Faculdade de Engenharia da Universidade do Porto

FEUP

Development of a Virtual Personal Video Recorder based on a Set Top Box

Pedro dos Santos Saleiro da Cruz

Final Version

A thesis submitted under the
Integrated Master in Electrical and Computers Engineering

Major in Telecommunications

Supervisor: Prof. Dr.-Ing. Dirk Elias

Co-Supervisor: Dr. Filipe Abrantes

July 2009

Abstract

Consumers demand interoperability between their consumer electronics (CE) devices and services. On the other hand, consumers are not skilled users. They should easily connect all the devices and immediately share their digital library. The existing Personal Video Recorders (PVR) systems are solutions restricted to the set top box (STB) hard disk.

Service Discovery and Configuration are the key of this project as they allow the system to hide the underlying technologies from the end user providing a better user experience. The *Virtual PVR* based on a STB takes advantage of UPnP and DLNA capabilities to allow users to choose where to store, when and where to reproduce their media content. Powerful and frontend user driven contents and services were implemented. Users are able to access and interact with highly intuitive, rich content and flexible interfaces.

Acknowledgements

I would like to thank everyone at Fraunhofer Portugal Research for the motivation and support they had shown, providing me a unique work environment. My special thank to Filipe Abrantes, for his patience and time availability. I could still remember my family and friends for the comprehension during this journey.

Table of Contents

List of Figures

List of Tables

Abbreviations

ACL	Access Control List
ACS	Auto-Configuration Server
API	Application Programming interface
AV	Audio-Visual
B2BUA	Back to Back User Agent
BRAS	Broadband Remote Access Server
BSP	Broadband Service Provider
BSS	Business Support System
CE	Consumer Electronics
CLI	Command Line Interface
CPE	Customer-premises Equipment
CPU	Central Processing Unit
CWMP	CPE WAN Management Protocol
DHCP	Dynamic Host Configuration Protocol
DLNA	Digital Living Network Alliance
DNS	Domain Name System
DVB	Digital Video Broadcasting
ED	End Device
EPG	Electronic Program Guide
GPL	General Public License
GUI	Graphical User Interface
HG	Home Gateway
HGI	Home Gateway Initiative
HN	Home Network
HTML	HyperText Markup Transfer
HTTP	HyperText Transfer Protocol
IMS	IP Multimedia Subsystem
IP	Internet Protocol
LAN	Local Area Network
LPCM	Linear Pulse Code Modulation
MMS	My Media System
NAS	Network Attached Storage
NGN	Next Generation Network
OS	Operating System
OSS	Operations Support System

PC	Personal Computer
PVR	Personal Video Recorder
QoS	Quality of Service
RMS	Remote Management System
SIP	Session Initiation Protocol
SP	Service Provider
SSDP	Simple Service Discovery Protocol
SOAP	Simple Object Access Protocol
SSID	Service Set Identifier
STB	Set Top Box
Telco	Telecommunications Provider
TCP	Transmission Control Protocol
UI	User Interface
UDP	User Datagram Protocol
UPnP	Universal Plug and Play
URL	Uniform Resource Locator
VoD	Video on Demand
WAN	Wide Area Netwotk
XBMC	XBOX Media Center
XML	Extensible Markup Language

Chapter 1

Introduction

1.1 - Motivation/Context

Nowadays we are living in the “digital” era. Homes were invaded by digital content, such as images, video and music which can be recorded on digital cameras and mobile phones or downloaded from the Internet. It is also possible to record TV series, for later viewing, in digital format or play our favorite movie in the DVD player. Consumers want more and more, so they ask why it is not possible to share all this content across the Home Network (HN) and play it when and where they want it, regardless from the source.

Personal video recorders enable users to record the TV programs for later viewing in digital format in a hard disk. Nowadays, they often offer the possibility to use the information from the electronic program guide (EPG) transmitted by the Telecommunications provider (Telco) to control the recording. The signal from the Telco also can be deviated to a computer, offering more possibilities for signal manipulation, but this requires a skilled user. The keyword is interoperability. On the other hand, consumers are not skilled users. They should easily connect all the devices and immediately share their digital library.

All these innovations must be accessed by everyone regardless of their technical knowledge. The main purpose of this project is to allow it to become true. It is only possible with an intuitive and friendly solution.

1.2 - Project Description

The idea of this project is to develop a proof-of-concept prototype of a *Virtual PVR*. The *Virtual PVR* concept refers to a solution that 1) enables the distribution of functionality typically residing in a Set Top Box among other devices in the home (e.g. NAS), and 2) allows the user to seamlessly and remotely access his PVR system through well-established web interfaces. A HTTP server in the Telco network or in the Home Network hosts a portal where the user can access the HN. Then the user can see the devices connected to his HN, like the NAS (Network Attached Storage) and the STB. In this case the *Virtual PVR* appears and the user can choose the programs to record and where to store it. The EPG transmitted by the Telco is used to build the Web interface that enables setting up the *Virtual PVR*. The user choices are used to configure the devices in the HN, through the STB. In this way, according to the user definitions, the STB streams the desired program to be stored in the NAS or in other media server. Thus, the PVR is virtual and accessible from any terminal with Internet access, from a TV to a cell phone. Powerful and frontend user driven contents and services will be implemented. Users will be able to access and interact with highly intuitive, rich content and flexible interfaces.

This project will develop a communication and networking infrastructure that will be used in future projects of Fraunhofer Portugal Research. This infrastructure will enable the developing of a wide range of solutions to:

- seamlessly connect and configure HN devices according to user pre-defined profiles and device capabilities;
- manage the HN from the Telco's network, including addressing devices;
- maintain, make diagnostics and troubleshoot the HN from the Telco's network.

This will enable the Telco to offer new services to the end user, like user-configurable services that take advantage of CE capabilities. Managed user-content backups, or home security are also examples of such services.

Finally this technical solution will provide a significant saving of expenses in terms of manpower and error/problems solutions, saving time and collaborating to a major user's satisfaction rate.

1.3 - Objectives

The first part of the work consists on studying the existing Service Discovery and Remote Management standards, Set-top-box operation and Media Centers available in the market. Then it is necessary to define an integrated solution of the *Virtual PVR* system. The specification of the complete solution is necessary to develop this project and for future developments on the Fraunhofer Portugal Research.

Below we describe the main features to be implemented in the proof-of-concept prototype:

- Personal Video Recorder (PVR) software prototype running on the set top box that makes UPnP devices in a Home Network visible and manageable.

- A user-friendly web interface that makes visible and streams all images, music and video available in the Home Network.

- Live TV service on the web interface: list and play all TV channels available in the STB.

- PVR service on the web interface: provide TV listings based on EPG to set up the recordings.

- The recorded shows should be stored on any network accessible storage (NAS) available in the user's Home Network (HN).

- The contents should be rendered by any display available in the HN.

- The storage and the renderer are neither fixed nor need to be manually configured by the user.

- Besides the STB prototype, all the CE devices only need to be UPnP/DLNA compliant in order to take part of the system.

1.4 - Document Structure

This Master Thesis Report is divided in six chapters. First of all there is a description of the problem's context and explanation of the main goals of this project. The second chapter is about service discovery and auto configuration, set top boxes operation and media centers. The third chapter defines an integrated solution for the *Virtual PVR* system which comprises UPnP devices and services, CPE WAN Management Protocol (CWMP) modules and a Web Interface. The fourth chapter is a presentation of the developed *Virtual PVR* prototype which implements some modules of the integrated solution explained in the third chapter. The fifth chapter describes some lab experiments that were performed in order to evaluate and validate the *Virtual PVR* prototype described in the fourth chapter. Last chapter is the conclusion which contains a synthesis of the developed work and describes some directions to future work on the *Virtual PVR* integrated solution.

Chapter 2

Background and State of the Art

2.1 - Service Discovery and Remote Management

2.1.1 - Overview

Service Discovery and Configuration are the key of this project as they allow the system to hide the underlying technologies from the end user providing a better user experience. In this Section we provide an overview of some of the technologies of this field: UPnP, DLNA, CPE WAN Management protocol and the Home Gateway Initiative.

- UPnP defines control protocols built on top of IP communication standards with the purpose of simplify configuration of home network by allowing devices to discover each other and announce services.

- DLNA is an UPnP-based standard defined by manufacturers of CE devices and it is focused on home entertainment. It extends UPnP by defining more specific device profiles.

- CPE WAN Management protocol enables network operator to diagnostic and troubleshooting services in the HN.

- Home Gateway Initiative guidelines present some solutions for remote access and management of devices inside Home Network through the Home Gateway. This architecture takes advantage of UPnP and SIP devices, as well as CPE WAN Management Protocol.

2.1.2 - Universal Plug and Play

The Universal Plug and Play (UPnP) Forum was founded in 1999 when some major PC and consumer electronics companies realized that interoperability was the future, as described in [1]. Today it has more than 800 associated companies from all kind of industries, like home automation, telecom and security.

The Forum decided that all network connectivity is based on TCP/IP. Web standards like HTTP, HTML, XML and SOAP provide the framework for device discovery, description, control and presentation. In order to define device and service profiles for specific device categories the Forum created working groups, such as Internet Gateway Device (IGD), Scanning, Printing, Lighting Control and Audio/Video (AV), among others. These working groups developed XML schemas that represents the principal set of functions and services that each device type is required to support.

So why is UPnP technology "universal"? UPnP networking is media independent and instead of using device drivers, it uses common protocols. When a device is first connected to the network it must search for a Dynamic Host Configuration Protocol (DHCP) server, it is the step 0 – ***addressing*** [2]. If it is available then the device must use the IP address assigned to it. If the device's DHCP client doesn´t find one DHCP server, it means the network is unmanaged and the device must auto configure an IP address using Auto-IP.

The step 1 is ***discovery*** [2] and it means that every time a device or a control point is added to the network, the UPnP discovery protocol allows that device to advertise its services to control point or that control point to search for devices on the network. In the case of the device, it must multicast a number of discovery messages advertising itself, its embedded devices and its services. When a control point is newly added to the network it may, if is interested in, multicast a discovery message searching for devices and/or services. The devices must respond if they match the criteria in discovery message. When the device's IP address is changed it must advertise using the new IP addressing, after revoking any earlier announcements. Obviously when a device leaves the network it should multicast a set of discovery messages revoking its earlier announcements, which means, declare to its root devices that embedded devices and services will no longer be available. The UPnP discovery protocol known as Simple Server Discovery Protocol (SSDP) uses part of the header field format of HTTP 1.1 however, it is not based on full HTTP 1.1, it uses UDP instead of TCP and it has its own processing rules. SSDP start line must be one of the following:

```
NOTIFY * HTTP/1.1\r\n

M-SEARCH * HTTP/1.1\r\n

HTTP/1.1 200 OK\r\n
```

In order to advertise its capabilities, a device multicasts a number of discovery messages. For instance, a root device should multicast:

Table 2.1 Root device discovery messages, as described in [2].

Notification Type	Unique Service Name
upnp:rootdevice	uuid:*device-UUID*::upnp:rootdevice
uuid:*device-UUID*	uuid:*device-UUID* (for root device UUID)
urn:schemas-upnp-org:device:*deviceType*:*ver* or urn:*domain-name*:device:*deviceType*:*ver*	uuid:*device-UUID*::urn: schemas-upnp- org:device:*deviceType*:*ver* (of root device) or uuid:*device-UUID*::urn: *domain-name*:device:*deviceType*:*ver*

The step 2 in UPnP networking is ***description*** [2]. This is very important because after the discovery step, control point knows very little about the devices newly added to the network. The UPnP description for a device contains vendor-specific and manufacturer information like the model name and number, the serial number, manufacturer name, URLs to vendor-specific Web sites and is expressed in XML. There is also a list of any embedded devices or services. The description includes, for each device, a list of the commands, or actions, to which the service responds, and parameters for each action. For modeling the state of the service at run time there are a list of variables which are described in terms of their data type, range and event characteristics.

