

## Mobile Media Content Sharing in UPnP-Based Home Network Environment

CHIH-LIN HU, WEI-SHUN LIAO\* AND YEN-JU HUANG

*Department of Communication Engineering*

*National Central University*

*Taoyuan, 320 Taiwan*

*\*Graduate Institute of Communication Engineering*

*National Taiwan University*

*Taipei, 106 Taiwan*

Many modern handheld devices feature functions of taking pictures, shooting video clips, or recording audio sounds. However, such devices are usually equipped with small display panels and poor acoustic equipments. In order to achieve better user experience with high quality panel or audio stereo, users often need to transfer or copy digital media contents to external powerful devices, like personal computers and home theater devices. Nevertheless, it can be awkward or difficult for users to follow up a series of operations of connection setup and media content transfer. In this paper, we present a mobile media content sharing mechanism based the UPnP framework and wireless communication technology in home network environments. As a result, users could conveniently share media contents stored in mobile devices and play them by other home networked player devices. The demonstration result shows that the developed mechanism is attractive and of practicability and ease of use.

**Keywords:** content sharing, mobile application, multimedia service, handheld device, Universal Plug and Play (UPnP), home network, mobile computing

### 1. INTRODUCTION

More and more mobile handheld devices, such as mobile phone and personal digital assistant (PDA), are equipped with digital still camera (DSC), audio recording and audio video (AV) processing modules, with that users can take pictures, shoot video clips, or record audio sounds conveniently and effectively. Notwithstanding, due to intrinsic dimensional limitation and power consumption concerns, such devices have smaller display panels and powerless AV playing functions. To enjoy media contents with high performance and quality, users often need to copy or transfer media contents to other external, powerful and professional player devices instead, through complicated manual setup steps with various data cables or connectors – ordinary content transfer operations which are troublesome and unfriendly for end users, empirically.

To mitigate such a circumstance, many research and industrial communities have contributed to realize the digital home vision. Among others are the Digital Living Home Alliance (DLNA)<sup>1</sup> and the Universal Plug and Play (UPnP) Forum<sup>2</sup>. The DLNA interoperability guideline [1] recognizes the UPnP device architecture [2] as the core network-

---

Received February 8, 2007; revised August 28, 2007; accepted January 4, 2008.

Communicated by Jonathan Lee.

<sup>1</sup>Digital Living Home Alliance (DLNA) website: <http://www.dlna.org/>.

<sup>2</sup>Universal Plug and Play (UPnP) Form website: <http://www.upnp.org/>.

ing framework used to develop interoperable home networked platforms and devices in home network environments. Particularly, the UPnP framework defines a base set of services for all home networked devices to follow, and conventions for describing devices and services. It is established on the top of IP layer, and leverages existing Internet standards, including TCP/IP, HTTP and Web technologies, to enable peer-to-peer proximity networking, device discovery, event notification, control action, and data transfer functions. Namely, it is independent of operating systems (OS) and hardware platforms. It can work with any type of physical-layer networking medium, in spite of wired or wireless one.

The increasing popularity of the UPnP technology makes clusters of networked devices, such as PC, entertainment equipment, and intelligent appliance, interconnected in home network environments. Hence, substantial interest in research and industrial communities comes into the examination of UPnP methodologies and performance [3-8]. It is noted that the UPnP architecture has a light-weight design and is of simplicity without complicating the system design. Unfortunately, there are still several shortcomings and problems found in different function layers, including problematic addressing transition [8], inefficient UPnP advertisement and power consumption [4], non-scalable M-Search [5], inflexible control action [6], and poor event notification [3, 7], which will be later described in section 2.3.

Over the past years, our work developed a UPnP middleware which successfully passed the UPnP Implementers Corporation (UIC) conformity [9] and was incorporated into BenQ™ home entertainment devices, for example, network TV. In this paper, we describe the design and the development of a mobile media content sharing mechanism in UPnP-based home network environments. We integrate the WLAN technology into the UPnP middleware, and noticeably implement an interesting mobile content sharing scenario on a specific mobile phone platform. In compliance with the DLNA interoperability guideline, we develop a mobile media server which is able to discover, interconnect with, and transfer media files to other UPnP-/DLNA-compliant devices. Accordingly, our developed mobile phones and home networked devices can communicate with each other through UPnP networking. Mobile phones can serve as “temporary” mobile content servers at any time they want to share stored media contents to be processed directly on external media player devices. As a result, users can conveniently and effectively share their favorite media contents anywhere and anytime, without need of unfriendly manual operations of connection setup and data file transfer, in a home network environment where the proposed mechanism is deployed.

The rest of this paper is organized as follows. Section 2 describes background knowledge, potential issues and related works about the UPnP technology. Section 3 mentions the proposed mechanism and its design issues. Section 4 presents the prototype implementation and discussion. Section 5 gives the conclusion and future research topics.

## 2. BACKGROUND AND RELATED WORK

We inspect several device and service discovery systems in section 2.1 and introduce basic knowledge about UPnP technology and its extension for multimedia services, UPnP AV [10], in section 2.2. Section 2.3 discusses the potential performance problems on UPnP Networking. Some related works are reviewed in section 2.4.

