
        <Declaration>
77          <NamedIndividual IRI="#CoolingLowly"/>
        </Declaration>
79      <Declaration>
            <NamedIndividual IRI="#HeatingDependOn"/>
81      </Declaration>
        <Declaration>
83          <NamedIndividual IRI="#HeatingHighly"/>
        </Declaration>
85      <Declaration>
            <NamedIndividual IRI="#HeatingLowly"/>
87      </Declaration>
        <Declaration>
89          <NamedIndividual IRI="#HighConsumtion"/>
        </Declaration>
91      <Declaration>
```

```
        <NamedIndividual IRI="#LowConsumtion"/>
    </Declaration>
    <Declaration>
        <NamedIndividual IRI="#MiddleConsumtion"/>
    </Declaration>
    <Declaration>
        <NamedIndividual IRI="#NoneConsumtion"/>
    </Declaration>
    <Declaration>
        <NamedIndividual IRI="#TEST"/>
    </Declaration>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <Class IRI="#TempControl"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <ObjectHasValue>
            <ObjectProperty IRI="http://www.semanticweb.org/ontologies↩
                /2010/11/response.owl#hasResponse"/>
            <NamedIndividual IRI="http://www.semanticweb.org/ontologies↩
                /2010/11/response.owl#Fast"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasCoolingPossibility"/>
            <NamedIndividual IRI="#CoolingHighly"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasHeatingPossibility"/>
            <NamedIndividual IRI="#HeatingHighly"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasPowerConsume"/>
            <NamedIndividual IRI="#HighConsumtion"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#CoolingPossibility"/>
        <Class IRI="#TempPossibility"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ElectricFan"/>
        <Class IRI="#TempControl"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ElectricFan"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasCoolingPossibility"/>
```

```
            <NamedIndividual IRI="#CoolingLowly"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ElectricFan"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasPowerConsume"/>
            <NamedIndividual IRI="#LowConsumtion"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ElectricFan"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasPowerConsume"/>
            <NamedIndividual IRI="#MiddleConsumtion"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ElectricRadiater"/>
        <Class IRI="#TempControl"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ElectricRadiater"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasHeatingPossibility"/>
            <NamedIndividual IRI="#HeatingLowly"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#ElectricRadiater"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasPowerConsume"/>
            <NamedIndividual IRI="#HighConsumtion"/>
        </ObjectHasValue>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#HeatingPossibility"/>
        <Class IRI="#TempPossibility"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#TempControl"/>
        <Class IRI="#HomeService"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#TempPossibility"/>
        <Class IRI="#TempControlDescriptor"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Window"/>
        <Class IRI="#TempControl"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Window"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasCoolingPossibility"/>
            <NamedIndividual IRI="#CoolingDependOn"/>
        </ObjectHasValue>
```

```
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Window"/>
        <ObjectHasValue>
            <ObjectProperty IRI="#hasPowerConsume"/>
            <NamedIndividual IRI="#NoneConsumtion"/>
        </ObjectHasValue>
    </SubClassOf>
    <ClassAssertion>
        <Class IRI="#CoolingPossibility"/>
        <NamedIndividual IRI="#CoolingDependOn"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#CoolingPossibility"/>
        <NamedIndividual IRI="#CoolingHighly"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#CoolingPossibility"/>
        <NamedIndividual IRI="#CoolingLowly"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#HeatingPossibility"/>
        <NamedIndividual IRI="#HeatingDependOn"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#HeatingPossibility"/>
        <NamedIndividual IRI="#HeatingHighly"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#HeatingPossibility"/>
        <NamedIndividual IRI="#HeatingLowly"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#PowerConsume"/>
        <NamedIndividual IRI="#HighConsumtion"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#PowerConsume"/>
        <NamedIndividual IRI="#LowConsumtion"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#PowerConsume"/>
        <NamedIndividual IRI="#MiddleConsumtion"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="#PowerConsume"/>
        <NamedIndividual IRI="#NoneConsumtion"/>
    </ClassAssertion>
    <ClassAssertion>
        <Class IRI="http://www.semanticweb.org/ontologies/2010/11/response↩
            .owl#Response"/>
        <NamedIndividual IRI="#TEST"/>
    </ClassAssertion>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#hasCoolingPossibility"/>
        <ObjectProperty IRI="#hasTempControlDescriptor"/>
    </SubObjectPropertyOf>
```