Control point may wish to retrieve the device description document and service description documents when it discovers a device in the network. This process is just a HTTP GET request to the URL in the discovery message, and receives device's description in the body of an HTTP response like in below:

Request

```
GET /descriptionPath HTTP/1.1
HOST: hostname:portNumber
ACCEPT-LANGUAGE: language preferred by control point
```

Response

```
HTTP/1.1 200 OK
CONTENT-LANGUAGE: language used in description
CONTENT-LENGTH: bytes in body
CONTENT-TYPE: text/xml; charset="utf-8"
DATE: when responded
Body
```

Control [2] is the step 3 in UPnP networking. Since Control point has retrieved a description of the device it can send suitable control messages to the control URL for the service, which is provided in the device description. Control messages are expressed in XML using the Simple Object Access Protocol (SOAP). The service returns any results or errors from the action, that is, action-specific values if there are changes in the variables that describe the run-time state of the service. These responses must be sent to the same IP address from which the request was received. In order to invoke an action on a device's service, a control point must send a request in the following format with method POST:

```
POST path control URL HTTP/1.0
HOST: hostname:portNumber
CONTENT-LENGTH: bytes in body
CONTENT-TYPE: text/xml; charset="utf-8"
USER-AGENT: OS/version UPnP/1.1 product/version
SOAPACTION: "urn:schemas-upnp-org:service:serviceType:v#actionName"
<?xml version="1.0"?>
<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<s:Body>
<u:actionName xmlns:u="urn:schemas-upnp-org:service:serviceType:v">
<argumentName>in arg value</argumentName>
<!-- other in args and their values go here, if any -->
</u:actionName>
</s:Body>
</s:Envelope>
```

The service must complete invoking the action and response within 30 seconds and in the following format:

```
HTTP/1.0 200 OK
CONTENT-TYPE: text/xml; charset="utf-8"
DATE: when response was generated
SERVER: OS/version UPnP/1.1 product/version
CONTENT-LENGTH: bytes in body
<?xml version="1.0"?>
<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<s:Body>
<u:actionNameResponse xmlns:u="urn:schemas-upnp-
org:service:serviceType:v">
<argumentName>out arg value</argumentName>
<!-- other out args and their values go here, if any -->
</u:actionNameResponse>
</s:Body>
</s:Envelope>
```

If the service detects any kind of error while invoking the action sent by a control point, it must send an error message within 30 seconds in the following format:

```
HTTP/1.0 500 Internal Server Error
CONTENT-TYPE: text/xml; charset="utf-8"
DATE: when response was generated
SERVER: OS/version UPnP/1.1 product/version
CONTENT-LENGTH: bytes in body
<?xml version="1.0"?>
<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<s:Body>
<s:Fault>
<faultcode>s:Client</faultcode>
<faultstring>UPnPError</faultstring>
<detail>
<UPnPError xmlns="urn:schemas-upnp-org:control-1-0">
<errorCode>error code</errorCode>
<errorDescription>error string</errorDescription>
</UPnPError>
</detail>
</s:Fault>
</s:Body>
</s:Envelope>
```

The step 4 in UPnP networking is ***enventing*** [2] and it is intimately linked to control where control points send actions to devices. The service publishes updates when variables that model the state of the service at run time change and control point may subscribe to receive this information. For that effect, service publishes updates by sending event messages, which includes names and values for all evented variables. There are two kinds of enventing: unicast and multicast. In unicast eventing the subscriber must send a subscription message to the service and wait for an answer. The answer says if it is accepted or not and if it gets a positive response also receives a subscriber time. Then subscriber may renew or cancel the subscription. In multicast eventing, every control points receive updates whenever an action is send to a service by a control point.

Unicast eventing messages must be in the following format:

```
NOTIFY delivery path HTTP/1.0
HOST: delivery host:delivery port
CONTENT-TYPE: text/xml; charset="utf-8"
NT: upnp:event
NTS: upnp:propchange
SID: uuid:subscription-UUID
```

```
SEQ: event key
CONTENT-LENGTH: bytes in body
<?xml version="1.0"?>
<e:propertyset xmlns:e="urn:schemas-upnp-org:event-1-0">
<e:property>
<variableName>new value</variableName>
</e:property>
Other variable names and values (if any) go here.
</e:propertyset>
```

Multicast eventing messages must be in the following format:

```
NOTIFY * HTTP/1.0
HOST: 239.255.255.246:7900  *** note the port number is different than SSDP ***
CONTENT-TYPE: text/xml; charset="utf-8"
USN: Unique Service Name for the publisher
SVCID: ServiceID from SCPD
NT: upnp:event
NTS: upnp:propchange
SEQ: monotonically increasing sequence count
LVL: event importance
BOOTID.UPNP.ORG: number increased each time device sends an initial
announce or update
message
CONTENT-LENGTH: bytes in body
<?xml version="1.0"?>
<e:propertyset xmlns:e="urn:schemas-upnp-org:event-1-0">
<e:property>
<variableName>new value</variableName>
</e:property>
<!-- Other variable names and values (if any) go here. -->
</e:propertyset>
```

The last step (5) in UPnP networking is ***presentation*** [2]. When a device has a URL for presentation is possible that control point retrieve a page from this URL. Then it loads the page into a browser, and depending on the capabilities of the page, it allows a user to control the device and view its status. The presentation URL element in the device description is used for presentations. The control point retrieves a presentation page by an HTTP GET request to the presentation URL.

The simplest action that users will perform is to renderer media content on a specific device. The UPnP sets up a rendering scenario with three distinct components. However, these components can be part of the same "physical" devices. The Media Server contains media content that users want to reproduce in a Media Renderer.

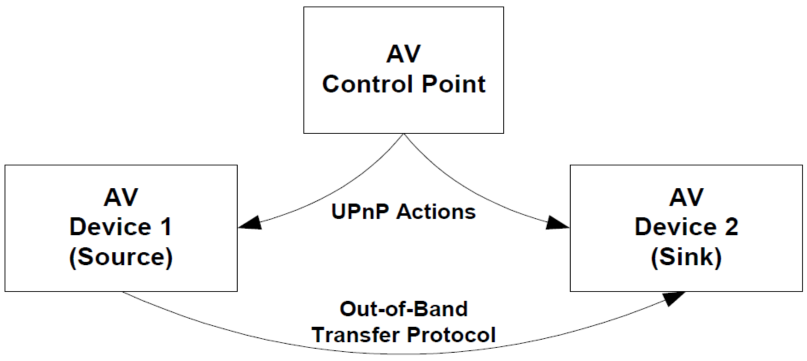


Figure 2.1- UPnP AV device interaction model, retrieved from [3]

The user interacts with an UPnP Control Point interface to find and select the desire content on the Media Server and to select the target Media Renderer [3]. The media server accesses its content and transmits it to another UPnP device using a non-UPnP transfer protocol and a data format understood by Media Server and Media Renderer. Media Server can support one or more transfer protocols and data formats for each content item. The Media Renderer gets content from a Media Server via network and the type of content depends on the transfer protocols and data formats that it supports [3]. The Control Point provides the user interface for the user to interact with, in order to coordinate and manage the operation between Media Server and Media Renderer. Examples of Media Servers are Set Top Boxes or NAS and examples of Media Renderers are TVs.

In [2] , there is a resume of the UPnP networking:

“Discovery is Step 1 in UPnP™ networking. Discovery comes after addressing (Step 0) where devices get a network address. Through discovery, control points find interesting device(s). Discovery enables description (Step 2) where control points learn about device capabilities, control (Step 3) where a control point sends commands to device(s), eventing (Step 4) where control points listen to state changes in device(s), and presentation (Step 5) where control points display a user interface for device(s).”

2.1.3 - Digital Living Network Alliance

Under DLNA’s leadership, the industry has cooperated in the development of workable guidelines for product design that define interoperable building blocks for devices and software infrastructure. It covers physical media, network transports, media formats, streaming protocols and digital rights management mechanisms. Standards for these areas are defined in many different forums and compliance with these standards has been an important first step.

The DLNA Networked Device Interoperability Guidelines were created in a unique cross-industry effort that combined the efforts of over 200 CE, PC, and mobile device manufacturers and other DLNA member companies from around the world, working together with the aim of achieving the world's first substantial platform for true interoperability between consumer electronics, personal computer and mobile devices, as described in [1].

In terms of digital media content, UPnP AV were the most significant guidelines so it became the basis for a new organization. The Digital Living Network Alliance (DLNA) was formed in 2003 by 21 companies including Microsoft, Intel, HP, IBM, Sony, Philips, Toshiba, Pioneer, Motorola and Nokia with the purpose of accelerating the a arrival of interoperable digital media devices at home. Current release of the DLNA Interoperability Guidelines is version 1.5 [4] and the DLNA device model is based on UPnP Forum device model composed by Devices, Services and Control Points.

The basic device model of UPnP was extended so all control still only passes between control point and device(s) but the devices interact with each other to pass digital content using a non-UPnP communications protocol. It could be an Ethernet connection between a PC sending audio to a stereo system or an s-video cable between a DVD player and a TV monitor.

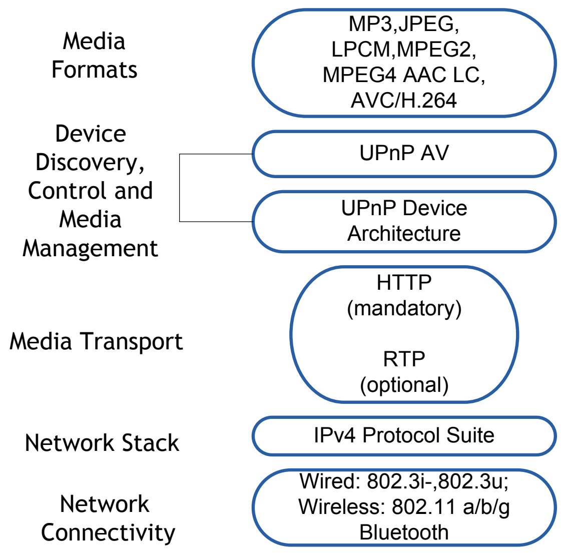

Media Formats
MP3,JPEG, LPCM,MPEG2, MPEG4 AAC LC, AVC/H.264
Device Discovery, Control and Media Management
UPnP AV
UPnP Device Architecture
Media Transport
HTTP (mandatory)
RTP (optional)
Network Stack
IPv4 Protocol Suite
Network Connectivity
Wired: 802.3i-,802.3u; Wireless: 802.11 a/b/g Bluetooth

Figure 2.2- DLNA Interoperability Guidelines Building Blocks, based on [4]

With the purpose of define the characteristics of devices and the services they offer, the DLNA Interoperability guidelines defined 12 device classes organized into 3 device categories [4].

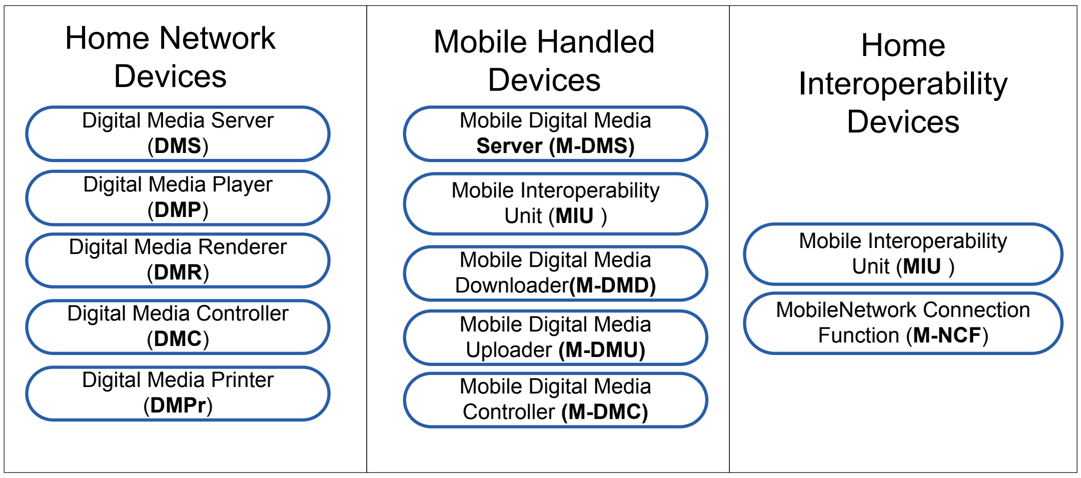

Home Network Devices
Digital Media Server (DMS)
Digital Media Player (DMP)
Digital Media Renderer (DMR)
Digital Media Controller (DMC)
Digital Media Printer (DMPr)
Mobile Handled Devices
Mobile Digital Media Server (M-DMS)
Mobile Interoperability Unit (MIU)
Mobile Digital Media Downloader(M-DMD)
Mobile Digital Media Uploader (M-DMU)
Mobile Digital Media Controller (M-DMC)
Home Interoperability Devices
Mobile Interoperability Unit (MIU)
MobileNetwork Connection Function (M-NCF)

Figure 2.3 Device categories, based on [4]

The Home Network Device (HND) category is made up of five device classes that are in use in the home network, and rely on the same media formats and network connectivity requirements.

