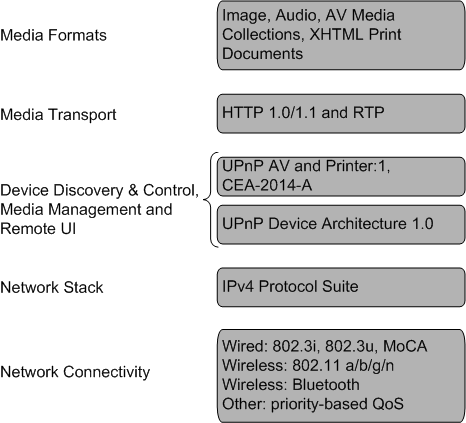
# DLNA总体体系架构



## Networking and Connectivity网络互联方式

包括物理连接的标准，有有线的，比如符合IEEE802.3标准的Ethernet，；有无线的 ，比如符合IEEE802.11a/g标准的WiFi，能做到54Mbps，蓝牙(802.15)等

## NetWorking Stack 网络协议栈

DLNA的互联传输基本上是在IPV4协议簇的基础上的。用TCP或者UDP来传都可以。

## Device Discovery and Control 设备发现&控制

DLNA用UPnP协议来实现设备的发现和控制。UPnP Device Architecture。

## Media Management 媒体管理

媒体管理包括媒体的识别、管理、分发和记录。

## Media Formats 媒体格式

规定了设备应该具有的格式支持能力。

|  |  |  |  |
| --- | --- | --- | --- |
| Video | Audio | | Images |
| MPEG-1  MPEG-2  H.263  MPEG-4 Part 2  MPEG-4 Part 10  WMV9  VC-1 | LPCM  MPEG-1/2 L2  MPEG-1/2 L3  MPEG-4 AAC LC  MPEG-4 AAC LTP  MPEG-4 HE AAC  MPEH-4 BSAC | AC-3  ATRAC3plus  WMA  WMA Professional  AMR  AMR-WB+  G.726 | JPEG  PNG  GIF  TIFF |

对于家庭的和移动终端的要求不同，具体如下：



## Media Transport 媒体传输

这一层用HTTP(HyperText Transfer Protocol)超文本传输协议。就是平时我们上网用的媒体传输协议。HTTP用TCP可靠传输，也有混合UDP方式的HTTP。现在HTTP的最新版本是HTTP1.1。可选协议是RTP。

### 媒体的传输方案

1、从DMS/M-DMS至DMP/M-DMP，即使不立即播放。

2、从一个DMS到另一个DMS，这时接收方DMS播放接收媒体内容，表现为一个DMP；也可以不立即播放，可能只是存储或者处理。

### 媒体传输模式

1)流传输。当DMR/DMP需要实时渲染接收媒体，媒体具时序性。

2)交互传输。不包含时序的媒体，如图片传输。

3)后台传输。非实时的媒体传输，比如上传下载等。

## Remote UI 远程界面

定义了如何描述UI的内容及格式，及如何将这些描述从一个设备传输到另一个设备。

这里对控制界面做了统一规范，使用的是一个xml，远程获取到这个xml就可以自己显示控制界面，如果设备的功能改变了，控制端更新xml刷新控制界面就行了，非常方便。

# UPnP AV(Audio/Video) Architecture 音视频架构

该架构定义了 UPnP 控制端与 UPnP AV设备基本交互，并且与特定设备类型，媒体内容格式与传输协议无关。

## 两种AV设备

UPnP AV MediaServer

UPnP AV MediaRender

## 两种AV设备要具备的服务

**Media Renderer必须包含的服务**：

1 、RenderingControlService渲染控制

这个动作允许CONTROL POINT来控制怎么还原RENDER得到的AV内容，其中包含了控制亮度，音量，对比度等属性。

2、ConnectionManagerService连接管理

这个服务是用来控制与其他设备的连接，其主要的动作是GetProtocolInfo() 其主要是让CONTROL POINT来列举其支持的传输协议和数据格式，并选择与SERVER一致的协议和格式。RENDER也调用PrepareForConnection()，其主要作用是让RENDER准备好建立新的连接，并返回一个ConnectionID让CONTROL POINT来获得连接信息。而且其也返回一个AVTransport InstanceID 来允许CONTROL POINT控制媒体的播放。

3. AVTransport Service（optional)

此服务主要是用来让CONTROL POINT来控制媒体的播放

**Media Server必须包含的服务**:

1 Content Directory Service内容目录

这个服务主要是为了让CONTROL POINT能够列举能够为用户提供的内容项目,其主要的动作就是Browse().它能够让CONTROL POINT了解到每一个内容的详细信息.