## 2.1 Device and Service Discovery

Device and service discovery is essential to enable devices and services to properly discover, configure, and communicate with each other. Specifically, *discovery* is a mechanism for dynamically referencing a resource on the network. Clients find resources automatically rather than needing pre-configured bindings to specific resources. Table 1 lists several discovery protocols we peruse to derive which is better designed for deployment in home environments [11, 12]. In terms of four major component categories, *i.e.*, topology, network scope, search and identity, the UPnP is deliberated better for home or small enterprise environments. Explicitly, Service Location Protocol (SLP) is oriented towards enterprise service discovery and heavyweight directories. Salutation is primarily used for special services with directory lookup and hard service states. Bluetooth is based on actual physical proximity rather than closeness in the IP routing domain, and only for profile-defined services are its effect of computation and bandwidth utilization. Jini rests on that all devices throughout a network support Java runtime capabilities, however, difficult to be realized. Bonjour popularized by Apple Inc. is an interesting proprietary solution which meshes closely with existing Internet standards, while operating in the absence of the traditional managed Internet infrastructure. However, its human-readable naming rule is not necessary to home appliances since users prefer automatic control by another intelligent device.

**Table 1. Comparison of discovery protocols with major component categories.**

System	Topology	Scope	Search	Identity
UPnP	P2P	subnet	simple	unique ID
Jini	P2P & Directory	bridgeable	medium	unique ID
Bonjour	P2P	subnet	simple	non-distinctive
Bluetooth SDP	P2P	subnet	medium	unique UUID
SLP	P2P or Directory	bridgeable	complicated	unique URL
Salutation	P2P or Directory	subnet	medium	unique
Others <sup>a</sup>	P2P or Directory	–	–	unique

<sup>a</sup>Others: Ninja, INS, eSquirt IR, RFID, and *etc.* [11, 12].

## 2.2 UPnP Device Architecture and its AV Extension

The UPnP device architecture [2] specifies three entities: *device*, *service* and *control point* (CP). A UPnP device can be any entity implementing the UPnP protocols. Services are functions which devices can provide to users, and what service a device shall implement depends on the device type description. A CP likes a client, while a device acts as a server offering services. A CP can invoke actions on services, and provide any required parameters to and receive any return value from devices.

A UPnP device performs six fundamental function layers, as shown in Fig. 1 (a).

- *Addressing* is an underlying function by which a device gets a unique IP address as newly joining a home network. The UPnP framework uses two address assignment methods: dynamic host configuration protocol (DHCP) [13, 14] and automatic IP con-

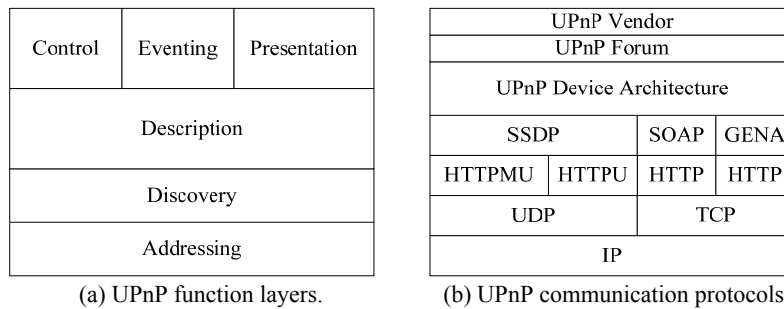


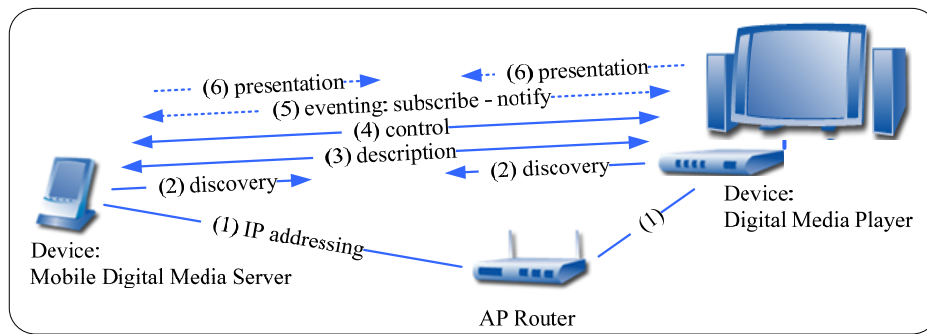
Fig. 1. UPnP function layers and communication protocol stacks.

figuration [15]. The latter is used as the DHCP service is not available.

- *Discovery* is the capability for a device to advertise its appearance and its services in a home network; so, CPs can find the device and its advertisement.
- *Description* is used by a device to present its services and capabilities in a well-defined format; so, CPs can parse its description file and know what it offers.
- *Control* is the capability for a device to handle requests from CPs and to invoke specific actions.
- *Eventing* is used by a device to notify the registered CPs of service state changes.
- *Presentation* is an HTML-based interface provided by a device for users to control or monitor the device directly.

On top of standard TCP/IP protocol stacks, the UPnP networking technology utilizes existing Internet protocols with compliant manipulations to support UPnP functions. Referring to Fig. 1 (b), we address the Simple Service Discovery Protocol (SSDP) [16], Simple Object Access Protocol (SOAP) [17] and General Event Notification Architecture (GENA) [18], as follows. The SSDP is a simple HTTP-based discovery mechanism used to discover local resources in a small, local area network with no need of centralized configuration, management, or administration. Each client directly queries the network, and each resource directly responds to the request. The SOAP integrates both HTTP and XML to provide a Web-based messaging and remote controlling mechanism. Its message body is used to specify control operations, and is expressed by XML and sent via HTTP. The GENA is in essence a publisher-subscriber system. A client (subscriber) registers at a service (publisher) for later notification about any changes of service states. The publisher responds a subscription ID and valid duration to the subscriber. Subsequent operations such as renewal and cancellation use the subscription ID to reference to the specific subscription. When the subscriber or publisher is no longer interested in the subscription, it will be cancelled by either.

Fig. 2 depicts an interaction procedure to ease exposition of UPnP functionality. In general, once a device is powered on, it tries to get an IP address first, organizes its description file, and broadcasts its advertisement by discovery function. When the device is found by a CP, it can be controlled by, send events to, and do presentation to a CP. Meanwhile, a CP can subscribe to services, later receive event notifications, and operate control actions. Note that addressing, description, discovery and control functions must be implemented imperatively; eventing and presentation parts are optional as needed.