1-Digital Media Server (DMS)

o PC Server on an Ethernet based home network
o Personal Video Recorder
o Advanced Set-top Boxes
o Dedicated Music Servers
o CD/DVD Jukebox
o Digital Cameras and Camcorders
o Cell Phones with Digital Cameras

2-Digital Media Player (DMP)

o Digital TVs
o Stereo / Home Theater Systems
o Game Consoles
o PDAs
o Multimedia Mobile Phones

3-Digital Media Renderer (DMR) – Unlike DMPs, they are unable to find content on the network, and must be setup by another network entity – a DMC.
4-Digital Media Controller (DMC)
5-Digital Media Printer (DMPr)

The Mobile Handheld Device (MHD) category:

6-Mobile Digital Media Server (M-DMS)
7-Mobile Digital Media Player (M-DMP)

8-Mobile Digital Media Controller (M-DMC)
9-Mobile Digital Media Uploader (M-DMU)
10-Mobile Digital Media Downloader (M-DMD)

The Home Infrastructure Device (HID) These devices are intended to allow HNDs and MHDs to interoperate.

11-Mobile Network Connectivity Function (M-NCF)
12-Media Interoperability Unit (MIU)

DLNA guidelines establish that the home network communications backbone is based on IP networking UPnP and IEC technologies. Next there is a description of the main blocks that are needed to achieve interoperability.

• "*Transparent connectivity between devices inside the digital home*" [4]: the objective is to allow end-to-end connectivity between devices that belong to the same home network. Devices directly connected to each other should have networking compatibility at the link layer (TCP/IP layer 2). Layer 2 bridging and layer 3 routing allows communication between devices of different layer 2 technologies.

•" *Unified approach for device discovery, configuration and control*" [4]: devices must be able to find other devices on the network. These capacities include configuring devices, services and control their operation.

• "*Interoperable media formats and streaming protocols*" [4]: different devices should agree on choosing appropriate streaming protocol and media formats. It is the only way to guarantee that media can be shared, consumed and rendered.

•*" Interoperable media management and control"* [4]: it includes the ability of organize, browse, search and select media items to be processed as well as controlling the operation of media streaming sessions.

• *"Compatible quality of service mechanisms"* [4]: The Quality of Service (QoS) is very important specially for transferring high-definition media streams. Even when there are not QoS mechanisms, devices must interoperate.

• *"Compatible authentication and authorization mechanisms for users and devices"* [4]: several authentication and authorization mechanisms that allow devices to request and/or grant access to particular devices and services are being considered by manufacturers and application developers.

DLNA interoperability guide lines are very explicit concerning media formats profiles.

They have attributes, parameters, system and compression level details defined in sufficient detail to ensure interoperability between DLNA compliant devices. The focus on media formats is a dividing line between the work of the UPnP Forum and DLNA. The UPnP Forum focused on achieving device interoperability, which was accomplished. But the lack of prescribed media profiles prevented the UPnP architecture from delivering media interoperability and led to the founding of DLNA.

Table 2.2 Media formats, based on [1].

Media Class	Mandatory Formats	Optional Formats
Image	JPEG	PNG, GIF, TIFF
Audio	LPCM	AC3, AAC, MP3,WMA9, ATRAC3plus
Video	MPEG2	MPEG1, MPEG4, VC1, MPV1

2.1.3.1 - Networking and Connectivity

There are many advantages in using IP in the digital home:

- IP allows applications running over different media to communicate transparently.
- IP can connect every device in the home to the Internet.
- IP connectivity is inexpensive.

2.1.3.2 - Media Transport

Supporting HTTP 1.1 is a mandatory requirement to DLNA devices that send and receive media content to and from the home network. As optional media transport protocol, RTP is available.

2.1.3.3 - Media Management

UPnP AV and Printer technology are the basis from DLNA media management. Content Directory, Connection Manage, AV Transport and Rendering Control are some services that this technology provides. In this way, media management enables devices and application to identify, manage and distribute content across network.

2.1.3.4 - Certification

To assure DLNA Certified Logo vendor, products needs to successfully complete certification testing. This logo guarantees that a device is DLNA complaint and ensures interoperability between other devices with this certification.

2.1.4 - TR-069

TR-069 is a DSL Forum specification for remote management of end user devices, entitled *Customer-premises Equipment Wide Area Network Management Protocol* [5]. An ACS (Auto-Configuration Server) is the broadband network component responsible for auto-configuration of the CPE (Customer-premises Equipment). CPE WAN Management Protocol defines a mechanism that enables secure auto-configuration of a CPE. This mechanism may include vendor specific provisioning parameters.

By using this protocol, network operators save time and money. Most end users are not capable to configure set top boxes, modems, routers so network operators provide and manage these CPEs.

CPE WAN Management Protocol provides[x]:

- Software/firmware image management
- Status and performance monitoring
- Diagnostics

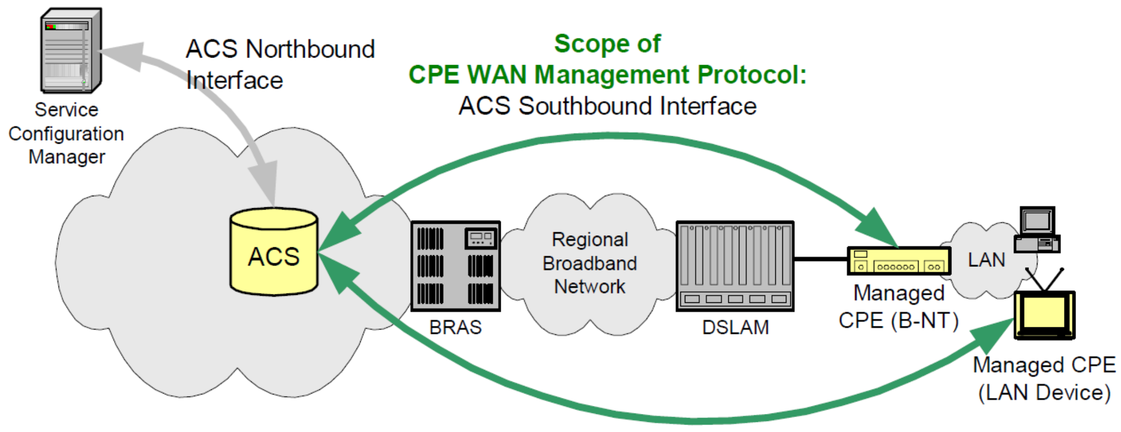


Figure 2.4- Positioning in the Auto-Configuration Architecture, image retrieved from [5].

TR-069 specifies several conditions that need to be met in order to determine who establishes and/or finishes the session, ACS or CPE.

The CPE WAN Management Protocol makes use of standard protocols but integrates several components that are unique to this protocol.

Table 2.3 Protocol layer summary, based on [5]

Layer	Description
CPE/ACS Application	This application is locally defined as it is not specified as part of the protocol.
RPC Methods	Specific CPE WAN Management Protocol methods.
SOAP	Standard XML-based syntax. CPE and ACS exchange SOAP messages.
HTTP	Standard HTTP 1.1 (CPE acts as the HTTP client and the ACS as the HTTP server).
SSL/TLS	Standard Internet transport layer security protocols.
TCP/IP	Standard TCP/IP.

2.1.5 - TR-111

TR-111 extends the CPE WAN Management Protocol defined in TR-069 to allow remote management of devices that are connected via a LAN through an Internet Gateway [6]. It specifies two mechanisms in order to achieve that.

The first mechanism is “Device-Gateway Association” [6] where both the gateway and the device are remotely managed (using TR-069) by the same ACS. It depends on the device’s use of DHCP so a gateway should inspect all DHCP requests and determine if there is the device’s identity on the request. By this mechanism, an ACS is able to manage a device to identify the associated gateway.

The second is “Connection Request via NAT(Network Address Translation) Gateway” [6] in order to enable an ACS to manage (using TR-069) a device operating behind a NAT gateway. In this situation, the Connection Request that is defined in the TR-069 specification cannot be used. This mechanism allows an ACS to issue the equivalent request but there are some requirements. For example, the CPE must know the public address and port associated with the open NAT binding and sends this information to the ACS. For this purpose, it defines the use of the STUN(Simple traversal of UDP over NATs) mechanism which requires a new UDP based Connection request.

2.1.6 - TR-140

TR-140 defines the TR-069 data model for provisioning auto configuration by an ACS to a CPE device that maintains a storage service such a NAS device.

“Remote Storage and Back-up Service” [7] is a service that a service provider can offer by using this data model. This backup operation can be initiated automatically if the user schedules it and the user’s content is stored remotely on a network server. SFTP and HTTPS are standard file transfer used in this procedure.

Another TR-140 function is “Remote access of Storage Service Content from a Remote Location” [7] which can be very useful for a user outside home to set up his TV recordings on the device’s web interface, for instance. Typically the storage service device is sitting behind a NAT gateway. To overcome this problem TR-111 provides some solutions.

2.1.7 - TR-135

TR-135, *Data Model for a TR-069 Enabled STB* [8] , as the name says, it is a data model for remote management of Digital Television functionality on Set Top Boxes devices.

The TV Operator Service Platform manages the access to the network and PVR content. The ACS monitors the STB configuration, operation and performance of STB. The goals of this specification [8] ,are the following:

- Enable configuration, by the ACS, of those objects and parameters that are not TV Operator Service Platform responsibility.
- Enable status monitoring and checking of specific parameters of an STB.
- Support various types of STBs.
- Estimates of QoS (quality of Service) and QoE(Quality of Experience) to performance monitoring.

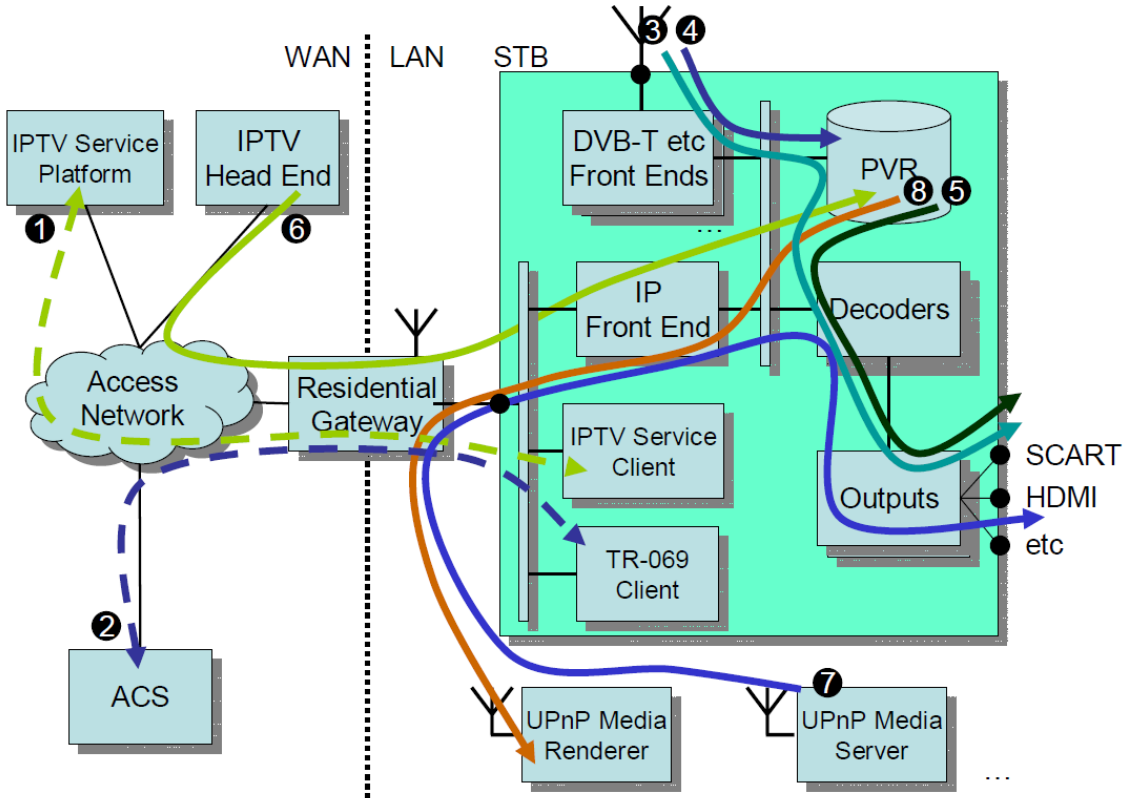


Figure 2.5- TR-135 STB context, image retrieved from [8]

2.1.8 - Home Gateway Initiative

2.1.8.1 - Overview

The Home Gateway Initiative (HGI) is composed by several Telecommunication providers and Technology makers with a non-profit purpose. This organization was created to define specifications and guidelines for broadband Home Gateways.

HGI assumes that broadband services will demand new requirements for the Home Gateway [9], such as, the need to manage the devices beyond the Home Gateway. This enables the right device to connect to the right service platform with the right service class, in order to create an “integrated home environment” [9].

