An Integrated Logical Structure-Based UPnP Architecture

Liang-Teh Lee
Department of Computer Science and Engineering, Tatung University
ltlee@ttu.edu.tw,

Wen-Ai Shang
Department of Computer Science and Engineering, Tatung University
tina.shang@gmail.com

Abstract

The UPnP architecture defines peer-to-peer network connectivity of intelligent appliances, devices, and control points[1,2]. It is designed to bring easy-to-use, transparent, standards-based connectivity to networks, whether these networks are in the home or small business. However, the weakness of the device communication is the problem of the UPnP architecture. For example, in the normal UPnP architecture, to play a video file must find out and navigate each Media Server. It is inconvenient to use and is quite a big work to spend much time to search it. In this paper, we propose an Integrated Logical Structure-Based (ILSB) UPnP architecture to improve the system performance and the proposed architecture will be compatible with standard UPnP devices.

Based on the UPnP architecture, the proposed architecture is constructed as a logical structure to integrate each device's service and provide the same capability with each other devices. Based on the architecture, the location of the device can be transparent and provides a more powerful data transmission capability. Once a user plugs a new UPnP device into the network, the capability of the new device will be also plugged into each UPnP devices. Comparing with the normal UPnP architecture, the ILSB-UPnP architecture can provide a higher availability, better performance, and easier-to-use, especially in an environment that devices are highly distributed.

Keyword: UPnP, ILSB-UPnP, discovery, description, control, eventing

1. Introduction

There are a few frameworks introduced by the eHome environment. They include UPnP, Salutation [3], Service Location Protocol (SLP)[4-8], Jini[9-12], and OSGI[13,14] which are summarized as shown in the following table.

	UPnP	Salutation	SLP	Jini
Entities	Control Point, Devices (Services)	Salutation Manager, Transport Manager, Client Server	Directory Agent, Service Agent, User Agent	Lookup services, client service
Service Repository	None	A set of SLMs	DA (directory agent)	Lookup service
Service Announcement	Multicast advertisement	Registering with local SLM	Service registration	Discovery/Join protocol
Access to Service	Invoking action to service	Service Session Management	Service type for discovered service	Service proxy object based on RMI
Service Description	Description in XML	Functional Unit and attributes within it	Service type and attribute matching	Interface type and attribute matching
Service Group	No	No	Scope	Group
Centralized	No	Yes	Scope	No
Event Notification	Service publishes event when state variable changes	Availability checking	SLP extension for event notification	Remote events
Other Features	Automatic configuration	Transport Independent	Authentication security feature	Java-centric architecture

In the eHome environment, we must consider the following issues for integration:

- Plug and Play Devices: In eHome environment, some devices are always power-on, and some are power-off. Maybe the others are by user's request. Hence the eHome device system couldn't be centralized. It must be more transparency and reliable.

- Standard Protocol and easy to build: Since most CE devices do not have much

memory and the performance is not good enough, the implementation must be simple, easy and more efficiency.

Based on above issues, the UPnP is a good choice for eHome environment. However its framework doesn't support any group service, a user must use UPnP control point to access a service by again and again. Hence we will base on the normal UPnP framework to improve it and make it be a more available and efficiency device.

1.1 UPnP Architecture

There are a few basic concepts introduced by the UPnP architecture. [15,16,17,18,19,20] This session introduces these concepts and the underlying UPnP object model, describing each of the different UPnP entities and their corresponding roles and responsibilities. Devices, services, and control points are the basic abstractions of the UPnP device architecture. A UPnP device can be any entity on the network that implements the protocols required by the UPnP architecture. Because UPnP standardizes the protocols through which a device communicates rather than the APIs that a programmer uses, any entity that behaves as a UPnP device by speaking the required protocols is a UPnP device. Thus, a device either can be a dedicated physical device, such as an Internet gateway, or a logical device, such as a PC, that has implemented the functionality required of an Internet gateway.

A UPnP device contains zero or more services; a service is a unit of functionality implemented by a device. Each service has a set of methods, or actions, each with a set of optional input and output parameters and an optional return value, much like a function in the C programming language. The services that a device must implement are determined by the device's type. The working committees of the UPnP standardize the set of services that particular device types must support. For example, an audio rendering device, such as a CD player, might have a service that provides the ability to play, stop, and pause audio content. A control point is an entity on the network that works with the functionality provided by a device. In the terminology of client/server computing, the control point is the client and the device is the server. Control points can invoke actions on services, providing any required input parameters and receiving any output parameters and possibly a return value. Control points can also request the device notification when the device states changes. Figure 1 shows a control point invoking an action on a UPnP device. The device has implemented a single UPnP device type that contains two services. Any entity that invokes the services of a UPnP device is a control point. In fact, UPnP technology-enabled devices may have control point functionality built in, so that they can invoke the services or monitor state changes in other devices. With this capability, devices can form a peer-to-peer network where devices take advantage of each other's service. Figure 2 shows a UPnP technology-enabled device that contains a control point that can invoke the services of other UPnP devices. A device such as an electronic picture frame could implement this pattern by having control point functionality build in, so that when a user presses buttons on the side of the picture frame to navigate through the pictures to be displayed, the control point application on the picture frame retrieves digital images from another UPnP device on the network. However, the control point application must know the location where the picture is. For example, if a user wants to find out a specific photo album, he/she must find each UPnP device, and navigate these. It is quite a big work and needs to spend much time.

