A Framework for Connecting Home Computing Middleware

Eiji Tokunaga, Hiro Ishikawa,
Makoto Kurahashi, Yasunobu Morimoto,
Tatsuo Nakajima
Department of Information and Computer Science,
Waseda University
3-4-1 Okubo Shinjuku Tokyo 169-8555 JAPAN
{eitoku, ishikaw, mik, morimo, tatsuo}@dcl.info.waseda.ac.jp

Abstract

In the future, micro processors will be embedded in various appliances such as home appliances, digital AV appliances, and personal appliances. These appliances will be connected to various types of networks, such as Internet, and communicate with each other. The communication of appliances would integrate some services provided by these appliances and make new services.

The future home computing environment requires home computing middleware on which we can control home appliances easily and develop new services without a great effort. Most of recent middleware for home computing such as Jini and HAVi has no compatibility and adaptability with each other. Therefore it is not easy for these appliances to communicate with other appliances which are controlled by any other middleware. Although there are some protocol bridges such as bridge between HAVi and Jini by Philips, Sony and Sun, these are almost only bridges of one and one. We need a framework with which we can integrate any middleware in a simpler way.

In this paper, we propose a framework for connecting home computing middleware. It enables any appliance under any middleware's control to communicate any other appliances. We show also a service integration framework with Internet service and home computing middleware.

1. Introduction

In the near future, various appliances around us will have microprocessors inside. In our home, micro processors would be embedded not only digital AV devices such as TV and VCR, and usual home appliances such as refrigerators and cleaners but also clothes and tableware. These computers will become invisible from us, and the use of home appliances become more comfortable.

Since embedded systems in these appliances have restricted resources and limited functions, they cannot provide effective services without cooperating with other servicesconnected by various types of networks. Accordingly, we need middleware components that provide high level abstraction and hide complex distribution, like Jini and HAVi. A programmer can build a home computing application without taking account distribution by writing it in the high level abstraction.

Various middleware will be working in the future home. Because there will be various types of networks such as Ethernet, Bluetooth and IEEE1394, and various devices such as PDAs, digital AV devices and other types of home appliances such as microwaves. New middleware has been proposed whenever new requirements are found, and we will have many home computing middleware in the future. Such middleware diversification causes two difficult problems. The first one is less interoperability of services which use different middleware each other. For instance, Jini-based client canft access HAVi-based service without any difficulty. The second one is less deployability of a service which uses several kinds of middleware. If we want to deploy a service which uses Jini-based service and HAVi-based service, this service have to use both of clients, Jini and HAVi.

We want to use and deploy services of home appliances without special conscious of diversification of networks forms and that of middleware. For example, letfs think about a smart home there are some kinds of networks and middleware which are a HAVi-based IEEE1394 network connecting a digital TV and VCR, a Jini-based Ethernet network connecting a refrigerator and an air conditioner. In there, we want to control the TV, the VCR, the refrigerator and the air conditioner from a PC without being conscious of heterogeneous forms of network and middleware. Moreover, we want to control these appliances from

Proceedings of the 22 nd International Conference on Distributed Computing Systems Workshops (ICDCSW'02)
0-7695-1588-6/02 $17.00 © 2002 IEEE

the GUI of the digital TV too. In such a case, the service discovery and the protocol conversion between Jini and HAVi should be managed out of userfs consciousness.

To accomplish the goal described above, we have to integrate heterogeneous middleware. In this paper, we propose a framework for connecting home computing middleware. Our framework provides an idea of a desirable environment in which services, based on any middleware, can connect any other services transparently. And also new middleware can be conneted in this framework effortlessly.

The reminder of this paper is structured as follows. In Section 2, we describe what service integration is. In Section 3 presents the design and the implementa tion of our framework. In Section 4, we describe current implementation status and experience. At the last, Section 6 concludes this paper with future work.

2. Service Integration

In this paper, we define a service as both of a function of an appliance connected via a certain form of a network and a network application provided by some servers, such as aWorld Wide Web service. A single service has a very restricted functionality because an appliance has very restricted resources [4]. A single service can increase its value by cooperating with other services. We define the service integration as making a new service from more than one service cooperating with each other.

For example, let us think a digital VCR which has a microprocessor and connected to a home network. It wouldn't be very useful to control the VCR from a PC by using a specialized application. But, if it is integrated with Internet services such as a TV program service and a video streaming service, its value will be extremely increased. For example, the service integration of a VCR control service with a TV program service on the Internet can provide an automatic video recording service that records TV programs according to user profiles on the Internet. Also, the integration with a video streaming service makes new service which records video streams on the Internet in local video tapes. They seem to be more useful than a VCR remote control service.

