A Standard Method-Based User-Oriented Integrated Architecture for Supporting Interoperability among Heterogeneous Home Network Middlewares

Hark Jin Lee[1] and **Sung Jo Kim**[2]
[1]Department of Computer Science & Engineering, Chung-Ang University
e-mail : gausslee@konan.cse.cau.ac.kr
[2]Department of Computer Science & Engineering, Chung-Ang University
e-mail : sjkim@cau.ac.kr

Abstract Although there exist many middlewares such as Havi, Jini, LonWorks, UPnP, and SLP, new middlewares specialized for diverse information appliances are expected to appear continuously as home networks evolve. In this paper, we first investigate an integrated architecture for supporting interoperability among heterogeneous middlewares on home networks. We then propose and implement a user-oriented integrated architecture based on standard methods for efficient home network. HOMI(HOmenetwork Middleware for Interoperability) architecture proposed in this paper provides an interface for identifying and utilizing services among heterogeneous devices using standard methods in order to support interoperability among heterogeneous middlewares. Non-standard methods provide interfaces that allows users to build and/or modify interoperable service scenarios using a script language called HOMIL(HOMI Language). Different from existing integrated middleware architectures, HOMI allows users to build and organize their scenarios through the interfaces. By allowing heterogeneous devices to control each other or to transfer data to others using standard methods, HOMI improves efficiency and convenience of interoperation between heterogeneous appliances for home network. Applying modified scenarios immediately to a home network environment, HOMI provides users with seamless services without installing new applications, updating the server, or rebooting in order to install new scenarios. Lastly, the overhead occurred in a centralized and integrated middleware architecture can be reduced by distributing agents into multiple devices.

Keyword: Home Network, Home Server, Home Gateway, Scenario

1. Introduction

The evolution of high-speed communication, Internet, the digital hardware technology has stimulated that information appliances become further intelligent. This has led to the development of middlewares such as UPnP[1], Jini[2], Havi[3], LonWorks[4], SLP[5], etc. supporting service discovery and interaction that simplify the installation of information appliances with an ability to discover other services(e.g. application softwares and devices which can be accessed and utilized by other applications or services) dynamically.

Although these middlewares have the same goal of discovering and controlling devices, there is heterogeneity in many respects. This heterogeneity has appeared because each technique is specialized to support domains of specific applications. For example, SLP focuses on scalability in order to provide services in an enterprise net-work environment while UPnP has been designed appropriately for SOHO environment. On the other hand, Havi focuses on interoperability among home AV appliances.

The heterogeneity of home network middlewares force service application developers either to develop different applications for each appliance which provide the same services considering properties of each middleware, to develop a large application that supports different types middleware simultaneously. It is expected that such heterogeneous home network middlewares will appear continuously and that such heterogeneous middlewares and protocols will mingles together in the future homenetwork environment[6][7].

The solution to the heterogeneity of middlewares is in a great demand in the field of home network research. In order to improve home network technology, mechanisms for supporting interoperation should be developed so that home users can use various home network devices efficiently. This service interoperation can provide the convenience and efficiency of life by supporting seamless connection among heterogeneous home network appliances.

For convenient and adaptable service interoperation in home network environment in which various services are needed, standardized service call/response methods among heterogeneous middlewares should be provided. After these standardized methods are published, users can receive various services conveniently without their intervention through interoperation among heterogeneous middlewares. This paper designs and implements a standard method-based user-oriented integrated architecture for heterogeneous home network middleware in order to support convenient and efficient home network environment. We then propose and implement a standard method-based efficient home automation.

"This research was supported by the MIC(Ministry of Information and Communication), Korea, under the HNRC(Home Network Research Center)-ITRC support program supervised by the IITA(Institute of Information Technology Assessment) (IITA-2006-C1090-0603-0035)"

This paper is organized as follows. In Section 2, we introduce researches in progress for the development of integrated middlewares which support interoperability among appliances under home network, analyze problems caused by them, and present our motivation of this research. In Section 3, we describe how to design architecture for standard method-based user-oriented integrated middlewares proposed in this paper, In Section 4, we show and explain the result of the framework implemented by us. Finally, in Section 5, we conclude the paper and describe the future work.