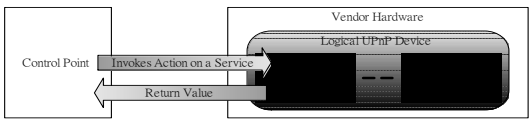


Figure 1 Control Point Invoking an Action

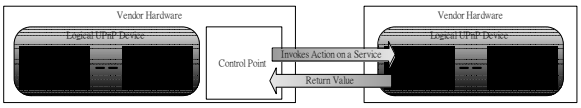


Figure 2 A Control Point in a UPnP Device

1.2 Motivations

In order to reduce the overhead of achieving UPnP device communication while supporting device discovering, service action, and notify event, we propose an integrated logical structure-based UPnP architecture in this paper.

First, the discovery process enables control points to find devices and service and retrieve information about them. Also, once a device has acquired an IP address, the device may advertise itself and its services on the network. Based on the Simple Service Discovering Protocol, we propose an improved device discovering service. The proposed architecture maintains an integrated logical structure-based list for each device and service on the network. Once a discover request occurs, the improved discovering protocol will take the place of all the devices to response the message.

Second, control is the phase of UPnP where control points invoke the actions provided by a device's service. A device's service receives a control message and acts upon it. The device may change state as a result of the operation, leading to the next UPnP phase, eventing. Thus we based on the above integrated device list to maintain each UPnP device description and its service interface invoke. Hence the Control Point might use any interface in one device as use it in any devices.

Third, eventing allows control points to monitor state changes in devices. The UPnP architecture uses a publisher/subscriber model where control points may subscribe to a service provided by a device. The device's service notifies all registered control points upon changes in state variables. Responding to state changes in this way enables a UPnP network of devices to be a dynamic, responsive, event-driven system. In this paper, we propose a mechanism to monitor the entire device in the network and a physical/logical event converting table for the publisher/subscriber mode. Based on the normal UPnP architecture, we construct an ILSB-UPnP Architecture to provide a more convenient environment for the end user to create, use, manage and share digital contents.

2. Integrated Logical Structure-Based UPnP Architecture

The growth of relevant UPnP technology has enabling normal equipment manufacturer (OEM) companies to equip today’s eHome environment with embedded platforms. As a result, general-purpose UPnP applications such as Network Light, Media Server, Printer, Network Gateway, and Digital Video Recorder (DVR) are highly promising. However, the deployment of such UPnP applications on a rapid and dependable scale over on-off computing and ad-hoc networks is proving to be extremely difficult. If a user wants to find out a specific media, he/she must find out the deployment of each UPnP devices, and navigate these. It is quite a big work and needs to spend much time and not convenient.

The object of this section is to investigate these efficiency issues in the design of scalable framework for UPnP architecture. The effort will include the implementation of an Integrated Logical Structure-Based for on-off computing and eHome environment to be as an ILSB-UPnP system. And specific research themes to be pursued are: (1) support service discovery and its description, (2) support service control as a proxy agent (3) support eventing notification.

We have developed an ILSB-UPnP system that integrates UPnP normal technology to enable a whole array of operations, such as support for increasing availability of services discovery, improving service control transparency and its grouping concept, and providing eventing notification awareness in eHome environment.

The basic concept of the Integrated

Logical Structure-Based system involves devices communicating with services through the use of proxy agents from UPnP device to another one. This concept is similar to a traditional 3-tier model from control point to device and from device to another one. However in the normal UPnP environment, a control point may send a request to a UPnP device for searching specific media. If result is fails, it would send the request again to another and step by step. These are presented in the Figure 3.

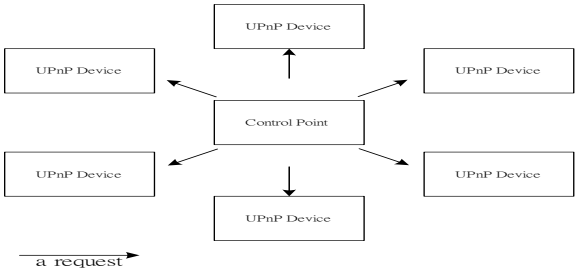


Figure 3 UPnP Control Point Communications

The proposed Logical Structure-Based system consists of three main mechanisms. The first mechanism comprises the ILSB-UPnP Discovery agents, i.e., entities that are expected to construct UPnP Logical Devices which mapping to another UPnP devices in the network and is made up each services to provide dynamic service. The second mechanism is the ILSB-UPnP Control agents, i.e., the agent that provides the platform form control point to execute, like to invoke a proxy server. The final mechanism is made up the UPnP eventing notification to control point. The mechanisms of Logical Structure-Based are illustrated in the Figure 4. The figure describes a typical sequence of interactions among different mechanisms within our provided.

