

Design and Implementation of SONICA (Service Oriented Network Interoperability for Component Adaptation) for Multimedia Pervasive Network

Hiroshi Hayakawa[†]

Takahiro Koita[‡]

Kenya Sato[‡]

[†]Graduate School of Engineering

[‡]Department of Information Systems Design

Doshisha University

1-3 Tatara Miyakodani, Kyotanabe-City, Kyoto, 610-0321, Japan

Abstract - *Recent advances in multimedia network systems have led to the development of a new generation of applications that associate the use of various multimedia objects. The connection of audio and video devices through high speed multimedia pervasive networks will make the use of these systems more convenient. For example, many home appliances, such as video cameras, display monitors, video recorders, audio systems, and so on, will be equipped with a communication interface in the near future. Recently some platforms (i.e. UPnP, Jini, HAVi and so on) have been proposed for constructing multimedia pervasive networks, however, there are still some issues to be solved before we can achieve various services that connect different equipment via the network. In case of UPnP based on SOAP, it is necessary to standardize a way to control any feature of a new (undefined) device connected to the network and update the software in the controller device. Meanwhile, Jini based on RMI technology, or HAVi developed for intelligent AV equipment, has a mechanism to upload software module from a new device connected to the network to the controller device. Although standardizing the new devices' features is not needed, HAVi devices require high CPU power and large memory. Considering that the targets of these home alliances are embedded systems, this situation raises issues of software and hardware complexity, cost, power consumption, and so on. In this study, we developed SONICA, a service oriented network architecture for the control and management of home appliances to address the issues described above.*

Keywords: *Pervasive Network, Multimedia, SOAP*

1. Introduction

As an increasing number of audio and video devices (e.g. video cameras, display monitors, video recorders, au-

dio systems, and so on) are becoming capable of connecting each other through high speed multimedia networks. Some platforms such as UPnP[1] adopted DLNA (Digital Living Network Alliance)[2], Jini[3, 4], HAVi[5] and so on, have been proposed for the construction of multimedia pervasive networks. However, the focus of these home appliance targets is on embedded systems, and these platforms are not able to solve embedded system-specific issues (i.e. software and hardware complexity, cost, power consumption, and so on). The technologies underlying UPnP Device Architecture include SOAP[6, 7, 8, 9] as a remote procedure call (RPC) mechanism. SOAP allows devices to offer complex and advanced services. However, each UPnP device must have a XML parser and needs to parse all communication messages described in XML. This results in inefficiency and requires additional software resources. In addition, with UPnP, it is necessary to standardize a way to control any feature of a new (undefined) device connected to the network, and update already installed software in a controller device to support the undefined device in advance. Jini network system that can automatically install communication interfaces requires interface standardization closely akin to HAVi's case. HAVi's sophisticated function offers connections among defined devices; however, the module must be loaded for use on an undefined device. This means HAVi requires additional memory and software resources to support this mechanism. To solve these issues, we have proposed and developed SONICA (Service Oriented Network Interoperability for Component Adaptation) for the control and management of home appliances over pervasive network. SONICA based on simple HTTP protocol with WebDAV technology[10], is a multimedia SOA (Service Oriented Architecture) application platform for use in embedded systems.

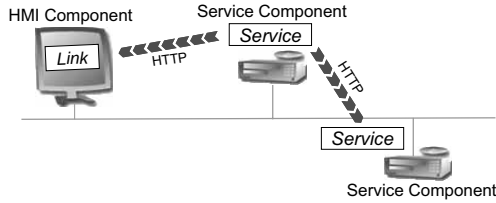


Figure 1. Device Interaction.

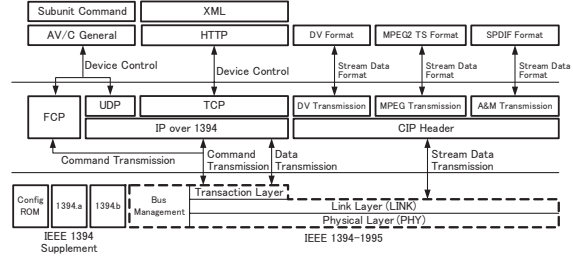


Figure 2. Protocol Stack.