2. Related Works

Firstly, this paper discusses what kind of effects the heterogeneity has on the interoperation of home network middlewares. In general, heterogeneity means that interfaces and architectures of diverse components that exist in a specific domain are different[8]. Heterogeneity can exist in many parts in a system.(for example, encoding methods of information, network protocols, data formats, etc.) If standardization is established among these parts, problems caused by the heterogeneity can be solved and various home network usage scenarios can be designed with seamless interoperation. However, it is not easy to standardize middlewares related to home network unlike existing middlewares such as TCP/IP because there exist too diverse devices and services, which is the property of the home network environment. Also, it is much more difficult to predict the future since it is possible that the existing services and devices are continuously developed. Even though it is predictable, there is a limit to solving fundamental problems of the heterogeneity. These problems usually appear at a semantic stage which will be discussed in Section 2.2, and it is not simple at all to solve and even occasionally to recognize the problems. Also, syntax problems caused by the difference between the service discovery and invocation mechanism should be solved as well for interoperability.

2.1 Interoperation Mechanisms Among Heterogeneous Home Network Middlewares

Interoperation mechanisms among heterogeneous home network middewares currently under investigation can be classified into two types such as an individual bridge and an integrated framework. The former one provides compatibility using one-to-one bridge protocol between middlewares in order to support interoperability among heterogeneous middlewares. The UPnP-to-Havi bridge[9] was developed by Thomson Multimedia and Philips and the interoperation of Jini and UPnP[10] had been re-searched at New Orleans University. Being useful for the interoperation between two specific middlewares, it has a scalability problem since it fails to provide a consistent way for the interoperation of various types of middlewares as the number of bridges increases and connections can be complicated as well with the appearance of new middlewares. The latter provides an abstracted common layer above various middle-wares and has an architecture that bridges each middleware based on the common layer. This type of architecture has an advantage that, even though new middlewares are developed, it can be easily integrated with other middlewares if an appropriate agent is implemented. At Waseda University, middleware integration has been at-tempted through SOAP(Simple Object Access Protocol)gateway configuration[11] and researches on the interoperation services among heterogeneous middlewares[12] have been under way at OSGi Alliance[13] and ETRI[14].

2.2 Analysis of Integrated Framework-based Mechanism

A model presented in this paper is based on an integrated middleware framework, and the following four issues should be considered.

1) How can devices adopting different middlewares find each other transparently?

Each middleware utilizes different service discovery mechanisms. Due to the differences between the mechanisms, even devices providing compatible services cannot recognize each other if each adopts heterogeneous middlewares.

2) How can services adopting different middlewares invoke each other?

A service invocation mechanism of Jini relies on Java RMI(Remote Method Invoke) which uses Java byte code. UPnP uses SOAP(Simple Object Access Protocol) to invoke a service transferring XML text stream. Problems caused by the difference between service invocation mechanisms must be solved to provide interoperability between heterogeneous middlewares. In order to solve the problem, syntax elements of middlewares(e.g., method name, the order of arguments, the type size of return value and arguments) should be adjusted. It is also necessary to convert calling methods according to service invoke mechanisms of each middleware.

3) How can the integrated middleware framework recognize that heterogeneous services are interoperable?

We cannot assume that interoperable services are provided when service interfaces(i.e. syntax elements) are identical. For example, suppose that Jini in Table 1. stores new data in a file using *PutFile* which is a storage service provider method. Similarly, suppose that a UPnP storage service in Table 1 stores files using a *PutFile* method in SCPD(Service Control Protocol Description). However, the UPnP storage service requires a user authentication process and new data can be stored in a file only after passing the process. If the authentication fails, it returns an error. In this case, syntax elements look identical, but, because of the difference between semantic elements, a method to adjust these elements is necessary for interoperability. A typical way to provide interoperability among services which are syntactically identical but semantically different components is to define a standard interface for integrated middleares for each service(e.g., TV, MP3, printer, etc.) and to use table-formatted translation bridges for each middleware according to the definitions. However, static bridging to such a single standard interface has a limit to reflect dynamic properties of home network middleare protocols whenever new functions or devices are inserted[15].

Moreover, whenever new devices appear, it is very time consuming to define standards and to develop new services adjusting to those standards as we have experienced previously. Instead, application developers should develop applications which are operable with all services under other middlewares whenever they develop services. But it is impossible in reality because new interfaces can continue to be defined owing to the advent of new devices after applications are developed.

Table 1. Jini and UPnP Syntax

Jini	void PutFile(String file);
UPnP	<action> <name>PutFile</name> <argumentlist> <name>file</name> <relatedStateVariable> newFile </relatedStateVariable> <direction>in</direction> </argumentlist> </action>

4) Are there enough demands for services that require interoperation between heterogeneous middlewares in a home network environment?