2.1. The Issues of Service Integration

One of the middleware, which aims to the service integration described in the previous section, is Jini [6]. Jini enables various computer devices such as embedded devices, handheld devices and PCs to be cooperated. Jini calls the cooperation "federation". The federation of devices integrates various functions provided by Jini-based devices. Thus, Jini defines service (Jini service) as the function of a device connected to a Jini network and the device itself. Jini runs on the Java framework.

HAVi [9] is also one of home computing middleware. The HAVi organization was founded by eight promoter companies. HAVi is a digital AV networking middleware that provides a home networking software specification for providing seamless interoperability among home entertainment products. Actually, the focus of HAVi is on the control and content of digital AV streams. IEEE1394 which is a high speed network for computing has been chosen to connect home appliances.

Service integration middleware such as we describe has problems to integrate all services over the world. For example, Jini is a Java-based system so that Jini devices must have the Java runtime. And, Jini is not good for integrating critical services such as multimedia streams because of Java's low performance and APIs. Also, the current implementation of Jini depends on the Java RMI technology in order that the Jini-based network to have to support RMI. So Jini can be used on a certain type of a network such as TCP/IP. And, HAVi, which is developed for digital AV devices, has various APIs to control digital AV devices and contents, but it doesn't have APIs to control information appliances like PDAs or cellular phones. Also, the network target of HAVi is only IEEE1394 networks. Thus, any legacy single middleware cannot integrate all services. Because there are many network protocols, data protocols and hardware protocolsaround us.

Furthermore, it is impossible to develop novel perfect middleware that integrates all services over the world. The diversification of protocols will be increased in the future. Because embedded devices, which will be increased unquestionably, will want to go on original platforms since the original functionality acquired by making original platforms have more important than the compatibility acquired by using legacy protocols and middleware. So, integrating all services over the world by the single perfect middleware is impossible.

2.2. Service Integration Middlew are

Although it is unrealistic to integrate all services described in the previous paragraph, we want to integrate various services which are under different middleware's control respectively. Since legacy middleware, such as Jini and HAVi, is useful enough to integrate certain services. Jini is an appropriate middleware to integrate Java-based services. HAVi is also appropriate to integrate AV-centric services on the IEEE1394 network. Jini and HAVi provide easy environments to deploy certain services for us. And there would be already legacy services using legacy middleware. We don't want to modify legacy services to access any other middleware. Then, we propose a framework for connecting

home computing middleware which integrates legacy middleware such as Jini and HAVi. The integration of legacy middleware connects a variety of services which are under different middleware's control. In this framework, we can use appropriate middleware to deploy certain services. For example, if we want to deploy a Java-based service at home, we can use Jini locally and this service can access other services which are controlled by other middleware. Also, we apply to integrate with the most important service middleware on the Internet.

3. Our Framework

The simple design goal of our framework is described below.

- We can use legacy service with legacy middleware easily.
- It is not necessary to change legacy clients and services.
- New middleware can be participated in our framework effortlessly.

Figure 1 illustrates the basic concept of our framework. As we can see from this picture, each middleware network has the Virtual Service Gateway which connects one middleware to another middleware using a certain protocol, the Protocol Conversion Manager which converts the local middleware protocol to the protocol of Virtual Service Gateway, and the Virtual Service Repository which records locations and functions of services. The next sections describe each component in detail.

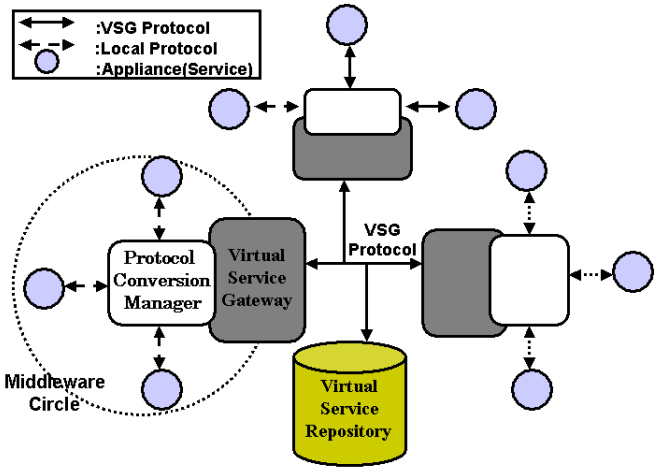

Figure 1. Connecting Middleware

3.1. Virtual Service Gatew ay

The Virtual Service Gateway (VSG) is a gateway which connects middleware to another middleware using certain protocol which decides the information of services such as interfaces, locations and data. How the protocol should we chose is demands on the purpose of service integration. To integrate some particular services such as a multimedia stream and a voice communication, the protocol must be able to connect streams and transport multimedia data. Besides, a simple protocol is enough to integrate simple services. We implement the prototype of our framework with SOAP [14], a simple protocol. We describe the reason of that in section 4 in detail.