Finally, the control point will see all the services of each UPnP devices are the same and all the capabilities of each UPnP devices are equal. In the following section, the proposed Logical Structured-Based system will be described in more detail.

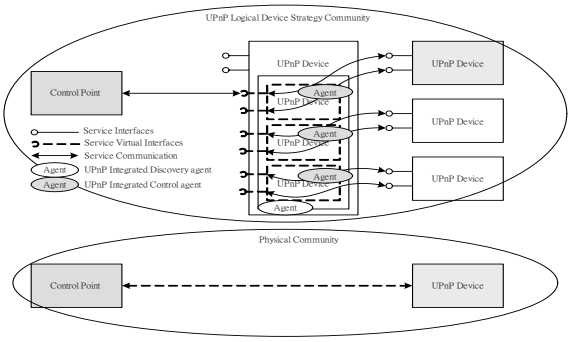


Figure 4 Logical Devices Strategy

According to the above deployment, all of the UPnP devices have the same services, which support discovery, control, and eventing. These are presented in the Figure 5.

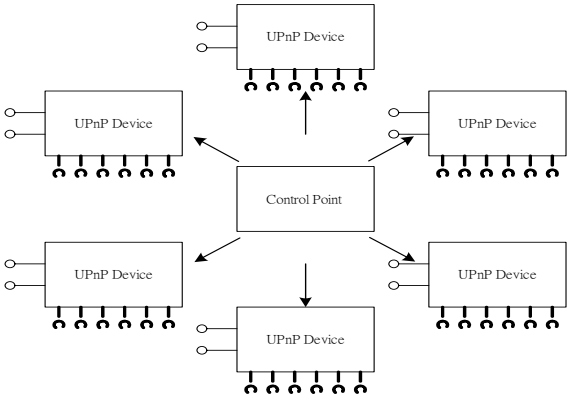


Figure 5 UPnP Control Point with Integrated Communication

2.1 Discovery for ILSB-UPnP

About the Discovery we inherit the UPnP Discovery protocol to procure the ILSB-UPnP Discovery protocol. When a device is added to the network, the ILSB-UPnP Discovery protocol allows that device to advertise all the ILSB-UPnP services on the network to control points. Similarly, when a control point is added to the network, the ILSB-UPnP Discovery protocol allows that control point to search for integrated devices of interest on the network. Hence any integrated device like a discovery gateway it advertise its and all the relative integrated device and allows control point to search any integrated device like normal device. Figure 6 is its sketch map.

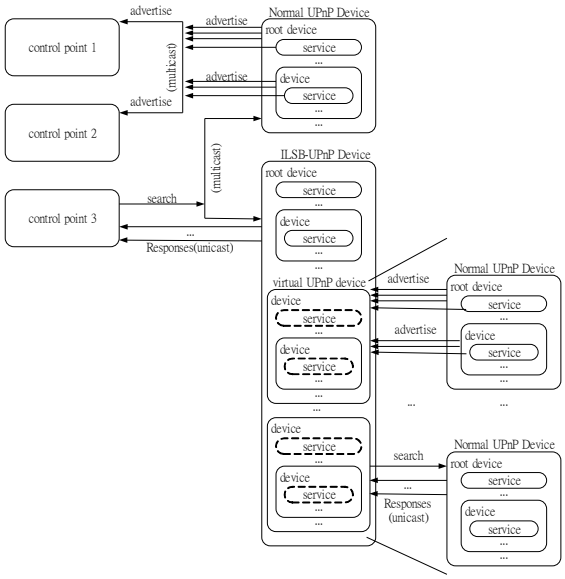


Figure 6 ILSB-UPnP Discovery Protocol

For constructing a Logical Devices into an embedded device, we induce two types of devices in the ILSB-UPnP Device. One is Normal Device and the other is Virtual Device in the network. Normal Device is a physical UPnP device. It is embedded into the normal UPnP Device for working special function, like UPnP Media Server, UPnP Media Renderer, and so on. Virtual Device is a virtual UPnP device, which does not exist in the ILSB-UPnP Device. It is like a proxy agent for all normal devices in the intranet. We use it to archive all the normal devices and let the ILSB-UPnP Device be an integrator. The following is the steps to construct it for some UPnP situation.

2.1.1 Steps to build ILSB-UPnP device

To construct a Logical Devices and embedded into an ILSB-UPnP device when it is available. All ILSB-UPnP devices must find out, which devices are normal and which are ILSBs. Hence a new add-on ILSB-UPnP device must collect the normal UPnP devices and then collect another type, ILSB-UPnP devices. Finally, to construct the logical device or virtual device list and then embed these list into device. The processing steps specified by the ILSB-UPnP device architecture are presented in Figure 7.

