DESIGN AND IMPLEMENTATION OF HOME NETWORK SYSTEMS USING UPnP MIDDLEWARE FOR NETWORKED APPLIANCES

Dong-Sung Kim, Jae-Min Lee, Wook Hyun Kwon
ERC-ACI, School of Electrical Engineering and Computer Science, Seoul National University, Korea

and

In Kwan Yuh
DVN Team, DM Lab., LG Electronics Inc., Korea

Abstract

This paper describes the design and implementation of a home network system using UPnP middleware and an embedded interface device. The developed home network system comprises a home server program, appliance emulators (e.g., for TV, refrigerator, and air-conditioner), and an embedded interface device for networked home appliances.

Keywords: Home network system, UPnP middleware, Networked home appliances, Appliance emulators, Embedded interface device.

1 Introduction

A home network system is a collection of networked home appliances, which are considered to be distributed embedded systems that are connected by networks. In a home network system, various networked home appliances and a home server system communicates with each other and share a common interface.

Home network systems must have an automatic configuration that allows devices to join and leave the network, and to learn about other networked home appliances. In addition, home network systems should enable pervasive peer-to-peer network connectivity of networked home appliances for distributed and open communication. Therefore, a practical home network system requires reliable home server programs that employ a proven middleware that covers the requirements outlined above [1][2]. In addition, an embedded interface device is required to provide a common interface between the networked home appliances.

Previous works on home network systems have explored a message specification for white goods, a three-tiered architecture and interface technologies such as a user interface for portable devices, and the design of an interface between consumer electronic bus (CEbus) and Internet protocol [3]-[13].

However, these studies have focused on the software implementation of common interface technologies, message specifications, and architectures of home network systems without considering middlewares such as universal plug and play (UPnP) [14][15].

In the present study, we develop a home network system using UPnP middleware and a compact embedded interface device for networked home appliances. The requirements that must be met by UPnP in the developed home network system are presented in Section 2.

This paper is organized as follows. An overview of UPnP is presented in Section 2. The architecture of the home network system is described in Section 3. Section 4 explains the developed home network system including the home server programs, appliance emulators and embedded interface device. Finally, conclusions are presented in Section 5.

2 UPnP Middleware in a Home Network System

In the present study, a home network system was developed by using UPnP middleware. The following requirements are discussed for the developed home network system.

- Automatic configuration
- Peer-to-peer network connectivity
- Use of common protocol stacks
- Support of various operating systems
- Availability of hardware resources

The UPnP architecture supports automatic-configuration, which allows networked home appliances to automatically join and leave the network as well as to learn about newly connected home appliances. In addition, when automatic-configuration is used networked home appliances leave the network without unwanted state information remaining behind.

UPnP architecture provides pervasive peer-to-peer network connectivity for networked home appliances, enabling distributed and open communication. In this non-PC centric architecture, each home appliance is simultaneously operated as a home server and client system. Thus, networked home appliances such as a refrigerator or TV are candidates for a home server system.

UPnP middleware uses common protocol stacks such as hypertext transfer protocol (HTTP) 1.0, simple

Manuscript received July 22, 2002

object access protocol (SOAP) and extensible markup language (XML). The use of these Internet technologies in UPnP enables seamless networking, control, and data transfer.

Operating systems such as Linux and Window CE can be used to implement UPnP-enabled devices. Because the UPnP development tool kit [16][17] simultaneously supports embedded Linux and Windows CE operating systems, the developer has a choice of operating systems for the UPnP-enabled embedded system.

UPnP middleware requires less hardware system resources than traditional PC-based solutions. Typically, networked home appliances are based on a low-cost micro controller, application-specific integrated circuits (ASICS), and around 500 and 3000 kilobytes of RAM and flash memory respectively for porting UPnP and application programs.

These advantages notwithstanding, UPnP may still need improvements in regard to recovery methods, setting methods of the time to live (TTL) parameter in the IP header, and additional bridges for non-IP protocols.

To support reliable operation of the overall home network, the home server system must be able to track all the changes and breakdowns of each networked home appliance. Therefore, home server systems must have a dependable recovery method.

However, UPnP v1.0 supports only simple recovery methods such as heartbeat checking and timeout. These recovery methods should be improved in the next version in order to improve the reliability of home network systems.

In addition, the fixed setting of the TTL field in the IP header can spread UPnP messages to areas of the local network where they are not wanted. To prevent this unwanted spreading of UPnP messages, systematic methods for adjusting the TTL field are required.

Finally, all networked home appliances do not require an IP stacks by its costs and limited resources. A networked home appliance that communicates with non-IP protocols needs additional bridges for communication.

3 Architecture of the Home Server System

Home network systems consist of server–client programs, a scheduler for home users, appliance emulators, and embedded communication devices using Linux and Window CE platforms. The home scheduler in the home server system includes appliance on/off scheduling modules and energy management modules.

In the home network system developed in the present work, two types of home server program are implemented. One is implemented using Microsoft Foundation Class (MFC) programming, and the other is implemented in an Internet browser style. The operation procedure of these programs is the same, and the former is explained in greater detail.

The overall architecture of the home network system is shown in Figure 1. The developed home network architecture has the ability to fully integrate and manage all appliances within a home using UPnP middleware-enabled high-speed power line communication (PLC) devices. A set of application program interfaces is specified, allowing the developer to construct the application programs for the home server system using the UPnP API development toolkit [16][17].