2. ConnectionManager Service连接管理

这个服务主要是用来掌控与特定设备的连接,其主要的动作是PrepareForConnection().它能够进行必要的设置使SERVER准备好进行新的连接.

3. AVTransport Service (Optional)音视频传输控制

这个可选服务主要是用来控制媒体的播放,其可被其他传输协议中自带的控制来取代.

## UPnP AV 设备的交互模型

## 2-Box Pull Model

这种情况下CP是MediaRenderer，它可以是一个智能手机。CP主动向MediaServer索取媒体内容，获得内容之后播放媒体，是拉(pull)的方式。

CP要做的是 获得媒体列表>选取所需内容>匹配协议 / 格式，MediaServer需要  选取所需内容>匹配协议 / 格式>开始传输。

### 2-Box Push Model

这种情况下CP是MediaServer，它可以是一个一体机。CP主动向MediaRenderer推送(push)媒体。

### 3-box pull model

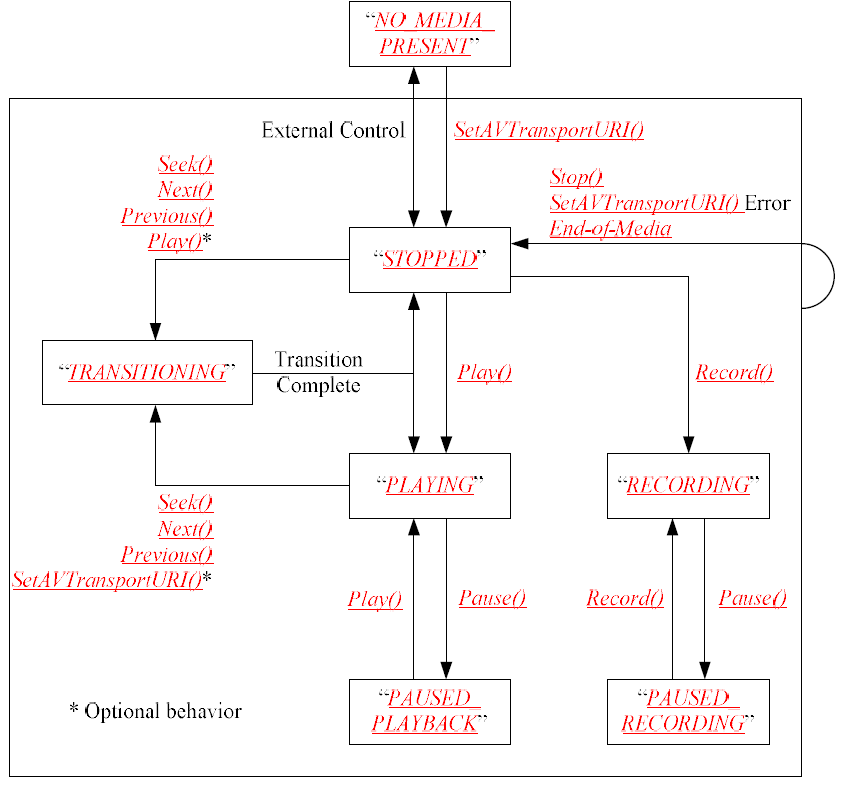
当pull方式时，CP向Renderer发送Server及Server上所需媒体内容的URL，让Renderer去取；

### 3-box pushmodel

当push方式时，CP向Server发Renderer的URL，让Server去向Renderer推送媒体内容。

# AV Transport Service 音视频传输服务

## MediaReader状态



NO\_MEDIA\_PRESENT

没有收到SetAVTransportURI报文，当前播放器没有关联到指定的多媒体资源上。

TRANSITIONING

收到了SetAVTransportURI指令，当前播放器已经关联到指定的多媒体资源上，但是需要一段准备时间才能播放多媒体文件，如下载图像文件中，正在缓冲视频。

PLAYING

当前正在播放由SetAVTransportURI指定的多媒体文件。

STOPPED

播放器关联到SetAVTransportURI指定的多媒体文件，当前已经停止播放。

PAUSED\_PLAYBACK

播放器关联到SetAVTransportURI指定的多媒体文件，当前已经暂停播放。

RECORDING

正存储

PAUSED\_RECORDING

暂停存储

## 传输请求

### SetAVTransportURI

请求设置传输的媒体文件的URI。

请求格式实例：

<?xml version="1.0" encoding="UTF-8" ?>

<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/" s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<s:Body>

