2009 International Conference on Embedded Software and Systems

Ontology-based Smart Home Solution and Service Composition

Jingjing Xu[1], Yann-Hang Lee[1], Wei-Tek Tsai[1], Wu Li[1], Young-Sung Son[2], Jun-Hee Park[2], and Kyung-Duk Moon[2]

[1]Computer Science and Engineering Dept.
Arizona State University
Tempe, AZ 85287

[2]Electronics and Telecommunications Research Institute
Daejeon, South Korea

Abstract

Given the diversity of home environment, appliances, and residents, the applications for smart homes must be configurable and adaptive. Instead of programming each household, we propose an ontology-based framework to facilitate the automatic composition of appropriate applications. The system composes appropriate services depending upon the available equipments in each individual household automatically. Meanwhile, it dynamically adjusts the environment parameters to match the customer needs and to encompass the available resource. With its supporting on customized function template editing, customers are able to specify their usual behavior templates as different mode.

1 Introduction

Smart homes have emerged as a focused application area of distributed embedded systems where digital appliances, sensor devices, PDAs, and handheld computers are integrated to facilitate intelligent services for households. Smart home is different than the traditional home on its ability to perform functions by integrating appropriate appliances automatically. Besides the automation, it also considers the different habits of the residences and adjusts the setting accordingly. For instance, to older people, the home temperature should not be too low. However, for a party, the guests will get sweaty with such a temporary level.

To realize the aforementioned intelligence in smart homes, there are three issues need to be addressed:

- Dynamically compose service plans to fulfill the requested functions base on available devices.
- Automatically supply and rank a list of functions that can be supported by a set of available devices.
- Dynamically adjust the system according to the environment and resident information.

This work was supported by the IT R&D program of MKE/IITA, 2006-S-066-02, Development of High Reliable Adaptive Middleware for u-Home.

Ontology has received significant attention in recent years with the emergence of semantic web [1][2][3][6]. It is an explicit specification of conceptualizations which organizes the semantic information as the knowledge base of the specific application domains [4][5][6]. Furthermore, its use is extended to Service-Oriented Computing (SOC) to facilitate more intelligent service discovery and composition [6][8]. The current ontology languages, such as RDF and OWL [9][10], are often represented in XML and can be processed by machines. They support the specification of concepts, relationships, and associated classification and reasoning mechanisms for data. Ontology has been established for context-aware and smart home applications [11]. For instance, in [12], a Getting up scenario is described to show whether the ontology-based model is valuable for description of context information in a home domain. In [12], an ontology-based model of the Tele-health Smart Home to initialize the Bayesian networks to recognize patient activity. Furthermore, ontology is used to model the relations between devices and services in a home environment and to manage software configurations [14].

In our smart home solution, ontology is introduced to deal with the three issues stated above base on the semantic information. Note that it is possible that a requested function cannot be fulfilled by a single device but a set of devices working together. Hence, in addition to the specification of static relationships between concepts in ontology systems, we introduce the application template [15] for the service composition specification. This paper presents our investigation of constructing an ontology-based knowledge representation for the semantic content and process model of smart home applications. In the following section, we propose an ontology-based architectural design of smart home. In Section 3, we go through the details of handling different scenarios that may occur in a smart home system. To illustrate the automatic function plan generation base on semantic information, a temperature control application scenario is used as case study in Section 4. Finally, the conclusion is made in Section 5.

978-0-7695-3678-1/09 $25.00 © 2009 IEEE
DOI 10.1109/ICESS.2009.60

2 Smart Home System Architecture

2.1 Ontology for Smart Homes

To alleviate the limitation of the keyword search, ontology is introduced to support the semantic discovery of devices in the SOC framework for smart homes. Four domain ontology systems – devices, function concepts, environment profile, and policy ontology, represent the knowledge base in smart home systems. Device ontology describes the concepts related to devices. Function ontology, as the core of the whole knowledge base, describes the concepts related to functions and templates are attached for each composable function. Policy/preference ontology describes the basic system constrains and preference rules about the use of devices and functions. Those rules may include conditions refers to environment ontology. Environment ontology describes the classification of environment profile, which includes natural environment profile and person profile.