The home server system provides network services, maintains the network database, and timetables the home server scheduler. The scheduler enables the user to manage all of the home appliances using its services.

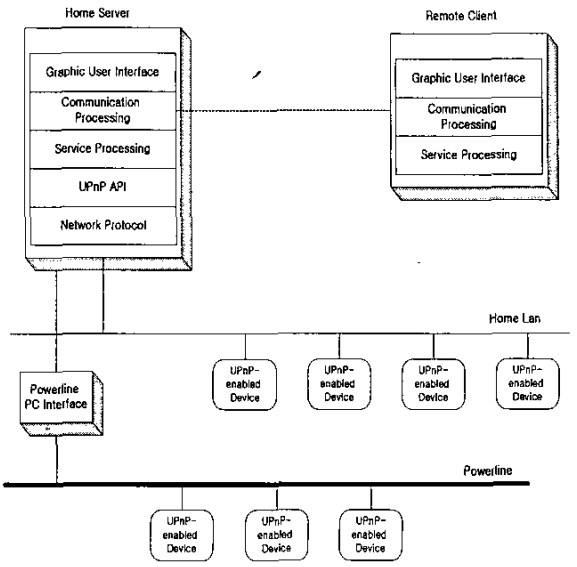

Figure 1. *Structure of the Home Network System*

Table 2. *Types of Networked Appliance Emulators*

Appliance Type	O.S	Display Type	Data rate
Refrigerator	Linux	Graphical	High
Microwave oven	WinCE	Text	Middle
Toaster	WinCE	Text	Low
Air Conditioner	WinCE	Graphical	Low
Washing Machine	Linux	Text	Middle
TV	Linux	Graphical	Very High

Table 2 presents the specifications of the appliance emulators used to implement the home network system. The home server checks all the UPnP-compatible appliances in the home network when the user invokes the search process using a command button.

When a UPnP-compatible appliance is found, a display icon of the appliance and its name appear in the tree view on the left side of the home server and home browser programs. The user receives information about the services and actions of the home appliance by double-clicking on the tree view. When an action service is invoked, the information of the arguments of the appliances in operation is displayed. The user can drag the appliance icon to the desired location on the screen's right-hand side, which shows a plan of the server program. As new home appliances appear in the window of home server program, the user can obtain information on these appliances by using the mouse to point to the desired information.

Figure 3 shows the activation status of a refrigerator, TV, and lighting system in a sub-network. Double-clicking on an appliance icon in the home server program invokes a presentation page for the appliance. Through the presentation page the user can directly control and monitor appliances. This page is designed to support more detailed information about the appliances, which can be displayed on request through the home server.

In the appliance section of the home server program display, simple information about each monitored appliance (the state variables) is displayed at the top. When the user invokes an action service using the action item in the device control frame, a message is sent to invoke the action. When successful, the updated state variables are displayed on the home server.

4 Implementation and Experimental Results

To implement application programs, UPnP API [15][16] is used to develop the home network system. UPnP API is designed to enable the simultaneous control of multiple networked home appliances.

The Windows shell user interface is a UPnP application that displays an icon for each networked home appliance that it discovers on the home network. In effect, it carries out a search for the UPnP-enabled home appliances, notifies the registers of all home appliances as they come and go on the home network, and displays the home appliances in the user interface.

Once the appliances are displayed, home appliance control is achieved by double clicking the mouse button on an appliance icon, which opens a presentation page. The presentation page has embedded script that calls UPnP API.

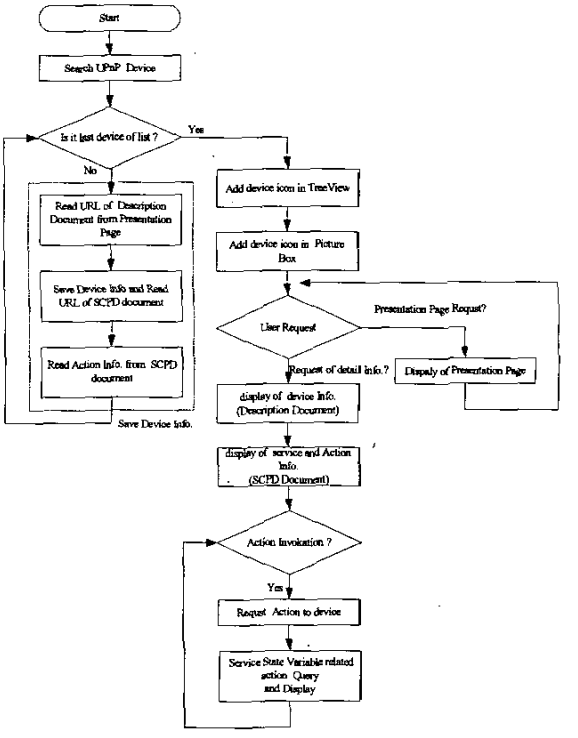

Figure 2. *Block Diagram of the Home Server Program*

4.1 Home Server using UPnP Middleware

In this section, the implementation results of the home server program are presented. Figure 3 shows the main home server program developed for the home network system using the MFC program.