```
    <SubObjectPropertyOf>
261         <ObjectProperty IRI="#hasHeatingPossibility"/>
            <ObjectProperty IRI="#hasTempControlDescriptor"/>
263     </SubObjectPropertyOf>
        <SubObjectPropertyOf>
265         <ObjectProperty IRI="#hasPowerConsume"/>
            <ObjectProperty IRI="#hasTempControlDescriptor"/>
267     </SubObjectPropertyOf>
        <SubObjectPropertyOf>
269         <ObjectProperty IRI="#hasTempControlDescriptor"/>
            <ObjectProperty abbreviatedIRI=":topObjectProperty"/>
271     </SubObjectPropertyOf>
        <FunctionalObjectProperty>
273         <ObjectProperty IRI="#hasCoolingPossibility"/>
        </FunctionalObjectProperty>
275     <FunctionalObjectProperty>
            <ObjectProperty IRI="#hasHeatingPossibility"/>
277     </FunctionalObjectProperty>
        <FunctionalObjectProperty>
279         <ObjectProperty IRI="#hasPowerConsume"/>
        </FunctionalObjectProperty>
281     <ObjectPropertyDomain>
            <ObjectProperty IRI="http://www.semanticweb.org/ontologies↩
                /2010/11/response.owl#hasResponse"/>
283         <Class IRI="#TempControl"/>
        </ObjectPropertyDomain>
285     <ObjectPropertyDomain>
            <ObjectProperty IRI="#hasPowerConsume"/>
287         <Class IRI="#TempControl"/>
        </ObjectPropertyDomain>
289     <ObjectPropertyDomain>
            <ObjectProperty IRI="#hasTempControlDescriptor"/>
291         <Class IRI="#TempControl"/>
        </ObjectPropertyDomain>
293     <ObjectPropertyRange>
            <ObjectProperty IRI="http://www.semanticweb.org/ontologies↩
                /2010/11/response.owl#hasResponse"/>
295         <Class IRI="http://www.semanticweb.org/ontologies/2010/11/response↩
                .owl#Response"/>
        </ObjectPropertyRange>
297     <ObjectPropertyRange>
            <ObjectProperty IRI="#hasCoolingPossibility"/>
299         <Class IRI="#CoolingPossibility"/>
        </ObjectPropertyRange>
301     <ObjectPropertyRange>
            <ObjectProperty IRI="#hasHeatingPossibility"/>
303         <Class IRI="#HeatingPossibility"/>
        </ObjectPropertyRange>
305     <ObjectPropertyRange>
            <ObjectProperty IRI="#hasPowerConsume"/>
307         <Class IRI="#PowerConsume"/>
        </ObjectPropertyRange>
309     <ObjectPropertyRange>
            <ObjectProperty IRI="#hasTempControlDescriptor"/>
311         <Class IRI="#TempControlDescriptor"/>
        </ObjectPropertyRange>
313 </Ontology>
```

```
315

317 <!-- Generated by the OWL API (version 3.0.0.1451) http://owlapi.↩
    sourceforge.net -->
```

A.2 Appliance ontology Modeling

```
<?xml version="1.0"?>


<!DOCTYPE Ontology [
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY xml "http://www.w3.org/XML/1998/namespace" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>


<Ontology xmlns="http://www.w3.org/2002/07/owl#"
     xml:base="http://www.semanticweb.org/ontologies/2011/0/import01.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     ontologyIRI="http://www.semanticweb.org/ontologies/2011/0/import01.owl">
    <Prefix name="xsd" IRI="http://www.w3.org/2001/XMLSchema#"/>
    <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="" IRI="http://www.w3.org/2002/07/owl#"/>
    <Prefix name="rdf" IRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
    <Prefix name="rdfs" IRI="http://www.w3.org/2000/01/rdf-schema#"/>
    <Declaration>
        <Class IRI="#Aircon"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Appliances"/>
    </Declaration>
    <Declaration>
        <Class IRI="#HomeService"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Ventilate"/>
    </Declaration>
    <Declaration>
        <Class IRI="#VentilatingFan"/>
    </Declaration>
    <Declaration>
        <Class IRI="#Window"/>
    </Declaration>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <Class IRI="#Appliances"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Aircon"/>
        <Class IRI="#Ventilate"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Ventilate"/>
```

```
        <Class IRI="#HomeService"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#VentilatingFan"/>
        <Class IRI="#Appliances"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#VentilatingFan"/>
        <Class IRI="#Ventilate"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Window"/>
        <Class IRI="#Appliances"/>
    </SubClassOf>
    <SubClassOf>
        <Class IRI="#Window"/>
        <Class IRI="#Ventilate"/>
    </SubClassOf>
</Ontology>



<!-- Generated by the OWL API (version 3.0.0.1451) http://owlapi.↩
    sourceforge.net -->
```

Appendix B

Link topology modeling and execution

B.1 Link topology execution file

The following file source is written by Python.

```
#Compostion.py

f=open("test.txt",'w')


DeviceInfo=[  [1, 'TV', 'DLNA', 'Input',  2,  'PZR900', 1],
      [1, 'TV',  'D4', 'Input',  3,  'PZR900', 1],
    [1, 'TV', 'HDMI', 'Input',  1,  'PZR900', 1],
    [1, 'TV', 'COMP', 'Input',  5,  'PZR900', 1],
    [1, 'TV', 'Svdo', 'Input',  4,  'PZR900', 1],
    [2, 'DVD',  'HDMI', 'Output', 1,  'BW930',  1],
    [2, 'DVD',  'DLNA', 'Output', 2,  'BW930',  1],
    [2, 'DVD',  'COMP', 'Output', 5,  'BW930',  1],
    [3, 'STB',  'HDMI', 'Input',  1,  'TZ-DCH', 1],
    [3, 'STB',  'DLNA', 'Input',  2,  'TZ-DCH', 1],
    [3, 'STB',  'COMP', 'Input',  5,  'TZ-DCH', 1],
    [3, 'STB',  'Svdo', 'Output', 4,  'TZ-DCH', 1],
    [3, 'STB',  'HDMI', 'Output', 1,  'TZ-DCH', 1],
    [3, 'STB',  'COMP', 'Output', 5,  'TZ-DCH', 1],
    [2, 'DVD',  'D4', 'Output', 3,  'BW930',  1],
    [4, 'TV', 'D4', 'Input',  3,  'KDE-P32',  1],
    [4, 'TV', 'Svdo', 'Input',  4,  'KDE-P32',  1],
    [4, 'TV', 'COMP', 'Input',  5,  'KDE-P32',  1]      ]


#print out the List of Appliance----------------------


for i in range(0,len(DeviceInfo)):
  f.write("\n")
  f.write("%s"%i)
  for j in range(0,len(DeviceInfo[0])):
```

```
    data=`DeviceInfo[i][j]`
35    f.write("     \t%s"%data)

37 f.write("\n\n\n\n")