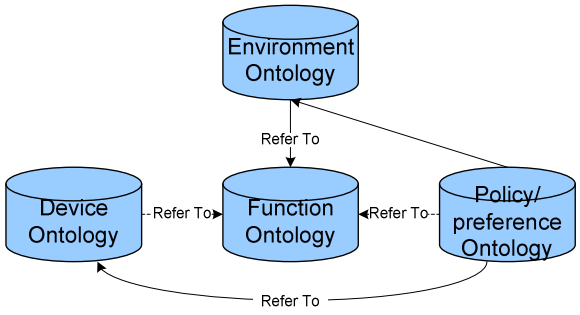

Figure 1 Ontology Systems

The ontology systems are cross-referenced to each other as shown in Figure 1. The knowledge base covers the necessary semantic information needed by a smart home system. Moreover, the application template is introduced in function ontology specification, by which the system is informed about how to assemble sub-functions to an integrated service.

2.2 Smart Home System Architecture

The proposed architecture supports the construction and execution of smart home applications based on the knowledge description in an ontology system. It includes four main components: Knowledge Base, Household Data Manager, Service Manager, and Plan Deployment as show in Figure 2.

Knowledge Base provides general knowledge representation related to smart home functioning. Household Data Manager is the data center of the whole system. It will store information about a specific house. Service Manager is responsible for all the checking and constructing the executable plan according to the information stored in Household Database. Plan Deployment sets the final parameter according to the resident profile and environment profile. It also responses for the resource control and monitoring. When exception happens, it is in charge of switching and deploying another executable plan to the system. Each part will be discussed in the following sections.

Knowledge Base

Knowledge Base supports the semantic knowledge specification and the reasoning.

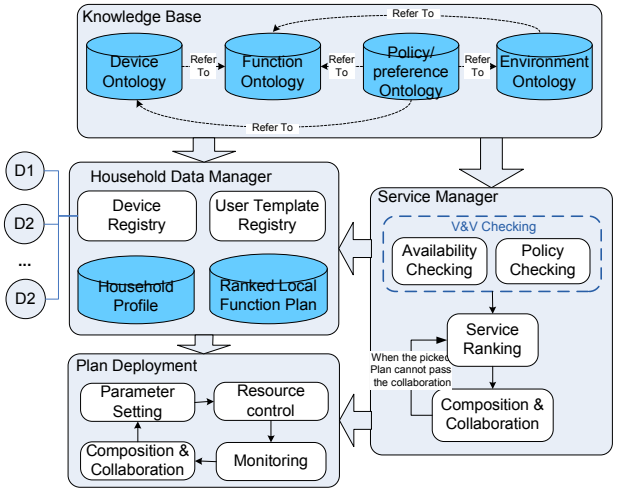

Figure 2 The Smart Home Architecture

As shown in Figure 1, function ontology is the core of the knowledge base. It is different from other ontology systems on its support of function template specification. There are two kinds of function concepts in the function ontology: atomic and composed. Both of them can be referred by other three ontology systems. The composed function concept has at least one function template associated to it. Besides ontology specification, it supports an important function: raw plan logic generation. When one function request comes, the inference query will be performed. All the atomic functions and composed functions, which can fulfill the request wholly or partially, are returned with ranking information.

Household Data Manager

Household Data Manager stores the detail information of the devices, user templates, the resident and environment profile, and ranked local function plans (RLFP) in a specific house.

The manager hosts a device registry which performs as a UDDI [16]. The household devices registration and the device discovery operation are all ontology based. Besides the basic device information, the associating protocols are also saved to enable the communication between devices. The other registry, user template registry, allows user register their

preferred functions and environment parameter combinations.

Except the device information, household profile keeps the other local information. It includes three main aspects: environment profile, resident profile, and policy. Environment profile records both static and dynamic information. The dynamic information can be obtained periodically via the web services, such as daily weather. Resident profile is more static. It includes the basic information and the self preference of the residents. Policy has two parts: the static policy and preference constrains, which is generated by system automatically based on the knowledge base and the local profile information. The RLFP are saved locally to speed up the recomposition process.