1. A device joins in a home network and is assigned with an IP address, either by DHCP or Auto-IP.
2. A device sends SSDP messages for device and service advertisement onto a special multicast address 239.255.255.250:1900.
3. A CP receives device advertisement from 239.255.255.250:1900.  
A CP requests the device for accessing the device and the service descriptions.  
The devices sends both device and service XML-based descriptions to the CP.
4. A CP sends SOAP action messages to control the device. A device responds to SOAP actions.  
For example, a media player (with CP included) can send a *browse* action to obtain the set of meta data of media objects that a media service can provide.
5. A CP subscribes to some services provided by the device.  
The device responds a subscription ID to the CP and later notifies it of GENA event messages (to inform any changes of service statuses).
6. The device may provide a Web page to present itself in a home network.

Fig. 2. The interaction procedure in a UPnP network environment.

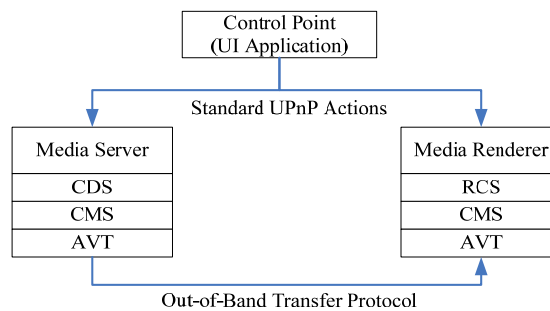


Fig. 3. UPnP AV architecture.

The UPnP AV [10], based on the UPnP framework, further defines three roles, *media server*, *media renderer* and *control point* (CP) in support of AV entertainment in home networks. As depicted in Fig. 3, the following describes each role with its service.

A *media server* serves as a media contents source in a home network. Its primary purpose is to allow CPs to enumerate media contents available for users to render them. Accordingly, three services, Content Directory Service (CDS), Connection Manager Service (CMS) and AV Transport Service (AVT), are defined below.

- *CDS* provides a set of actions that CPs invoke to enumerate media contents. For example, the “browse” action allows CPs to obtain detailed meta-data about media items available on the media server. The meta-data records the supported transfer protocols and data formats, and so CPs use it to determine if an indicated media renderer is able to render a selected media item.
- *CMS* manages the connections associated with a particular device. A CP invokes the primary “preparing-for-connection” action to notify a media server to prepare for upcoming media transfer. This action returns an AVT instance ID to the CP for later controlling the transfer flow and releasing the connection.
- *AVT* controls the player operations, such as stop, pause, seek, *etc.* Specially, it is allowed that AVT can take proprietary out-of-band transfer protocols for media content transfer [10].

A *media renderer* is to render media contents available in a home network. It allows a CP to control what and how media contents are rendered. Accordingly, three services, Rendering Control Service (RCS), Connection Manager Service (CMS) and AV Transport Service (AVT), are defined.

- *RCS* provides rendering actions for a CP to control the rendering of media contents, for example adjustment of brightness, contrast, and volume, for fine tune.

The CMS and AVT are the same as those of a media server. Minor difference in CMS is a CP checks if a media renderer is capable of rendering a specific media item.

### 2.3 UPnP Potential Problems

Prior works have examined the performance of the UPnP solution and found several problems in terms of discovery, control and eventing. Liong *et al.* [4] manifested that periodic SSDP-based discovery messages can impact network congestion and power dissipation of small electronics in an asymmetric home network where a Bluetooth Personal Area Network is bridged to an Ethernet network or Wireless LAN with higher bandwidth. Though UPnP M-Search can control the jitter bound to avoid implosion caused by huge multicast queries and responses, it may fall in a weak performance in large home networks. Mills *et al.* [5] devised an adaptive jitter control method that characterized performance of various combinations of jitter bound and network size.

As for UPnP control, SOAP was criticized to conflict with the REST principles [19] in designing an ideal Web-based network service. It obscured the semantics of HTTP POST without promotion to the original functionality. To improve SOAP response time, Newmarch [6] designed a REST-based method that can coexist with the existing one. It was shown that this method outperformed the original in aspects of request-response time, size of request/response payload, and memory consumption.

From the view of service-oriented architecture [20], Mazuryk *et al.* [7] examined the UPnP and remarked the scalability problem in discovery, inflexibility of fixed TCP time-out and impractical binding of arguments in control, inefficiency of eventing over TCP, and anonymity in UPnP API. The analytic result showed UPnP eventing was the weakest spot. They decomposed the eventing protocol into UDP-based transport level

and application level protocols, thereof avoiding TCP time-out and tearing down connections. Hu *et al.* [3] devised a compatible multicast complement which improved the efficiency of UPnP eventing without ambiguity and heavy implementation.

## 2.4 Related Works

Many academic and industrial researches have worked on related areas. Kim *et al.* [21] developed a home network system prototype for networked appliances by UPnP middleware. They focused on controlling traditional home appliances, not mobile devices. Al-Ali *et al.* [22] designed a Java based automation system for home devices. They focused on using Java platform and controlling by PC-based web page, which was different from UPnP middleware. Rich *et al.* [23] designed a collaborative system for home networked devices. They designed an interface for controlling traditional home appliances; however, no mobile or portable devices were involved. Choi *et al.* [24] developed a context-aware middleware for controlling home devices by Open Service Gateway initiative (OSGi) framework. Their work presented a control mechanism based on higher level service structure instead of UPnP technology. H. Kuriyama *et al.* [25] developed a translator for remotely controlling conventional home appliances without networked function, rather than developing a new device type. In this paper, we design a mobile media content sharing mechanism on the mobile phone platform and accordingly introduce a new device type, *i.e.*, mobile media server. The proposed framework is compliant to the UPnP and UPnP AV specifications. Therefore, users enjoy their mobile media contents via any home networked media players in a convenient way.