39 #-------------------------------------------------------------


41

   Index=0
43 DeviceName=1
   Capability=2
45 Direction=3
   Value=4
47 DevID=5
   Flag=6
49

   start=2
51 end=4

53

   #find object device-----------
55 def fun_object(o_index):
     for i in range(0,len(DeviceInfo)):
57     if DeviceInfo[i][Index] == o_index:
         return i
59


61 #find start device-----------------------------------------

63 def shc_capable_output(dvice_start):
     list_devout=[]
65   temp=0
     for i in range(0,len(DeviceInfo)):
67     if DeviceInfo[i][Index]==dvice_start and DeviceInfo[i][Direction]=='↩
         Output':
         if DeviceInfo[i][Flag] != 0:
69         list_devout.insert(temp,i)
           temp=temp+1
71   return sort_list(list_devout)

73 #-------------------------------------------------------------
   #Sort the list allow to QoS-----------------------------
75

   def sort_list(list):
77   sort=[]
     for i in range(0,len(list)):
79     sort.insert(i,0)

81   for i in range(0,len(list)):
       temp=0
83     for j in range(0,len(list)):
         if DeviceInfo[list[i]][Value] > DeviceInfo[list[j]][Value]:
85         temp=temp+1

87     sort[temp]=list[i]

89   return sort
```

```
91 #-------------------------------------------
   #find input device ------------------------------
93
   def shc_capable_input(dvice_out):
95   temp=0
     possible_devin=[]
97   for i in range(0,len(DeviceInfo)):
       if DeviceInfo[i][Capability]==DeviceInfo[dvice_out][Capability] and ↩
           DeviceInfo[i][Direction]=='Input':
99       if DeviceInfo[i][Index]!=DeviceInfo[dvice_out][Index] and DeviceInfo↩
             [i][Flag] != 0:
           possible_devin.insert(temp,i)
101        temp=temp+1
103
     return possible_devin
105
   #-----------------------------------------------
107

109#Function of print out ------------------------------

111def fun_disp(index):
     for j in range(0,len(DeviceInfo[0])):
113    data=`DeviceInfo[index][j]`
       f.write("%s   \t"%data)
115  f.write("\n")
   #-----------------------------------------------
117
   #Function of print out the list ------------------------
119
   def print_out(output):
121  for j in range(0,len(output)):
       fun_disp(output[j])
123
   #-----------------------------------------------
125#store the searched route list ------------------------------

127
   ListCount=0
129List_up=[]
   crr_list_out=[]
131crr_list_in=[]
   Level=0
133Result=[]

135
   def fun_list_up(devNo):
137  global ListCount
     global List_up
139  List_up.insert(ListCount,devNo)
     ListCount=ListCount+1
141#-----------------------------------------------

143
   #Temp_stack Function for momorizing previous searching ---------
```

```
145
def fun_tempout(devNo,count):
147   global crr_list_out

149   t_size=len(crr_list_out)
      if t_size > count:
151     crr_list_out[count]=devNo
      else:
153     crr_list_out.insert(count,devNo)

155 def fun_tempin(devNo,count):
      global crr_list_in
157
      t_size=len(crr_list_in)
159   if t_size > count:
        crr_list_in[count]=devNo
161   else:
        crr_list_in.insert(count,devNo)
163
#-------------------------------------------------------------------
165

167 #Main Function to searching the course-------------------------

169 def main_ff(s_index,e_index,crr_cnt):
      global Level
171   global ListCount
      global Result
173   global List_up
      global crr_list_out, crr_list_in
175
      object=fun_object(e_index)
177
      DeviceOut=shc_capable_output(s_index)
179
      if len(DeviceOut)==0:
181     f.write("Dead-end<----------\n")
        ListCount=0
183     List_up=[]

185
      for i in range(0,len(DeviceOut)):
187
        find_way=shc_capable_input(DeviceOut[i])
189
        fun_list_up(DeviceOut[i])
191
        for j in range(0,len(find_way)):
193

195       fun_tempout(DeviceOut[i],crr_cnt)
          fun_tempin(find_way[j],crr_cnt)
197
          if DeviceInfo[find_way[j]][DevID] != DeviceInfo[object][DevID]:
199         fun_disp(DeviceOut[i])
            fun_disp(find_way[j])
201
```

```
          fun_tempin(find_way[j],crr_cnt)

          cnt=crr_cnt+1

          resch=DeviceInfo[find_way[j]][Index]
          main_ff(resch,e_index,cnt)

        else:
          fun_disp(DeviceOut[i])
          fun_disp(find_way[j])

          fun_list_up(find_way[j])

          presteck=[]
          for k in range(0,crr_cnt):
            presteck.append(crr_list_out[k])
            presteck.append(crr_list_in[k])

          List_up=presteck + List_up
          Result.insert(Level,List_up)
          Level=Level+1
          ListCount=0
          List_up=[]

          f.write("%s----------\n"%'succes')

#-------------------------------------------------------------------------

def serch_result(result):
  f.write("RESULT LIST----------\n\n")

  maxlength=0

  for i in range(0,len(result)):
    if len(result[i]) > maxlength:
      maxlength=len(result[i])
  f.write("%s\n"%`maxlength`)


  for i in range(0,len(result)):

    listlength=len(result[i])
    Effic=float(listlength)/maxlength*100

    Quality=0
    for j in range(0,len(result[i])):

      fun_disp(result[i][j])
      if DeviceInfo[result[i][j]][Value] > Quality:
        Quality=DeviceInfo[result[i][j]][Value]


    f.write("Q=%s,  "%`Quality`)
    f.write("E=%s%%\n"%`Effic`)
    f.write("%s----------\n"%'OK')

```

```
259 f.write("%s  "%`start`)
    f.write("%s\n"%`end`)
261
    main_ff(2,1,0)
263 serch_result(Result)