2. Network Architecture

2.1. Device Interaction

In SONICA, devices interact with each other using the partner's service. Figure 1 shows an outline of device interaction in SONICA. When a device (service component) is connected to the network, functional information (including link information for performing services) offered by the connected device is sent to other devices. In this case, only functional information is sent and the service module is never sent. Another device (HMI component) generates a menu with the link information received from the connected service component. When a user selects a service on the menu, the HMI component that has the menu sends a message to the requested service component, and the service component provides the requested service. The user does not have to know from where the service is provided; only that it is available on the HMI component. Devices, such as HMI components do not need to implement complex software modules such as the service mechanism, running processes, other machine architectures. With this kind of information based on the SONICA mechanism, there is no need to define service control messages in advance, and there is no difference for any devices, whether the method of accessing a device is defined or undefined. As a result, the system can be easily constructed with Plug & Play while using the hardware resources effectively, and network traffic can be suppressed. UPnP and HAVi adjust to undefined devices by loading or updating software modules, however, it is difficult to support many kinds of devices because the device's resources are limited.

The SONICA interaction mechanism is based on HTTP verbs (e.g. GET, POST, PUT, and DELETE). The service based on HTTP verbs is simple and can be easily used from other devices without specific software modules. In addition, WebDAV technology is used to achieve interactions between devices.

2.2. Components

In SONICA, the system is composed of a human machine interface component (HMIC) as a GUI (graphical

user interface) and other components called service component (SC). An SC consists of an HTTP server function and some of the services provided by equipment such as a DVD recorder, a tuner, an audio amplifier, and a video camera. Those services are provided to a user after the user makes a request to an HMIC. A single physical device may contain one or more components. Each component keeps tabs on the device components when a device is connected or removed. To use limited bandwidth efficiently, a bandwidth management mechanism is needed. A resource manager reserves required bandwidth and assigns the bandwidth according to the device's demands. The SCs also become a resource manager or a DHCP server in the network.

An HMIC including XML browser function works as an interface. A user can operate SCs connected to the network by the interface. The HMIC generates a menu from the functional information offered by the SC. It is easy to control the components with user interfaces that have a high user affinity, such as GUI. However, it is possible to accept a simple user interface without graphical images (e.g. an ordinal remote controller) to control the SCs.

2.3. Network

In this study, SONICA is implemented to introduce IEEE 1394 (a.k.a. iLink or FireWire)[11, 12, 13] as a network interface, because the Ethernet is not capable of supporting the transport of real-time multimedia streaming data over the network, which is an essential function for home AV appliances. The IEEE 1394 is a serial interface that supports up to 800Mbps data transmission, and has isochronous transmission mode for real-time streaming data, as well as an asynchronous transmission mode. The IEEE 1394 allows SONICA to construct AV appliances network without a PC.

Figure 2 shows the protocol stack for SONICA. SONICA constructs an IP network using the IPover1394[14] protocol in order to realize high affinity with a PC and the Internet. To achieve QoS guarantees in the network, isochronous transmission mode is processed in the low layers up to link layer and can be accelerated by hardware.

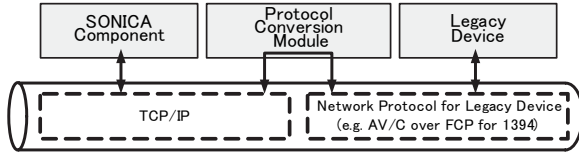


Figure 3. Legacy Device Support Mechanism.

2.4. Supporting Legacy Device

A legacy device that does not correspond to SONICA can be available using a protocol conversion module (PCM). Figure 3 shows how a legacy device that uses AV/C commands is used. In SONICA, HTTP is adopted to transmit requests and responses. PCM converts HTTP requests and responses into an original command acceptable for a legacy device. HTTP messages can coexist with other control protocol like FCP (function control protocol) or data transmission because IPer1394 uses asynchronous transmission mode. Thus, if a device using a PCM for AV/C commands is connected to the network, all devices connected to the network can use the device as PCM function.