HGI establishes some use cases like the remote access to some home devices and services, which they assume as “one of the latest consumer market trends” [9]. One use case is the remote access to PVR service to initiate a record remotely. Other use case has the specific requirement of upload/download media content to outside home network. This means that Home Gateway guidelines need to cover security concerns when accessing to the home network resources.

2.1.8.2 - Home Gateway Requirements

Ease of use and simplicity - HG and home network associated devices and services must be easy to install and configure which means plug and play and zero-touch configuration capabilities.

Security - Firewalls, personal monitoring and access control are the main security functions that must be provided by the Home Gateway.

Remotely managed gateway / home network - The Home Gateway needs to be a managed device in order to enable remote operations and provisioning of managed services at home.

Performance Monitoring and Diagnosis - HGI approach is based on acts after a problem is reported by the customer, however, after this report a remote diagnosis must be performed. Monitoring service performance is also a requirement.

The Home Gateway Initiative guidelines define three approaches for remote access of services and devices [10].

2.1.8.3 - IMS based Remote Access

HGI presents an IMS (IP Multimedia Subsystem) approach to enable remote access to home devices, including UPnP and non-UPnP devices, as shown in the next figure.

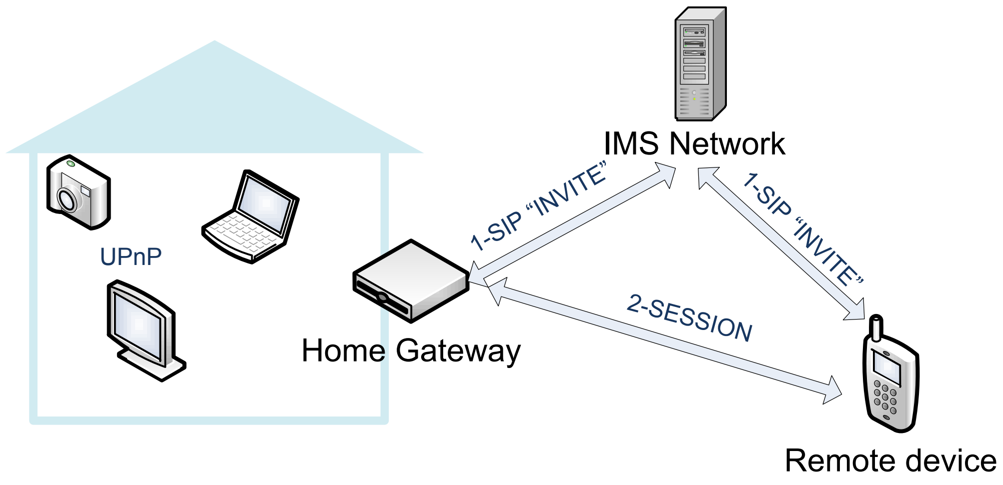


Figure 2.6 – IMS high level Remote Access architecture, based on [10].

Generic IP devices represent the device types of interest to the IMS remote access approach. In this way, UPnP devices can perform the signaling between the HG and the home device. On the other side, SIP devices need B2BUA (Back to Back User Agent) functionality in the IMS enabled Home Gateway. The principle of this approach is that IMS control plane is the responsible to route the "INVITE" message to the HG. It is also used to set up the session between the remote device and the Home Gateway on the IMS media plane.

The Administrator can be either the end-user or the service provider which means that, there is a partitioning into end-user devices and operator managed services or devices.

2.1.8.4 - Web based Remote Access

Web-based approach is defined for non-IMS based networks and it assumes that there is a web server within home network. It shall support SSL/https for remote access. The Access Control List (ACL) specifies the rights of each user in accessing specific end-devices.

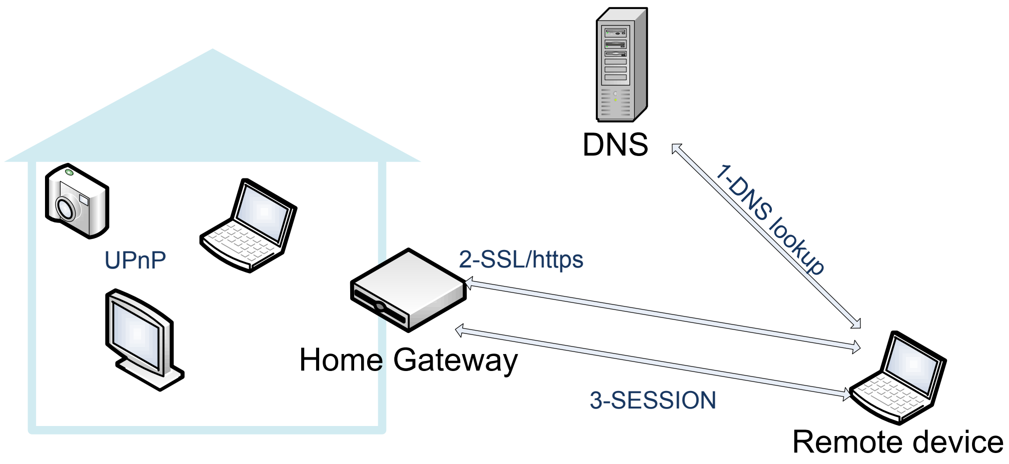


Figure 2.7 - Web high level Remote Access architecture, based on [10].

A Dynamic DNS service is required to find the HG with a unique name instead of an IP address. In this way, the remote user can access the Home Gateway web server and log in, then the secure session is initiated using SSL/https and the user authorized devices become visible. The source IP address of the remote media access is also inspected.

2.1.8.5 - Operator controlled Remote Access

This approach relies on the Broadband Service Provider (BSP) control of the entire remote access process. The requirements from this solution are the capability of the BSP server opening pinholes through the firewall for secure access to a device. The configuration of the Home Gateway is done from the Remote Management System (RMS) which is conceptually different than the previous approaches.

2.1.8.6 - Management Architecture

The Home Gateway Initiative management approach is fully based on TR-069 which defines the CPE Remote Management Protocol (CWMP) and the Auto Configuration Server (ACS) to manage TR-069 enabled devices in the Home Network (HN). This means that devices are managed from the ACS via the CWMP protocol.

The Auto Configuration Server (ACS) implements device configuration, software updates, device diagnostics, remote operations and reboot requests as defined in TR-069 [5]. The Home Gateway is controlled by a single Remote Management Service (RMS) at any one time. The RMS is the management entity that includes the ACS and other additional management functionalities (resource and device inventory, diagnostics and troubleshooting, event notification and alarm management).

There is the possibility of the user has an interface to tune configuration of the Home Gateway. This interface may have one part based in the HG itself and the other located in the service provider's portal (OSS/BSS).

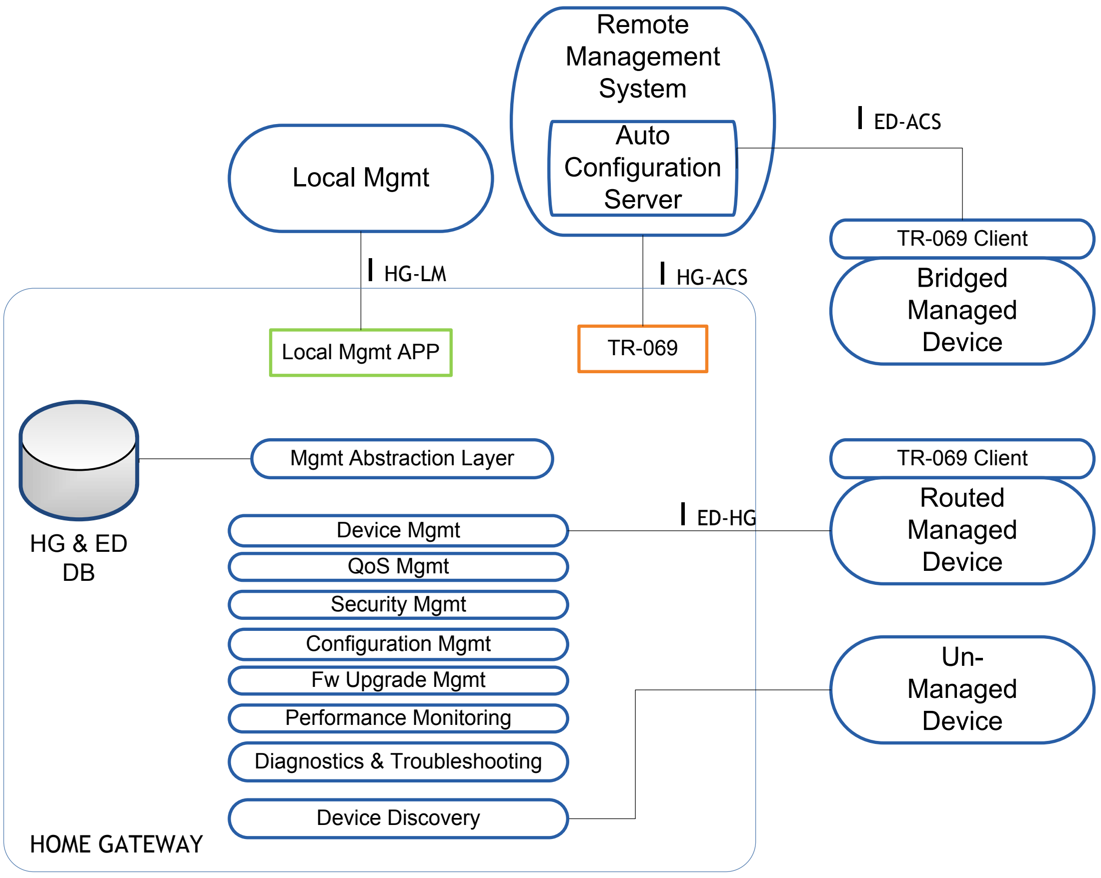


Figure 2.8 - Management architecture, based on [9].

The main Home Gateway interfaces are I_{HG-LM} for local management (HTML interface) and I_{HG-ACS} for remote management (CWMP interface). The end-device interfaces are I_{ED-ACS} for remote management of bridged end-devices (CWMP interface) and I_{ED-HG} is a communication interface HG/end-device (DHCP or UPnP interface).

The Home Gateway needs to enable some remote management of simple end devices that do not support TR-069 but UPnP or even only DHCP. SIP devices in the Home Network should also be discovered.

ARP cache, DHCP repository, UPnP Control Point cache and SIP registrar cache provide the ID from connected end devices to HG. An UPnP device can be an embedded device which means that is more like a logical entity than a physical device. Either root or embedded UPnP devices shall be discovered to obtain a trusted view of Home Network. Every UPnP device can be identified by UUID. The ID information fills a Managed Devices Data Base that is accessed by the remote management server.

2.1.9 - Conclusion

UPnP and DLNA guidelines together with IP provide a wide range of features and possibilities to explore. For example, a PC or an advanced Set Top Box may stream media content to a television in the living room through an Ethernet cable to an 802.11 Access Point and then wirelessly to the television that is in the kitchen. With IP, the media server and the television are unaware that the media content travels over two separate physical media.

CWMP provides remote management of end user devices which can be done at low cost, reducing the calls to helpdesk and the number of technicians needed.

Finally, Home Gateway Initiative defines guidelines for integration of UPnP and SIP devices inside Home Network with a remote management infrastructure based on CPE WAN Management Protocol.

2.2 - Set Top Box and Media Centers

2.2.1 - Overview

Usually embedded appliances are perceived as a “box” that cannot be modified beyond the manufacturers distribution, as described in [11]. This is the opposite of a PC, especially if it is running a Linux distribution. Recently, many embedded systems manufacturers adopted the Linux as embedded operating system and changed to most standard hardware. The first and most important reason for this change is the cost reduction which leads to more “open” embedded devices.

Even with this changes, developing software solutions for embedded devices are still not attractive for some developers. However a set top box might have some advantages when comparing to a powerful PC:

- Lower power consumption which is a very important feature nowadays.
- Rarely requires active cooling so it is much more quiet system.
- Allows less experimented users to have access to media content easily.
- It is cheaper.

These reasons justify why may be it is not so convenient to have a PC always on when it could be a set top box.

A set top box (STB) can be understood as a small computer that enables a television set to become a user interface to digital media content [12]. The term “set top box” is referred to a box that is set on top of a TV. Set top box can also receives encoded digital signals from the signal source (network operator) and decodes those signals in order to be displayed on a TV set.

It can be viewed as a small computer because it usually has only a CPU, a DRAM and a flash memory. The STB accepts commands from the user, commonly via a remote control, and transmits its commands back through the return path to the network operator. Nowadays set top boxes have return path capability for two way communication.

There are lots of set top box features, such as receive and display TV signals, Internet access, interactive program guides, streaming, VoIP or PVR.