According to the survey conducted by KISDI(Korea Information Strategy Development Institute)[16], home network usage patterns of users can be classified into data network, entertainment, and home automation and the number of users who are interested in entertainment and data communication is not few as shown in Fig. 1. In spite that the potential market demand for entertainment and data communication in home network is very large, researches on communication among heterogeneous middlewares are restricted to provide static services; furthermore, researches on architecture which supports adaptively service changes requested by users have not been materialized yet. For supporting efficient and convenient services under diverse middlewares, mechanisms for interoperability among heterogeneous home network middlewares should be developed. Such interoperability mechanisms need to support services by the form of user's requests no matter what kind of service change is required.

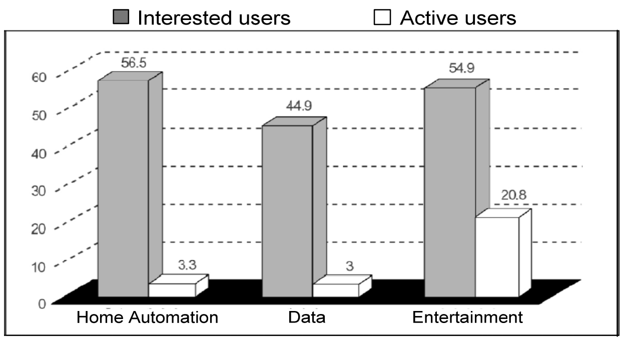

Fig. 1. Home Network Application and Interest Areas

3. HOMI Architecture

HOMI is a standard method-based user-oriented integrated middleware framework designed in consideration of service usage patterns. In order to solve semantic problems caused by the difference of heterogeneous middleware interfaces during the interoperation process as pointed out in Section 2.2, HOMI defines standard methods among middlewares and provides an interpretive language called HOMIL(HOMI Language) which supports to design preferred interworking scenarios in a simple way to end users for non-standard methods.

HOMI consists of five main modules:

- Agent Manager
- Standard Method Manager
- State Manager
- HOMIL Analyzer
- Context/Event Manager

HOMI can distribute each Agent into various servers in order to solve the bottleneck problems encountered in a centralized architecture. Also, all protocols have an open architecture using XML and provide utility classes and APIs based on C++ and JAVA to assist in developing agents for newly defined middlewares.

Fig. 2 shows the HOMI architecture.