265
    f.write("—>%s\n"%`Result`)
267 f.write("—>%s\n"%`crr_list_out`)
    f.write("—>%s\n"%`crr_list_in`)
269 f.write("%s\n"%`List_up`)
    #print out the List of Appliance————————————————
271

273 for i in range(0,len(DeviceInfo)):
      f.write("\n")
275   f.write("%s"%i)
      for j in range(0,len(DeviceInfo[0])):
277     data=`DeviceInfo[i][j]`
        f.write("      \t%s"%data)
279
    f.write("\n\n\n\n")
281
    #————————————————————————————————————————————
283



285



287 f.write("\n\n%s"%`shc_capable_output(3)`)
    #f.write(`DeviceOut`)
289 #f.write("\n%s"%`find_object`)

291
    f.close()
```

appendix/Composition.py

B.2 A result of link topology

The following list is result of link topology from DVD "BW930" to TV "PZR900".

```
   2  'DVD'  'HDMI'  'Output'  1  'BW930'  1
 2 1  'TV'  'HDMI'  'Input'  1  'PZR900'  1
   succes——>Total Score:3
 4 faild
   3  'STB'  'HDMI'  'Input'  1  'TZ-DCH'  1
 6 3  'STB'  'HDMI'  'Output'  1  'TZ-DCH'  1
   1  'TV'  'HDMI'  'Input'  1  'PZR900'  1
 8 succes——>Total Score:3
   3  'STB'  'Svdo'  'Output'  4  'TZ-DCH'  1
10 1  'TV'  'Svdo'  'Input'  4  'PZR900'  1
   succes——>Total Score:8
12 4  'TV'  'Svdo'  'Input'  4  'KDE-P32'  1
   succes——>Total Score:10
14 3  'STB'  'COMP'  'Output'  5  'TZ-DCH'  1
   1  'TV'  'COMP'  'Input'  5  'PZR900'  1
16 succes——>Total Score:16
   4  'TV'  'COMP'  'Input'  5  'KDE-P32'  1
18 succes——>Total Score:18
   2  'DVD'  'DLNA'  'Output'  2  'BW930'  1
20 1  'TV'  'DLNA'  'Input'  2  'PZR900'  1
   succes——>Total Score:7
22 faild
   3  'STB'  'DLNA'  'Input'  2  'TZ-DCH'  1
24 3  'STB'  'HDMI'  'Output'  1  'TZ-DCH'  1
   1  'TV'  'HDMI'  'Input'  1  'PZR900'  1
26 succes——>Total Score:3
   3  'STB'  'Svdo'  'Output'  4  'TZ-DCH'  1
28 1  'TV'  'Svdo'  'Input'  4  'PZR900'  1
   succes——>Total Score:8
30 4  'TV'  'Svdo'  'Input'  4  'KDE-P32'  1
   succes——>Total Score:10
32 3  'STB'  'COMP'  'Output'  5  'TZ-DCH'  1
   1  'TV'  'COMP'  'Input'  5  'PZR900'  1
34 succes——>Total Score:16
   4  'TV'  'COMP'  'Input'  5  'KDE-P32'  1
36 succes——>Total Score:18
   2  'DVD'  'D4'  'Output'  3  'BW930'  1
38 1  'TV'  'D4'  'Input'  3  'PZR900'  1
   succes——>Total Score:13
40 4  'TV'  'D4'  'Input'  3  'KDE-P32'  1
   succes——>Total Score:15
42 2  'DVD'  'COMP'  'Output'  5  'BW930'  1
   1  'TV'  'COMP'  'Input'  5  'PZR900'  1
44 succes——>Total Score:21
   faild
46 3  'STB'  'COMP'  'Input'  5  'TZ-DCH'  1
   3  'STB'  'HDMI'  'Output'  1  'TZ-DCH'  1
48 1  'TV'  'HDMI'  'Input'  1  'PZR900'  1
   succes——>Total Score:3
50 3  'STB'  'Svdo'  'Output'  4  'TZ-DCH'  1
   1  'TV'  'Svdo'  'Input'  4  'PZR900'  1
52 succes——>Total Score:8
```

