UPnP AV
The technology basis of DLNA
July 2014
Contents

• Audio video use cases
• UPnP Technical basics
• Technical basics applied to AV
• UPnP Forum
• DLNA
Audio/Video Use Cases
TV uses Remote control to select content, and plays locally the content

- TV has control point
- PC has Media Server
Pull scenario (Generalized)

Player devices can be in any shape and size
- TV
- Video/Audio Media Players
- Phones

Server devices can be in any shape and size
- PC/Laptop
- Networked Attached Storage (NAS)
- Phones
Phone selects content on PC, and pushes the content to the TV so that the TV plays the content

- PC has media server
- Phone has control point (UI) for selecting content on Media Server
- Phone has control point for pushing content to Media Renderer
- TV has Media Renderer
Control points can also have any shape and size
- PC
- Phone
- Tablets

Media Renderers can also have any shape and size
- TV
- Digital Media Renderers
What is needed to realize these scenarios?

Devices
• Devices should announce themselves on the network
• Devices should announce their capabilities
• Actions should be called on Devices
• State changes should be evented from the Devices

Control points
• Should detect devices automatically on the network
• Should use capabilities to perform a functionality
• Should invoke actions
• Should listen to events
AV specifications
UPnP AV Devices:
• Media Server
• Media Renderer

No 1 to 1 mapping with real world devices/boxes
• PC: Media Player + Media Server + Printer
• TV: Media Player + Media Renderer
• Phone: Media Player + Media Server

Any combination can exist!
(but does not always make sense)
Media Server can have:

- Content Directory Service (CDS):
  Group of actions that allows for browsing and searching the content tree. (CDS is the most important service of a media server, hence often used as synonym for server)
- Connection Manager Service (CMS):
  Group of actions to support initial connection setup and transport type definitions
- AV Transport Service (AVT):
  Group of actions to control streaming and playback
- Scheduled Recording Service (SRS):
  Group of actions that allow to set up timed recordings (AV:2 option)

Media Renderer can have:

- Connection Manager Service (CMS) – reuse!
- AV Transport Service (AVT) – reuse!
- Rendering Control Service (RCS):
  Group of actions that affect how content is rendered (playback)
The *part* of the UPnP stack that *discovers devices* in the network and allows *sending commands* to these devices, and *receive events* from these devices.

A control point is **NOT** a UPnP *device*.

A control point is **NOT** visible on the network.

A control point is **NOT** a control point 😊

It is used to control other devices, hence the implementation depends on *what* can be controlled.

E.g.: Control point for Media server

   Control point for Media server + Media Renderer

   Control point for light control
The Content Directory Service (CDS) provides a *tree of meta data* objects.

The meta data objects describe *object properties* and provides *links to the content*.

The hierarchy of meta data objects can be defined by the application (server).

The presentation of the meta data objects is determined by the UI of the client.

*It is not a file system and not a file system hierarchy.*

Important functions:

- Browse command
- Search command

These API functions will return a list of objects defined by the hierarchy of the server, the list is bound to a subset of a container.

The Media Server (CDS) has no state, except for changes in the CDS hierarchy.
2 box model

- Media Server
  - Connection manager service
    - GetProtocolInfo
  - Content Directory service
    - Browse

- Control Point
  - Player

Issue command
Stream data

© 2014 UPnP Forum
2-Box Play Scenario (1)

CP

Select Server

Select Item
Select resource

Play Stream

Browse

Http-get “Url”

Server
## Media Server

### Functions

<table>
<thead>
<tr>
<th>Service</th>
<th>Function</th>
<th>M</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection Manager Service</td>
<td>GetSearchCapabilities:</td>
<td>M</td>
<td>On what meta-data can we search</td>
</tr>
<tr>
<td></td>
<td>GetSortCapabilities:</td>
<td>M</td>
<td>On what meta-data can we sort</td>
</tr>
<tr>
<td></td>
<td>GetSystemUpdateId:</td>
<td>M</td>
<td>Return current SystemUpdateID for polling for changes in stored content</td>
</tr>
<tr>
<td></td>
<td>Browse:</td>
<td>M</td>
<td>Browse tree of meta-data descriptions</td>
</tr>
<tr>
<td></td>
<td>Search:</td>
<td>O</td>
<td>Search CDS tree, returns list</td>
</tr>
<tr>
<td></td>
<td>more functions…</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>Content Directory Service</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AV Transport Service</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