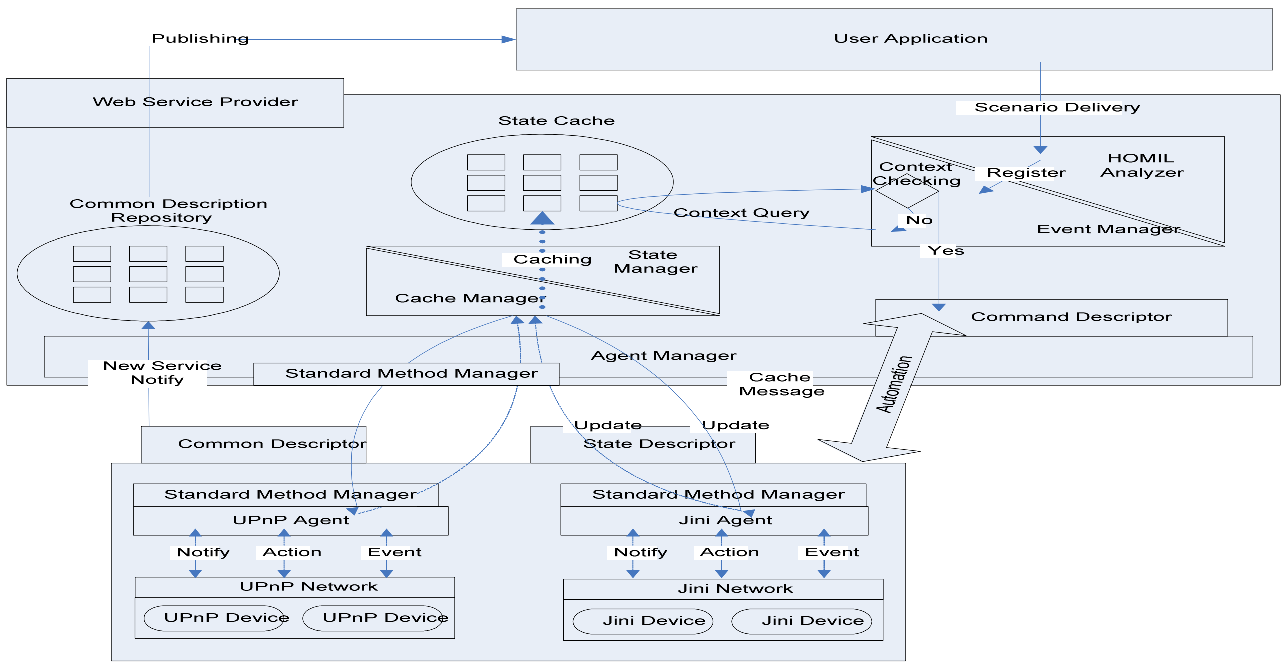

Fig. 2. HOMI Architecture

3.1 Agent Manager

It has been already mentioned previously that the home network middlewares for the operation of home appliances cannot be interoperable due to the different protocols and execution mechanisms. A layer is required to abstract different middlewares into one for integration.

For this paper, we define an abstract layer called Common Description in this paper. This layer consists of the minimum number of components essential to home appliances to contain all the common parts of diverse middlewares. Appliances with an capability to operate under home network environment must have at least three components; Services Description, Service Method(Action), Service State. Service Description is a service specification which can be understood by human beings and Service Method is a function performed by the service. A appliances must have their own Service States and provide mechanisms that notify its status to the outside or assist the outside to recognize the status. It is possible for an appliance to interoperate with other appliances as long as this kind of mechanism is provided either from the outside or inside of the appliance. Currently, both UPnP and Jini have an architecture that supports this kind of mechanism.

When each middleware Agent discovers services in a network, after analyzing service names, interface names, argument names, and of the device status, it converts them to a Common Description format, and then sends them to Agent Manager. In addition to Common Descriptor, Agent Manager also defines other templates that agents follow. Because these templates are all defined by XML, they are independent of particular languages or OS. Agent and Agent Manager are reliable and stable because they communicate through TCP/IP. Moreover, by separating Agent from Agent Manager as shown in Fig. 3, the HOMI architecture which supports intercommunication through TCP/IP can solve the overhead problem which may occur in the centralized integrated architecture.

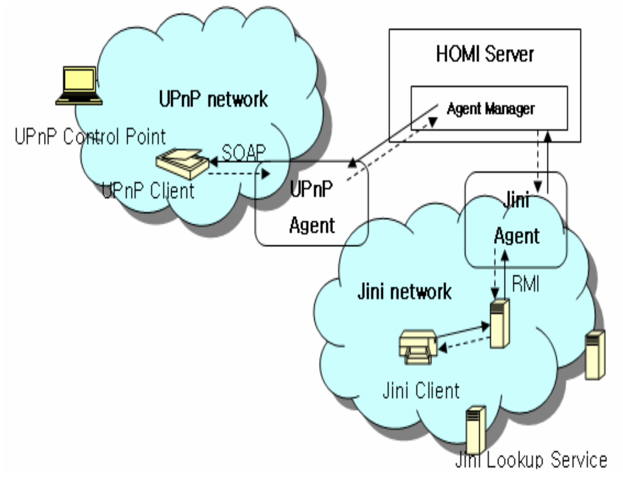

Fig. 3. Distributed Agent Architecture of HOMI

3.2 Standard Method Manager

A standard among heterogeneous middlewares is required for an interoperable service among heterogeneous devices in home network. Services represented with the same syntax elements shown in Section 2.2 do not necessarily mean that they are interoperable, but they are interoperable only when their semantic elements are also identical. Therefore, this paper attempts to implement interoperation among heterogeneous devices using standard methods. Standard Method Manager notifies each Agent of standard methods related to each middleware. The Agent creates a service if it is one of services provided by HOMI and defined by standard methods, and notifies it to its network. If information appliances look up and find an agent which supports a service needed, they can utilize the service provided by heterogeneous middlewares as if they were provided by homogenous middlewares. (cf. Fig. 4, Fig. 5)