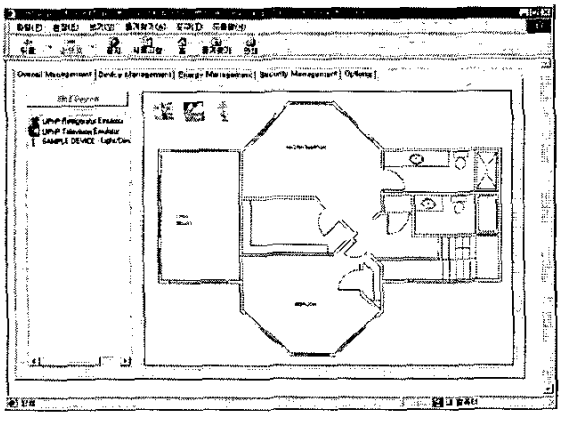

Figure 3. *Screenshot of the Home Server Program*

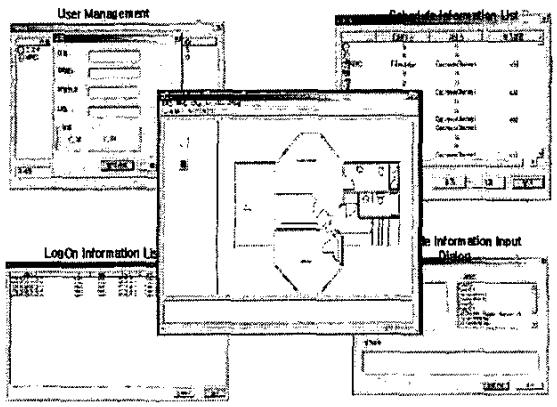

Figure 4. *Additional Modules in the Home Server Program*

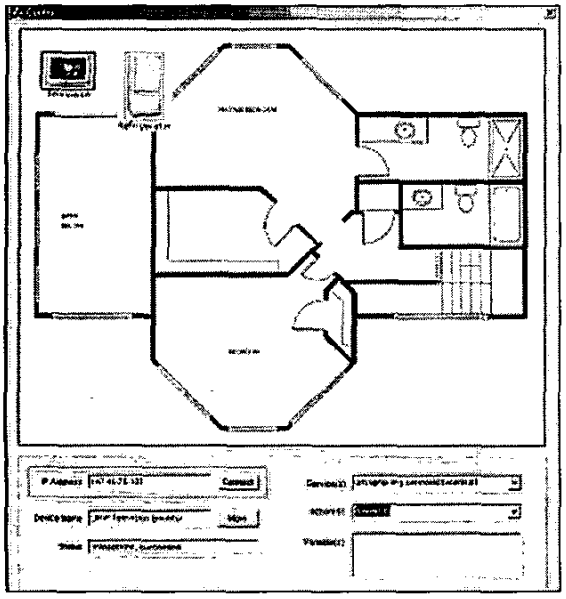

Figure 5. *Discovery Operations in the Home Server Program*

In the home server program, each home appliance can transfer messages to, or control data streams from any other appliance. The control point sends a discovery request to a newly connected home appliance and the appliance responds in Step 1. When an appliance is connected to the network, it must automatically configure itself. The appliance then announces its presence to the other appliances already on the network using a simple discovery protocol based on the Internet HTTP, and is immediately ready to share its services with any appliance that requests them.

The user control point (UCP) [1] requests a description document of a home appliance that sends it in Step 2. The UCP can retrieve a description document by issuing an HTTP GET service on a description for the Uniform Resource Location (URL).

This URL returns data on the location header of a simple service discovery protocol (SSDP) announcement. An HTTP GET service is used to retrieve sub-elements of a description document that are expressed as URLs. In Step 3, the UCP requests additional information such as appliance icons, the names of appliances, and the available service information. When the user selects an appliance icon, the presentation server of the controlled appliance sends the appliance user interface in Step 4. In Step 5, the user can completely control and monitor the appliance using the User Interface (UI).

These operation procedures in home server program are compatible to UPnP specifications [1].

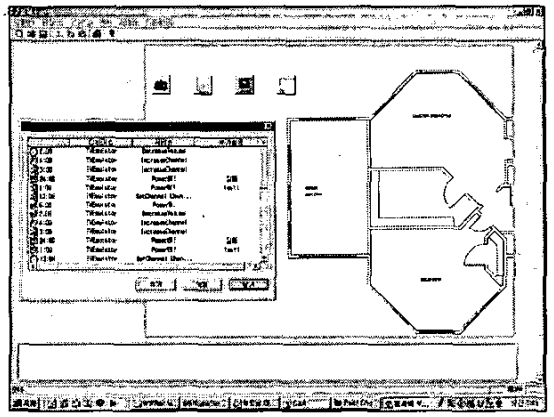

Figure 6. *Send and Request of Networked Home Appliance Description*

There are six types of data class in the developed home server program: ClogInfo, CuserInfo, CscheduleInfo, CUHome, CdevInfo, and CsvcInfo.

The ClogInfo class is used to log information, the CuserInfo class is for user management, the CscheduleInfo class is for the schedule service of the home appliance, the CUHome class is for information on the inner home structure and location information on the home appliances, the CdevInfo class is for information on each home appliance, and the CsvcInfo class is for the services of each home appliance.

In the client section, the information of the networked home appliance is transferred to the home server system when it is demanded. This function uses the following classes for the transferring operation of home appliance data in the client section: UDevice, property, services, and variables. The following pseudo-code presents each class in the server and client section respectively.