## 3. MOBILE MULTIMEDIA SHARING MECHANISM

This section describes the design of the proposed mechanism as well as design issues in course of developing the mobile media server and media renderer devices.

### 3.1 Design Abstract

The objective of this mechanism design is to make home networked devices interconnected and to enable mobile content sharing anywhere in a household playground. Consider a user scenario: a user can use a mobile device with storage units to easily and conveniently share stored media contents to home networked media player devices. Both mobile device and media player device involved are required to support UPnP and UPnP AV mechanisms, as mentioned in section 2.2. The mobile device is the media server, and the networked player device is the media renderer. A control point can be combined with either of them or as a dedicated device, and so used to interact with them and to negotiate media formats and transfer protocols.

Fig. 4 illustrates the user scenario to be developed and demonstrated in this paper. When a user took media contents, a picture or a video clip for example, and stored it in the mobile device such as mobile phone, the user can share media contents by means of the UPnP protocol via WLAN to a networked player device, for example, a network TV (BenQ™ DTV), which was fulfilled in our previous work [9]. To extend the utility of

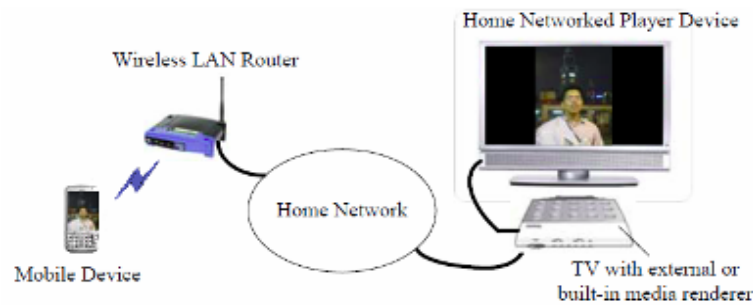


Fig. 4. Proposal of mobile content sharing scenario.

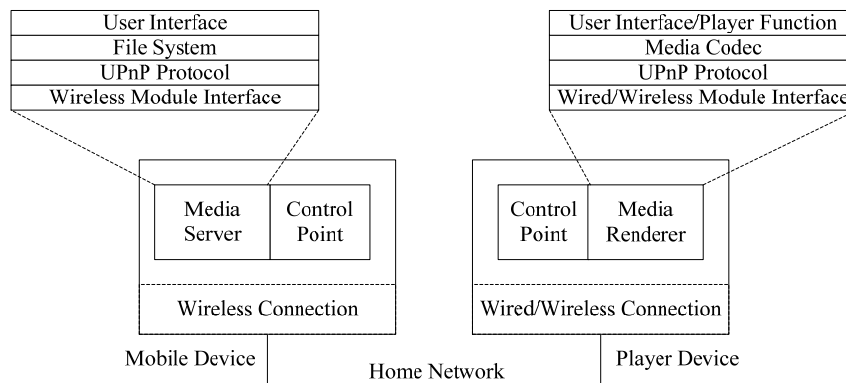


Fig. 5. System block diagram.

each device, the control point function is implemented on both mobile device and player device. In this way, a user can select the desired media item from either mobile server side or player side, therefore making this scenario more effective and convenient.

### 3.2 Media Server on Mobile Device

The design of the media server on a mobile device follows UPnP AV specification and uses WLAN as its transportation medium. On the top of the UPnP AV are an effective file system and a friendly user interface. As shown in Fig. 5, there are four significant layers: user interface, file system, UPnP protocol, and WLAN networking function. We briefly address design considerations of each layer, as follows.

- User interface (UI) is a top application. The most important design principle is user friendliness. There are two specific UIs: browse UI and control UI. Since a mobile device serves a mobile media server, a user can browse the file system through the browse UI to select the media content directories and files to be shared. The design of browse UI can classify the media content by media types, such as photo, music and video, so that the user can quickly and effectively find out the target media content. In

addition, the control UI displays available media renderers from which the user can select a favorite one to play indicated media contents.

- The file system is often related to the operating system (OS) chosen and is embedded-system-dependent. So, the primary design concern on file system is whether the OS adopted is advisable or whether there is adequate application program interface (API), *e.g.*, file descriptions and access operations, for development. Otherwise, the developer shall implement API porting layers to some extent to integrate the file system onto the target platform.
- The developed UPnP protocol layer herein contains both functions of media server and CP device classes. Significantly, we shall consider several system design issues. First of all, due to the limited memory space on mobile device, the code size should be compact and optimized. Second, task priority and manipulation are crucial because UPnP can invoke a number of tasks to handle different function modules, including web client and server, XML parser, network I/O, *etc.* Third, different sorts of UPnP messages may have various sizes. The UPnP kernel should be sophisticated to manage kernel resource carefully and avoid fatal exceptions like heap overrun and out of memory. Finally, independence design should be clear to maintain concurrence and stability since media server and CP run simultaneously.
- The WLAN function adopts IEEE 802.11b/g technology. In practice, this layer is highly hardware dependent. Its performance strongly depends on wireless chipset solution, embedded hardware architecture, configuration of network input/output interface, and driver implementation.

### 3.3 Media Renderer on Player Device

The media renderer on a player device is devised according to the UPnP AV in spite of what network medium used. As shown in Fig. 5, the media renderer includes user interface and player function, media codec, UPnP protocol, and physical network layers. Most functions are similar to those in the media server. We specially address several design considerations in user interface and player function, and media codec layers.