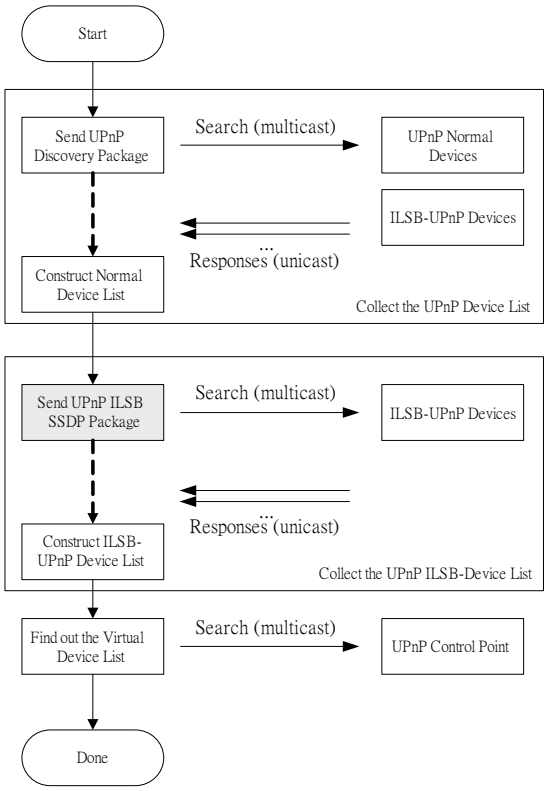


Figure 7 The Processing Steps of the UPnP Virtual Device

- Try to Collect UPnP Device List via UPnP SSDP
  First, an ILSB-UPnP device must try to collect the UPnP Device List via SSDP. If the device successfully acquires a device list, it is ready to continue with next phases. It must thereafter only be concerned with updating the virtual device list when any SSDP advertise event occurs.
- Try to Collect ILSB-UPnP Device List via ILSB-UPnP SSDP
  Next, an ILSB-UPnP device must try to collect the ILSB-UPnP Device List via ILSB-SSDP. If the device successfully acquires a device list, it is ready to continue with finding out virtual device list. It must thereafter only be concerned with update the virtual device list when any integrated advertises event occurs.
- Find out the Virtual Device List
  The first step, the collected device list includes UPnP Normal Devices and ILSB-UPnP Device. Hence the second step, we collect the integrated device

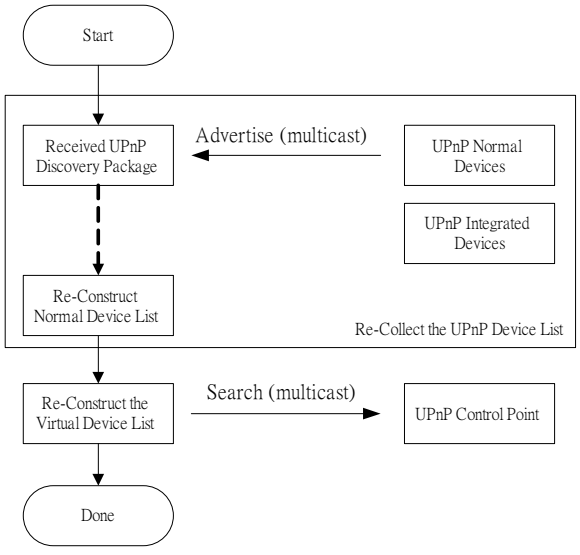
list. If we want to find out the Normal Devices for building the Virtual Device List, we just only subtract Normal Device from ILSB-UPnP Device.

- Send the Device available – NOTIFY with ssdp:alive
  The last step, send the device available notification via UPnP SSDP and the ILSB-UPnP SSDP.

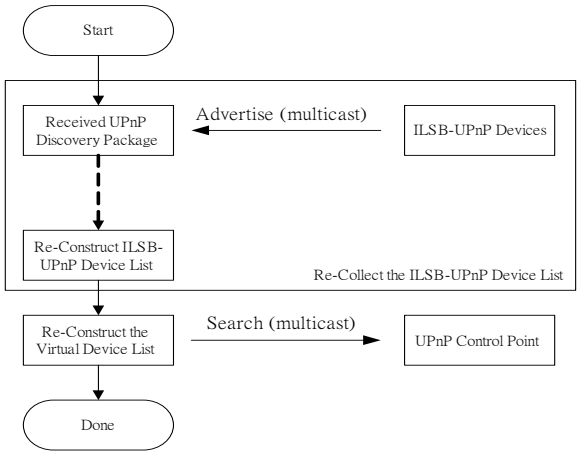
2.1.2 Modify the ILSB-UPnP device dynamically

Once a device is available, all ILSB-UPnP devices must have the capability to change Logical Devices dynamically when receiving any SSDP or ILSB-UPnP SSDP notification. We propose the following steps to re-construct virtual UPnP device. Figure 8(a) shows the processing flow.

- Try to Re-Construct the normal and Virtual UPnP Virtual Device List when received UPnP SSDP notification:
  First, when the device received a UPnP SSDP notification ILSB-UPnP device must try to re-construct the UPnP Virtual Device. Figure 8 shows the processing flow of this step.