3. Implementation

3.1. Components

We implemented all the previously described components on PCs (Fedora Core 3 with customized kernel based on 2.6.9, Pentium 4 3GHz) to evaluate the SONICA network architecture. The system for the evaluation consists of an HMIC, a contents manager as an SC and an IIDC PCM as PCM to controls the IEEE 1394 video camera (Apple's iSight). An HMIC generates a menu from functional information offered by the SC. For simplicity, the HMIC directly accesses the SC's control menu. The content manager offers a service to save images sent from other components. Servlet is used for functions such as generating a control menu and managing HTTP requests and responses. IIDC PCM converts HTTP requests and responses to IIDC requests and responses for IEEE 1394 digital cameras. The IIDC PCM allows iSight to connect to the SONICA network. We used iSight as a legacy device. We developed a library named libdc1394j in order to use the IEEE 1394 digital camera that controls the library called libdc1394 from Java. IIDC PCM uses libdc1394j and libdc1394 to converts HTTP messages into IIDC messages.

3.2. Libdc1394j

On Linux, libdc1394 is available as an IIDC control library. Libdc1394 is described in C Language and it is im-

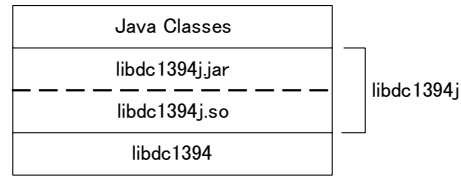


Figure 4. Libdc1394j Software Stack.

possible to use the library directly in Java. To use the library directly in Java, we implemented a shared library that includes a controlling mechanism described in the native language and a stub that loads this library from Java with JNI (Java native interface). This stub can be used as if the Java library. Figure 4 shows a libdc1394j software stack. Libdc1394j consists of the stub (interface of libdc1394 and original libraries). The original libraries include the image format translation method implemented in the native language. In libdc1394, error handling is strongly dependent on applications, however, it is difficult to manage errors occurred in the native methods. Thus, libdc1394j manages errors in the native methods, and informs the Java applications. Call by reference is heavily used in the original source code and JNI does not define the passing of pointer variables. In the Java application, we handle the pointer variables as integer variables.

3.3. Model Network

We constructed the network shown in Figure 5. iSight is a legacy IEEE 1394 digital camera that does not correspond to SONICA. HTTP requests and responses for iSight are managed by IIDC PCM, and IIDC PCM controls iSight according to the requests. IIDC PCM is a PCM for IIDC and does not save the images taken by iSight. To save images, IIDC PCM uses the contents manager service with a WebDAV function.

4. Evaluation

We actually implemented the functions of the SONICA network system and confirmed its behavior. Figure 6 shows an overview of the communication procedure within the SONICA network. The following sequence is our confirmation process.

- (1) A user accesses the IIDC PCM control menu that shows the functions and legacy components available now.
- (2) The user selects iSight from the menu. Using libdc1394j, the IIDC PCM servlet allows iSight to take a picture.
- (3) The captured picture and the components providing 'save' function are shown in the HMIC browser.
- (4)

Table 1. Evaluation of Transfer Method (SONICA and UPnP).

Method	SONICA	Serialized SOAP Envelope	Attachment
Detail	PUT method (WebDAV)	Serialize integer array with SOAP envelope	Attached SOAP message
Overhead	Small	Large	Medium
Transfer Time (Average)	20 ms	5 minutes	2 seconds

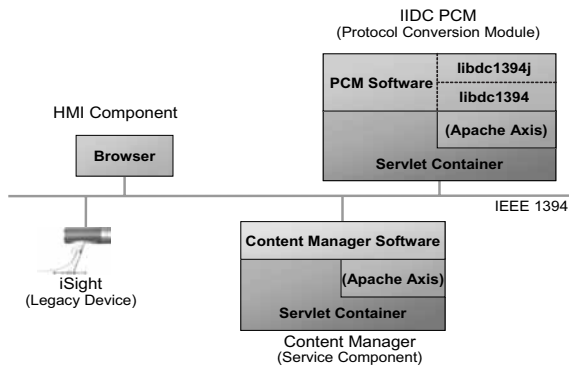


Figure 5. Model Network and Component Software Stack.