Client Classes

```
class UDevice : public CObject
    {
      protected:
        Property* pProperty;
        CObList m_listService;   }

  class Property
    {
      protected:
        CString strUDN;
        CString strFriendlyName;
        CString strDeviceType;
        CString strPresentationURL;
        CString strManufactureName;
        CString strManufactureURL;
        CString strModelName;
        CString strModelNumber;
        CString strModelDescription;
        CString strModelURL;
        CString strUPC;
        CString strSerialNumber;

    }
    class Service : public CObject
    {
      protected:

        CString strServiceType;
        CString strServiceID;

        CObList m_listVariable;
        CStringList m_listAction;
    }
    class Variable : public CObject
    {
      protected:
        CString strName;
        CString strType;
        CString strValue;   }
```

Udevice is the class implemented for UPnP-enabled appliances such as refrigerators, TVs, etc. Because each appliance comprises one property and several services, the protected field of Udevice declares one property-type point. CObList is used for saving the information describing several variables of the types of service.

The Property class is composed of each appliance's property. One Udevice has one property. The protected field of the appliance saves information on a variable of CString type such as unique device name (UDN), friendly name, appliance type, presentation URL, manufacturer's URL, model name, model number, model description, model URL, and serial number.

The service class is composed of the various services of each appliance. Because one appliance has several services, Udevice declares an m_listService using a linked list as a CObList type. Variable classes represent the variables of the appliances. Because one service has several variables, the service declares m_listVariable type of CObList type using a linked list. The protected field is declared as type CString using strName, strType, and strValue. This field is used for saving each variable's name, type, and value. For the home browser program, an XML-based browsing scheme is used in the five-step simulation. The procedure for the browsing scheme in the developed home network system is shown in Figure 7.

Server Classes

```
class CLogInfo : public CObject
    {
        CString m_Logintime;
        CString m_ID;
        CString m_Name;
        CString m_IP;
        CString ctrlDevice;
        CString JobControlInfo;
        CString JobResultMessage;
    }
class CUserInfo : public CObject
    {
        CString m_ID;
        CString m_Age;
        CString m_Name;
        CString m_Sex;
        CString m_Password;
        DECLARE_SERIAL(CUserInfo)
    }
class CScheduleInfo : public CObject
    {
        CString m_Time;
        CString m_Device;
        CString m_Service;
        CString m_Contents;
    }
class CUhome : public CObject
    {
        CString homeName;
        int R_LTX[10];
        int R_LTY[10];
        int R_RBX[10];
        int R_RBY[10];
        CString R_Item[10];
        CString I_flag[10];
    }
CSvcInfo::CSvcInfo()
    {
        iStVarNumber = 0;
        iActionNumber = 0;
        bstrVarNameArray = new BSTR[MAX_STATEVAR];
        varStVarArray = new VARIANT[MAX_STATEVAR];
        pre_varStVarArray = new BSTR[MAX_STATEVAR];
        bstrActionArray = new BSTR[MAX_STATEVAR];
    }
class CDevInfo
    {
        int iSvcNumber;
        VARIANT_BOOL vbHasChildren;
        BSTR bstrUdn;
        BSTR bstrUfn;
        BSTR bstrType;
        BSTR bstrPresenUrl;
        BSTR bstrManuName;
        BSTR bstrManuUrl;
        BSTR bstrModelName;
        BSTR bstrModelNumber;
        BSTR bstrDescription;
        BSTR bstrModelUrl;
        BSTR bstrUpc;
        BSTR bstrSerialNumber;
        BSTR bstrServiceId;

        CSvcInfo* pSvcArray[MAX_SERVICE];
    }
```

For the discovery service, the advertisement processes will time out unless refreshed; the timeout is determined by the device and can be refreshed indefinitely. For appliance control, each action must be completed within 30 seconds; if no response is received within this time, the control point cannot assume the device is still responsive. For the event service, event subscriptions time out unless refreshed; the timeout is requested by the event subscriber, is determined by the event publisher, and can be refreshed indefinitely.

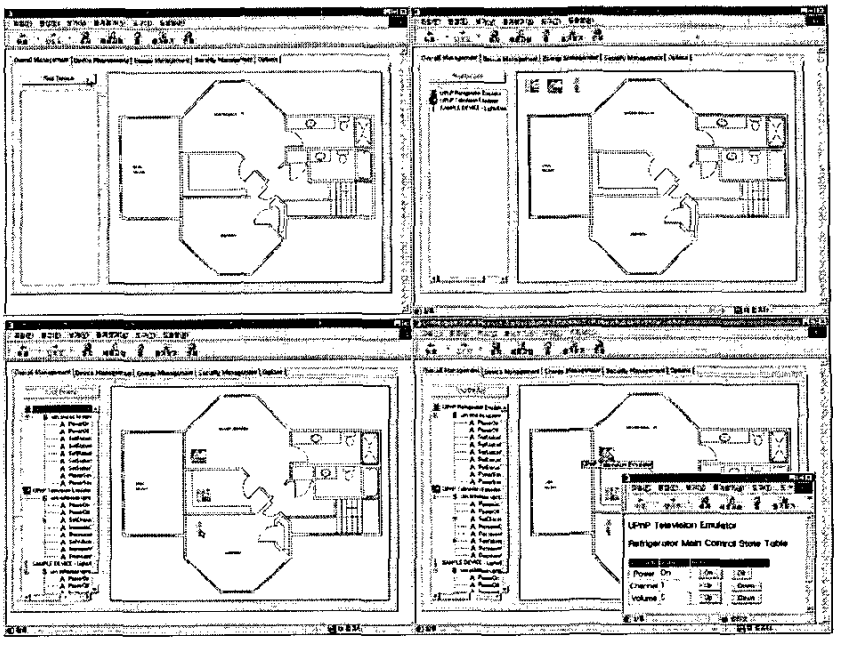

Figure 7. *Screenshot of Home Browser in the Test-bed*

4.2 Implementation of Appliance Emulators

UPnP-enabled appliance emulators are shown in Figure 9. In the case of the refrigerator, the refrigerator emulator displays the open status of the internal compartment. Air conditioner, toaster, and TV emulators are also implemented. These appliance emulators can be operated on a UPnP-enabled embedded device or PC using a Linux or Win CE platform. These home appliance emulators are improved from the implementation results in [4].