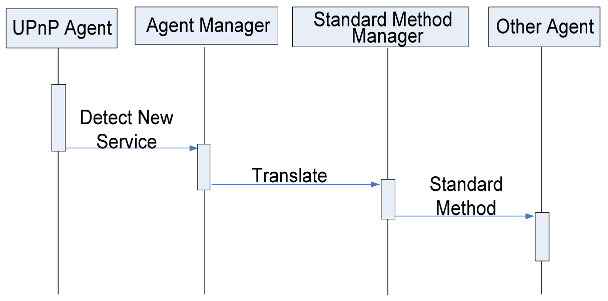

Fig. 4. Transmission Diagram of Standard Method

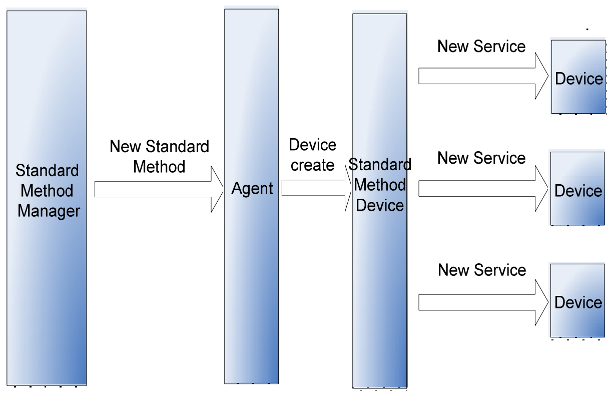

Fig. 5. Transmission Diagram of Standard Method Command

3.3 State Manager

A state of devices is an important element for the interoperation of appliances on the home network. As noted in the scenario in Section 2.3, when the status is changed after the execution of a service of a particular device, it acts as an event to trigger the next scenario. To solve problems caused by the difference between state advertisement mechanisms[1][2] of each middleware, each Agent in HOMI monitors the status of appliances under its control and notifies State Manager of the HOMI server of the result.

3.3.1 State Transmission Mechanism of UPnP

The state advertisements of UPnP adopts event-based Publisher/Subscriber scheme. When a new service is found, UPnP requests for a subscription to the device to monitor the status of the discovered device and receives SID(subscription identifier) as a result. If the subscription request is successful, UPnP Agent is notified of the changed status of the subscribed device and sends the result back to HOMI State Manager.

The state transmission mechanism of UPnP is described below. (cf. Fig. 6)

1. UPnP Agent requests a subscription so that status changes of devices to be monitored can be notified.
2. If the subscription request is succeeded, UPnP Agent receives SID(subscription identifier) as a result.
3. If the status of the UPnP device is changed, the device notifies all its clients that have sent subscription requests of the event.
4. UPnP Agent which has been notified of the status changes sends State Manager the value of changed status through Agent Manager.
5. State Manager updates the table of state variables using the values of changed status.

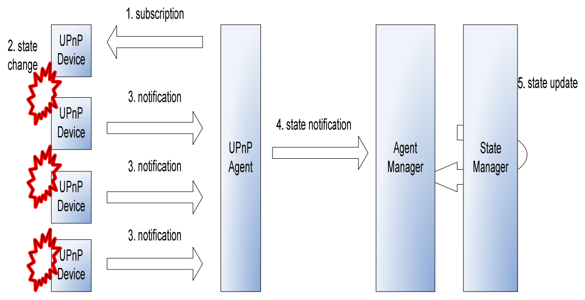

Fig. 6. State Transmission Mechanism of UPnP Agent

3.3.2 State Transmission Mechanism of Jini

There have been no standard mechanisms to transfer the status in Jini. To solve this problem, Jini makes it possible to notify status changes of a device using attributes[2] provided by Jini. State Manager of HOMI is notified of status changes through events that occur when the service attributes change.

The state transmission mechanism of Jini Agent can be summarized as below. (Fig. 7)

1. Jini Agent registers Listener to Jini Lookup Service in order to be notified of status changes of devices to be monitored.
2. If the registration request is succeeded, serviceChanged Event is transferred to Jini Agent.
3. When status is changed, the Jini device notifies all its clients, which have requested registration to itself, of the event.
4. Jini Lookup Service notifies clients, which have registered Listener to device changes, that the service has been changed.
5. Jini Agent, which has been notified of status changes, informs State Manager of the changed value of status through Agent Manager.
6. State Manager updates status variables using the changed value of status.