2.2.2 - Hardware Architecture

First the set top box starts tuning to one of many input channels. It is the way of selecting the appropriate broadcast TV information. There are satellite, cable and digital terrestrial applications.

Quadrature Phase Shift Key (QPSK), Quadrature Amplitude Modulation (QAM) and Orthogonal Frequency Division Multiplexing (OFDM) are used for satellite, cable and terrestrial applications respectively. The demodulator processes the information in the selected RF channel and produces an MPEG-2 Transport Stream (TS) containing the audio, video and all information relative to TV Program [13].

Usually, there is a modem responsible to send and receives interactive data. Cable set top boxes have a cable modem while conventional telecommunication modems are used in satellite and terrestrial.

When there is some kind of restriction for accessing paid services it means the MPEG-2 TS may be encrypted. The MPEG demultiplexer selects and decrypts the compressed audio and video using decryption keys supplied by the Conditional Access Subsystem (CASS). A CA module embedded inside a set top box means that it can no longer be considered to have an open architecture [13].

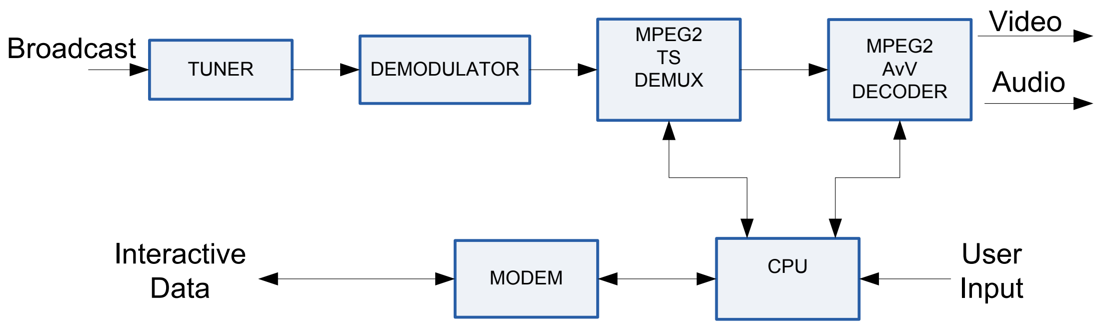


Figure 2.9 – STB architecture overview, based on [12].

The Central Processing Unit (CPU) is responsible for controlling the whole process and performs specific data manipulation function.

There are different processors families available on set top boxes market [12]. The most common are PowerPC, ARM and MIPS processors. However there are some set top boxes that have x86 family processors, especially if we are talking about Internet and media center set top boxes. Of course, there are some software applications that, at the moment, are not compatible with some processors.

Digital set top boxes tuners are under a Digital Video Broadcast (DVB) open standards for digital television like DVB-S, DVB-C and DVB-T for satellite, cable and terrestrial transmission, respectively [12]. There are also IP-TV set top boxes that receive the TV stream over Internet Protocol. In some areas, where IP-TV cable service cannot be available is commonly used IP-TV over satellite and the set top boxes need to integrate both IP and DVB requirements.

2.2.3 - Operating Systems

A set top box, like any computer hardware, needs an operating system to work. It is responsible for managing the resources in a STB. As a device with a few hardware resources, a set top box needs an operating system reliable and robust in order to avoid undesirable reboots. It needs to be multi-tasking and multi-threading to process incoming digital MPEG streams and check security messages.

As an embedded device, set top box usually has a Real Time Operating System (RTOS) on top of hardware abstraction layer [11]. STB OS is arranged in layers and the kernel layer is its core layer, which is stored in ROM. When the STB is powered on, it will be the first to be loaded and remains in memory until the STB is switched off again. The kernel deals with memory resources, real time applications and high speed data transmission. The STB also needs drivers for each hardware component. In this case a driver is a program that translates commands from the TV viewer to a recognizable format for hardware devices.

A set top box operating system needs to incorporate an Application Programming Interface (API) which is used by developers to create high level applications for a specific API, as described in [12].

2.2.3.1 - Microsoft Windows Embedded CE

Windows Embedded CE [14] is an operating system from Microsoft and it is aimed to be used on minimalistic computers and embedded systems. It runs on x86, MIPS, ARM and SH4 processors and it is a real time operating system. A Windows Embedded CE kernel may run in under a megabyte of memory.

Unlike other Microsoft operating systems, large parts of it are offered in source code form. This feature allows manufacturers to adapt Windows Embedded to their hardware specifications. However, a number of core components which do not need adaptation to specific hardware are not available in source code form. This operating System is not free.

2.2.3.2 - Microsoft Windows XP Embedded and Embedded Standard 2009

This version of Windows contains the full feature set of Windows XP Professional but has restrictions on licensing. An original equipment manufacturer is free to choose the components needed. In opposition with Windows Embedded CE, XP Embedded [14] provides the full Windows API and only runs on x86 processors. Microsoft released a new version called Windows Embedded Standard 2009 which provides extra multimedia and development features. Windows Embedded Standard 2009 includes Silverlight, .NET Framework 3.5, Internet Explorer 7, Windows Media Player 11 and Network Access Protection. Both versions are not free and the source code is not publicly available.

2.2.3.3 - LINUX

Linux [15] is the most common Unix-like computer operating system. There are many Linux distributions which are developed under the terms of the GNU GPL and other free licenses. The system is free and open source, i.e., all the source codes are public. In this way, developers can find and eliminate programming bugs, as well as being able to integrate new functions easily. This feature enables quickly integration of drivers for new adapters.

Linux distributions are installed in a wide range of computer hardware, such as supercomputers, servers, embedded devices or mobile phones. The continuous development and support for advanced hardware, combined with enhanced multimedia capabilities allowed Linux to be used in gateway devices and set top boxes.

There are different ways of controlling a Linux-based system. It can be done through controls on the device itself, text based command line interface (CLI) and graphical user interface (GUI). The last is usually the default for desktop, however there are some distributions having just the CLI as interface.

Nowadays, Linux is probably the most common set top box operating system.

2.2.3.4 - MAC OS-X

Mac OS X [16] is a Unix-like operating system. Its core is a POSIX compliant operating system built on top of the XNU kernel. A free and open source operating system using this set of software was released under the name of Darwin. Aqua Interface and the Finder among other components layered on top of Darwin, complete the GUI based operating system that is Mac OS X. Apple TV and iPhone have specific versions of MAC OS X. It runs on x86, PowerPC and ARM processors, the last are only used for iPhone.

2.2.3.5 - Android

Android [17] is a software platform consisting in an operating system, middleware and key applications. It is based on the Linux kernel and developed by Google and the Open Handset Alliance. It is available as open source since October 2008. Android SDK allows developers to create applications written using the Java programming language and compiled into Dalvik bytecodes. Dalvik is not a standard Java Virtual Machine and it is special designed for mobile device use.

Android is a modern software manager. It was conceived for a touch screen interaction with a very friendly user interface.

2.2.4 - Middleware

Middleware [18] is a layer of software that acts as a conversion or translation layer and runs on top of set top box operating systems. It mediates data exchange between two or more separate software applications and/or manufacturers, which mean supporting, open and proprietary standards.

Middleware usually includes a virtual machine (application manager), the interactive engine, the libraries and databases [12]. It becomes particularly useful when there are a number of different programs, platforms and software in use. The set top box middleware is often associated to the concept of Resident Application. In this case, it is a program or programs that are built into the memory of the set top box. Often the network operator automatically updates the Middleware via the data stream received by the set top box.

Thus set top applications don't need to be concerned about underlying network protocol which reduces complex of content development. Applications can be created to take advantage of a common API. Early generation of set top boxes had no API but as costs were reducing and processing power has increased, set top boxes have included Application Programming Interfaces. Electronic Programming Guide (EPG) for navigating through channels broadcasting is just an example of an application that requires an API.

Each network provider has its own "closed" middleware software which sometimes leads to lack of interaction between different network operator's solutions.

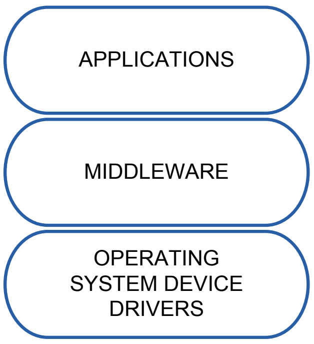


Figure 2.10 – Software layers of a STB

Multimedia Home Platform (DVB-MHP) is an open solution but it was designed only for interactive digital television (digital terrestrial). It integrates Java based applications on a TV set and it is a model used by several European broadcasters. It is independent of the underlying environment and is based on the use of Java virtual machine. Cable industry from U.S. has decided to develop its own middleware largely based on MHP that is OCAP (OpenCable Applications Platform).

MythTV

MythTV [19] is a GPL licensed suite of programs that turns a computer with the necessary hardware into a digital multimedia home entertainment system, frequently known as Home Theater Personal Computer (HTPC). MythTV is GPL licensed which means that is free and open source. It works on Linux , Mac OS X and recently on Windows.

MythTV structure is divided in frontend and backend [19]. These applications can be both installed in one regular desktop or physically separated.

The frontend is the client application GUI from MythTV with several themes that change the look of the interface. It is also responsible for rendering the video that is displayed which can be non trivial if we are talking about high definition content. Each frontend is able to communicate to one or more backend servers. This communication determines which recorded shows are available or which TV tuners are available. It also provides access to system status and there are several pug-ins available to install like MythMusic or MythVideo.

MythTV backend is the server application which means managing the MySQL database and the capture cards. It can be a powerful PC ready to host recording files. It streams the data to be displayed by any frontend and performs routine maintenance on any files generated during the recording process.

MythTV implements an UPnP Server (“UPnP AV Media Server” device) in order to allow an UPnP client to automatically see and renderer all content available in the MythTV system.

XBMC

XBMC [20] was formerly named as XBOX Media Center but since 2003 it is named only as XBMC because of its widespread use over several platforms. It is written in the C++ program language and runs on Windows, Linux and Mac OS X. It is distributed under GNU GPL license.

XBMC has many multimedia features due to its Python Scripts Engine and WindowXML application framework. This allows users to develop new functionalities to XBMC.

XBMC uses MPlayer to play video/audio files. MPlayer get most of its codec support from the FFmpeg codec-suit.

As database engine, XBMC utilizes SQLite to store all music, video and program databases. It has a built in UPnP client ("UPnP AV MediaServer Control Point" device) which is able to detect any UPnP server on the local network. Of course that is only possible as long as that UPnP server is not behind a firewall/router or the necessary ports are opened for UPnP. XBMC also has a built in UPnP server, however other UPnP clients must support receiving broadcasts because XBMC doesn't support multicast. XBMC built in UPnP server and UPnP client are compatible with DLNA CERTIFIED devices.

There are some projects that are based on XBMC. It happens because XBMC is an application framework platform for developers to base their own media center software on. Boxee is one of the XBMC based projects.

MMS

My Media System [21] is a free, open source and runs on Linux. The target platforms of this application are set top boxes and HTPCs. It does not integrate a TV watching and record feature but it can be integrated. It lets other applications, such as MPlayer, VDR or Xine run together and integrates them into one system.

FREEVO

Freevo [22] is mainly a personal video recorder application for Linux and Mac OS X. It is free and open source distributed under the GNU GPL license.

Freevo is written in Python and unlike other media center projects, reuses existing software as mplayer, xine or vlc. Its API is designed to split the logic into several independent modules that can be used by other projects.

Neuros OSD

Neuros OSD [23] has a media center application specially conceived to run on Neuros OSD set top box. As any media center it archives, organizes and plays media content.

Neuros OSD comes with a complete software stack of Linux, Qt as the user interface and VLC as media framework. In this way, developers are able to keep improving features for this open source media center.

Boxee

Boxee is not conventional media center software. It is a social network [24] where the users view and recommend media content with their friends. In order to be part of this social network, users must have a Boxee account. It is developed by a commercial start up company with natural profitable ambitions.

Boxee is a fork from the XBMC project [20] which means that is distributed under the GNU General Public License. However its social networking library libboxee is closed source as it deals with the Boxee's online backend server. This backend server handles with user account information and social communication network between users.

Boxee is currently under development and there is a Alpha release that runs on Linux and Mac OS X for computers with x86 processors. As well as on XBMC project there is a Python plug-in system for new multimedia improvements.

Boxee announced plans for a set top box releasing in 2009 and also plan to license their Boxee media center in order to be used by other companies in their hardware.

2.2.5 - STB solutions

2.2.5.1 - Overview

Different solutions can be implemented depending on the STB used to develop the prototype. There are many different STBs brands and models however it is not possible to develop this project in all of them. Usually STBs are closed devices where is not possible to change its software applications. Processing power and memory are some of the constraints as well as the operating system.