3.2. Protocol Con version Manager

The Protocol Conversion Manager (PCM) converts the protocol of a local middleware component into that of VSG, also VSG into a local middleware component. The PCM has two proxy modules, the Server Proxy module and the Client Proxy module as depicted in the Figure 2. The Server Proxy (SP) module provides the interfaces of remote services to the local services. Then, Client Proxy (CP) converts the interfaces of local services into the VSG services. Remote clients request the VSG services and CP invokes local services in order to transmit the request of remote client. For example, if the VSG protocol is SOAP and a local middleware component is Jini, the SP converts SOAP services into Jini services and the CP converts Jini services into SOAP services. Then, a Jini client sends a query to SP, and it calls a remote SOAP service through VSG. And a remote SOAP client (VSG) sends a query to CP through VSG, and it calls a Jini service.

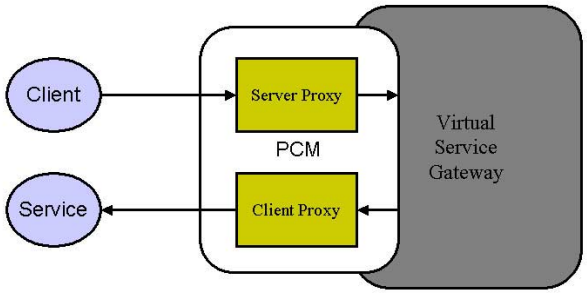

Figure 2. Proxy Modules

3.3. Virtual Service Repository

The Virtual Service Repository (VSR) is a virtual database which has a lot of information of heterogeneous services such as service locations and service contexts. The VSG and the PCM use this component to detect services or aware contexts. The implementation of this component demands on the protocol of VSG. For example, if the protocol of VSG is SOAP, the VSG will be implemented with WSDL and UDDI.

Using this framework makes it possible to use and integrate legacy services. Legacy clients and services don't have to special operations to communicate with other services on any middleware. Then, expected middleware can be used for legacy services.

4. Implementation and Discussion

This section describes the implementation of a prototype system to verify our framework. We have implemented our prototype system based on commodity software, which is recognized generally and ported various platforms such as embedded devices and PCs. For example Java and Linux are commodity software.

To develop portable software with low cost is extremely important because embedded devices may need to be executed on a lot of heterogeneous platforms. And middleware and software do not run within only PCs but such appliances. We have implemented our prototype system with Java and Linux. Java is a portable language which is appropriate to implement complex middleware. Also, Linux, which runs on various CPUs, and is easily ported to new platforms since the Linux community has strong software development ability, is suitable for future embedded systems.

4.1. Current Implemen tation Status

In the current implementation, we have used Apache SOAP, a product of IBM developer works, for VSG. Currently, the protocol of VSG is SOAP as described before. SOAP, which is an XML/HTTP-based protocol, has several advantages described below.

- It is simple protocol so it is easy for implementation and light-weight for network.
- It uses HTTP which has high-quality scalability based on the Internet.
- It uses existing infrastructure that does not depends on paticular company such as HTTP and XML.

The prototype system using our framework has four types of PCM that of Jini, X10 [15], HAVi and Internet Mail service as shown in Figure 3. Automatically we can generate a proxy object, such as client proxy and server proxy, for certain service using the interface of that service. The proxy automatic generation is implemented by Javassist [10] that is a load-time reflective system for Java. It is a class library for editing bytecodes in Java; it enables Java programs to define a new class at runtime and to modify a class file when the JVM loads it.

Through these proxy objects, each client can access any other services transparently as the services are implemented on the same middleware. For example, we describe the transaction and the protocol conversion between Jini and X10 as shown in Figure 4. Of course, each client of these middleware can access the SOAP Web service transparently too. Currently VSR has been implemented by WSDL which is an XML format for describing network services and Universal Description, Discovery and Integration (UDDI) which is used to describe the repository.