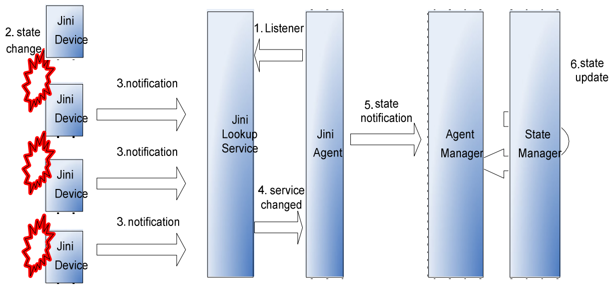

Fig. 7. Status Transmission Mechanism of Jini Agent

3.4 HOMI Language(HOMIL) Analyzer

It is sufficiently feasible for users to construct home network usage scenarios by themselves because their scope is limited to home appliances and the range of usage context is not broad unlike ubiquitous environment. In the respect of these properties of home network environment, we have designed a script language called HOMIL(HOMI Language) for the convenience of users when constructing scenarios. We are now working on developing a scenario builder as well which assists scenario construction more conveniently. HOMIL is an event-driven processing type of script language with an ability to design complex interoperable scenarios with a stream of successive events occurred based on a specific event. A scenario constructed with HOMIL is sent to HOMI after being converted into a XML format through the HOMIL parser.

Table 2. HOMIL Usage Example

```
a) execute Aservice.Method1
b) execute Bservice.Method2 when time == 7:00
c) execute Cservice.Method3, Dservice.Method4 if
   Bservice.state1 == Xstate
```

Table2 shows pseudo-code using HOMIL. To be specific, a) is a command using only *execute clause*, which executes *Aservice's Method1*. On the other hand, a **when** *condition clause* in b) specifies a specific time to trigger *Bservice's Mehtod2*. c) demonstrates that, when the status of *Bservice* becomes *Xstate*, *Method2 of Cservice* in *execute clause* and *Method3 of Dservice* are triggered asynchronously.

3.5 Context/Event Manager

Contexts of home network for supporting home automation can be regarded as conditions which affect the execution of a series of successive scenarios. We have classified contexts into *Time, Synchronization,* and *Asynchronization*. For example, a scenario "the alarm clock turns on at 7:00" is related to the time context.

Context Manager and Event Manager always operate in pairs. After classifying scenarios according to contexts, Context Manager transmits them to Event Manager. Event Manager manages scenarios for each context with queues and triggers an appropriate scenario when the conditions in the context of the scenario are satisfied. The result of the executed scenario changes the status of the device, and the table of state variables of State Manager is updated due to the changed status. This change might generate an event to trigger the next scenario.

3.5.1 Types of Contexts

Three types of contexts are provided by HOMI as below.

1) Time Context

This context means that a specific event is triggered by time and is controlled and activated by the global time management module in HOMI. The time context is represented with a *time condition* in HOMIL as below.

- execute Aservice.Method1 when time == 7:00

2) Synchronization Context

This context means that a specific event is triggered when the result of the *condition clause* in a command is true. This context is represented with a *condition clause* in HOMIL as below.

- execute Aservice.Method1 if Bservice.State1 == Xstate and Dservice.State2 == Ystate

3) Asynchronization Context

This context means that a variety of services runs in parallel under specific conditions regardless of their actual ordering. It can be represented as below.

- execute Aservice.Method1, Bservice.Method2 if Cservice.State1 == Xstate

4. Standard Method and Scenario Test

HOMI proposed in this paper as a standard method-based user-oriented integrated architecture can modify scenarios easily according to user's demands. We have constructed a testbed as Fig. 8. in order to demonstrate the interoperability among heterogeneous services for supporting home automation.

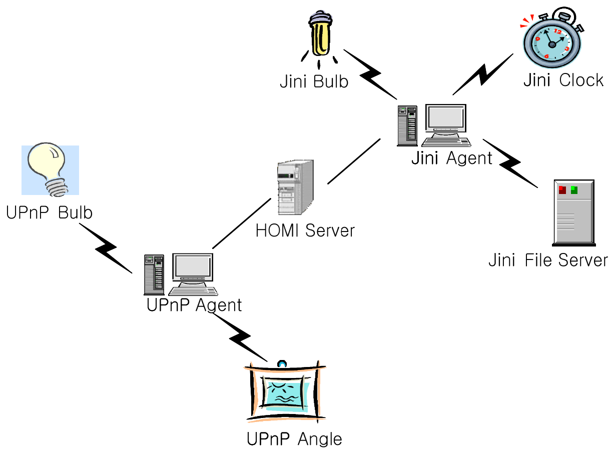

Fig. 8. System Configuration of Testbed

In order to demonstrate the interoperability among heterogeneous middleware services, we first developed applications as Jini devices to simulate Jini-Clock which allows users to set the time and turn on and off the clock, Jini-File Server which stores and retrieves files, and Jini-Light which can remotely turns on and off the light. We also developed a UPnP digital picture frame on PXA-255-based ARM board as a UPnP device and used UPnP Network light[17] provided by Intel as a UPnP bulb.