(a)



(b)

Figure 8 UPnP Virtual Device Flowchart

- Try to Re-Construct Virtual UPnP Device List when received ILSB-UPnP SSDP notification:
  Next, an ILSB-UPnP device must try to re-construct the ILSB-UPnP Device List once the ILSB-UPnP SSDP notification is received.
- Send the UPnP SSDP and ILSB-UPnP SSDP notification again:
  The last step, send the UPnP SSDP and ILSB-UPnP SSDP notification again to broadcast to every device of the changed message.

The processing flow of the last two steps is shown in Figure 8(a), and Figure 8(b) shows the guideline for the UPnP Virtual Device and how to build the ILSB-UPnP Device II into the ILSB-UPnP Device I. The whole concept is shown in Figure 9.

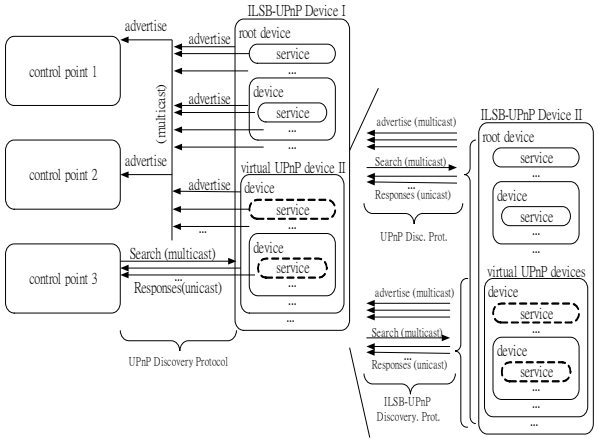


Figure 9 ILSB-UPnP Discovery Protocol

2.2 Description for ILSB-UPnP

After an ILSB-UPnP control point discovers a device, it has only the information contained in the discovery message, the device's type, its universally-unique identifier, and a URL to its description document. In this situation the ILSB-UPnP Device is already fabricate Virtual UPnP devices. Hence UPnP control point could find out more about the device, including the services and actions it supports, the control point retrieves description documents from the device even these items is not in ILSB-UPnP Device.

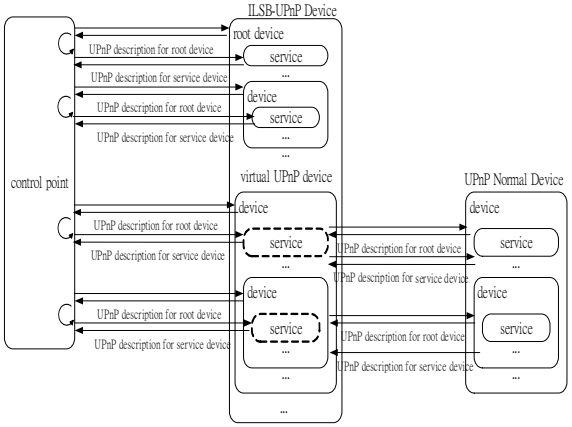


Figure 10 ILSB-UPnP Device Description Flow

The UPnP description for a device is partitioned into two parts, which is shown in Figure 10. One is a device description describing the physical and normal containers, and one or more service descriptions describing the capabilities exposed by the local device. The other is a device description describing the virtual and fake logical containers and one or more service descriptions describing exposed by another UPnP device. Likewise these descriptions are like standard UPnP description includes vendor- specific, manufacturer information like the model name and number, serial number, manufacturer name, URLs to vendor-specific Web sites, etc.

When a UPnP Control Point sends a request to get device description, the UPnP will response a device description including normal and virtual device description (The virtual device list is built by UPnP Integrated Discovery). Hence now we must consider how to build the virtual device description. The steps, as specified by the ILSB-UPnP device description, are presented in the flowchart shown in Figure 11.

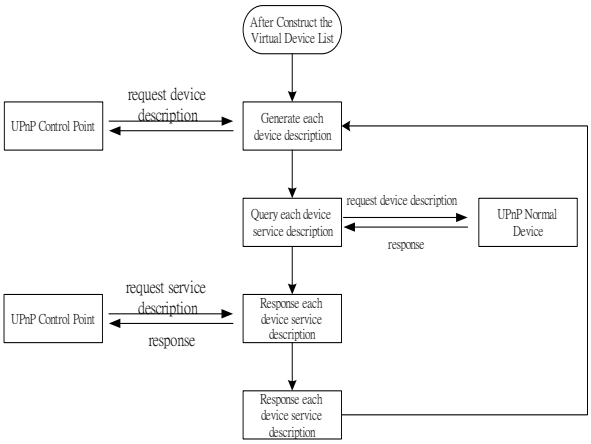


Figure 11 Device Descriptions and Service Control Flow

First, an ILSB-UPnP device must generate a Virtual Device Description via Virtual Device List, and response it for the UPnP Control Point request. Next, the ILSB-UPnP device must try to cache each device's service and build a service list for request. When a service request is not the cache, the ILSB-UPnP device will send the service request to the background Normal UPnP device and cache it after received its response message. It is like a service description agent. However when the Virtual Device List is changed by received a UPnP SSDP or an ILSB-UPnP SSDP message. The ILSB-UPnP Device Description must be re-organized and the cache agent must be reset.