© 2014 UPnP Forum
Connection Manager Service:1

- GetProtocolInfo R
- PrepareForConnection O
- ConnectionComplete O
- GetCurrentConnectionIDs R
- GetCurrentConnectionInfo R

Possible formats

Only Interested in GetProtocolInfo, because it provides Information about what the media renderer can play or media server can provide

Based on the protocolInfo definition:
<Transport:*:Mime:AdditionalFlags>
Content Directory Service:1

- GetSearchCapabilities R
- GetSortCapabilities R
- GetSortExtensionCapabilities O
- GetFeatureList R
- GetSystemUpdateID R
- Browse R
- Search R
- DestroyObject O
- UpdateObject O
- MoveObject O
- CreateReference O
- CreateObject O
- ImportResource O
- ExportResource O
- DeleteResource O
- StopTransferResource O
- GetTransferProgress O
Objects: Hierarchy

CDS exposes tree of meta data objects

- Hierarchy maintained by @id and parentID
- Hierarchy Starts with Object “0”
- Rest of @id will automatically be retrieved by browse for children

- Objects can be Containers or Items
- Containers can contain Containers or Items
Have required properties:

- **@id**
  Unique object identification in the hierarchy
- **@parentID**
  The objectID of the container above
- **dc:title**
  The title to present to the user
- **upnp:class**
  Definition of what kind of object it is
  (Example: videoltem, albumContainer)

- Many more properties according the class specification
**Container object classes**

`object.container`, the generic container object

Can be extended with more specific definitions:

- `object.container.album.musicAlbum`: contains an audio collection
- `object.container.album.imageAlbum`: contains an image collection
- `object.container.channelGroup.videoChannelGroup`: containing video broadcast items
- `object.container.epgContainer`: containing EPG items

A more precise definition restricts the content in a container

Example:

```xml
<container id="4" parentID="1" childCount="3" restricted="false">
  <dc:title>Brand New Day</dc:title>
  <upnp:class>object.container.album.musicAlbum</upnp:class>
</container>
```
Item object classes

- *object.item* is the generic item
- Items have a defined set of meta data properties
- The items can have resources, a reference to an URL to be played
- Dependent on the class definition the item should have:
  - 0 or more resources
  - set of metadata properties

Example:

```xml
<item id="8" parentID="3" restricted="false">
  <dc:title>Drown</dc:title>
  <dc:artist>Smashing Pumpkins</dc:artist>
  <upnp:class>object.item.audiolItem.musicTrack</upnp:class>
</item>
```
Items can contain resource descriptions

- A Resource contains the URL and attributes to define a “file” which can be retrieved for playback
- Items can have zero or more resources, designating to the same content, but in different (content) formats, sizes, transport protocols, etc
- Protocol info:
  <Transport::*:Mime:DLNAFlags>

```xml
<item id="24_0" parentID="0/IROOT/IALL" restricted="1">
  <dc:title>109-0974B_IMG</dc:title>
  <upnp:class>object.item.imageItem</upnp:class>
  <dc:date>2004-02-21T16:41:13</dc:date>
  <res protocolInfo="http-get::*:image/jpeg:*" size="888322" resolution="1600x1200" colorDepth="24">http://130.145.203.202:49153/content/C:/wouter/content/mixed/109-0974B_IMG.jpg</res>
  <res protocolInfo="http-get::*:image/jpeg:*" size="48322" resolution="320x240" colorDepth="24">http://130.145.203.202:49153/content/C:/wouter/content/mixed/109-0974B_IMG_TN.jpg</res>
</item>
```
Example of a music object

Properties for object.item.audioItem
• upnp:genre
• dc:description
• dc:longDescription
• dc:publisher
• dc:language
• dc:relation
• dc:rights
• At least one resource element

Extra Properties for object.item.audioItem.musicTrack
• upnp:artist
• upnp:album
• upnp:originalTrackNumber
• upnp.playlist
• upnp:storageMedium
• dc:contributor
• dc:date