2.2.5.2 - Dreambox

The Dreambox STB [25] are Linux supported DVB satellite, terrestrial and cable digital television decoders. Dreambox STBs support own conditional access system which enables receiving and storing encrypted content. Through the built in Ethernet interface networked computers are able to access the internal hard disks on some Dreambox models and the opposite is also possible.

The DM 600 PVR is built around IBM STBxx25xx integrated controller and it allows adding an HDD drive. The tuner module is plug_and_play so it is to change between satellite, terrestrial or cable versions. It features 32MB flash and 96 MB of RAM(64 MB user-accessible).

The DM 800 PVR HD is very similar to DM 600 but it supports high definition. It features DVI to HDMI cable and USB 2.0. It is based on a 300 MHz MIPS processor, 64 MB of Flash and 256 MB of RAM memory.

The Linux distribution used by Dreambox is provided by Tuxbox project and is mostly available under the GNU GPL. It uses standard Linux APIs like *Linux DVB API* and *Linux Infrared Remote Control*. These configurations allowed developers to modify its functions in the form of so called images such as *PLi* and *Gemini*.

As conventional PowerPC and MIPS set top boxes, Dreambox models may not be able to support a media center installation. This means that it would be necessary to develop the project from the existing Dreambox middleware. Dreambox models do not have a good processing power or great graphical solutions. As conventional set top box it is not so completely open environment as Neuros.

2.2.5.3 - Scientific Atlanta

Some Scientific Atlanta IP-TV set top boxes [26] are compatible with Linux and Windows Embedded CE operating systems. It is usually a pay TV companies set top box supplier. It is a set top box aimed for end user utilization.

This STB has proprietary software and is not conceived to developers as it is not going to be considered in this project.

2.2.5.4 - Buid a set top ox

Another option is to build a Set top Box based in a home theatre PC. In order to build this solution some hardware requirements are necessary like the following list:

- TV capture card
- Case
- Power supply
- CPU
- Motherboard
- Video output card
- Memory
- Hard disk drive

- Infrared receiver

This solution has the advantage of being a completely free developer choice, however the project is aimed to use an off the shelf STB. On the other hand, there are two possibilities to develop the project using a Home Theater PC:

1. Installing Linux distribution so it could be the same as working in a desktop.

2. Installing Android and develop UPnP capabilities. In this case would be necessary to compile the UPnP code in Android which could represent more time reserved for this task. Using Android could be more appropriated to develop a friendly user interface as it is conceived for touch screens.

2.2.5.5 - Neuros Link and OSD

Neuros set top boxes can be realized as open solutions [23]. Neuros Link and Neuros OSD are two different models but both have a regular Linux distribution.

Neuros LINK [23] is a set top box that enables user to develop several media applications. It has an AMD Athlon LE-1600 processor and a modified version of Ubuntu Intreprid and it is a media player focused on internet interaction. Neuros Link has 1GB DDRAM memory which is very powerful for a STB. However, in order to support a MythTV frontend application it is necessary at least 256MB of RAM memory. It has a software stack composed by Adobe Flash, MPlayer, VLC and Xine. It has firewire which allows controlling other set top boxes and receives broadcast for view and changing channels.

Neuros OSD [23] like a PC is open and expandable. It is aimed for multimedia solutions developers. It has an ARM processor using Texas Instruments DaVinci 6446 chipset. Neuros OSD has its own media center installation.

Neuros Link and OSD models don't have any HDD drive but they support one internal ATA-6 3.5 HDD drive.

Both Neuros Link and Neuros OSD provide a good solution to develop the project because of the processing power and the memory available in both models as it is like using a common desktop.

Neuros is a pure development solution although it is not possible to use XBMC (or other media center, see 2.2.6) as framework on Neuros OSD because it has an ARM processor and XBMC at the moment is not supported by ARM processors. Neuros Link has the advantage of having an AMD Athlon LE-1600 so it supports XBMC or MythTV frontend application.

However, Neuros OSD does not need a media center installation because it already has media center features. It is open source and it makes use of VLC and Qt applications.

Neuros STBs don't have any kind of tuner so it is necessary to connect a TV tuner for watching and record TV. Due to physical restrictions this project needs to be developed using a DVB-S2 TV tuner. Technisat Skystar-2 is a conventional DVB-S2 TV tuner that supports LINUX. Neuros OSD do not have available PCI slots to connect a TV Tuner while Neuros Link has. This can be a great advantage to Neuros Link choice.

2.2.6 - Conclusion

Table 2.4 STB models comparison

	Neuros Link	Neuros OSD	DM 600	DM800
Processor	AMD Athlon LE-1660	ARM 256.5 MHz	250 MHz PowerPC	300 MHz MIPS
Memory	1GB DDR2 SDRAM	256 MB RAM	32 MB Flash, 96 MB Ram total	64MB Flash 256 MB RAM
Storage devices	X	X	X	X
Ports	USB 2.0;IEE 1394 (firewire); HDMI; PS/2; eSATA	HDMI; USB; RJ-45 Ethernet;	RS232	**RS232, DVI,USB**
Network	802.11 g/b WiFi; Gigabit Ethernet	RJ-45 Ethernet	10/100Mbit Ethernet	,10/100Mbit Ethernet
OS	Linux	**Linux**	Linux	Linux
TUNER	X	X	DVB-S, DVB-C, DVB-T	DVB-S, DVB-C, DVB-T

This project intends to use an off the shelf STB but as described before STBs are perceived as "closed" devices which can not be modified or extended. Neuros Link is not a conventional STB and it is similar to a regular desktop with a Linux distribution installed. This is a developer friendly configuration that allied with a DVB-S2 TV tuner like Skystar 2 provides the requirements to simulate a STB "behavior" and develop properly a *Virtual PVR* prototype.

Table 2.5 Media Centers comparison, based in [27]

Feature	MythTV	Freevo	XBMC	Boxee	MMS	Neuros OSD
Project Started	2002	2002	2002	2008	2002	2003
Operative System	Linux	Windows Linux Mac	Windows Linux Mac	Linux Mac	Linux	
Video player/handling	FFmpeg, mplayer, vlc, xine	xine, vlc, mplayer	mplayer, FFmpeg	Mplayer FFmpeg	CXFE, xine, mplayer	vlc
Audio player/handling	FFmpeg, mplayer, vlc, xine	xine, mplayer	PAPlayer, FFmpeg	PAPlayer, FFmpeg	Alsaplayer xine, gstreamer	vlc, xmms2
TV Watch/Recording	✓	✓	*	–	*	✓
Multiple TV Tuner Cards	✓	✓	–	–	–	–
Full HD support (1080p)	✓	✓	✓	✓	✓	*
Video Encoding	MPEG4, MPEG2, RTjpeg,	MPEG4, MPEG2, RTjpeg	–	–	–	ASF, MPEG-4
Schedule timed recordings	✓	✓	*	–	–	✓
TV Guide navigation	✓	✓	*	–	✓	–
Commercials Removal	✓	✓	*	–	–	–
Remote Control support	✓	✓	✓	✓	✓	✓
VoIP	✓	–	–	–	–	–
Web Browser	✓	–	✓	–	–	*
Email	*	–	✓	–	–	–
Plug-in support	✓	✓	✓	✓	✓	*
UPnP support	✓	✓	✓	✓	–	✓

✓	Available
–	Not Available
*	Under development

2.3 - Development tools and libraries

2.3.1 - Overview

There existing free and open source UPnP SDKs allow the development of UPnP Media Servers, Control Points and Media Renderers. The same happens with streaming solutions, EPG grabbers and DVB initial setup and configurations tools. As this project intends to put together standard technologies and extend them, these development tools and libraries will be useful for the project. Below we describe some of the existing modules.

2.3.2 - UPnP libraries

Platinum

Platinum UPnP is a development kit written in C++ based on UPnP AV v1 specifications. This framework is free and open source software which makes it easy to create Control Points, Media Servers and Media Renderers. Platinum provides UPnP capabilities to some well known media applications, such as, XBMC and Boxee.

Portable SDK for UPnP Devices (libupnp 1.6.6)

This library is written in C and it is adopted by MediaTomb and Geexbox in order to provide UPnP capabilities. Portable SDK for UPnP Devices is a fork from libupnp project which was discontinued. As it is not written in object oriented programming language it can offer some disadvantages when compared with Platinum.

2.3.3 - Media tools

2.3.3.1 - VLC

VLC media player is part of VideoLAN streaming solution which is free and open source software. VLC supports various streaming protocols and it acts as server and as client to stream and receive network streams. VLC supports DVB capture devices as well as, advanced stream output options, as transcoding and various kinds of output supports. VideoLan Manager

(VLM) is one of the most important features of VLC, in a way that it is a small media manager which is able to control multiple streams with only “one instance of VLC” [28]. It is a way of configuring VLC from text files instead of complex command lines. VLM can be only controlled by the telnet interface or http interface of VLC. It allows multiple streaming and video on demand (VoD).

VLM is composed by Media objects which have a list of inputs (usually video and audio streams), an output and some options. There are two types of Media: broadcast and vod. A broadcast media is commonly like a TV program where the client as no control over this media. The administrator launches, stop, pause and repeat the stream. A vod media is available and launched only when the client asks for it [28]:

```
% vlc --ttl 12 -vvv --color -I telnet --telnet-password videolan --
rtsp-host 0.0.0.0:5554 --vlm-conf vod.conf
```

- *12* is the value of the Time To Live of the IP packets
- *telnet* launches the VLC´s telnet interface
- *videolan* is only a password to connect to telenet interface.
- *0.0.0.0* is the rtsp host address.
- *5554* is the port on which it streams.
- *vod.conf* is the text file which contains the media object.

A simple VLM file can be like:

```
new Test vod enabled
setup Test input my_video.mpg
```

In order to access the stream, the client just needs to launch vlc and specify the *vod* session he wants to stream. *Server* is the IP address of the streaming server:

```
% vlc rtsp://server:5554/Test
```

VLM is capable of perform scheduled actions. *Schedule* is a script that is launched when the script date is reached.

VLC uses its own multiplexers and demultiplexers but it uses libavcodec library from the FFmpeg project. It also supports codecs that are not included in FFmpeg.

2.3.3.2 - MPlayer

MPlayer [29] is a media player that runs on many systems and supports a wide range of supported output drivers. It is free and open source software under the GNU GPL license.

MPlayer is a command line application that has many optional GUIs for each of its supported operating systems. It is available for LINUX, Microsoft Windows and Mac OS X. The most common GUIs for Linux operating systems is gMPlayer which is written in GTK+.

2.3.3.3 - DVB tools

The dvb-apps package contains Linux DVB API applications and utilities towards the operation of a DVB device, regardless it is of hardware decoding or software decoding class [30]. The *scan* utility scans for channels and creates a channels.conf file which can be used by the *zap* utility in order to tune a given channel.

2.3.3.4 - TV grab DVB

TV Grab is a simple application that decodes EPG TV listings from DVB signal and outputs them in a XMLTV format file. It reads a channels.conf file available in its current directory to match the channel ID and the display name of the channel. This channel.conf needs to be provided from a DVB channels finder. After tuning a DVB channel using a zap utility, TV Grab catches the EPG from wide range of channel frequencies [31].

2.3.3.5 - XMLTV format

In order to provide an EPG service to set up the recordings it is mandatory to catch this information from the TV signal. XMLTV format is widely used for describing TV listings.

It is composed by multiple <channel> tags which contains an ID, title of the tv programmes, start time, stop time, language and optionally a description attribute [32]. However, there are some differences between each EPG decoding application.

2.3.3.6 - FFmpeg

FFmpeg [33] is a command line tool that can record, convert and stream digital audio and video in several formats. It is composed by a collection of free and open source software, like libraries as *libavcodec* and *libavformat*.The first is an audio/video codec library and the second is an audio/video container multiplexer and demultiplexer library. The command line interface (CLI) is supposed to be intuitive because the user only has to specify the desired target bitrate. It is able to convert from any sample rate to any other and resize de video, for instance. FFmpeg is hosted on the MPlayer project server as they are connected projects.

2.3.3.7 - VDR

Video Disk Recorder [34] is an open source application for Linux. It provides TV features by recording and replay TV programming using the computer’s hard drive. It also operates as a media player.

2.3.4 - Conclusion

There are development tools and libraries that can be very useful to this project. The study of the existing tools allows a better understanding of how some tasks can be performed and which tools can be changed or extended. The *Platinum* library is going to be used as Framework to develop the Media Servers and the Control Point of the system. The library is written in C++ and has good documentation which are major reasons to choose *Platinum*. VLC can be a media streaming server and a media client. It is free and widely used in multimedia projects. It has the VLM which is a great tool to manage the VLC instances. TV GRAB DVB is going to be used as EPG capture tool as it creates a TV listings file in the XMLTV format

Chapter 3

Virtual PVR Proposal

3.1 - Overview

In order to achieve the objectives of this project, described in the section 1.3, a prototype consisting of a STB will be implemented. However, the system is more than a software application running on a STB. It is necessary to create a Web interface that allows users to access the system. There are also software modules that need to be developed enabling the TV provider to remotely manage the system. The complete solution comprises UPnP devices and services, a web interface and CPE WAN Management Protocol modules. The solution is based on the Home Gateway Initiative guidelines, besides the fact that this project does not handle with SIP devices.