Service Manager

Service manager mainly supports three functions: verification and validation (V&V) checking, plan ranking, and the composition and collaboration. V&V checking performs availability checking and policy checking. Service ranking is performed base on the preference rules. The highest ranked function plan will be selected and the details of protocol matching will make sure that the plan is actually executable.

Plan Deployment

Plan deployment handles the remaining issues after an executable function plan is generated. It includes parameter setting, resource control, monitoring, and recomposition and collaboration. After the parameter setting and resource control phases, the selected function plan will become executable and can be performed in the smart home. The plan execution status is monitored. If the current plan is no longer feasible for the system due to device or network failures, a new plan will be applied to continue fulfilling the function. Since the generated plan is stored, the system does not need to go through the plan generation path again.

3 Service Composition

When system is running, it will meet three kinds of situations: a function request comes from user, a running device becomes unavailable, and a new device is added into the system or an unused device is removed. For the first two situations, system has to initial/adjust a running function plan, while for the third situation, RLFP database has to be maintained. The detailed methodologies are discussed in the following subsections respectively.

3.1 Initial Function Construction

The available devices in a house are not fixed. Devices will be removed/added from time to time. These changes may affect the function support of a system. To save unnecessary time on the function plan updating, the system only generates specific function plans when they are requested. Function plan is generated base on device ontology, function ontology, and current available devices information. The control flow of initial function construction is shown in Figure 3. It involves three phases: knowledge query, plan construction, and plan deployment.

System enters into knowledge query phase when a new function request comes in. It includes two steps. First, all related function concepts are extracted from function ontology with weight by inference query. For the composed function, the corresponding plan logic will also be included in the query results. Then, all related device concepts are extracted from device ontology base on the query result from the first step. The elements in the plan logic in this stage are still concepts on the knowledge level.

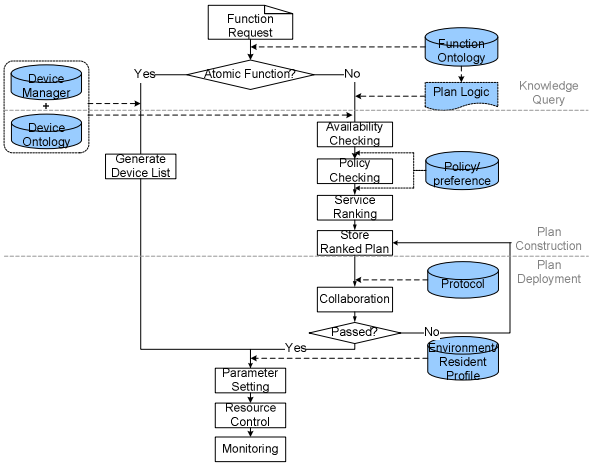

Figure 3 Initial Function Construction

After collecting needed knowledge from knowledge base, system goes into plan construction phase, where the result function concepts are all replaced by real devices and various checking is performed. For example, availability checking will filter out unfeasible function or function plan, and policy checking will make sure new function plan will not cause any system confliction. Meanwhile, all the rest function plans will be ranked base on the weight generated from the first phase and also the reliability and other attributes. Among all function plan candidates, system will pick the best one to deploy and save the whole list into RLFP database. Then, system enters into the last phase: plan deployment.

In plan deployment phase, the collaboration details, such as communication protocol and signal format, are determined and established to make sure the selected plan is executable. If any problem is found, the system will go through the collaboration part again until an executable plan is confirmed. After parameter setting

and resource control, the system will deploy the function plan finally and monitor it.

3.2 System Recomposition

When a running device fails, instead of going through initial function construction process again, system recomposition process will be invoked to achieve constantly function supply. System recomposition has the same plan deployment phase as in initial function construction. Because of pre-stored reusable function plans, the knowledge query and plan construction phases are eliminated, which accelerates system response time.