```
   4  'TV'  'Svdo'  'Input'  4  'KDE-P32'  1
54 succes--->Total Score:10
   3  'STB'  'COMP'  'Output'  5  'TZ-DCH'  1
56 1  'TV'  'COMP'  'Input'  5  'PZR900'  1
   succes--->Total Score:16
58 4  'TV'  'COMP'  'Input'  5  'KDE-P32'  1
   succes--->Total Score:18
60 4  'TV'  'COMP'  'Input'  5  'KDE-P32'  1
   succes--->Total Score:26
```

Appendix C

Results of the service composition service

C.1 Parent room

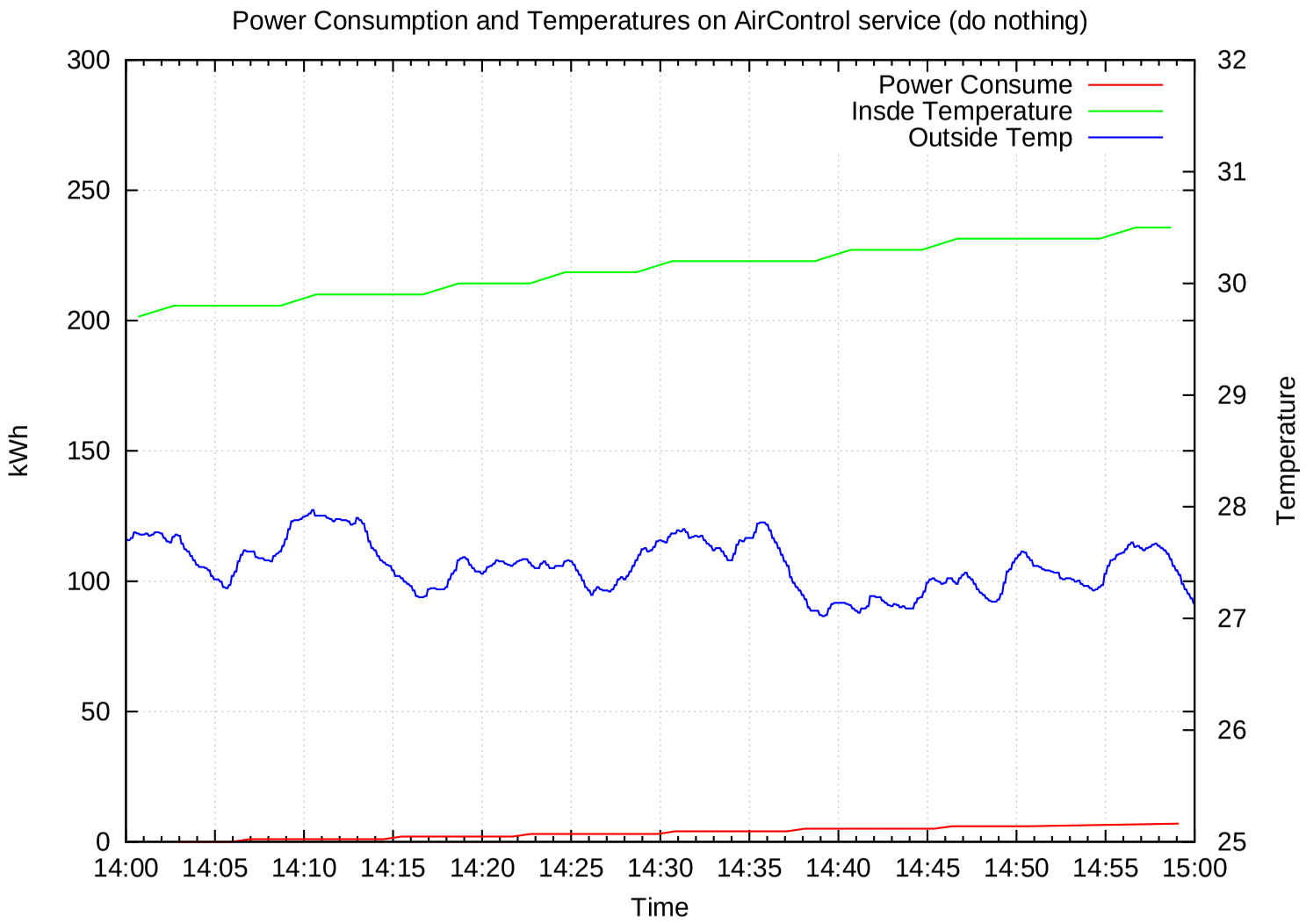