2.3 Control for ILSB-UPnP

After a UPnP Control Point acquired the ILSB-UPnP Device Description and its Service Description, Control Point will then invoke any of the actions provided by the device's services. In this section, we will introduce how to invoke these ILSB-UPnP Device controls.

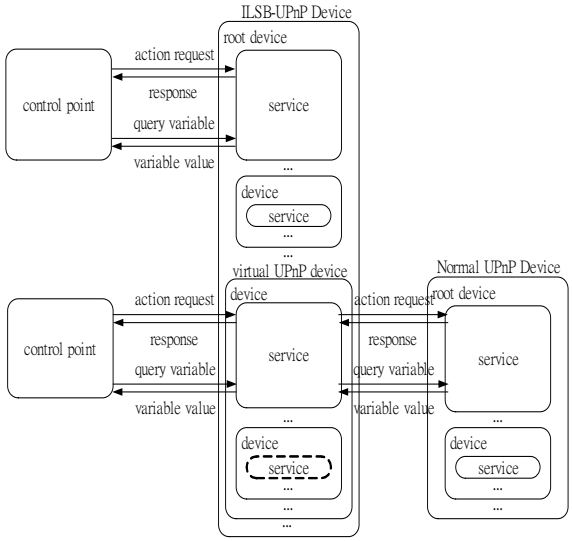


Figure 12 ILSB-UPnP Device Description Flow

The ILSB-UPnP Control Protocol shown in Figure 12 is partitioned into two parts. One is a device controller controlling the physical or normal embedded device, and one or more service controller controlling the capabilities exposed by the local device. The other is a device controller controlling the virtual or fake logical containers and one or more service controller controlling exposed by another UPnP device. Likewise these controllers are like UPnP Control including action invoke, response, query invoke, and query response.

When a UPnP Control Point sends a request to invoke an action, the ILSB-UPnP device will distinguish which type of action. Is it a physical invoke action or logical invoke action. If it is a logical invokes action, the ILSB-UPnP device will be like a proxy agent and send the same invoke message to another UPnP device. After the response message from another UPnP device, the ILSB-UPnP device will send these response messages to its sender. These are presented in the Figure 12. Though the control action, and query variables are corresponded UPnP control action and query variables norm. Since ILSB-UPnP Device is similar to a proxy agent, we will provides the proxy procedure policy in the following. Three steps explain this.

- When receiving the control invoke message
  When receiving the control invoke message, the agent will reproduce this message and modify some related fields for using it to invoke another UPnP device.
- Using reproduce message to invoke another UPnP device and waiting for its response
  Using reproduce message to invoke another UPnP device, we can find out the UPnP device address via UPnP Virtual Device List (It is a mapping table between Virtual Device Service and Physical Device Service).
- When receiving response message from another UPnP device
  When receiving response message, the agent will reproduce this message and modify some related fields and response it for its requestor.

2.4 Eventing for ILSB-UPnP

This section explores ILSB-UPnP eventing, starting first with a brief overview of event notification in a distributed system and the publisher/subscriber model. After a UPnP Control Point acquired the ILSB-UPnP Control Protocol, Control Point will then sends a subscription message to subscribe eventing. In this section, we will introduce how to subscribe the ILSB-UPnP eventing.

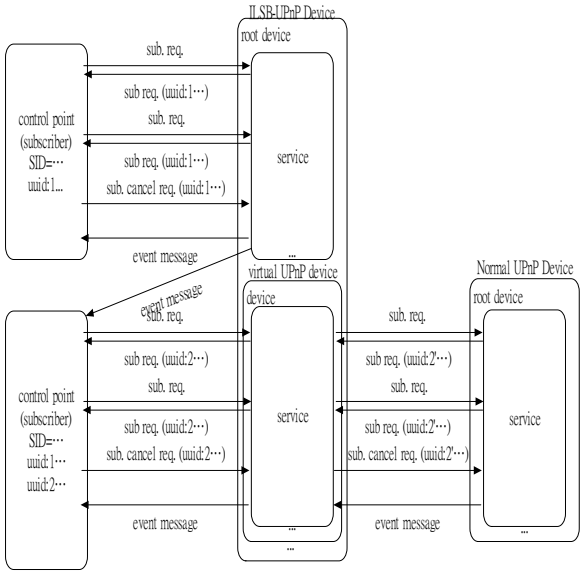


Figure 13 ILSB-UPnP Eventing Flow

Figure 13, the UPnP Integrated Eventing, is partitioned into two parts. One is a device eventing from the physical or normal embedded device, and one or more eventing capabilities exposed by the local device. The other is a device eventing from the virtual or fake logical containers and one or more eventing exposed by another UPnP device. Likewise these eventing are like UPnP standard eventing including subscribe, unsubscribe, and eventing message.