- There are two specific user interfaces: browse UI and control UI. The browse UI must be friendly for a user to easily browse or search on-line available media servers on the network. Moreover, browse UI must be *proactive* in transparently distinguishing or filtering out media content of which media formats are not supported by the renderer. So, a user will not inadvertently play unsupported media content and be prompted by exception or warning messages. As for control UI, its design has to support graphic interaction about basic operations, like play, stop and pause, corresponding to the control and decoding progress at media codec layer.
- The design of the media codec module should be extensible to support addition/removal of media formats on the media player device, and to open/close the support of any specific media format. This function layer should develop a structural, uniform interface for media codec management.

Same as the media server, the utilization of system resource is critical. The concurrency control must be considered when CP and media renderer run simultaneously on the

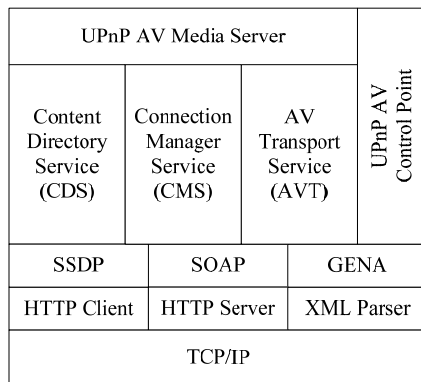
player device. Again, we develop UPnP middleware and applications on software platforms, and performance of network medium layer will be hardware-dependent.

#### 4. PROTOTYPE IMPLEMENTATION AND DISCUSSION

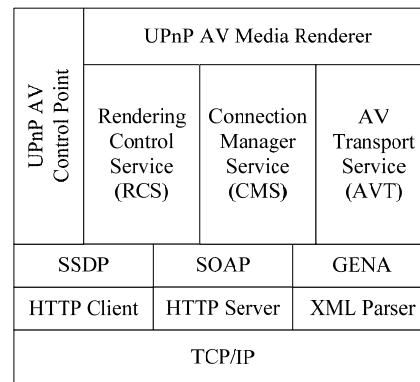
This section presents the prototype implementation and discusses related issues. Sections 4.1 and 4.2 mention the implementation and concerns about mobile device and player device. Section 4.3 exhibits a prototype demonstration. In light of the DLNA interoperability guidelines, section 4.4 addresses several remarks about gateway device, media transport and flow control in UPnP-based home network environments.

##### 4.1 Mobile Device Implementation

As depicted in Fig. 6 (a), the UPnP middleware on the mobile device is built on the base of TCP-IP layer. Upon it, we implement CDS, CMS, and AVT services as well as media server functions as mentioned in sections 2.2 and 3.2.



(a) Middleware stack on mobile device.



(b) Middleware stack on player device.

Fig. 6. Middleware stack.

For simplicity and compatibility of AVT function, we develop a proprietary out-of-band media transfer mechanism by means of HTTP communication protocols. It is employed on both media server and media renderer to stream media objects between them. Specifically, the HTTP GET and POST transfer methods are utilized to behave in a request-response manner. One side can send an HTTP GET message to request an indicated media object. The other side responds an HTTP POST message which encapsulates the binary content of that media object. As a result of HTTP communications, the media transfer acts like in an interactively streaming manner.

The CDS service checks all shared media items and summarizes their information into meta-data. Since the representations of object identifier and parent identifier of each media item are not standardized in UPnP AV, we customize CDS service to meet our requirements of media content management.

As for the file system layer, because the native file system, provided by the mobile handheld platform, is usually too simple to satisfy developers' and users' demands, we develop a virtual file system instead with friendly and flexible UI structures for end users to process media content browsing and other operations. Referring to the design of browse UI, the virtual file system can catalog media items into image, audio, or video directories according to their formats rather than putting them into ordinary directories.

## 4.2 Player Device Implementation

The UPnP middleware implemented on player devices, as shown in Fig. 6 (b), has similar function blocks as those on the media server, except media renderer functions. The CMS and AVT services, and media codec layer are specially addressed below.

Getting protocol information is a key action in CMS. The CMS checks available media codec modules on the media renderer, and specifies supported media formats and file types in CMS protocol information. With CDS and CMS services, a CP negotiates a common media format for the media server and renderer to play an indicated media file.

The AVT service on media renderer is symmetric to that on mobile server. By using the HTTP Range header field to specify the access data range of a content binary file, we implement play, stop and pause actions on mobile devices as well as trick action, like fast forward/backward. Note that the implementation of trick mode is complicated somewhat. The newly launched DLNA v1.5 guideline [26] further specifies two implementing methods: client-oriented HTTP Range method and server-oriented method. The former is that the player device manipulates the values of Range header field in sequential HTTP GET requests to randomly access the target content binary. In the latter, the media server handles the target response according to time seek or play speed specified in the DLNA-defined TimeSeekRange.dlna.org or PlaySpeed.dlna.org header field.

We observe that the performance of HTTP-based method can be degraded by the intrinsic properties of HTTP and limited bandwidth throughput that mobile devices can offer. Relatively, the implementation effort of the server-side method is mostly on media server side. This method is thus applicable if the media server runs on a powerful desktop PC or media gateway device, but may unfortunately make a mobile device overloaded, especially as streaming video content. At present, we fulfill the HTTP-based method to perform the AVT service, and leave the server-side method till the mobile device has enough network capacity and computing resource. Incidentally, as mentioned before, AVT is out-of-band and has less influence on system design. We consider the alternative use of real-time transport protocol, like RTP/RTSP, in the future implementation.