Figure C.1: Result of scenario 0

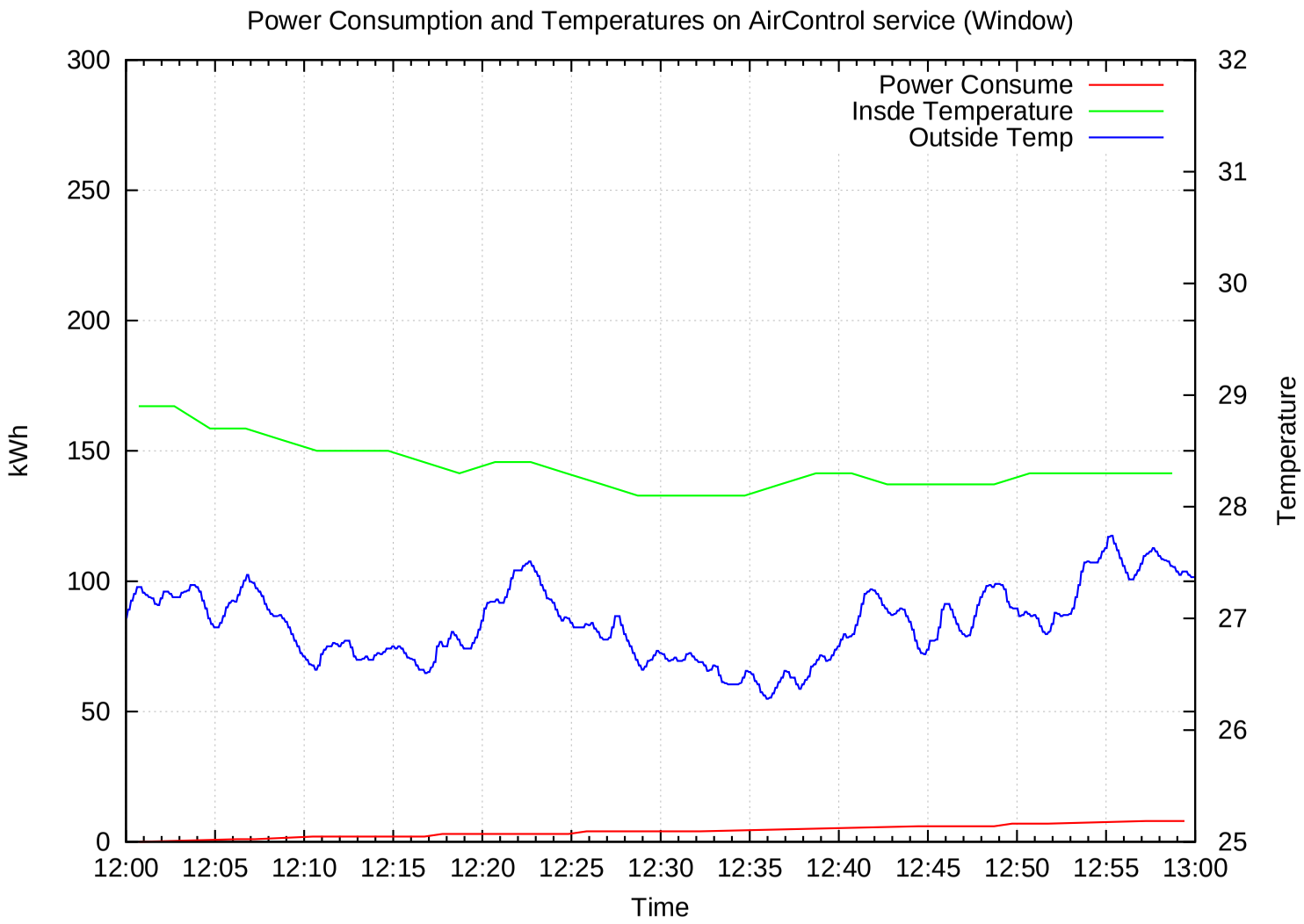


Figure C.2: Result of scenario 1

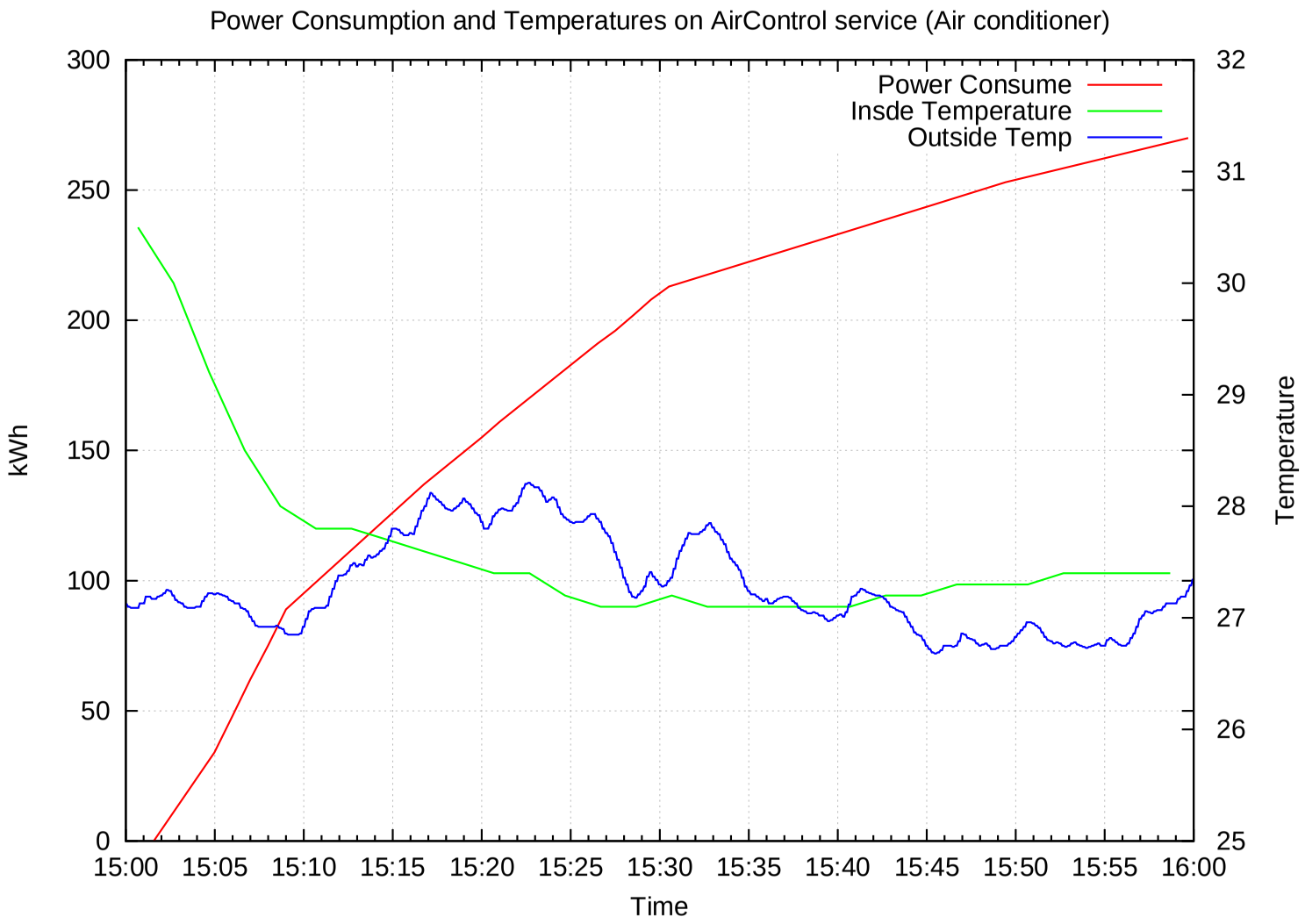


Figure C.3: Result of scenario 2

C.2 Child room