<u:SetAVTransportURI xmlns:u="urn:schemas-upnp-org:service:AVTransport:1">

<InstanceID>0</InstanceID>

<CurrentURI>http://192.168.0.103:52100/Music/F2078749.flac</CurrentURI>

<CurrentURIMetaData>

<DIDL-Lite xmlns="urn:schemas-upnp-org:metadata-1-0/DIDL-Lite/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:upnp="urn:schemas-upnp-org:metadata-1-0/upnp/" xmlns:dlna="urn:schemas-dlna-org:device-1-0" xmlns:av="urn:schemas-sony-com:av" xmlns:pv="http://www.pv.com/pvns/">

<item id="F2078749" parentID="0" restricted="1">

<dc:title>Get It On</dc:title>

<upnp:class>object.item.audioItem.musicTrack</upnp:class>

<upnp:artist>Grinderman</upnp:artist>

<upnp:artist role="Performer">Grinderman</upnp:artist>

<upnp:artist role="AlbumArtist">Grinderman</upnp:artist>

<dc:creator>Grinderman</dc:creator>

<upnp:album>Grinderman</upnp:album>

<upnp:genre>Popular</upnp:genre>

<upnp:originalTrackNumber>1</upnp:originalTrackNumber>

<duration>0:03:07.000</duration>

<size>23271070</size>

<dc:date/>

<pv:lastPlayedTime/>

<pv:addedTime>1325001459</pv:addedTime>

<pv:modificationTime>1234569393</pv:modificationTime>

<upnp:albumArtURI dlna:profileID="JPEG\_TN">http://192.168.0.103:52100/AArt/2078749.jpg</upnp:albumArtURI>

<res protocolInfo="http-get:\*:audio/flac:DLNA.ORG\_PN=FLAC;DLNA.ORG\_OP=01;DLNA.ORG\_CI=0;DLNA.ORG\_FLAGS=21700000000000000000000000000000" duration="0:03:07.000" size="23271070" nrAudioChannels="2" sampleFrequency="44100" bitsPerSample="16" bitRate="176400">http://192.168.0.103:52100/Music/F2078749.flac</res>

<res protocolInfo="http-get:\*:image/jpeg:DLNA.ORG\_PN=JPEG\_TN;DLNA.ORG\_OP=00;DLNA.ORG\_CI=1">http://192.168.0.103:52100/AArt/2078749.jpg</res>

</item>

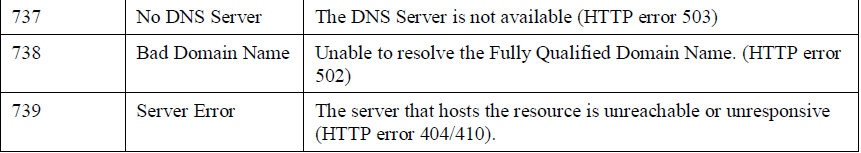
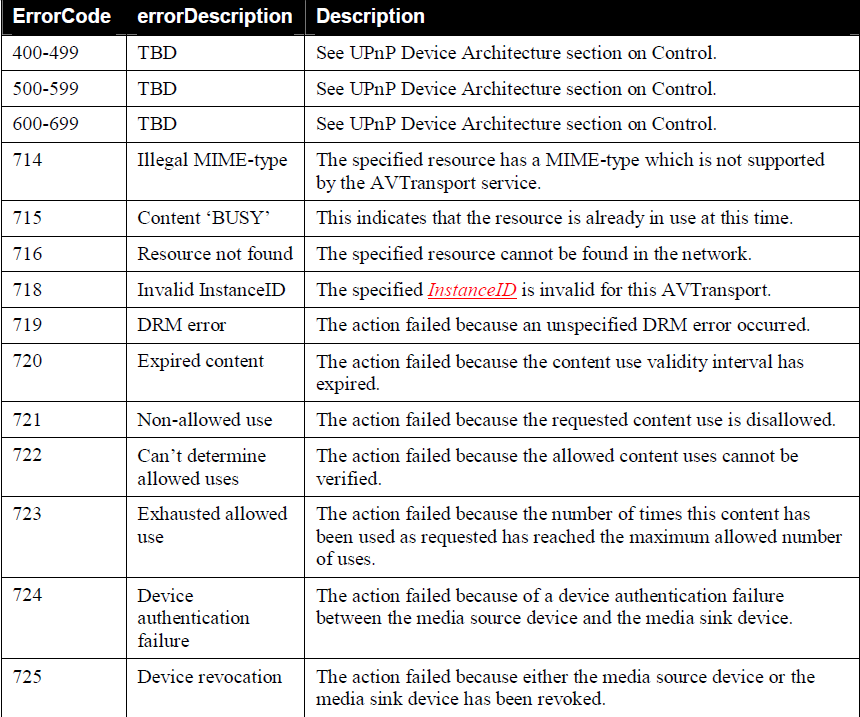
</DIDL-Lite>