Figure 4 shows the work flow of the recomposition process. Instead of generate function plan list again, system will pick function from RLFP database to replace the broken one. Because there might be more than one device are broken, or the broken device is applied to fulfill more than one functions at the same time, system will first run dependency checking to get affected function list. Then, it will keep getting the next candidate function plan until it can enable the collaboration among the devices in the plan. If there is more than one function to be re-composited, system will perform confliction checking among the plan set, such as power and bandwidth limitation, or if the an exclusive device is used in two function plans. Once a function plan is selected, the deployment phase is carried out same as in initial function construction scenario.

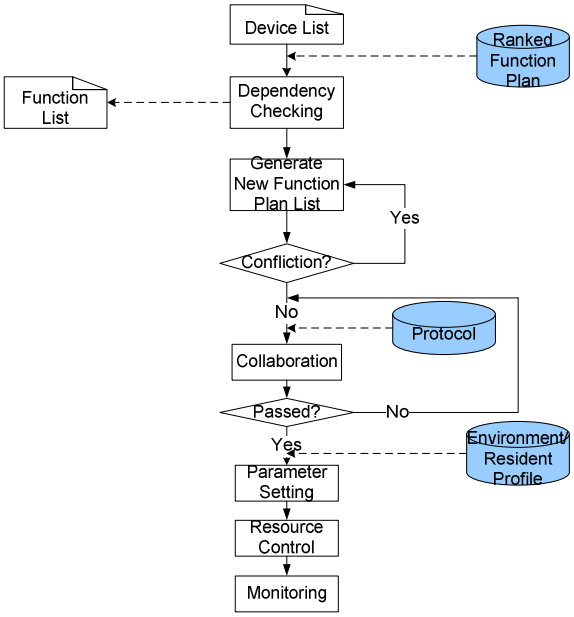

Figure 4 System Recompostion

3.3 Device Registration and RLFP Maintenance

The system available devices pool changes from time to time. Some existing ones may be removed while new devices will be added. Device upgrading can be treated as device removing and adding. Not all of them will affect stored RLFP. For example, if the removed device does not have dependency with any requested function, no matter the function is running or pending, system will simply remove its registration information. More details will be discussed in the following.

Removing device

When a device is removed from system for any reason, system has to response accordingly to what kind of device it is. With the dependency relations between the function template and device ontology, the system can trace from the device concept, under which the removed device registered, back to the affected function concepts, then the RLFP. If the device is not related with currently requested function, its registration will be simply removed. If the device is currently running, system has to perform recomposition as shown in section 3.2. If it is not used currently, however in the candidate function plan, the related function plan will be removed from RLFP.

Adding device

When a new device is added into system, it has three different cases. Its processing logic is shown in Figure 5. Same as in device removing, the dependency checking will be performed first.

If the device concept is not related with any current requested function, system will simply keep the registration information for future use.

If the device concept is related with currently requested function, no matter if the function has a feasible plan already running or not. The system has to update the RLFP database. System will perform dependency checking and find out if there is any function that is affected by this change. If there is any, system will further check if the affected function is in pending status.

If any new function plan can be composed because the new device being added, the system will go through the plan construction phase as in section 3.1. The generate function plan will be stored in the RLFP database when the function request is fulfilled by the current system already. Otherwise, user will be notified that the function is ready. If user chooses to apply it, system will deploy the new generated function plan as in plan deployment phase in initial function construction.