DLNA has also a list of properties per class definition
Class tree for items

- `object`
  - `item`
  - `container`

- `imageItem` → `photo`
- `audioItem` → `music Track` → `audio Broadcast` → `audio Book` → `movie` → `video Broadcast` → `music VideoClip`
- `videoItem` → `playlist Item` → `textItem` → `bookmark Item` → `epgItem` → `audio Program` → `video Program`

- `Vendor defined class extensions`
Using

• Browse
  • Normal traversal over the predefined hierarchy
• Search
  • Searching for item in the hierarchy

Getting content from the server
Browse Command

```
Browse(ObjectID, BrowseFlag, Filter, StartingIndex, RequestedCount, SortCriteria, Result, NumberReturned, TotalMatches, UpdateID)
```

- **ObjectID:** @id: the identification in the tree
- **BrowseFlag:** browse the current object (for metadata) or the descendants
- **Filter:** Filtering of the parameters in the result
- **StartIndex:** Limiting output: start giving the result back from StartIndex
- **RequestedCount:** Limiting output: amount of objects in the result
- **SortCriteria:** the sorting order of the items in the Result
- **Result:** the list of objects returned
- **NumberReturned:** the amount of objects in the list
- **TotalMatches:** the amount of items in the container
- **UpdateID:** the current updateID of the CDS
Search Command

Search(ContainerID, SearchCriteria, Filter, StartingIndex, RequestedCount, SortCriteria, Result, NumberReturned, TotalMatches, UpdateID)

- **ContainerID:** @id: the container to search from
- **SearchCriteria:** browse the current object (for metadata) or the descendants
- **Filter:** Filtering of the parameters in the result
- **StartIndex:** Limiting output, start giving the result back from StartIndex
- **RequestedCount:** Limiting output, amount of objects in the result
- **SortCriteria:** the sorting order of the items in the Result
- **Result:** the list of objects returned
- **NumberReturned:** the amount of objects in the returned list
- **TotalMatches:** the total amount of items specified by the searchCriteria
- **UpdateID:** current updateID of the CDS
- Contains the AV Transport Service (AVT)
  Is being used for player control functionality
  - Setting an URL for playback: `SetAVTransportURI(URL,Item)`
  - `Play(speed),Stop,Pause, Next, Previous`
  - Trickmodes are done by issuing play speeds with the Play command
    - Should be implemented on client, but can make use of server side impl.

- Contains the Connection Manager Service (CMS)
  Informs control points of the capabilities of the media renderer
  - Codec support
  - Transport protocol support
    - `GetProtocolInfo(Source,Sink)`

- Rendering Control Service (RCS)
  - Control of how the playback is rendered
    - Presets, Audio and Video settings
## Media Renderer

<table>
<thead>
<tr>
<th>Connection Manager Service</th>
<th>Rendering Control</th>
<th>AV Transport Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetAVTransportURI</td>
<td>M</td>
<td>Tell the renderer from where a stream should be obtained</td>
</tr>
<tr>
<td>Play:</td>
<td>M</td>
<td>Start Playing the URI</td>
</tr>
<tr>
<td>Stop:</td>
<td>M</td>
<td>Stop Playing URI</td>
</tr>
<tr>
<td>Previous:</td>
<td>M</td>
<td>Play previous URI</td>
</tr>
<tr>
<td>Next:</td>
<td>M</td>
<td>Play Next URI</td>
</tr>
<tr>
<td>SetNextAVTransportURI:</td>
<td>O</td>
<td>Set next URI already</td>
</tr>
</tbody>
</table>

**Semantics**

- M: Mandatory
- O: Optional

Plenty more optional functions...
AV Transport Service: 1

- SetAVTransport
- Play
- Stop
- Pause
- Next
- Previous
- Seek
- SetNextAVTransportURI
- SetPlayMode
- Record
- GetMediaInfo
- GetMediaInfo_Ext
- GetTransportInfo
-GetPositionInfo
- GetTransportSettings
- GetCurrentTransportActions
- GetDeviceCapabilities
- SetRecordQualityMode
- GetDRMState
- GetStateVariables
- SetStateVariables