UPnP devices and services are the only devices to take into account inside the Home Network. As this project intends to have a zero-configuration, the end user only needs to worry about media contents. The overall UPnP AV architecture [3] forms the foundation for the UPnP devices interaction of the system. The general interaction between UPnP Control Points, Media Servers and Renderers is independent of a particular device type, media format or transfer protocol. A variety of devices supports it, such as TVs, VCRs, game consoles, stereo systems, MP3 players, camcorders, electronic picture frames and the PC. Multiple media formats (like, MPEG2, MPEG4, JPEG, WMA, bitmaps PAL, ATSC, NTSC, etc.) are allowed by the AV architecture [3], as well as transfer protocols (such as, HTTP GET,HTTP POST, RTP, RTSP, TCP/IP, etc.).

3.2 - Architecture

The system architecture is divided in two environments: Home Network and Telco Network. The goal of this project is to implement the first. The Home Network has UPnP devices and services that are announced and discovered automatically.

The system is scalable in a way that several UPnP Media Servers and Renderers can be connected at the same time. The end user buys a new external disk or a new TV that is UPnP/DLNA compatible. Then he just needs to connect it to the Home Network, in order to the new device take part on the system. The Control Point detects the new device, which can be a Media Server, Renderer or even another Control Point, and searches for all its services and capabilities. This information is send to the Home Network database (DB) where all UPnP devices and services data are stored. CPE is the CWMP client present in the Home Network and has full access to the data stored in the DB. CPE sends periodic CWMP messages to ACS which is the CWMP server of the system. These messages are used to fill and actualize the data present in the Telco DB. This process assures remote access to the devices and services available inside Home Network.

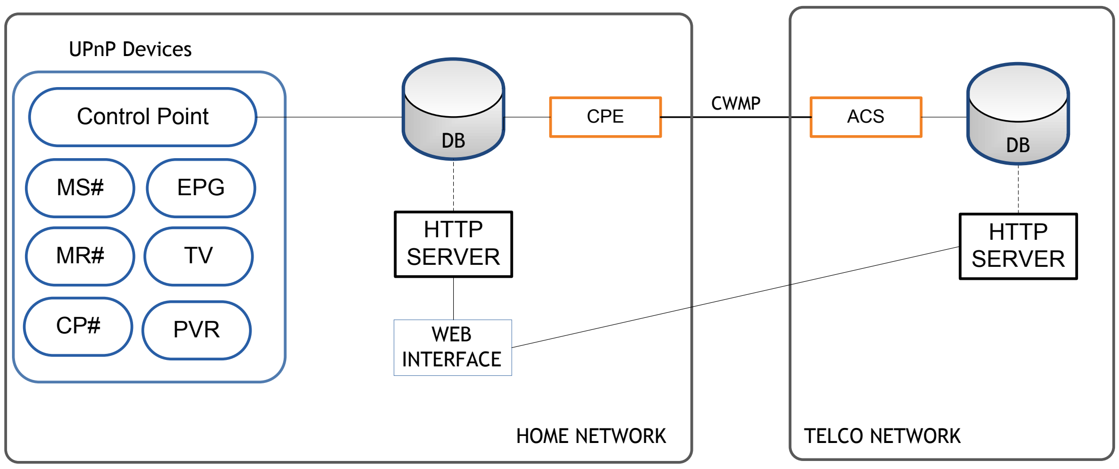


Figure 3.1- *Virtual PVR* integrated architecture.

The figure above illustrates the architecture of the *Virtual PVR* integrated solution. It is important to notice that MS#, MR# and CP# represent one or more Media Server, Media Renderer and Control Point. These devices are conventional UPnP devices that can take part on the system. The Control Point, EPG, TV and PVR are specific UPnP devices developed by this project. The end user accesses the system through a Web Interface that is provided by a HTTP Server that can be installed in the Telco Network or in the Home Network. The web interface is different whether the user is accessing the portal from a terminal inside or outside Home Network. Accessing the Web Interface outside Home Network means that the end user can see all the UPnP devices and services present in the HN but not actually access the media content. This happens due to security and bandwidth requirements. Besides these reasons, there are some services that can be provided directly from the Telco Network, like streaming and recording a TV channel, without accessing the STB. The project scope resides on the services offered by the devices present in the Home Network. Streaming media contents is only possible if the user is accessing the Web Interface from a terminal inside Home Network.

3.3 - UPnP Devices and Services

TV, EPG and PVR represent the UPnP AV Media Servers that this project intents to implement. These UPnP AV devices are logical entities running on the STB. An UPnP AV Media Server allows UPnP Control Point to browse content items available for the user to render [3]. Content Directory Service, Connection Manager Service and AV Transport Service are the main services that a Media Server contains, as defined in the UPnP guidelines [2].

The *Virtual PVR* system allows users to see live TV channels available on the STB through the Web Interface. Each channel is announced as a media file which means that it is necessary an UPnP AV Media Server (TV) to announce the TV channels. The TV application launches the streaming server that is going to broadcast the TV channels to the Home Network. Once the streaming server is launched, the TV initiates its UPnP session, announcing services and capabilities through the Home Network.

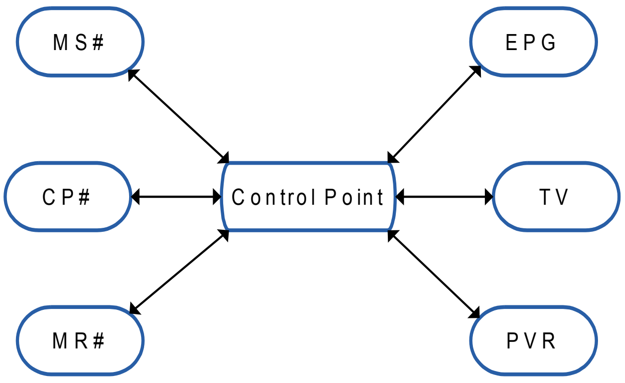


Figure 3.2- UPnP devices interaction

The recording service is performed by the EPG and the PVR devices. It is logical to separate the service in two Media Servers due to make the system more efficient. The EPG device is responsible to create a file for each TV channel, containing the TV listings description, like the name, start and stop time of a programme. There are some TV channels not supporting the Electronic Programme Guide. In this case, an empty file is created by the EPG device. The EPG files are announced to the Home Network and each TV programme represents a media item to be recorded. The PVR device receives the user order to record a specific TV programme. The request is processed and when the start time is reached the PVR device launches a media player. The TV channel stream is opened and stored in any available UPnP Media Server. The PVR announces the history of recordings which allows the user to control the recordings that are going to be launched and those that are already available for streaming. This recording concept makes one Media Server announcing the TV programmes to be recorded and the other announcing the recordings requested by the end user.

The Control Point is the core application responsible to connect UPnP devices and Services with other non-UPnP applications of the system. It acts as a bridge that allows the Web Interface to access media content available in the UPnP Media Servers.

3.4 - CWMP modules

The CWMP modules comprise a CPE that can be located in the STB and an Auto Configuration Server located in the Telco Network. Usually the CPE accesses the DB inside Home Network and search for devices names and capabilities. It creates a file for each device, containing a description of all items announced by the device. Then the CPE initiates a connection with the ACS and sends the name of each device and the URI of each file associated to it. This information is sent using the parameters defined in the TR-135 guidelines and using some extensions created specially for this purpose. TR-069 allows each vendor to extend the CWMP protocol with specific vendor parameters. The ACS stores this information on the Telco's DB and this process is repeated periodically.

3.5 - Interface

As described before, the web Interface can be installed inside Home Network or in the Telco Network.

Considering that the Web Interface is provided from the Telco Network, the DB provides the data necessary to show the available devices in the Home Network. If the user is accessing from a terminal outside Home Network, he can probably see a list of the media items available but streaming is not possible. However, if the user terminal is connected to the Home Network the process is quite different. The Web application checks the address of the description files available in the Telco's DB. This address is used to transfer the description files from the Home Network to the Telco Network. After this, the Web application processes the information of each description file, like media items name and URI. As the user terminal and Media Servers belong to the same sub-network, it is possible to stream the media items.

In a scenario where the web Interface is provided from the Home Network, it is not necessary to transfer any description file because the Web Interface is able to access those files directly.

3.6 - Conclusion

One of the major goals of this thesis is to define an integrated solution to the *Virtual PVR* system. The system architecture has two environments: Home Network and Telco Network. The thesis is focused on the Home Network environment which comprises UPnP devices and services and a Web Interface. However, it was necessary to specify these modules as being part of a solution that is also composed by the CWMP modules.

Chapter 4

Virtual PVR Implementation

4.1 - Overview

The *Virtual PVR* developed is composed by UPnP devices and services and an Interface. The developed work consists in a *Virtual PVR* system available to the end user through he Web Interface located in the Home Network.

The *Platinum* library provided the necessary framework and tools to implement UPnP Media Servers and Control Points. There are some reasons that were considered to choose this UPnP library: it is written in C++ (object oriented programming language) and some well known media center applications (XBMC, Boxee, etc.) takes advantage of the Platinum library to provide UPnP support. The C++ programming language was used to implement the UPnP devices and services. The Web Interface was build using php and javascript.

4.2 - Physical Architecture and Components

The *Virtual PVR* test architecture simulates a disposal of media devices in a conventional Home Network. It comprises the implemented STB prototype, a Media Server, Media Renderer, a router, and terminals with a WWW browser.

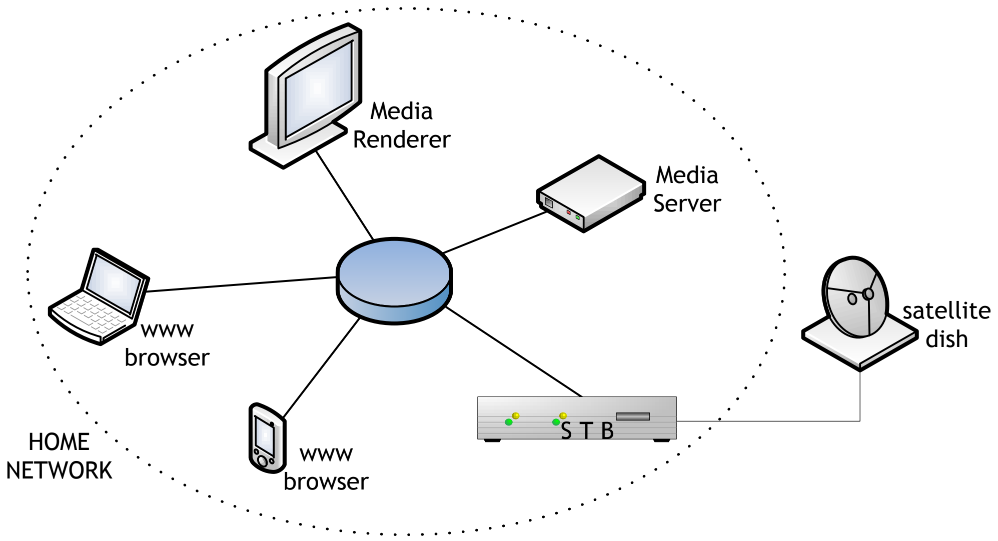


Figure 4.1- *Virtual PVR* device architecture.

The STB prototype is based on a Neuros Link with a TV tuner Technisat Skystar 2 connected to a satellite dish. Due to restrictions of the satellite dish physical support and building orientation, only the HISPASAT 1C satellite is visible. The HISPASAT 1C satellite orientation is 30º west. The *Virtual PVR* prototype uses only free to air channels. The STB prototype contains some resident applications developed to achieve the *Virtual PVR* solution. It has the TV, EPG and PVR UPnP Media Servers, the main Control Point and, besides these UPnP devices, a HTTP Server (Apache HTTP Server) with the Web Interface. The *Virtual PVR* Control Point and the Web Interface were developed to allow portability, which means that they can be running on other devices without damage to the system.

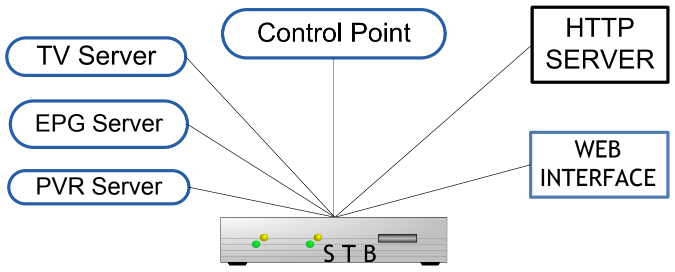


Figure 4.2- Logical entities running in the STB.