In order to test a standard service, we utilize a scenario in which UPnP-Digital Picture finds a picture in Jini-File Server and displays it. We register the format of a picture file at the Jini File Server and a scheme to access it as standard methods. As a UPnP Agent registers getFile of Jini-File Server as a standard method, a UPnP device can find it.

A scenario for home automation used in this test is given as shown below.

"When Jini-Clock strikes 8:00:00, UPnP-Light and Jini-Light are turned on. When UPnP-Light is on, one of the new stored photos replaces the old one periodically."

This scenario is represented using HOMIL as shown in Table 3.

Table 3. A Scenario Represented by HOMIL

Execute Light.SetTarget(TRUE), JiniLight.PowerOn() when time == 8:00 Execute UPnPAngle.AutoOn if Light.state == TRUE

Fig. 9 shows the result of XML conversion for the scenario.

```
<?xml version="1.0"?>
<root xmlns="konan:schemas-homi-org:service">
<statementlist>
  <statement>
    <asynccontext>
      <service>
        <name>Light</name>
        <uuid>3081</uuid>
        <action>SetTarget</action>
        <returntype />
        <argumentlist>
           <argumenttype>bool</argumenttype>
           <argumentvalue>true</argumentvalue>
        </argumentlist>
        <stateVarName>target</stateVarName>
      </service>
      <service>
        <name>JiniLight</name>
        <uuid>3082</uuid>
        <action>PowerOn</action>
        <returntype>void</returntype>
        <argumentlist />
        <stateVarName>status</stateVarName>
      </service>
    </asynccontext>
    <timecontext>
      <hour>8</hour>
    </timecontext>
  </statement>
  <statement>
    <asynccontext>
      <service>
        <name>UPnPAngle</name>
        <uuid>3083</uuid>
        <action>AutoOn</action>
        <returntype />
        <argumentlist />
        <stateVarName>IsAuto</stateVarName>
      </service>
    </asynccontext>
    <synccontext>
      <service>
        <name>Light</name>
        <uuid>3081</uuid>
        <action>GetStatus</action>
        <returntype />
        <argumentlist />
        <stateVarName />
      </service>
      <operator>equal</operator>
      <operand>
        <type>string</type>
        <value>true</value>
      </operand>
    </synccontext>
  </statement>
<statementlist>
</root>
```

Fig. 9. Test Scenario Converted to XML

The execution procedure of the test scenario can be described as below.(See Fig. 10)

1. If a new scenario is delivered, it is analyzed.
2. An event of the analyzed scenario is registered.
3. At 8:00, the registered events(UPnP-Bulb On, Jini-Bulb On) are triggered executed.
4. When an execution command is delivered from State Manager, it finds a service in Common Description Repository and requests its execution to an appropriate Agent.
5. Agent parses a command transmitted from Agent Manager and requests its execution to a device registered at Agent.
6. The device notifies Agent of its status(UPnP-Bulb On, Jini-Bulb On) changes as a result of the execution.
7. Agent Manager notifies State Manager that the status of the bulb has been changed to On

As the status of the UPnP bulb is changed to On, the next event which triggers an automatic change of the UPnP digital picture frames. The process is identical as above.

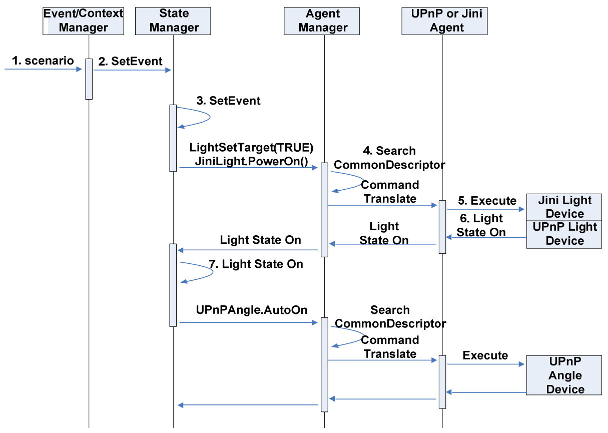

Fig. 10. Transmission Diagram of Test Scenario

The experiment proved that heterogeneous devices operate successively interacting with each other following the constructed scenarios. Also, we could make sure that a user can manage each device efficiently and conveniently using web services.

5. Conclusions and Future Works

For the efficient utilization of home network, problems raised by the heterogeneity of middlewares must be solved. Also, it is very important to support the reconstruction of diverse home automation services according to the user's need through the interoperation of heterogeneous middlewares.