Media Controls:
- R
- R
- R
- O
- R
- R
- O
- O

Reporting Capabilities on current play
- R
- R
- R
- R
- R
- R
- O
- O
- O

Reporting Capabilities for next play
- R
- R
- O
- O
- O
AVT has State

State changes will be evented

- **“STOPPED”**
  - Transition Complete
  - Play()
  - Pause()
  - Record()
  - SetAVTransportURI()
  - Stop()
  - SetAVTransportURI()
  - Error

- **“PLAYING”**
  - Seek()
  - Next()
  - Previous()
  - Play()
  - Transition Complete

- **“RECORDING”**
  - Record()
  - Pause()

- **“PAUSED PLAYBACK”**
  - Play()

- **“PAUSED RECORDING”**
  - Pause()

- **“NO MEDIA PRESENT”**
  - SetAVTransportURI()

* Optional behavior
## Media Renderer

### Connection Manager Service
- **ListPresets**
- **SetPresets**

### Rendering Control
- **ListPresets**
- **SetPresets**
- **Get/Set VideoOptions**
- **Get/Set AudioOptions**

### AV Transport Service
- **ListPresets**
- **SetPresets**

<table>
<thead>
<tr>
<th>Service</th>
<th>Action</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>ListPresets</strong></td>
<td>List presets that the device supports</td>
</tr>
<tr>
<td></td>
<td><strong>SetPresets</strong></td>
<td>Set a preset</td>
</tr>
<tr>
<td></td>
<td><strong>Get/Set VideoOptions</strong></td>
<td>set a video playback option</td>
</tr>
<tr>
<td></td>
<td><strong>Get/Set AudioOptions</strong></td>
<td>set an audio playback option</td>
</tr>
</tbody>
</table>

© 2014 UPnP Forum
Use Get/Set actions only when you are controlling the output device.

- Video options should be implemented on a device with a screen
- Audio options should be implemented on a device with a Amplifier
3 Box Model Play

Server
DMS

Client
DMR

Client
Control Point

Play

Browse
User finds
nice content

Instruct
Client to start
playing URL

URL
2-3 box interactions

- Media Server Device
  - Connection manager service
    - GetProtocolInfo
  - Content Directory service
    - Browse

- Control Point
- Media Renderer Device
- Player

- Issue command
- Stream data
AV versioning

• Currently 4 sets of AV specifications exist.
• Each new version is fully backward compatible with a previous version
  • No changes, only additions and clarifications are made
• 5th version in progress
<table>
<thead>
<tr>
<th>version</th>
<th>Service</th>
<th>Addition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AV2</td>
<td>SRS</td>
<td>Scheduled Recording Service</td>
</tr>
<tr>
<td>AV3</td>
<td>CDS</td>
<td>Tracking Changes Option</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Foreign Meta data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FreeFormQuery (XML based Search)</td>
</tr>
<tr>
<td>AV4</td>
<td>CDS</td>
<td>Richer Content Description (ResExt)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enriched Content Navigation (Content Segmentation)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Content Privacy (depends on DeviceProtection)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Resource management (depends on DeviceProtection)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Exclusive ownership</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Time Shift Support (of recordings)</td>
</tr>
<tr>
<td>RCS</td>
<td></td>
<td>Renderer Side Content Transforms</td>
</tr>
<tr>
<td>AVT</td>
<td></td>
<td>Playback Synchronization (precision timed)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enhanced Playback Support</td>
</tr>
<tr>
<td>CMS</td>
<td></td>
<td>Renderer Content Matching (with DRM)</td>
</tr>
</tbody>
</table>
UPnP Audio Video Architecture defines:

• Media Servers; that are discoverable on the network, and expose multi-media content by providing meta-data about the content.

• Media Renderers; that are discoverable on the network, and expose API to play content.

• Control Points which can be used to find other devices, send commands. For example, browsing the content hierarchy of a media server for selecting a song for playback.
DLNA
• DLNA: builds on UPnP to specify the interoperability of devices that share Images, Audio, and AV content on the home network.