Home appliance emulators can exchange many types of messages or control data. The home server sends a discovery request to a newly connected appliance emulator and the appliance responds. When an appliance emulator is plugged into the network, the appliance emulator must automatically configure itself. The appliance emulator announces its presence to the other appliances already on the network using a simple discovery protocol based on HTTP, and is immediately ready to share its services with any home appliance that requests them.

For example, an air conditioner emulator is implemented to be compatible with the architecture of the UPnP Development Kit [16]-[17]. The operational procedure of the air conditioner is as follows. When an air conditioner emulator is plugged into the network, it sends device information to advertise its existence. Then, after the home server receives information about the home appliance using the XML page, the action service presentation page is activated

In the case of the toaster emulator, the operational procedure is simplified. In the case of the air conditioner's emulator, when it is in the state "Power On", all related variables are included in the state table for the next operation; each action is required to search the state table using several processes. This makes the implementation of the air conditioner's emulator difficult. Therefore, in the case of a toaster emulator, we simplified the initialization process by creating the state table when the emulator is invoked.

In the action services, the values of the state variables are changed by the user.

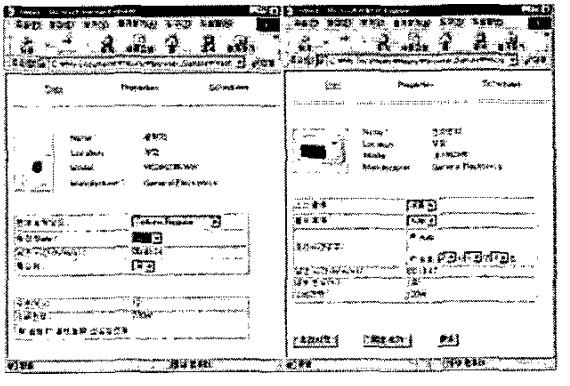

Figure 8. *Appliance Emulator Type 1: Washing Machine and Microwave Oven*

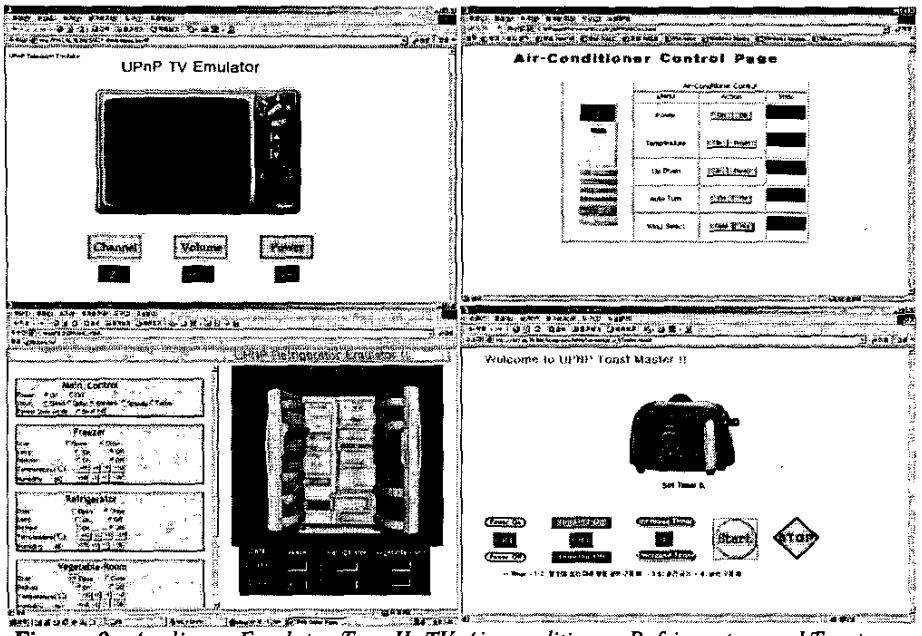

Figure 9. *Appliance Emulator Type II: TV, Air-conditioner, Refrigerator, and Toaster*
(The program sources of emulators can be downloaded at http://i.am/homenetwork/emulators)