In this paper, we have proposed a standard method-based user-oriented integrated architecture in order to provide interoperation among heterogeneous devices and meet the demand of home network users for various home automation services. Unlike the existing integrated middleware architectures, HOMI has promoted the interoperability of heterogeneous devices by not defining a standard for home network service but defining standards for each method. It is possible to transfer data and interoperate among heterogeneous middleware installed on information appliances by this architecture. And for the interoperability among heterogeneous appliances HOMI classifies operation services of users into three contexts which are *Time Context*, *Synchronization Context*, and *Asynchronization Context*. It has an architecture which supports to execute the next service according to contexts when a specific event occurs. This kind of architecture provides a convenient and efficient environment which allows users to reconstruct services in flexible ways. Also, HOMI assists users with seamless services by applying modified scenarios to home automation environment in real-time without installing new applications, updating the server, or rebooting in order to adapt new home network usage scenarios. Lastly, we solved the overhead problem occurred in a centralized architecture of integrated middlewares by distributing Agents.

We need to extend contexts for the execution of home automation services under diverse environments and HOMIL to support the extended contexts as well. Also, in order to integrate heterogeneous home network middlewares, fault tolerance of the HOMI server using a centralized mechanism is a major issue. A fault tolerance system which can operate when the HOMI server does not operate by recognizing high-performance appliances under home network and distributing important services into high-performance appliances should be investigated in the future. Finally, a scenario builder to support users to construct scenarios conveniently and new APIs for developers to quickly develop new Agent for middlewares are in need.

References

[1] UPnP Forum. http://www.upnp.org.

[2] Sun Microsystems. Jini Architecture Specification. http://www.sun.com/jini/.

[3] The Havi Organization, Havi Version 1.1 Specification. http://www.havi.org.

[4] Echelon Co., LonTalk Protocol Specification, Ver 3.0, 1994.

[5] E. Guttman, C. Perkins, J. Veizades and M. Day, Service Location Protocol, Ver 2, 1999.

[6] B Rose, "Home Networks: A Standards Perspective," IEEE Communications Magazine, pp. 78-85, Vol. 39, December 2001.

[7] G. O'Driscoll, The Essential Guide to Home Networking Technologies, Prentice-Hall, 2001.

[8] S. Huhns, Service-Oriented Computing, WILEY, 2005.

[9] B. Guillaume, R. Kumar, B. Helmut, and S. Thomas, "Methods for Bridging a HAVi Sub-network and a UPnP Subnetwork and Device for Implementing said Methods," Thomson Multimedia, 2002.

[10] J. Allard, V. Chinta, S. Gundala, G. Richard III ,"Jini Meets UPnP: An Architecture for Jini/UPnP Interoperability," Symposium on Applications and the

Internet, pp. 268-275, January 2003.

[11] D. Box, "Simple Object Access Protocol 1.1" available at URL http://www.w3.org/TR/SOAP/.

[12] E. Tokunaga, H. Ishikawa, M. Kurahashi, Y. Morimoto, and T. Nakajima, "A Framework for Connecting Home Computing Middleware," ICDCSW, pp.765-770, July 2002.

[13] OSGI Alliance. http://www.osgi.org/.

[14] K. Moon, Y. Lee, Y. Son, and C. Kim, "Universal Home Network Middleware Guaranteeing Seamless Interoperability among the Heterogeneous Home Network Middleware," IEEE Transactions on Consumer Electronics, Vol. 49, August 2003.

[15] A. R. Ponnekanti and A. Fox. "Application Service Interoperation without Standardized Service Interfaces," Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, pp. 30-40, March 2003.

[16] KISDI, home network market analysis and prospect, 2003.12.

[17] Intel® Software for UPnP Technology. http://www.intel.com.

Authors

Hark Jin Lee

Received the B.S. degree from the Chung-Ang University in 2005, Currently, he is a M.S. of the school of Computer Science and Engineering at Chung-Ang University. His research interests include embedded system, home network, and Linux system.

Sung Jo Kim

Received the B.S. degree from the Seoul National University in 1975, the M.S. degree from the Korea Advanced Institute of Science and Technology in 1977, the Ph.D. degree from the University of Texas at Austin in 1987, respectively. Currently, he is a Professor of the school of Computer Science and Engineering at Chung-Ang University. His research interests include embedded system, mobile computing system, home network, and Linux system. He is a member of IEEE and ACM.