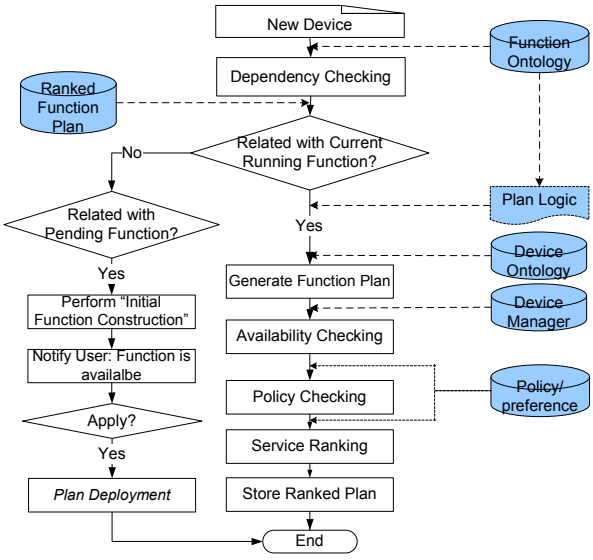

Figure 5 New Device Adding

3.4 Ontology System Maintenance

Normally, the changes over ontology specification will not affect the running function plan. However, ontology maintenance may need to be performed for the future use. There are three basic changes over ontology system: adding a new item, deleting an existing item, and modifying an existing item.

Adding any new information requires significant update and checking. An item added must satisfy the constraints in the ontology system. If a new relation is added onto existing classes, the original and derived relations needs to be examined to make sure there is no violation among them. If a new item is added under a given concept, all the relations associated with the concept must be verified with the new item. In addition, the consistency checking needs to be performed over updated relations.

Deleting an existing item from the ontology in general is difficult and should be discouraged because the ontology systems are cross-referenced. Deleting an item will not only affect the current ontology system but the changes may propagate to the other referenced ontology systems. Also, if a device concept is removed, any device that references to this item will realize it has referenced to a null item and will never be visible to the system.

There are two kinds of modifications. One is moving existing item. This kind of modifying is difficult because moving is essentially removing an item or a relation from one place, while adding the same item in another place. Thus, all the complexities of both adding and deleting will be there. The other is modifying the attributes of the item. The modified attribute must satisfy the constraints in the ontology system. Any propagated changes need to be checked for the system consistency.

4 Case Study

In this section, a "temperature control" example is used to show how the proposed smart home system generates function plan automatically and performing service recomposition.

4.1 Device and Function Ontology

The example device ontology of a sample smart home is illustrated in Figure 6. Devices are categorized into two classes: small power and larger power. The categorization is base on the technical parameters of the devices.

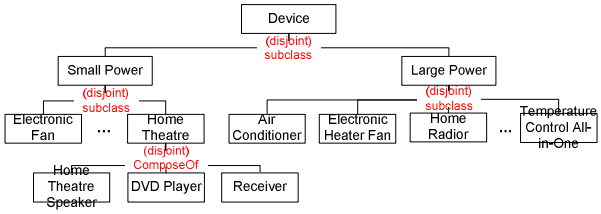

Figure 6 Device Ontology

Figure 7 shows the function ontology of the smart home system. Functions that a smart home can supply are divided into five classes: access control, security, appliance control, information supply, and sensor. The composed function "Temperature Control" is the subclass of "Appliance Control". It is composed by two functions: heating and cooling. The function template shown in Figure 8 describes how the temperature control is composed using heating and cooling services.

4.2 Policy Ontology

Figure 9 shows a simple policy classification. Under each class, there will be several policy templates. For example, every house has some constraints about power consumption. It can be a specific maximal AC current or expressed as "if two devices are running in parallel, at most two of them can be large-power devices according to device ontology definition". An example formal expression is as following.

$$Status(A, \text{"running"}) \cap Status(B, \text{"running"}) \cap Status(C, \text{"running"}) \Rightarrow \neg (LargePower(A) \cap LargePower(B) \cap LargePower(C))$$

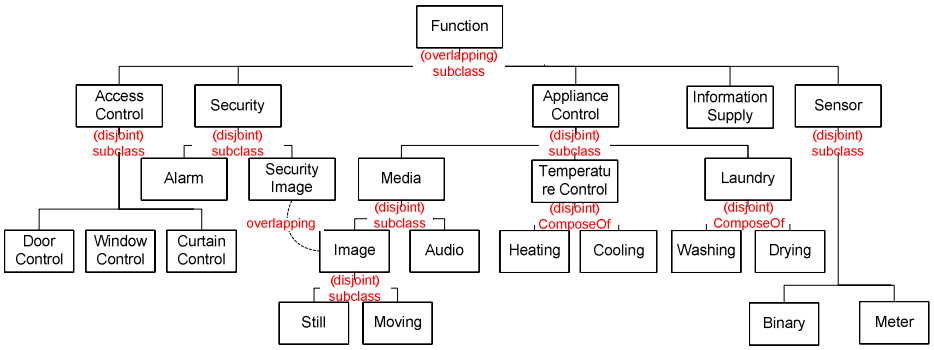

Figure 7 Function Ontology

The other example of policy specification related with temperature control is expressed in the following, which states that "when the temperature control function is running, the window needs to be closed",