Figure 10 shows examples of the XML description of an air conditioner and a toaster. All appliance emulators are compatible with the specification of UPnP ver 1.0 [1].

```
<?xml version="1.0" ?>
- <scpd xmlns="urn:schemas-upnp-org:service-1-0">
  - <serviceStateTable>
    - <stateVariable sendEvents="0">
        <name>Power</name>
        <dataType>Boolean</dataType>
        <defaultValue>0</defaultValue>
      </stateVariable>
    - <stateVariable sendEvents="0">
        <name>Temperature</name>
        <dataType>i4</dataType>
      - <allowedValueRange>
          <minimum>-50</minimum>
          <maximum>50</maximum>
          <step>1</step>
        </allowedValueRange>
        <defaultValue>0</defaultValue>
      </stateVariable>
    - <stateVariable sendEvents="0">
        <name>UDDirection</name>
        <dataType>i4</dataType>
      - <allowedValueRange>
          <minimum>1</minimum>
          <maximum>10</maximum>
          <step>1</step>
        </allowedValueRange>
        <defaultValue>0</defaultValue>
      </stateVariable>
    - <stateVariable sendEvents="0">
        <name>AutoTurn</name>
        <dataType>Boolean</dataType>
        <defaultValue>0</defaultValue>
      </stateVariable>
```

```
<?xml version="1.0" ?>
- <scpd xmlns="urn:schemas-upnp-org:service-1-0">
  - <serviceStateTable>
    - <stateVariable sendEvents="0">
        <name>Power</name>
        <dataType>Boolean</dataType>
        <defaultValue>0</defaultValue>
      </stateVariable>
    - <stateVariable sendEvents="0">
        <name>Thawing</name>
        <dataType>Boolean</dataType>
        <defaultValue>0</defaultValue>
      </stateVariable>
    - <stateVariable sendEvents="0">
        <name>Timer</name>
        <dataType>i4</dataType>
      - <allowedValueRange>
          <minimum>0</minimum>
          <maximum>10</maximum>
          <step>1</step>
        </allowedValueRange>
        <defaultValue>0</defaultValue>
      </stateVariable>
  </serviceStateTable>
```

Figure 10. *XML description of Air-conditioner and Toaster*

Additional modules are combined with the scheduler and the energy management module is implemented for the user interface.

In the home server system, a digital signature and encryption are used for security. However, the security issues associated with home server systems are not considered in the present work.

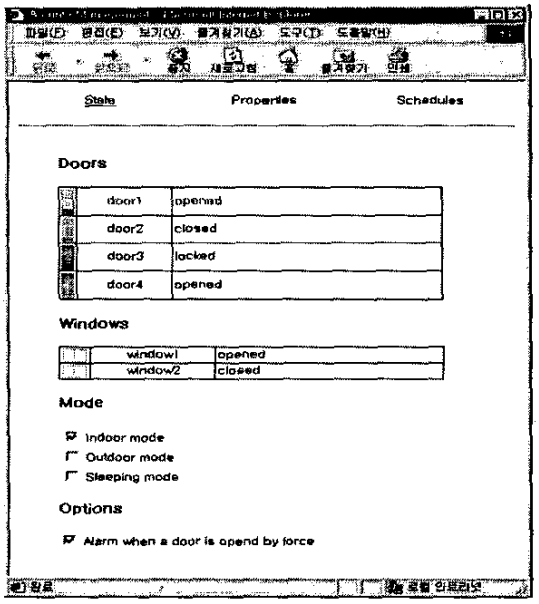

Figure 11. *Energy Management Module in the Home Server*

4.3 Implementation of the Embedded Interface Module for Networked Appliances

Most off-the-shelf home automation modules are add-on PLC modules that sit between the power outlets and the home appliances that are to be remotely controlled [18]. Therefore, in this paper, PLC is considered as the main medium for home network systems.

To implement the embedded interface module between the home appliance and the PLC modem, the embedded interface module was designed and implemented (Figure 12). This module can be combined with a PLC modem that has an embedded operating system with UPnP such as Win CE [16] or embedded Linux [17]. UPnP is ported to the ROM of this embedded interface module for the home networking system.

The design specifications for embedded interface module are as follows.

- CPU: 32-bit CPU operating above 100 MHz
- Main Memory: SDRAM 32MB
- Boot ROM: Flash 16MB
- I/O
  - i. Ethernet Controller
  - ii. Embedded USB Controller
  - iii. Serial Port
  - iv. 28 PIO
- ROM Program
  - i. support JTAG programming (Flash memory)
  - ii. Ethernet protocol interface
- Operating System
  - i. Embedded Linux
  - ii. Win CE 3.0
- Application Program
  - i. UPnP porting
  - ii. Appliance Emulator porting

For the PLC communication, an Ethernet-based high-rate commercial PLC modem and the PLC gateway are used. They support a high data rate (up to 2 Mbps) for home appliances such as TVs, refrigerators, and washing machines. The embedded interface module is designed using a four-layer schematic to minimize hardware size.

It can be applied as add-on modules, such as a refrigerator that is used as a kitchen server in the next step. For this Web-connected application, a 32-bit microprocessor, main memory, boot ROM, I/O for Ethernet, and serial USB are used. Because the PLC modem is installed, the test-bed supports the Ethernet protocol and no additional bridge is required to communicate to a serial interface. In most cases, the PLC modem is connected using a serial interface. Using this PLC modem, the home server system connects and controls the appliance emulators. To connect to a public network, the home server is connected to a LAN or ADSL modem.