When a UPnP Control Point sends a request to subscribe an eventing, the ILSB-UPnP device will distinguish which type of eventing. Is it a physical or logical eventing. If it is a logical eventing, the ILSB-UPnP device will like a proxy agent and send the same eventing request to subscribe another UPnP device eventing. After the response message from another UPnP device, the ILSB-UPnP device will send this response message to its subscriber. These are presented in the Figure 13. However the subscribing and un-subscribing eventing are corresponded UPnP eventing norm. Hence the ILSB-UPnP Device is like a proxy agent, we will define these policies in the following.

- When receiving the subscribing/ un-subscribing eventing message, the agent will reproduce this message and modify some related fields for using it to subscribe/un-subscribing eventing to another UPnP device.
- Using reproduce message to subscribing/un-subscribing eventing to another UPnP device and waiting for its response
  Using reproduce message to subscribing/un-subscribing eventing to another UPnP device, we can find out the UPnP device address via UPnP Virtual Device List (It is a mapping table between Virtual Device Service and Physical Device Service).
- When receiving subscribing/ un-subscribing response message from another UPnP device
  When receiving subscribe/un-subscribe response message, the agent will reproduce this message and modify some related fields and response it for its subscriber.
- When receiving eventing message from another UPnP device
  When receiving eventing message from another UPnP device, the ILSB-UPnP will reproduce this message, modify some related fields, and send it to the eventing subscriber.

3 Integrated Logical Structure-Based UPnP Analysis

This section presents the comparison of the proposed ILSB-UPnP Architecture with the standard UPnP. We also analyze the availability, efficiency, transparency, and easy-to-use of the proposed architecture.

3.1 Availability

Here, we assume that all UPnP device nodes have the same up-probability *p*, which is the probability that a single node is up operational and there are *n* nodes in the eHome environment. Thus, the availability of the UPnP normal device is *p*. It is very clear to analyze this availability. Since a control point always accesses a device node at one time. If the probability of this node is *p*, then the availability of eHome UPnP architecture is *p* at the same time.

Next, we analyze the availability of the ILSB-UPnP Architecture. Let *AV(i)* be the function for evaluating the availability of the *i*th node environment. If the network consists of only one node, it degenerates to a central controller and the availability of itself, i.e., $AV(1) = p$ where $0 \leqq p \leqq 1$. Next, for the two nodes the availability is at least one node being up. Thus:

$$AV(1) = p \quad (1)$$

$$AV(2) = p2 + 2p(1 - p) = p(2 - p) \geqq AV(1) \geqq p \quad (2)$$

$$\because 0 \leqq p \leqq 1. \quad (3)$$

$$\therefore AV(2) \geqq AV(1) \quad (4)$$

$$\therefore AV(n) = pAV(n-1) + (1-p)AV(n-1) + p(1-p)^{n-1} \quad (5)$$

$$\because AV(n-1) \geqq p \quad (6)$$

$$AV(n) \geqq AV(n-1) \quad (7)$$

Finally, let $AV(n-1) \geqq p$ be the probability that *n-1* nodes in the network and the *n*th node is added to the network. Hence the availability of *n* nodes can be presented in eq. (7). According above we discussion. The availability is greater than p. Thus, the availability of the ILSB-UPnP architecture is better than normal. Figure 14 illustrates the *AV(n)*-availability of the ILSB-UPnP between *n*=1 to *n*=20, where *p* is between 0.1 to 0.9. Obviously, as *n* is increased, the *AV(n)*–availability is increased for a fixed *p*. Moreover, as *p* is increased, the *AV(n)*-availability is increased for a fixed *n*.

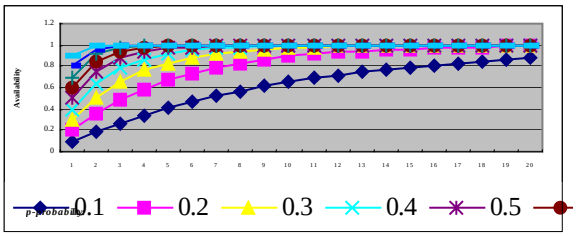


Figure 14 The availability of the ILSB-UPnP Architecture

3.2 Efficiency

In this section, we define a UPnP environment to verify the efficiency. Suppose there are n Media Servers, like media storage server, in the network and each service would be spent t milliseconds. Hence in the normal UPnP architecture, to play a video file must find out each server and navigate it. Thus we analyze each steps of the normal UPnP architecture and our proposed via discovery, description, control, and event protocol. We suppose a normal UPnP device will spend t milliseconds for each service. Hence in the normal UPnP device, the operator will spend t(ms)*n in discovery step, t(ms)*n in description, t(ms)*n in control step and the total time is 3tn(ms) since in the discovery and description must have to operate each UPnP device. However in the integrated device, the operators only operate a integrated device. Hence it will spend t(ms)*1 in discovery step, t(ms)*1 in description. In the control step the operators spend time is the same since the integrated is like a proxy and service for control point. The following is the comparison to spend time between theirs.