Implementing media codec layer is critical. The following describes the flow of media codec process. In the beginning, the process extracts the HTTP MIME-type (data format) and file extension of the media item from parsed meta-data. Only mime-type is enough to identify its data format in most cases. For reliability, our implementation double checks the file extension to avoid an unwarned application break or fatal exception. Then, the process maps the resolved format to a specific program-defined value, instead of using the originally extracted string of data format, to facilitate program processing. The media codec checks what type the received data stream is, image, audio or video, and in turn calls the proper media codec to process the media item. Otherwise, the error handler will be called if the incoming data type is unknown or unsupported. After the

media codec finishes decoding the incoming data stream, the player outputs the decoded data to the target rendering devices such as display or audio speaker.

### 4.3 Prototype Demonstration

We describe the development environments and show the mobile media content sharing architecture in reference to the user scenario aforementioned in section 3.1.

#### 4.3.1 Prototype development environment

Our development takes into account two situations. First, we have to make reuse of UPnP code module and platform migration (exporting UPnP source library onto different target platforms). Second, only C software development environments most embedded devices support. Thus, we implement the UPnP and UPnP AV code modules in ANSI C language, and port them onto target mobile media server and media player devices.

The prototype of mobile media server is implemented on the Windows PocketPC 2005 environment. We use Microsoft MFC to develop UI components. All source components are synchronized to BenQ™ P51 PDA phone as an experimental device with IntelPXA27x with WMMX, 520 MHz CPU, 128 MB RAM and Windows PocketPC 5.1 Operating System. Incidentally, we experience depreciated UI in small part when we integrate the MFC UI into Windows PocketPC 2005. It is because GUI and graphic library packets are practically platform-dependent.

For the role of media renderer, we use a previously developed media renderer [9] with Phillips PNX1500, 300 MHz CPU, 64 MB Flash RAM and pSOS Operating System. Our media renderer device currently supports several media formats, including (image) JPEG, PNG, BMP and TIF, (audio) LPCM, MP3, MPEG1-L2, (video) MPEG1, and MPEG2. As for the establishment of demonstration context, any wireless AP router with support of IEEE 802.11 a/b/g and DHCP server function can be adopted to interconnect the mobile media server and media renderer devices in a home network.

#### 4.3.2 Scenario demonstration

Fig. 7 presents the snapshots of scenario demonstration. We run a mobile media server on BenQ™ P51 PDA phone. Through the browse UI, a user can select to share media contents stored in the phone or external memory card. As the storage unit is indicated, media contents are classified by their media formats into image, audio and video categories. In this way, users can have a friendly content browsing experience.

The capability of CP is respectively integrated with the media server and the media renderer, as mentioned before, to facilitate user's operations. There are two ways to play a media item. First, a user with a mobile media server in P51 can browse the local media contents, through the browse UI on the CP functions, and *push* a media item to the indicated renderer devices. Second, a user with a media renderer can browse the remote media contents in P51. As the play action is activated, the target renderer device can immediately *pull* a media item for rendering.

As illustrated in Fig. 7, on the media renderer side (output to TV screen), a user can use a remote IR controller to operate the CP to search available media servers on the network. The mobile media server on P51 is highlighted. The user can further browse the

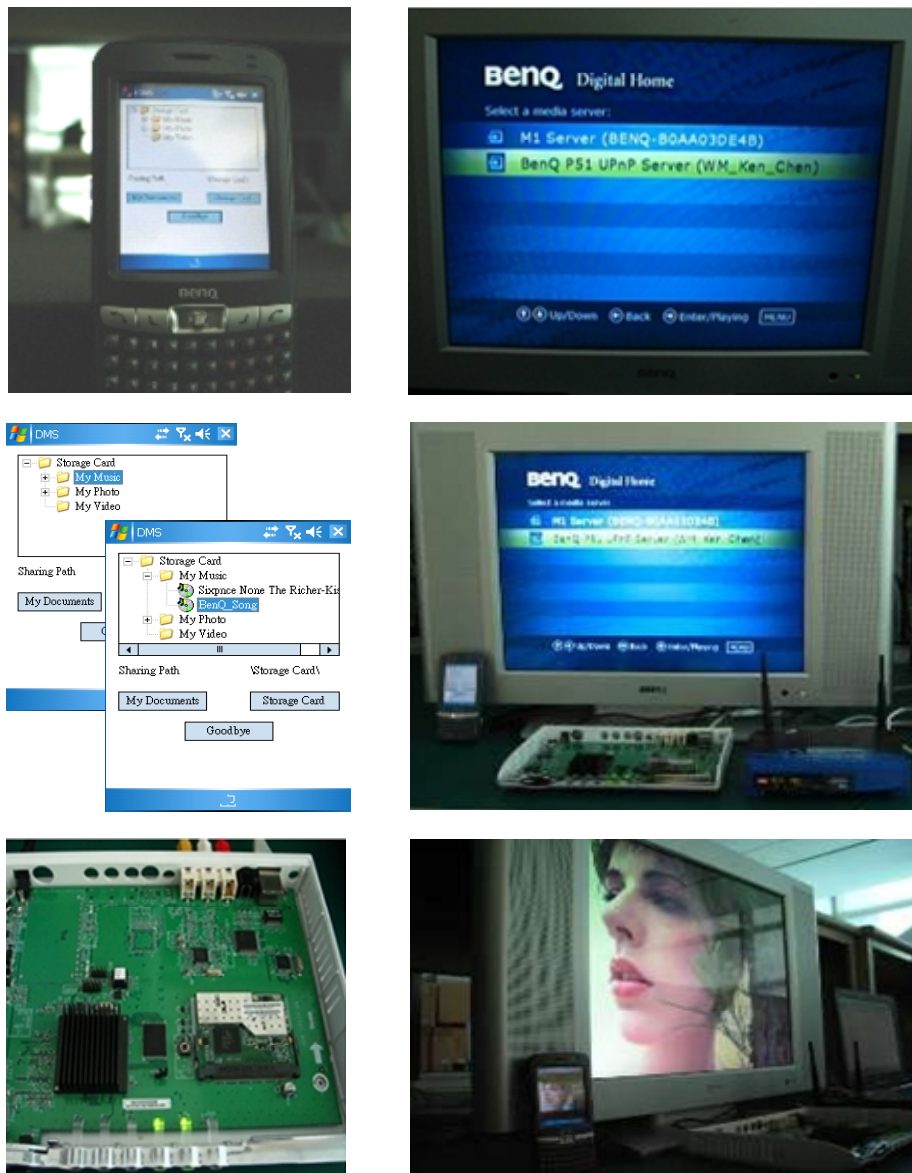


Fig. 7. Scenario demonstration.