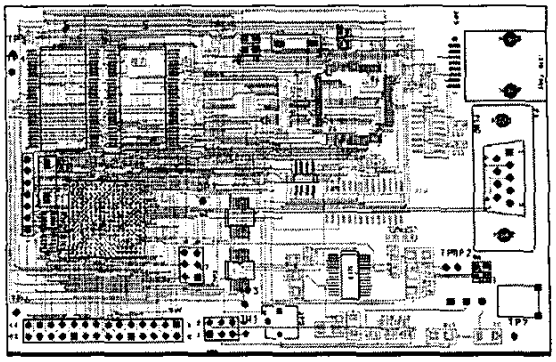

Figure 12. *Prototype of the embedded Interface Circuit Module*

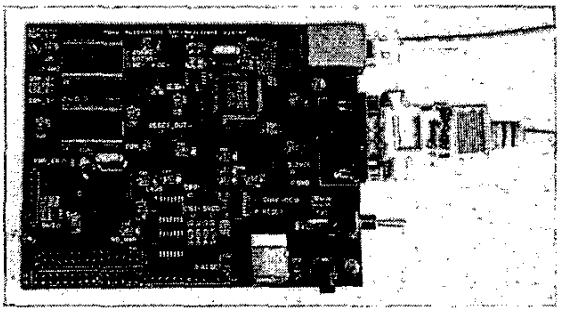

Figure 13. *Printed Experimental Board: Embedded Interface Module*

The prototype of the developed module is shown in Figures 12 and 13. To combine home appliances, the implemented modules will be modified. The design specifications can be adjusted to the requirements of each home appliance.

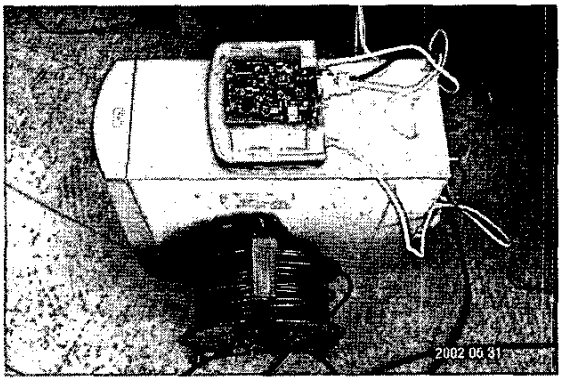

Figure 14. *Connection to Interface Module and PLC modem in the Test-bed*

The developed embedded interface device is applied to the control and monitoring of networked home appliances using UPnP-enabled applications.

The embedded interface module is connected to the PLC modem by a twist pair line, and the data are sent through the power line (Figure 14).

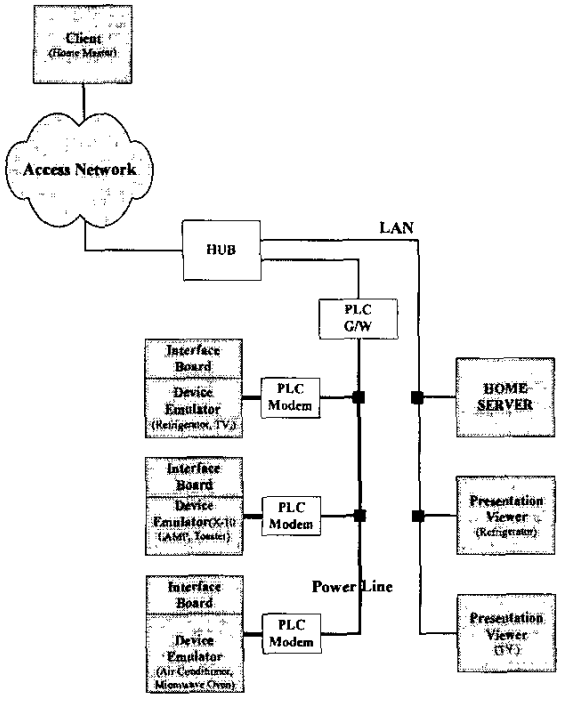

Figure 15. *Testing Environment in the Laboratory*

The testing environment of the home network system is shown in Figure 15. It comprises the developed home server program and hardware devices for power line communication.

Finally, the demonstration video file of the developed home network system can be downloaded at http://asri.snu.ac.kr/~dskim/seoul.asf.

5 Conclusions

In this paper, a home network system for the control and monitoring of networked home appliances is designed and implemented.

The developed home network system includes home server programs, home appliance emulators, and an embedded interface device that uses UPnP middleware.

The primary goal of this work was to design and implement a UPnP-based home network system with extended functions for networked home applications.

The results of this paper are useful and provide guidelines for the design of home network systems using UPnP middleware and embedded interface modules for networked appliances.

Future studies should develop the security and energy management modules that are necessary components of a reliable home network system.

Acknowledgements

The authors acknowledge the financial supports from the Ministry of Commerce, Industry and Energy and KERI, Korea. They also thank the editors and UPnP forum members for their time and efforts.

References

[1] Universal Plug and Play Device Architecture Reference Specification Version 1.0, Microsoft Corporation, June 2000, Available to: http://www.upnp.org.

[2] Sun Microsystems, "Jini 1.1", 2000, Available to: http://developer.java.sun.com/developer/products/jini/.

[3] H. Ishikawa, E. Tokunaga, and T. Nakajima "A case study of implementing home appliance middleware on Linux and Java", *Symposium on Applications and the Internet Workshops*, 2002, pp.31 -34.

[4] D.S. Kim, W.H. Kwon, Y.H. Kim, H.J. Shin, and Y.I. Kwan, "Home Network Message Specification for White Goods and Its Applications", *IEEE Transactions on Consumer Electronics*, Vol.48, No.1, 2002., pp.1-10.

[5] P.M. Corcoran, "Mapping home-network appliances to TCP/IP sockets using a three-tiered home gateway architecture", *IEEE Transactions on Consumer Electronics*, , Vol. 44, No. 3, Aug. 1998, pp. 729 -736

[6] P.M. Cocoran, F. Papai and A. Zoldi, "User Interface Technologies for Home Appliances and Networks," *IEEE Transactions on Consumer Electronics*, Vol.44, No.3, Aug. 1998, pp.679-685.