</CurrentURIMetaData>

</u:SetAVTransportURI>

</s:Body>

错误码：



DIDL-Lite元素介绍

有且仅有一个 <DIDL-Lite> 元素

有且仅有一个 <item> 或者一个 <container> 元素

有且仅有一个 <dc:title> 元素，并有属性值

零个或者一个 <dc:creator> 元素，并带有属性值

有且仅有一个 <upnp:class> 元素，并带有属性值

至少有一个 <res> 元素

DIDL-Lite文本和片段，且不允许出现xml的注释：<！...>

DIDL-Lite的值不应设在为空或者完全为空格，这样可能导致对端实体无法处理这些值。所以必须使用非空和非空格值。

**dc:title，upnp:class**字段的属性值必须不能大于256个byte,且必须是utf8编码。

所有的**uri**类型字段的长度限制为1024字节，且是utf-8编码。

相应的**dc:class**推荐填充的属性

|  |  |
| --- | --- |
| object.item.imageItem | dc:creator, upnp:album, upnp:genre,  res@duration, res@size |
| object.item.audioItem | dc:date, res@resolution, res@size |
| object.item.videoItem | dc:date, upnp:genre, res@duration,  res@size |
| object.container.album.musicAlbum | dc:creator, upnp:genre, @childCount |

**DLNA.ORG\_OP**：两位的参数，每一位只能是0或1。前一位表示byte seek,后一位表示time seek

01 仅支持time seek  
 10 仅byte seek

11 time seek和byte seek都支持

在protcolInfo的字段中，只有 DLNA.ORG\_PN是限制必须的，其他的都是可选的。

### GetTransportInfo()

获取当前Render的状态。随时都可以发送这个请求，不依赖于播放器的状态。



### GetPositionInfo()

获取当前播放进度。

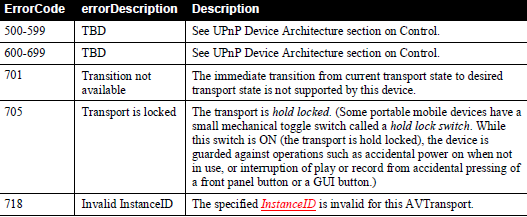
错误码



### Stop()

停止播放。除了 NO\_MEDIA\_PRESENT状态外，其他状态下都可以执行这个请求。状态变更为STOPED.

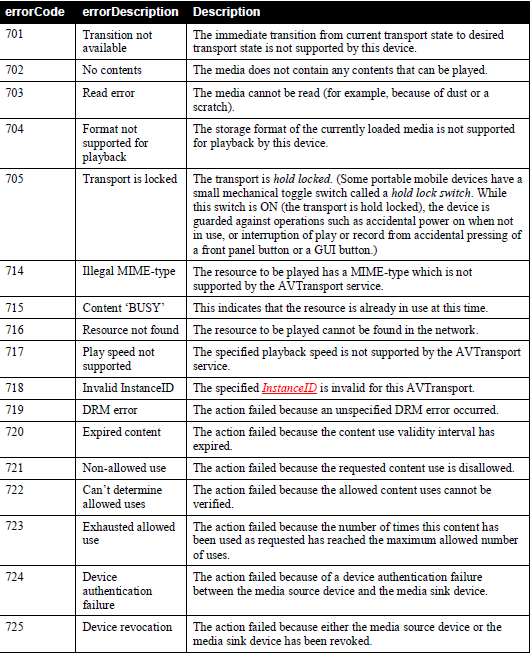
错误码：



### Play()

播放。仅可以在 “STOPPED”, “PLAYING”, 和“PAUSED\_PLAYBACK”做这个请求，否则会收到701错误。

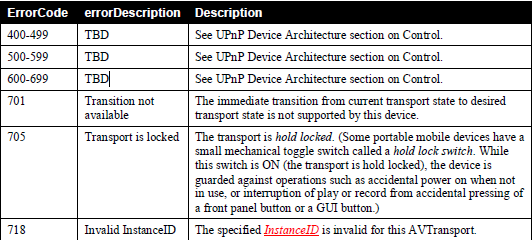
错误码：



### Pause()

停止播放。处于PLAYING和RECORDING状态可以发出这个请求，其他状态则会返回701错误码。

错误码：



### Seek()

播放定位。定位有几种类型：

TRACK\_NR：代表从头开始

ABS\_TIME

绝对时间

REL\_TIME

相对时间。