media contents and select an interested media item. For example, a user shows a picture, stored in the mobile media server, on the TV screen, outputted by the media renderer. There are not manual operations of connection setup and media content copy/transfer.

#### 4.4 Additional Remark

The above has mentioned in place several design and development challenges and issues. We further take some remarks about the gateway device and flow control below.

#### 4.4.1 Gateway device in transcoding, transrating, or scaling

In light of the UPnP and DLNA specifications, the task of decoding media is indeed acted at the play device side. A mobile device can just stream media items to the player device which takes charge of decoding service. In practice, this behavior is much similar to the media file transfer. On the other hand, the role of home gateway service (on a powerful PC or dedicated device) is attractive in digital home vision. Mobile devices with limited computing and storage resource can leverage its content transformation ability, like transcoding, transrating or scaling of content binary, in distributing media content. Therefore, the newly launched DLNA interoperability guideline v1.5 [26] defines the home infrastructure device (HID) category that includes two device classes, mobile network connectivity function (M-NCF) and media interoperability unit (MIU), to support network interoperability and media format interoperability. Notwithstanding the design of a home gateway device is orthogonal to the proposal in this paper, it is complementary to mobile media content sharing scenarios.

#### 4.4.2 Quality of service (QoS) in flow control

Since the DLNA interoperability guideline v1.0, the HTTP communication protocols are used as the media transport baseline for media transfer across a home network [1]. With the support of standard Multipurpose Internet Mail Extension (MIME), the content receiver is able to identify the type of binary data contained in a streaming file, and invoke the corresponding decoding function to process the received data. However, due to device differentiation about system designs and restrictions, UPnP allows the use of proprietary out-of-band media transport protocols for AVT Service. The DLNA v1.5 hence adds the RTP as an optional media transport [26]. At present, we deliberate the parallel development of RTP protocols onto other target devices, subject to the significant limitations of embedded device system and resources.

In accordance with the DLNA v1.0, our design delivers the media file in a default, best-effort quality of service (QoS) fashion for immediate playback over the HTTP communications. To support the increasing system usages, the DLNA v1.5 defines a structure of priority-based QoS control and a set of transfer modes. There are three basic media transfer modes: streaming, interactive and background modes. Each transfer mode can be associated with different priority levels. A DLNA-defined application-level QoS tag, *i.e.*, DLNAQS\_UP = 0, 1, 2 or 3, is so used to correlate the underlying settings of “802.1Q user priority,” “Wi-Fi wireless multimedia access category,” and “differentiated services code point” to a particular DLNA traffic type. We notice that the media transfer in the DLNA v1.0 is closely mapped to a streaming transfer mode in terms of functionality.

### 5. CONCLUDING REMARK AND FUTURE WORK

Many modern mobile handheld devices enable users to take pictures, shoot video clips, or record audio sounds, and to store them in mobile devices. However, due to the intrinsic dimensional limitation and resource and power limitation, mobile devices cannot offer users high-performance and high-quality multimedia enjoyment. Although us-

ers can have external, powerful or professional media player devices, unfortunately, they often suffer from an awkward, inconvenient way to share and transfer multimedia contents stored in mobile devices to external high-end media player devices. To resolve this situation, in this paper we have developed an efficient mobile content sharing architecture that is based on the UPnP and UPnP AV technologies and is compliant with the DLNA interoperability guidelines. It is appealing that our architecture supports a convenient and effective way for users to share their media contents on mobile devices to other home networked devices. We have demonstrated that with the control point design, users can browse and select media items on mobile devices and render them on the player devices; meanwhile, through the friendly UI design, the network connection and media streaming are transparent to users.

Currently, we are investigating several design issues, development challenges and requirements on our mobile content sharing architecture. The future work will include several aspects. First, we will study those UPnP essential issues and propose solutions to upgrade the performance and robustness of UPnP technologies. Second, we will continue to extend the provision of more effective media codec functions on either the media server or media renderer, associated with high-definition content formats for the upcoming trend of high-quality multimedia. Third, we will develop suitable media streaming protocols, such as RTP/RTSP, deployed on both media server and media renderer sides to support various playing operations, hereby satisfying user requirements. Finally, premium content protection, such as Digital Rights Management (DRM) on content source or Digital Transmission Contents Protection over IP (DTCP-IP) [27] on transmission channel, is not supported yet. The future work will incorporate the above into our developed home-networked multimedia content sharing architecture.