[7] S. Koutroubinas, T. Antonakopoulos, and V. Makios, "A New Efficient Access Protocol for Integrating Multimedia Services in the Home Environment," *IEEE Transactions on Consumer Electronics*, Vol. 45, No. 3, Aug. 1999, pp.481 - 487.

[8] A. Chandra, V. Gummalla, and J.O. Limb, "An Access Protocol for a Wireless Home Network," *International Conference on Wireless Communications and Networking*, Vol. 3, 1999, pp.1392-1396.

[9] D.L. Waring, K.J. Kerpez and S.G. Ungar, "A Newly Emerging Customer Premises Paradigm for Delivery of Network-Based Services," *International Journal of Computer Networks*,

Vol.31, No.4, 1999, pp.411-424.
[10] A. Dutta-Roy, "Networks for homes", *IEEE Spectrum*, Vol. 36, Dec. 1999, pp.26-33.
[11] S. Teger, and D.J. Waks, "End-user perspectives on home networking" *IEEE Communications Magazine*, Vol. 40, No. 4 , April 2002, pp. 114-119.
[12] I. Abdul-Fatah and S. Majumdar, "Performance of CORBA-based client-server architectures", *IEEE Transactions on Parallel and Distributed Systems*, Vol.13, No.2, Feb. 2002, pp.111 –127.
[13] T. Kurioka, H. Minami, H. Okuda, J. Numazawa, and A. Yanagimachi "Television home server for integrated services-toward the realization of ISDB anytime services", *IEEE Transactions on Consumer Electronics*, Vol.44, No.4, 1998, pp. 1195 –1200.
[14] B.A.Miller, T. Nixon, C. Tai, and M.D. Wood, "Home networking with Universal Plug and Play ", *IEEE Communications Magazine*, Vol.39 , No. 12, Dec. 2001, pp.104-109.
[15] S. Moyer, D. Maples, S. Tsang, S and A. Ghosh, "Service portability of networked appliances " *IEEE Communications Magazine*, Vol. 40, Issue 1, 2002, pp.116 -121.
[16] Microsoft Device Kit for UPnP device, Microsoft Corporation, 2002, Available to: http://www.microsoft.com/hwdev/upnp
[17] Intel UPnP SDKs for Linux, Intel Corporation, 2002, Available to: http://www.intel.com/labs/connectivity/upnp
[18] J. Newbury and W. Miller, "Potential Communication Services Using Power Line Carriers and Broadband Integrated Services Digital Network," *IEEE Transactions on Power Delivery*, Vol.14, No.4, 1999, pp.1197-1201.
[19] S. Ramanathan and R. Gusella, "Home Network Controller for Providing Broadband Access to Residential Subscribers," *IEEE Transactions on Consumer Electronics*, Vol.41, No.3, Aug. 1995, pp. 859 -868
[20] Geoff Coulson, Gordon S. Blair, Michael Clarke, and Nikos Parlavantzas, "The design of a configurable and reconfigurable middleware platform", *Distributed Computing*, Vol.15 , No. 2, 2002, pp.109-126.

BIOGRAPHIES

Dong-Sung Kim: Dong-Sung Kim was born in Korea, in 1969. He received B.S. and M.S. degrees in the Department of Electronic engineering from Han Yang University, Korea, in 1992 and 1994, respectively. From 1994 to 1998, he worked as a full-time researcher in engineering research center for advanced control instrumentation at Seoul National University. Since 1998, he has been pursued a Ph.D. degree at EECS, Seoul National University, Korea. From 2000.9 to 2001.12, he was a part-time lecturer at Dong-guk University, Seoul, Korea. His main research interests are currently the design of a protocol for network systems, scheduling of real-time communication, and networked control systems. He has been an IEEE/ACM member since 1998.

Jae-Min Lee: Jae-Min Lee was born in Korea on August 6, 1971. He received the B.S. and M.S. degrees in Electronics from Kyungpook National University, Daegu, Korea, in 1997 and 1999, respectively. Since 1999, he pursues the Ph.D. degree from Seoul National University, Seoul, Korea. His main research interests are currently home network system and industrial networks.

Wook Hyun Kwon : Wook Hyun Kwon was born in Korea on January 19, 1943. He received the B.S. and M.S. degrees in electrical engineering from Seoul National University, Seoul, Korea, in 1966 and 1972, respectively. He received the Ph.D. degree from Brown University, Providence, RI, in 1975. From 1976 to 1977, he was an adjunct assistant professor at University of Iowa, Iowa City. Since 1977, he has been with the School of Electrical Engineering, Seoul National University. From 1981 to 1982, he was a visiting assistant professor at Stanford University, Stanford, CA. Since 1991, he has been the Director of the Engineering Research Center for Advanced Control and Instrumentation. His main research interests are currently multivariable robust and predictive controls, statistical signal processing, discrete event systems, and industrial networks.

In Kwan Yuh: Yuh In Kwan was born in Korea on April 16, 1967. He received the B.S. and M.S. degrees in Electronic engineering from Korea University, Seoul, Korea, in 1988 and 1999, respectively. Since 1989, he is research engineer in LG Electronics Digital Media Research Lab. He is currently senior research engineer in LG Electronics Digital Media Research Lab. His main research interests are currently a home gateway system.