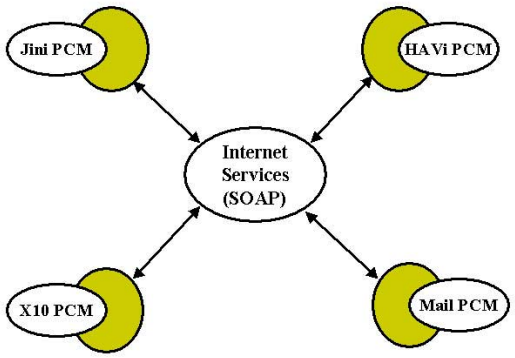

Figure 3. Protype of Integration System

4.2. Prototype Technical Experiences

We've developed several applications using our middleware. One of the applications is Universal Remote Controller. It is an X10 remote controller that allows us to control not only X10 devices but also Jini and HAVi services that are connected via our middleware. The picture is depicted in Figure 5. The person in the picture is controlling a Jini Laserdisc with an X10 remote controller, and he can also control a HAVi DV camera. We could develop this application without any difficulties since VSGs and PCMs hide the differentiation between these middleware.

Besides, we have tried to develop the event-based multimedia system, which manages multimedia streams and send

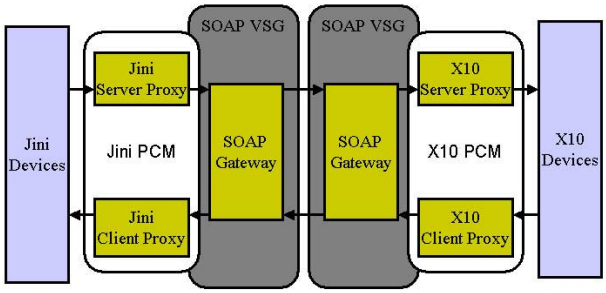

Figure 4. Conversion between Jini and X10

multimedia data to appropriate I/O devices, with X10 motion sensors and HAVi and Jini AV systems. But, there are some difficulties such as multimedia data conversion and dynamic service activation because of the limitation of HTTP. HTTP is inherently a client/server protocol, which does not map well to asynchronous notification scenarios.

Additionally, current HTTP must run over TCP, and a TCP stack is large and complex. This can be an issue in small devices or appliances with stringent memory and processing requirements.

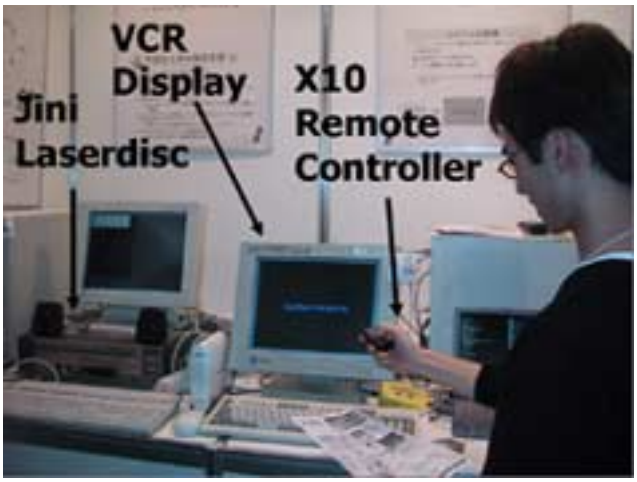

Figure 5. Universal Remote Controller

5. Related Work

Royal Philips Electronics, Sony Corporation and Sun Microsystems have collaborated to bridge the HAVi and Jini network architecture. The companies aim to provide a solution that links HAVi appliances to services provided by the Jini technology. It would allow not only digital AV appliances to access remote network services, such as a video storage service, but also allow users to remotely operate digital AV appliances across a Jini network. The experiment has solved some problems, described above, but most of middleware diversification problems remain yet. In the future, new middleware will be developed one after another. So it is not enough to develop a single bridge that connects two specific middleware one to one. So the framework for the integration of home computing middleware is necessary, like we described in this paper.

Universal Plug and Play (UPnP) [3] project at the UPnP forum defines common protocols and procedures to guarantee the interoperability among network-enabled PCs, appliances, and wireless devices. But, as we described below, it is impossible that UPnP connects all services over the world because the protocol will not be adpoted for all future appliances. We can connect the UPnP service to other middleware by developing a PCM for UPnP.

JXTA [7] technology by Sun Microsystems is a set of open, generalized peer-to-peer protocols that allow any connected devices on the network from personal appliances such as cell phones and PDAs to desktop computers such as PCs and servers to communicate and collaborate in a peer to peer manner. The concept of the peer-to-peer is very interesting. We should think about the service integration of critical peer-to-peer services such as a file sharing service.