functionStatus(temperatureControl, "running") => Status(window, "closed")

The function can be attached to the policy enforcement mechanism. For instance, the function WindowControl.closeAll is attached to the 2nd policy. To be compliant with this policy, the corresponding function, i.e., WindowControl.closeAll, will be performed.

4.3 Environment Profile Ontology

Environment profile includes two kinds of profiles: Natural environment profile and person profile. Natural environment indicates the characters of the weather in different area. Person profile indicates the age range and some other issues related with people and will affect the preference of smart home parameter setting. The example profile ontology is shown in Figure 10.

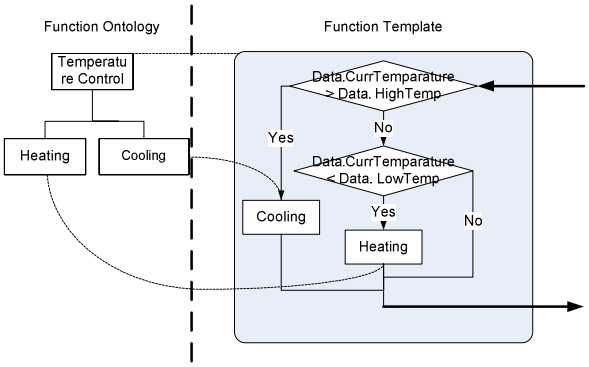

Figure 8 Function Template Sample

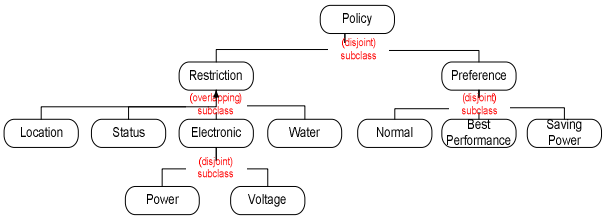

Figure 9 Policy Ontology

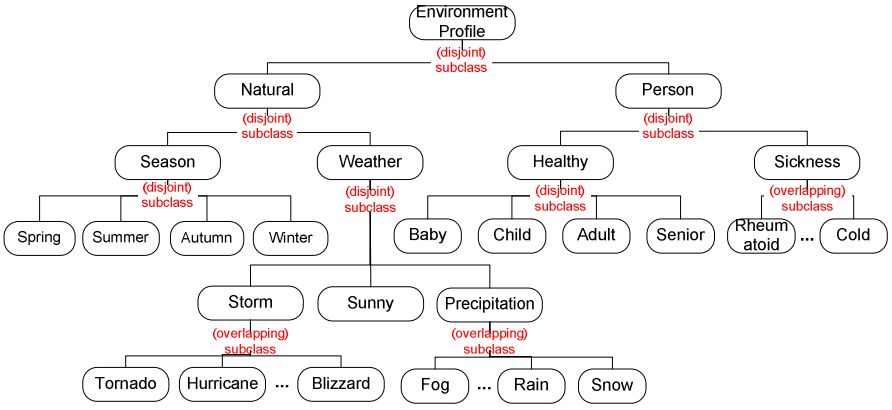

Figure 10 Environment Profile Ontology

Once the ontology is specified, it refers to several parameter tables. For example, for four seasons specified in Table 1, the best indoor temperatures are set differently to provide comfort indoor atmosphere.

Table 1 Temperature parameter setting 1

	Spring	Summer	Autumn	Winter
LTemp	16	17	16	16
HTemp	28	25	28	25

The temperature will be further adjusted according to person profile as shown in Table 2.