## REFERENCES

1. Digital Living Network Alliance, "DLNA home networked device interoperability guidelines version 1.0," 2004.
2. UPnP Forum, "UPnP device architecture 1.0 version 1.0.1," 2003.
3. C. L. Hu, Y. J. Huang, and W. S. Liao, "Multicast complement for efficient UPnP eventing in home computing network" in *Proceedings of IEEE International Conference on Portable Information Devices*, 2007.
4. Y. L. Liong and Y. H. Ye, "Effect of UPnP advertisements on user experience and power consumption," in *Proceedings of the 2nd IEEE Consumer Communications and Networking Conference*, 2005, pp. 91-97.
5. K. Mills and C. Dabrowski, "Adaptive jitter control for UPnP M-Search," in *Proceedings of the 38th IEEE International Conference on Communications*, Vol. 2, 2003, pp. 1008-1013.
6. J. Newmarch, "A RESTful approach: clean UPnP without SOAP," in *Proceedings of the 2nd IEEE Consumer Communications and Networking Conference*, 2005, pp. 134-138.
7. Y. Mazuryk and J. J. Lukkien, "Analysis and improvements of the eventing protocol for universal plug and play," in *Proceedings of the IASTED Conference on Communications, Internet and Information Technology*, 2004, pp. 419-424.

8. C. L. Hu, *et al.*, "Network address transition methods and systems," R.O.C. Patent No. I-281335, May 11th 2007 to April 7th 2025.
9. BenQ™ DTV, UPnP Implementers Corporation (UIC) certified device, device name: BenQ DTV, device type: MediaRenderer v1.0, device model: DTVWM1100RV1, <http://www.upnp-ic.org/certification/default.asp>.
10. UPnP Forum, "UPnP AV architecture: 0.83: for universal plug and play version 1.0," 2002.
11. W. K. Edwards, "Discovery systems in ubiquitous computing," *IEEE Pervasive Computing*, Vol. 5, 2006, pp. 70-77.
12. F. Zhu, M. W. Mutka, and L. M. Ni, "Service discovery in pervasive computing environments," *IEEE Pervasive Computing*, Vol. 4, 2005, pp. 81-90.
13. RFC2131, "Dynamic host configuration protocol," 1997.
14. RFC2132, "DHCP options and BOOTP vender extensions," 1997.
15. RFC3927, "Dynamic configuration of IPv4 link-local addresses," IETF, 2005.
16. Internet-Draft, "Simple service discovery protocol/1.0 operating without an arbiter," IETF Internet-Draft draft-cai-ssdpv1-03.txt, 1999.
17. W3C Consortium, "Simple object access protocol," W3C Consortium: [www.w3.org/TR.2000/NOTE-SOPA-20000508](http://www.w3.org/TR.2000/NOTE-SOPA-20000508), 2000.
18. Internet-Draft, "General event notification architecture base: client to arbiter," IETF Internet-Draft draft-cohen-gena-pbase-txt, 2000.
19. R. Fielding, "Architectural styles and the design of network-based software architecture," <http://www.ics.uci.edu/fielding/pubs/dissertation/top.htm>.
20. G. Bieber and J. Carpenter, "Introduction to service-oriented programming," <http://www.openwings.org/download.html>, 2001.
21. D. Kim, *et al.*, "Design and implementation of home network systems using UPnP middleware for networked appliances," *IEEE Transactions on Consumer Electronics*, Vol. 48, 2002, pp. 963-972.
22. A. Al-Ali and M. Al-Rousan, "Java-based home automation system," *IEEE Transactions on Consumer Electronics*, Vol. 50, 2004, pp. 498-504.
23. C. Rich, *et al.*, "Collaborative help for networked home products," in *Proceedings of IEEE International Conference on Consumer Electronics*, 2005, pp. 209-210.
24. J. Choi and D. Shin, "Research and implementation of the context-aware middleware for controlling home appliances," *IEEE Transactions on Consumer Electronics*, Vol. 51, 2004, pp. 301-306.
25. H. Kuriyama, *et al.*, "Home appliance translator for remote control of conventional home appliance," in *Proceedings of the 20th International Conference on Advanced Information Networking and Applications*, 2006, pp. 346-350.
26. Digital Living Network Alliance, "DLNA home networked device interoperability uidelines version 1.5," 2006.
27. Digital Transmission Licensing Administrator (DTLA), "Digital transmission content protection specification: volume 1 (revision 1.4)," 2005.



**Chih-Lin Hu (胡誌麟)** received the B.S. and M.S. degrees in Computer Science from National Chengchi University and National Chung Hsing University, and the Ph.D. degree in Electrical Engineering from National Taiwan University in 1997, 1999 and 2003, respectively. He was a researcher at BenQ and Qisda Advanced Technology Centers, Taipei, Taiwan from 2003 to 2007. He had the honor to get the best paper award in IEEE ICPADS-2000, and BenQ Innovation Awards in 2006 and 2007. Since 2008, Dr. Hu is currently an Assistant Professor in Department of Communication Engineering, National Central University, Taoyuan, Taiwan. His research interests include mobile agent technology, home networking technology, broadcast information systems, mobile data management, and mobile and pervasive computing systems. Dr. Hu is a member of IEEE and a member of IICM.



**Wei-Shun Liao (廖偉舜)** received the B.S. and M.S. degrees in Electrical Engineering from National Taiwan University, Taipei, Taiwan, R.O.C., in 2000 and 2002 respectively. In 2002, he joined BenQ Cooperation as an engineer in Networking and Communication Business Group, and then as a researcher in Advanced Technology Center in 2004. In 2006, he joined the Graduate Institute of Communication Engineering, National Taiwan University, and is currently a Ph.D. candidate. His research interests include discrete-time signal processing, wireless communication systems, wireless networks, and wideband communication systems.



**Yen-Ju Huang (黃彥儒)** received the B.S. and M.S. degrees in Electrical Engineering from Yuan Zu University in Taiwan and then joined BenQ Mobile System Company, Hsinchu, Taiwan from 2003 to 2004 to be an engineer in Digital Signal Processing Department. Afterward, at the end of 2004, he joined BenQ Advanced Technology Center, Taipei, Taiwan as a senior engineer until 2007. His research interests include wireless communication, mobile communication network and home networking technology.