Table 1 Comparison of normal and ILSB- UPnP

	Normal	Integrated
Discovery	*t* (ms)**n*	*t*(ms)*1
Description	*t* (ms)**n*	*t*(ms)*1
Control	*t* (ms)**n*	*t*(ms)*1+ *t*(ms)*(*n-1*)
Event	N/A	N/A
Total	3*tn*(ms)	*2t+tn*(ms)

Based on above table; in practical, if the t is 300(ms) and n is 20. The spend time of Normal is 18 seconds and the spend time of Integrated is 6.6 seconds. The spend time of ILSB-UPnP Architecture is less than normal UPnP. Hence the efficiency of the ILSB-UPnP architecture is better than normal.

3.3 Transparency

Based on the above A/V environment, in the normal UPnP architecture, the probability to find out a special video file from n Media Server is 1/n. However in the ILSB-UPnP is 100%. Since it is like a proxy and can invoke any request to backend for control point. Thus its probability is 100%. That means the end user doesn't need to known where the Media Server is and how to control the data path. Hence the transparency of ILSB-UPnP architecture is better than normal.

3.4 Easy-to-Use

Since the architecture of the ILSB-UPnP strategy is like a proxy which cans communication with each other. These communications construct an integrated system in the eHome environment. When a new UPnP device is plugging in and then talking to share their services. These

behaviors will establish a whole system. Each ILSB-UPnP Device has the same capability and share mutual message. For the control point, it is more convenient and don't care where the device is and how to submit the data transmission. Thus the data transfer and control is more transparent. Hence this architecture is more easy-to-use than normal.

4. Conclusion

In this paper we have proposed a Logical Devices Strategy, called ILSB-UPnP system that integrates UPnP normal technology to enable a whole array of operations, for increasing the availability of services discovery, improving the service control transparent and its grouping concept, and providing the eventing notification awareness in eHome. We have enhanced the SSDP to collect each normal UPnP device and integrated device lists in the network and provided a logical structured-based device node as virtual device nodes to the UPnP device. Comparing to the normal UPnP architecture, the ILSB-UPnP architecture can provide a higher availability, better performance, and easier-to-use, especially in an environment that devices are highly distributed. However, the ILSB-UPnP SSDP protocol will increase the traffic of the network and impact a running service quality, for it will interrupt the running service and reconstruct a new logical device list based on the proposed strategy.

Reference

[1] "Universal Plug and Play Device Architecture", http://www.upnp.org/download/UPnPDA10_20000613.htm

[2] Michael Jeronimo and Jack Weast, UPnP Design by Example, A Software Developer's Guide to Universal Plug and Play, Intel Press, April 2003

[3] "About Salutation", http://www.salutation.org/about.htm

[4] "An Introduction to the Service Location Protocol (SLP)", http://www.openslp.org/doc/html/IntroductionToSLP/

[5] "What is SLP?", http://www.openslp.org/

[6] "Appendix B. Introduction to SLP and Open SLP", http://www.caldera.com/support/docs/volution/vm/ig/appendixb.html

[7] "Introduction",http://www.zvon.org/tmRFC/RFC2614/Output/chapter1.html

[8] "SLP WHITE PAPER TOPIC", http://www.ipsi.fraunhofer.de/mobile/teaching/mobinkom_ws0304/5Qos/slp_white_paper.html, May 1997

[9] Bill Venners,"Introduction to Jini", http://www.artima.com/javaseminars/modules/IntroToJini/, May 26, 2005

[10] Bill Venners, "What Is Jini Technology?", http://www.artima.com/objectsjini/introJini.html, March 12,2000

[11] SDSU & Roger Whitney San Diego State University, " Jini Intro", http://www.eli.sdsu.edu/courses/spring99/cs696/notes/jiniIntro/jiniIntro.html#Heading3, March 1999

[12] Phil Bishop, " Jini Technology Introduction, Part One", http://www.jini.org/Newsletter/DesignCorner/jini_intro_may05.html, May 2005

[13] "OSGi Technology", http://www.osgi.org/osgi_technology/index.asp?section=2

[14] "About the OSGi Alliance", http://www.osgi.org/about/index.asp?section=1

[15] "Open Source Linux SDK for UPnP Devices 1.2.1 (libupnp)", http://upnp.sourceforge.net/

[16] "Universal Plug and Play in Microsoft Windows XP", http://www.microsoft.com/technet/prodtechnol/winxppro/evaluate/upnpxp.mspx

[17] "Intel(R) Software for UPnP Technology", http://www.intel.com/technology/upnp/

[18] Jiunn-Der Wu, Tse-Min Chen and

Chi-Yang Hu, "UPnP MP3 Player Design and Implementation".
[19] Y. Mazuryk and J. J. Lukkien, "Improved Eventing Protocol for Universal Plug and Play".
[20] Y. Mazuryk and J. J. Lukkien, "Aanlysis and Improvements of the Eventing Protocol for Universal Plug and Play".