Table 2 Temperature parameter setting 2

	Baby	Child/Adult	Senior
LTemp	LTemp+1	LTemp	LTemp+1
HTemp	HTemp-1	HTemp	HTemp+1

4.4 Composed Function Plan

In the device ontology specification, the concepts defined in function ontology are referred. When the device is registered into system, since it is based on the device ontology, the device is registered with its function information implicitly. As shown in Figure 11, five devices are registered into system with functions related with "temperatureControl" directly or indirectly. They will be used when the "temperatureControl" function is composed.

The system composed four different function plans for "temperatureControl" according to the template shown in based on the available devices. As an example, one of them is depicted in Figure 12.

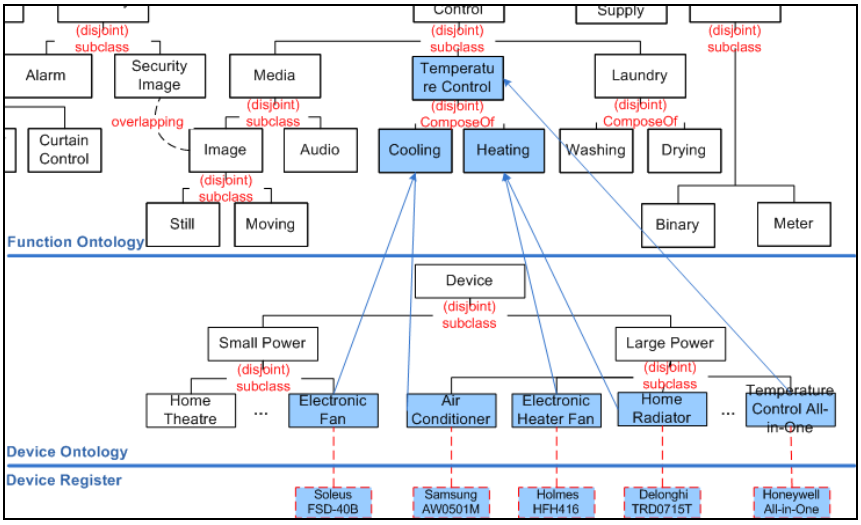

Figure 11 References between function ontology, device ontology, and device register

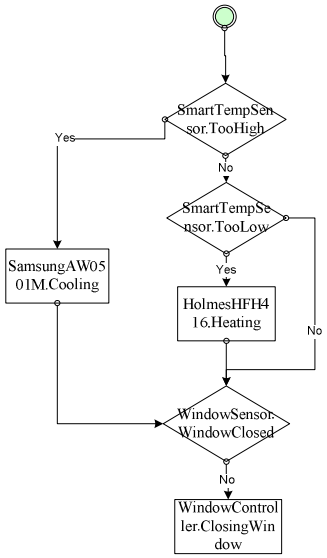

Figure 12 A sample plan generated automatically

The running log of the system is shown in Figure 13. The system automatically adjusts the running plan to fulfill the required "temperatureControl" function. When the deployed function cannot be satisfied, system proposes to invoke the most related function "cooling", base on the function ontology definition, such that it may be able to partially fulfill the requested function.

5 Conclusion

Given the diversity of home environment, appliances, and residents, the applications for smart homes must be configurable and adaptive. In this paper, we proposed a system architectural design for smart home ontology and service composition. The proposed system consists of a smart home knowledge-base (ontology), a household database, a service manager, and a plan deployment component. A generic knowledge representation is used to facilitate the composition of appropriate applications based on user profile and available appliances.

There are further optimizations and functionalities can be addressed in the future. In addition, automated

code generation for a selected execution platform will be the immediate tasks. This will allow us to test any integration issues that could be raised when the ontology and the process composition are parts of the smart home systems.