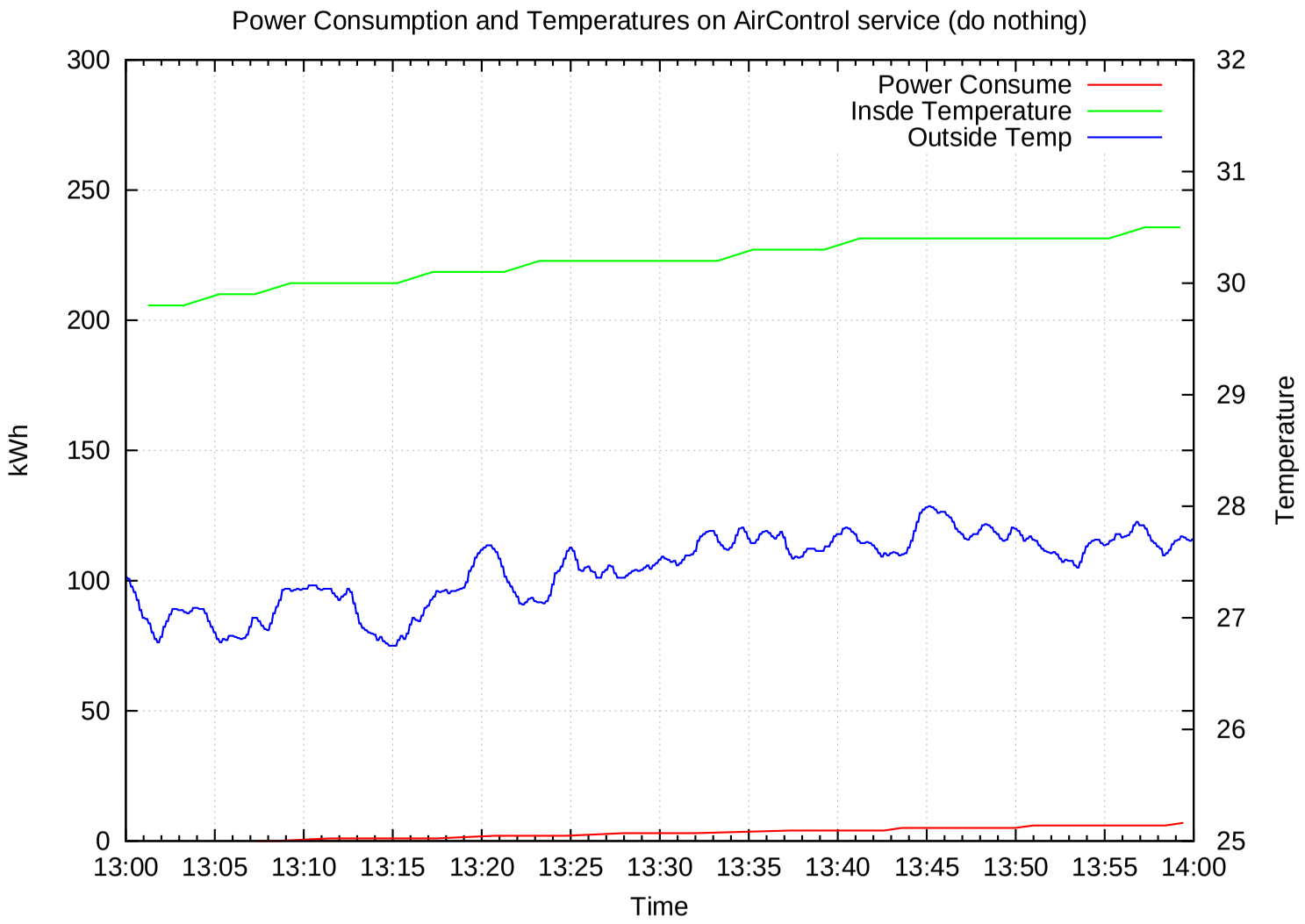


Figure C.4: Result of scenario 0

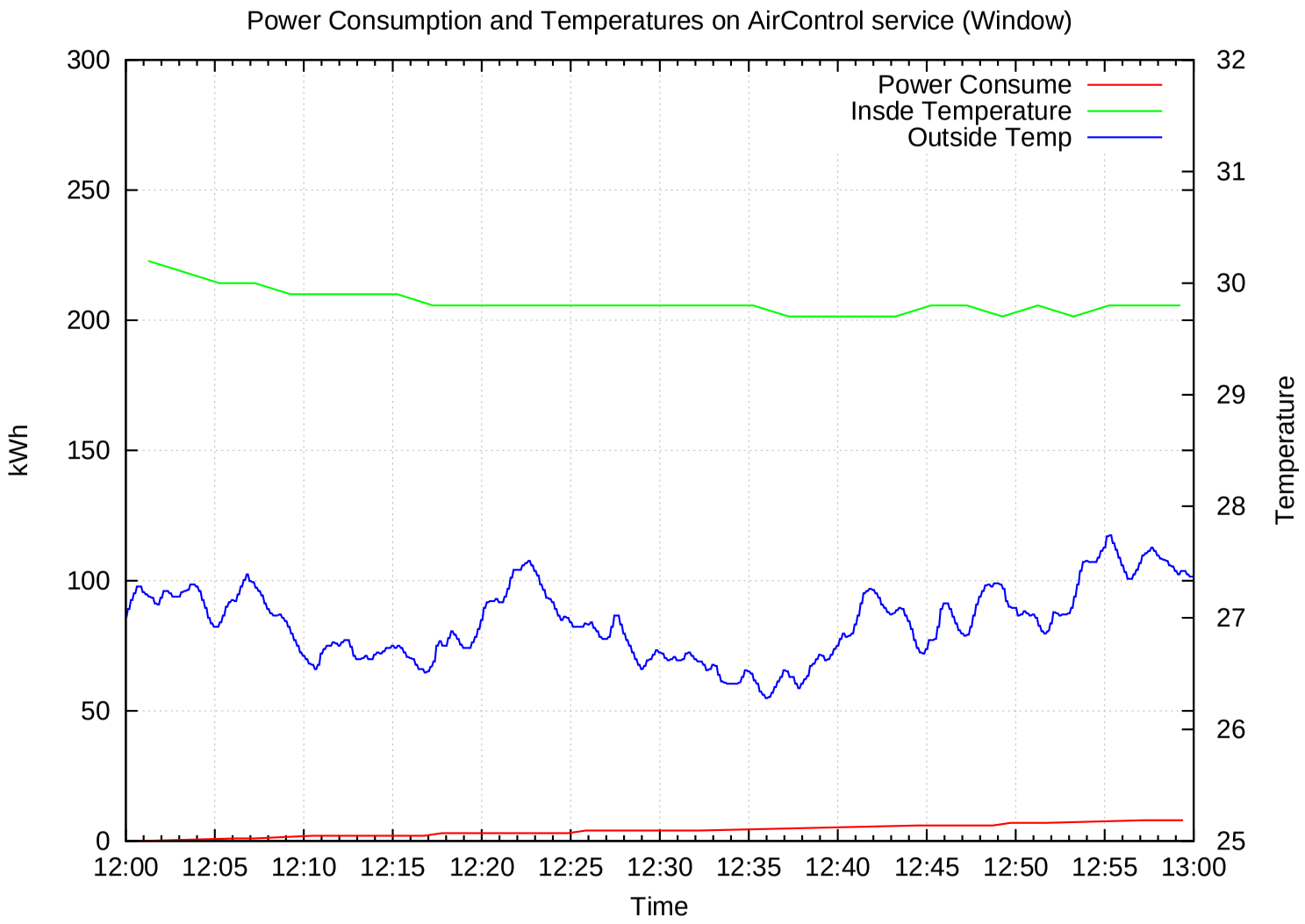


Figure C.5: Result of scenario 1

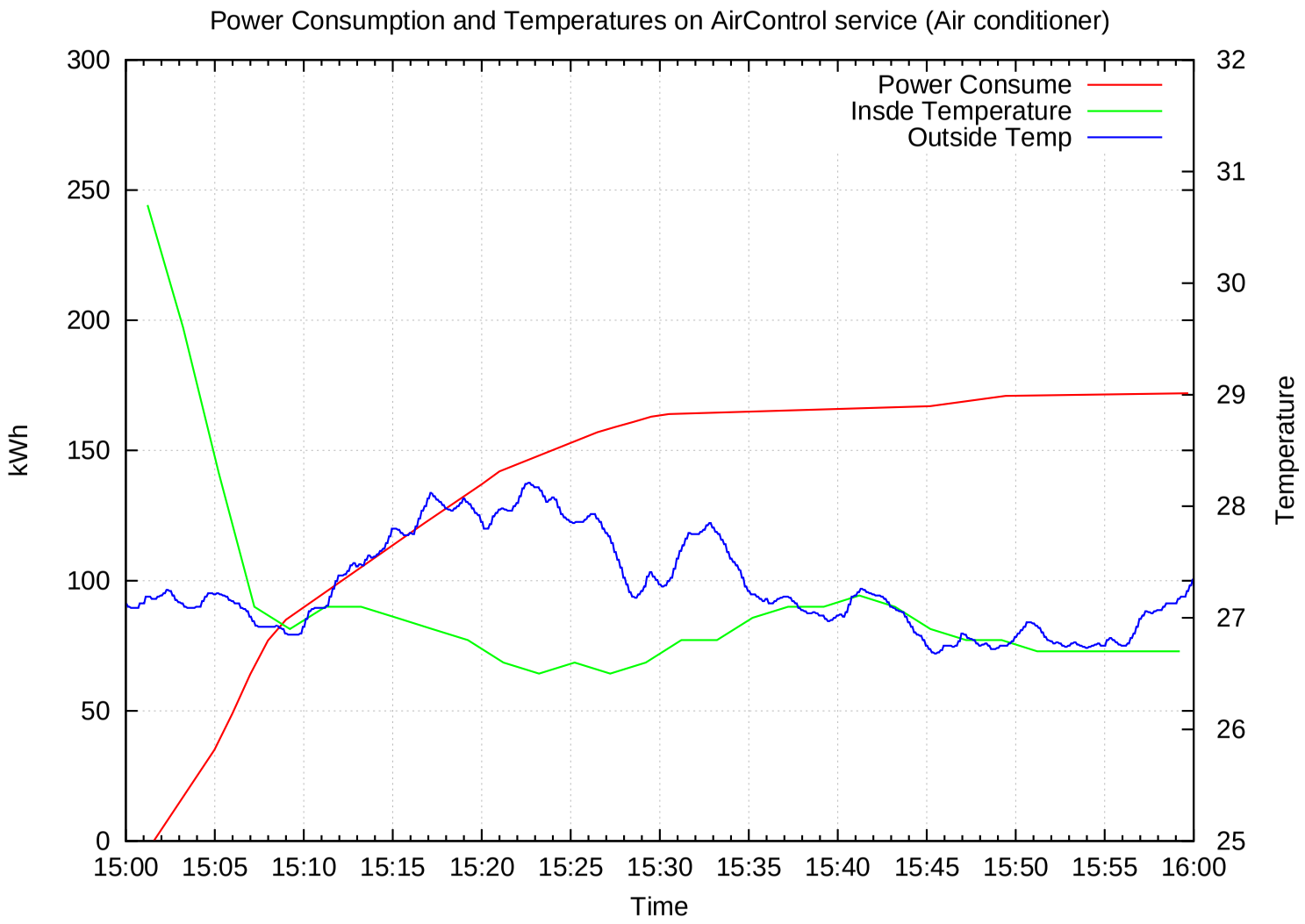


Figure C.6: Result of scenario 2