• Refers to other existing specifications

• Focuses on system usages (use cases)
System Usages Overview

- 2-Box Pull System Usage
- 2-Box Push System Usage
- 3-Box System Usage
- 2-Box Printing System Usage
- 3-Box Printing System Usage
- Download System Usage
- Upload System Usage
- Download Synchronization System Usage
- Upload synchronization System Usage
- 2-Box RUI Pull System Usage
- 3-Box RUI System Usage
- Recording and EGP System Usage
### DLNA: Interoperability at All Layers
Narrowing the plethora of standards to a mandatory small set

<table>
<thead>
<tr>
<th>Layer</th>
<th>Standard(s)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link Protection</td>
<td>DTCP-IP</td>
<td>How commercial content is protected on the Home Network</td>
</tr>
<tr>
<td>Media Formats</td>
<td>MPEG2, AVC/H.264, LPCM, MP3, AAC LC, JPEG, XHTML-Print + optional formats</td>
<td>How media content is encoded and identified for interoperability</td>
</tr>
<tr>
<td>Media Transport</td>
<td>HTTP&lt;br&gt;Quality of Service</td>
<td>How media content is transferred</td>
</tr>
<tr>
<td>Media Management</td>
<td>UPnP AV 1.0&lt;br&gt;UPnP Print Enhanced 1.0</td>
<td>How media content is identified, managed, and distributed</td>
</tr>
<tr>
<td>Discovery &amp; Control</td>
<td>UPnP Device Architecture 1.0</td>
<td>How devices discover and control each other</td>
</tr>
<tr>
<td>IP Networking</td>
<td>IPv4 Protocol Suite</td>
<td>How wired and wireless devices physically connect and communicate</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Wired: Ethernet 802.3, MoCA&lt;br&gt;Wireless: Wi-Fi 802.11, Wi-Fi Protected Setup</td>
<td></td>
</tr>
</tbody>
</table>
Technical Additions on top of UPnP

- DLNA signaling in Device description
- DLNA protocolInfo extensions
- DLNA defined actions
- DLNA defined state variable values
• Extra signaling in the device description that UPnP devices has an Supported DLNA profile:
<dlna:X_DLNADOC xmlns:dlna="urn:schemas-dlna-org:device-1-0">DMS-1.50</dlna:X_DLNADOC>

• Extra signaling in the device description that UPnP devices has an Supported features:
<dlna:X_DLNACAP xmlns:dlna="urn:schemas-dlna-org:device-1-0">av-upload,image-upload,audio-upload</dlna:X_DLNACAP></device>

having the upload capabilities signaled means that you have to accept HTTP-POST as upload commands.
Extra DLNA information in the protocolInfo 4th field

• More specific mimetype definition: DLNA.ORG_PN
  • More than 100 definitions already defined
  • Most important: Split up of Mpeg Transport and Program stream

• Extra transport settings: DLNA.ORG_OP
  • Time range capable
  • Byte range capable

• Server side trick modes: DLNA.ORG_PS=1,2
  • Play speed normal, double, done on the SERVER side
    The player plays back at normal speed...

• FLAGS
  • Binary transport settings flags

• Example
  “http-get:*:video/mpeg:DLNA.ORG_PN=MPEG_PS_PAL_XAC3;DLNA.ORG_OP=01,
   DLNA.ORG_PS=1,2;DLNA.ORG_FLAGS=03100000000000000000000000000000”
DLNA defined actions

Media Server Extensions

• Upload extension:
  CDS:X_GetDLNAUploadProfiles()
  Function to indicate the DLNA media profiles that will be accepted by CDS:CreateObject()

• Synchronization extension:
  CDS:X_GetTakeOutGroupNames()
  Function identifies content tree to be synchronized.

Media Renderer Extensions

• Extensions for Byte Seek
  AVT:X_DLNA_GetBytePositionInfo()
  Function to indicate the current byte position in the stream
Eventing of possible trick modes supported by a Digital Media Renderer

- AVT.CurrentTransportActions state variable:
  - X_DLNA_PS
  - X_DLNA_SeekTime

Signaling DLNA specific seek instructions

- AVT.Seek input arguments:
  - X_DLNA_REL_BYTE
  - X_DLNA_SeekTime
  - X_DLNA_SeekByte
For the interconnected lifestyle