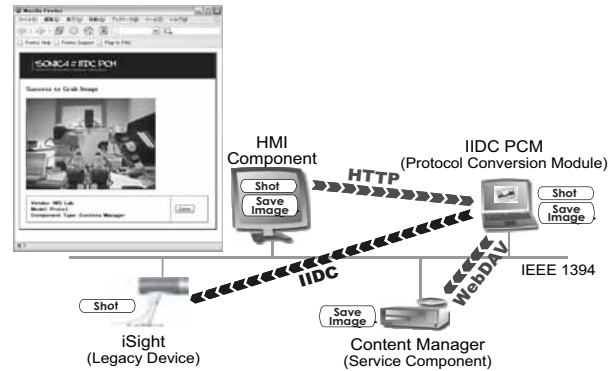


Figure 6. Communication Procedure.

When the user selects the content manager to save the image, IIDC PCM uses the content manager’s ‘save function’ and transfers the picture with WebDAV. (5) The user accesses the contents manager’s control menu. (6) The user confirms that the transferred image has been saved.

We implemented two kinds of the methods as well as SONICA to transfer the picture to evaluate the UPnP model based on SOAP. Apache Axis as a SOAP engine is implemented in the servlet container of IIDC PCM. The first method is the serialized SOAP envelope method. In this method, an array of pixel data are serialized and sent in this method. The second method uses an attached SOAP message. DIME + WS-Attachment are used as the attachment method.

As a result, we confirmed that the proposed model works as designed. Table 1 shows the result of the three transferal methods to transfer 320 x 240 pixels data. In case of SONICA with WebDAV technology, it takes 20 milliseconds to transfer the data. With the serialized SOAP envelope method, the integer array of 320 x 240 pixels data is transferred. This method requires 180MB heap memory, which cannot work under the Java VM default setting (64MB). Moreover, it takes about 300 seconds per transfer. This is because the SOAP envelope is expanded by a lot of tags in a serializing sequence. On the other hand, the

transferring method with an attached SOAP message takes 2 seconds.

5. Conclusion

The connection of audio and video devices with high speed multimedia networks is expected to make the use of these systems more convenient for home appliances. In this study, we proposed SONICA, a service oriented network architecture for pervasive network to address the issues of UPnP or HAVi that includes software and hardware complexity, and so on. According to this proposal, we implemented the model network on IEEE 1394, and confirmed the functions.

References

- [1] UPnP Forum, UPnP Device Architecture Version 1.0 (2000). http://www.upnp.org/download/UPnPDA_10_20000613.htm.
- [2] DLNA (Digital Living Network Alliance), Overview and Vision White Paper 2006 (2006). http://www.dlna.org/about/dlna_white_paper_2006.pdf
- [3] Sun Microsystems, Jini Network Technology. <http://www.sun.com/software/jini/>.

- [4] Jini.org, Jini Standards. <http://www.jini.org/standards/>.
- [5] Home Audio Video Interoperability (HAVi) Organization, White Paper, HAVi, the A/V digital network revolution. <http://www.havi.org/pdf/white.pdf>.
- [6] W3C Recommendation, SOAP Version 1.2 Part 0: Primer (2003).
<http://www.w3.org/TR/soap12-part0/>.
- [7] W3C Recommendation, SOAP Version 1.2 Part 1: Messaging Framework (2003).
<http://www.w3.org/TR/soap12-part1/>.
- [8] W3C Recommendation, SOAP Version 1.2 Part 2: Adjuncts (2003).
<http://www.w3.org/TR/soap12-part2/>.
- [9] W3C Recommendation, SOAP Version 1.2 Specification Assertions and Test Collection (2003).
<http://www.w3.org/TR/2003/REC-soap12-testcollection-20030624/>.
- [10] Internet Engineering Task Force (IETF), HTTP Extensions for Distributed Authoring and Versioning (Web-DAV), Request for Comments (RFC) 2518 (1999).
<http://www.ietf.org/rfc/rfc2518.txt>.
- [11] IEEE, IEEE Std 1394-1995, IEEE (1995).
- [12] IEEE, Standard for High Performance Serial Bus Amendment 1, IEEE std 1394a-2000 (2000).
- [13] IEEE, Standard for High Performance Serial Bus Amendment 2, IEEE std 1394b (2002).
- [14] Internet Engineering Task Force (IETF), IPv4 over IEEE 1394, Request for Comments (RFC) 2734 (1999). <http://www.ietf.org/rfc/rfc2734.txt>.