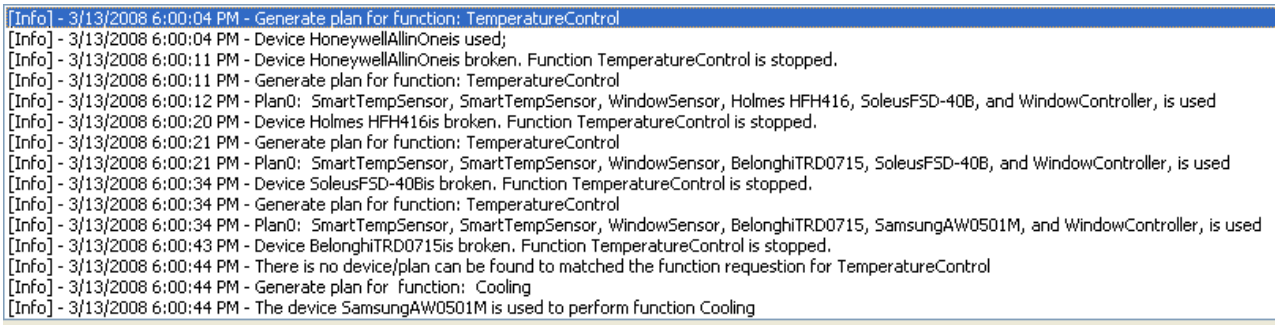

Figure 13 System running log

Reference

[1] N. F. Noy, M. Sintek, S. Decker, M. Crubezy, R. W. Fergerson, and M. A. Musen. “Creating Semantic Web Contents with Protege-2000”, IEEE Intelligent Systems 16(2):60-71, 2001.

[2] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara. “Semantic Matching of Web Services Capabilities”, 1st International Semantic Web Conference, 2002.

[3] R. Chinnici, J. J. Moreau, A. Ryman, and S. Weerawarana. “Web Services Description Language (WSDL) Version 2.0. June 27, 2007. http://www.w3.org/TR/wsdl20/.

[4] D. Fensel. “Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce”, Springer-Verlag, Berlin, 2001.

[5] “What is an Ontology,” http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.

[6] A. Bernaras, I. Laresgoiti, and J. Corera. “Building and Reusing Ontologies for Electrical Network Applications”, ECAI96, 12th European conference on Artificial Intelligence Ed., John Wiley & Sons Ltd., pp. 298-302

[7] M. P. Papazoglou and G. Georgakapoulos. “Service-Oriented Computing”, CACM, October 2003, 46(10).

[8] W. T. Tsai, "Service-Oriented System Engineering: A New Paradigm", IEEE International Workshop on Service-Oriented System Engineering (SOSE), October 2005, pp. 3 - 8.

[9] “OWL-S: Semantic Markup for Web Services”, http://www.w3.org/Submission/OWL-S, Nov 22, 2004.

[10] “OWL Web Ontology Language Reference, Feb 10, 2004 ”, http://www.w3.org/TR/owl-ref/

[11] T. Gu, X. H. Wang, H. K. Pung, D. Q. Zhang. "An Ontology-based Context Model in Intelligent Environments", In Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, California, January 2004.

[12] E. Kim and J. Choi. “An Ontology-Based Context Model in a Smart Home”, Workshop on Ubiquitous Web Systems and Intelligence (UWSI 2006), pp. 11-20.

[13] F. Latfi, B. Lefebvre1 and C. Descheneaux, “Ontology-Based Management of the Telehealth Smart Home, Dedicated to Elderly in Loss of Cognitive Autonomy”, Third International Workshop, OWLED 2007, OWL: Experiences and Directions.

[14] E. Meshkova, J. Riihijarvi, P. Mahonen, and C. Kavadias. “Modeling the home environment using ontology with applications in software configuration management,” International Conference on Telecommunications, 2008. ICT 2008, pp. 1-6.

[15] W. T. Tsai, B. Xiao, Q. Huang, Y. Chen, and R. Paul, "SOA Collaboration Modeling, Analysis, and Simulation in PSML-C", Proc. of the Second IEEE International Symposium on Service-Oriented Applications, Integration and Collaboration (SOAIC’06), October 2006.

[16] UDDI Technical Committee. Universal Description, Discovery and Integration (UDDI) http://www.oasis-open.org/committees/uddi-spec.