There are several projects [8] [5] using SIP [2] to connect networked appliances which are connected to several middleware such as Jini, UPnP and X10. SIP allows abstract naming, provides end-to-end security, and can cary a flexible payload. These features are suitable to connect end-to-end services. Also, SIP supports asynchronous calls and call forwarding which is not supported by HTTP. We think that is also effective choice to use SIP with some modification to connect various appliances. And it is interesting that SIP enables us to integrate VOIP services such as Internet Telephony easily. SIP may be more suitable than other protocols such as HTTP for service integration. But the problem is few popularization of SIP.

6. Conclusion and Future Work

This paper has presented a conceptual framework which integrates home computing middleware. In comparison to related activities our framework provides the transparent access to many services and the easy integration of new middleware. With our approach, we can select the most appropriate middleware to develop specific services. For example, we can develop a DV control service on IEEE1394 with HAVi and make it possible this service to communicate with other middleware such as Jini and X10. Moreover, new

middleware can participate in our framework smoothly, by developing new PCM which converts the middleware protocol to VSG protocol. This type of middleware, such as our prototype, is a kind of Meta middleware. We can connect home computing middleware easily on our Meta middleware framework.

However, as described before, we will not be able to solve some problems by our prototype. For instance, we can't integrate of multimedia streaming and dynamic service activation. But it is impossible to solve all problems by single Meta middleware as same as it is impossible to connect all services by single middleware. And we don't have to solve all problems at the same time. That is, we think another Meta middleware should be developed for some critical applications such as multimedia services and sensing services. We are working on the deployment of novel CORBA-based middleware which applies dynamic service activation, conversion of multimedia streams for multimedia application and interface mapping of physical object with sensing devices. And the middleware would be able to coexist with our framework described in this paper, at the same area. We aim to integrate some services on demand by these Meta middleware. We will continue to do work on middleware which connects and integrates legacy middleware.

References

[1] D.A.Norman. "The Invisible Computer". *The MIT Press*, 1998.

[2] M. Handley et al. "SIP: Session Initiation Protocol". *IETF RFC 2543*, Mar. 1999.

[3] UPnP Forum. "Universal Plug and Play". http://www.upnp.org.

[4] H.AIzu, I.Sato, D.Ueno, and T.Nakajima. "A Virtual Overlay Network for Integrating Home Appliances". *SAINT-2002 The 2002 International Symposium on Applications and the Internet*.

[5] T. Kanter. "Adaptive Personal Mobile Communication, Service Architecture and Protocols". *Doctoral Dissertation, Royal Institute of Technology, Stockholm, Sweden*, Dec. 2001.

[6] Sun Microsystems. "JINI Architecture Specification". http://www.sun.com/jini/.

[7] Sun Microsystems. "Project JXTA". http://www.jxta.org/.

[8] Stan Moyer, Dave Marples, and Simon Tsang. "A Protocol for Wide-Area Secure Networked Appliance Communication". *IEEE Communication Magazine*, Aug. 2001.

[9] The Havi Organization. "Havi Version1.1 Specification". http://www.havi.org.

[10] Michiaki Tatsubori, Toshiyuki Sasaki, Shigeru Chiba, and Kozo Itano. "A Bytecode Translator for Distributed Execution of Legacy Java Software". *ECOOP 2001 –Object-Oriented Programming,LNCS 2072, Springer VerLag*, 2001.

[11] T.Nakajima. "System Software for Audio and Visual Networked Home Appliances using Stateless Thin-Client Architecture". *In Proceedings of the 2nd International Workshop on Ubiquitous Computing and Communication*, 2001.

[12] T.Nakajima, H.Ishikawa, E.Tokunaga, and Frank Stajano. "Technology Challenge for Building Internet-Scale Ubiquitous Computing". *In Proceedings of the Seventh IEEE International Workshop on Object-oriented Real-time Dependable Systems*, 2002.

[13] T.Nakajima, K.Soejima, M.Matsuda, T.Iino, and T.Hayashi. "A Framework for Building Audio and Visual Home Appliances on Commodity Software,". *In Proceedings of the IASTED International Conference on Internet, Multimedia Systems, and Applications*, 2001.

[14] W3C. "Simple Object Access Protocol (SOAP) 1.1". http://www.w3.org/TR/SOAP/.

[15] X10 WirelessTechnology, Inc. "CM11A programming protocol". ftp://ftp.x10.com/pub/manuals/cm11a_protocol.txt.