The STB does not have any HDD storage unit, so the recordings are streamed to the Media Server available at the moment. Any network storage with UPnP Media Server capabilities can be connected to the Home Network and automatically takes part on the system. The SMB (Server Message Block) protocol is used to perform a local mount action. The Media Server is mounted, like an external disk, enabling the STB to write data into the Media Server storage unit. Iomega Home Media Network Storage has a built in Twonky UPnP Media Server and is the NAS used for testing the *Virtual PVR* prototype.

The router simulates a regular router present in a conventional Home Network. This project uses an Asus WL-500W model because it supports OpenWRT which can be useful in future improvements, like porting the Control Point to the router.

The Media Renderer is useful to demonstrate the capability for reproducing the available media items on any Media Renderer through the Web Interface. Only software Media Renderers running on different PCs were tested but any TV with a built-in UPnP Media Renderer can be used.

In our prototype we used a DVB-S2 receiver, however it is important to note that DVB-C (or others…) could also be used. The system was designed to be as modular as possible so that other receivers can be plugged with minimal modifications. The specific configuration and parameters of the receiver technology are not propagated outside of its module. The DVB-S2 reception has some specific parameters that must be configured, as frequency and signal rate. In order to provide a flexible *Virtual PVR* solution, the DVB configuration parameters are stored in a configuration file. Adapt the solution to a DVB-C or DVB-T receiver only means to replace the configuration file to a DVB-S or DVB-T specific one.

Accessing the system just requires a WWW browser which means a laptop, the STB or a Smartphone connected to the Home Network.

4.3 - UPnP Media Servers

4.3.1 - Overview

The sections 4.3.1.1 - 4.3.1.3 expose the specific functions of each Media Server implemented in the *Virtual PVR*: TV, EPG and PVR. These devices perform common UPnP actions and specific tasks depending on the device function and purpose. They have common UPnP actions that are going to be explained.

The *Platinum* library has been used to build the UPnP Media Server. There are methods to implement the Content Directory and the Connection Manager services. The first thing to do is set the UDP port 1900 as the broadcast port of the Media Server like defined in the UPnP guidelines. The broadcast through UDP port 1900 allows a single message to be received and heard by all UPnP devices present in the same sub-network. The Media Server announces its capabilities by issuing a UDP message to the multicast IP address 239.255.255.250 aimed at port 1900. A new `PLT_FileMediaServer` object is initialized defining a new UUID, a path to the folder which files are going to be announced and a device friendly name. Then, it is necessary to define the model description, URL, model number, name and manufacturer information to fill the UPnP Media Server XML description file.

Once defined the device description parameters the *upnp.Start*() is set and the UPnP Media Server starts announcing services and capabilities. The path to the folder containing the files to be announced and the UUID are the two parameters responsible for distinguish the TV, EPG and PVR Media Servers.

4.3.2 - TV

The TV device is responsible to tune the TV channels and consequently, to launch the streaming server. VLC and the VideoLan Manager (VLM), as described on 2.3.3.1 section, provide tools to perform these tasks.

The VLM allows putting the DVB-S2 configuration parameters on a single text file, the *vlm.conf*. Recalling Section 2.3.3.1, there are two types of Media: *broadcast* and *vod*. In this work each channel is described as a *vod* Media object. It happens because the *Virtual PVR* allows users to view a TV channel present on the channels list, stop and change between channels like a video on demand session. The configuration file *vlm.conf* was created in an iterative process where different DVB parameters were tested for each free to air channel. The Hispasat 1C broadcasts 22 free to air TV channels which mean that 22 *vod* sessions were created. The Lyngsat web site provides a list of HISPASAT 1C channels and DVB-S2 parameters for each channel [35]which were very useful during the initial tuning.

Next there is an extract of the configuration parameters of three TV channels.

```
new   Parlamento.tv  vod enabled
setup Parlamento.tv  option dvb-frequency=11972000
setup Parlamento.tv  option dvb-srate=27500000
setup Parlamento.tv  option dvb-inversion=2
setup Parlamento.tv  option dvb-caching=300
setup Parlamento.tv  option dvb-adapter=0
setup Parlamento.tv  option dvb-device=0
setup Parlamento.tv  option dvb-satno=0
setup Parlamento.tv  option dvb-voltage=18
setup Parlamento.tv  option no-dvb-high-voltage
setup Parlamento.tv  option dvb-tone=-1
setup Parlamento.tv  option dvb-fec=3/4
setup Parlamento.tv  option dvb-lnb-lof1=0
setup Parlamento.tv  option programs=7
setup Parlamento.tv  input  "dvb-s://"


new   TV5Monde.tv    vod enabled
setup TV5Monde.tv    option dvb-frequency=12092000
setup TV5Monde.tv    option dvb-srate=27500000
setup TV5Monde.tv    option dvb-inversion=2
setup TV5Monde.tv    option dvb-caching=300
setup TV5Monde.tv    option dvb-adapter=0
setup TV5Monde.tv    option dvb-device=0
setup TV5Monde.tv    option dvb-satno=0
setup TV5Monde.tv    option dvb-voltage=18
setup TV5Monde.tv    option no-dvb-high-voltage
setup TV5Monde.tv    option dvb-tone=-1
setup TV5Monde.tv    option dvb-fec=3/4
setup TV5Monde.tv    option dvb-lnb-lof1=0
setup TV5Monde.tv    option programs=98
setup TV5Monde.tv    input  "dvb-s://"


new   ZON.tv    vod enabled
setup ZON.tv    option dvb-frequency=11771000
setup ZON.tv    option dvb-srate=27500000
setup ZON.tv    option dvb-inversion=2
setup ZON.tv    option dvb-caching=300
setup ZON.tv    option dvb-adapter=0
setup ZON.tv    option dvb-device=0
setup ZON.tv    option dvb-satno=0
setup ZON.tv    option dvb-voltage=13
setup ZON.tv    option no-dvb-high-voltage
setup ZON.tv    option dvb-tone=-1
setup ZON.tv    option dvb-fec=5/6
setup ZON.tv    option dvb-lnb-lof1=0
setup ZON.tv    option programs=934
setup ZON.tv    input  "dvb-s://"
```

The example above shows the minimum set of options to stream 3 TV channels using VoD session. Each channel is a media object with a unique name (*Parlamento.tv, TV5Monde.tv,* and *ZON.tv*). The DVB-S2 parameters refer to different options:

- **dvb-frequency**: is the frequency to tune to in KHz. In this case, as HISPASAT 1C operates in the Ku band, it varies between 10.7 GHz and 13.25 GHz. Different channels can have the same DVB frequency.

- **dvb-srate**: specifies the symbol rate of the modulated signal, in symbols/s .

- **dvb-inversion**: it defines whether the signal is inverted or not (value *2* represents automatic detection).

- **dvb-adapter**: in case of having several adapters, this parameter specifies the adapter to use (default is *0*).

- **dvb-device** and **dvb-satno**: usually it is equal to *dvb-adapter* and specifies the name of the DVB device to use.

- **dvb-voltage**: is the voltage to apply on the LNB (low-noise block converter) and defines the polarity. When supplied with 13 V LNB selects vertical polarity and horizontal polarity with 18 V.

- **no-dvb-high-voltage**: means disable the special mode of the DVB adapter to compensate for the voltage loss in long cables.

- **dvb-tone**: *-1* is the default value that VLC uses when the frequency supplied is in the Ku band and it determines whether to send a 22 KHz pulse tone to the LNB in order to switch to high band or not.

- **dvb-fec**: represents the code rate to use for FEC (Forward Error Correction).

- **dvb-lnb-lof1**: specifies the frequency of the first oscillator.

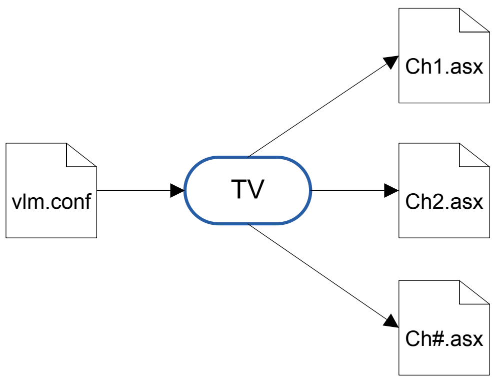
- **program**: represents the Program ID used to specify the program to be selected on a given frequency. It allocates the video and audio streams.

The *Virtual PVR* streaming server is launched using the command line described below. The Telnet interface is used since VLM works only trough the Telnet or HTTP Interfaces. The RTSP protocol is responsible to control the streaming server. It initiates and controls media sessions between end points. RTSP is widespread used in entertainment and communications systems. It uses RTP protocol to the transmission of streaming data.

```
   %vlc -I telnet --telnet-password videolan --rtsp-host 192.168.10.102:25000
--vlm-conf vlm.conf
```

The RTSP host defines the address that streaming clients shall use to receive the stream. The path to *vlm.conf* is passed to the command line. VLC parses it and executes the specified instructions. After parsing the file, VLC does not need to access it during the media session. VLC performs an initial tuning of all channels defined on the *vlm.conf* file to check if there is any anomaly and if everything is correct, the Telnet interface is initiated.

Summarizing, a vlm.conf file must be provided to the TV device in order to launch the TV channels streaming. After launching VLC, starts the UPnP services and capabilities announcing the *.asx* files representing each TV channels *vod* session.



The Advanced Stream Redirector (ASX) format is a XML metafile that stores one or more files to play in a multimedia session. Each ASX file has the name of a channel and points to the RTSP address of the streaming server. The streaming server has the ability to streams the channels since the address requested by the client matches with the name of the Media object. This one of the great advantages of using a single VLM configuration file (*vlm.conf*). However, it is not possible to have multiple clients viewing different channels. There is only one TV tuner connected to the STB so only one client is able to receive a TV channel.

The example below shows the content of three ASX files announced by the TV device.

File TV5Monde.asx :

```
<ASX version="3">
<Entry>
```

```
<ref href="rtsp://192.168.10.102:25000/TV5Monde.tv" />
</Entry>
</ASX>
```

File Parlamento.asx :

```
<ASX version="3">
<Entry>
<ref href="rtsp://192.168.10.102:25000/Parlamento.tv" />
</Entry>
</ASX>
```

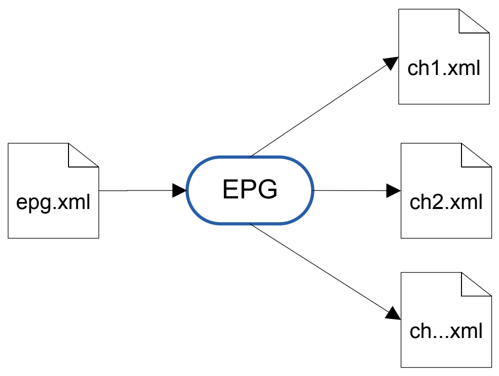
File ZON.asx :

```
<ASX version="3">
<Entry>
<ref href="rtsp://192.168.10.102:25000/ZON.tv" />
</Entry>
</ASX>
```

The ASX media format has some advantages. It hides the audio/video content source which means that the client cannot download a copy of the content and the streaming is not send via HTTP like usual web contents. Open one of the *.asx* files, announced by the TV device, makes the default media player reproduce the TV channel stream.

4.3.3 - EPG

As explained on section 3.3, the EPG device announces files containing the TV listings which represent media items to be recorded.
The diagram below represents the input/output files of the EPG device.



The input of the EPG application is a XML file *(epg.xml)* containing all the next 7 days programmes of all channels on the frequency interval 11.6-12.3 GHz. The EPG device, as all the system is DVB-x independent in a way that the only EPG requirement is a XML file containing the TV listings in the format specified below:

```
<tv generator-info-name="dvb-epg-gen">
<channel id="765.dvb.guide">
	<display-name>TV5 Monde</display-name>
</channel>
<channel id="774.dvb.guide">
	<display-name>TV Galicia</display-name>
</channel>
<channel id="91.dvb.guide">
	<display-name>TeleMadrid Sat</display-name>
</channel>

<programme channel="811.dvb.guide" start="20090531223000"
stop="20090531231500">
	<title lang="eng">Os Contemporâneos</title>
</programme>
<programme channel="811.dvb.guide" start="20090531231500"
stop="20090601010000">
	<title lang="eng">Programa A Designar</title>
</programme>
```

The file begins with the channels ID, the display name of each channel and then the TV listings description. The channels ID are used to identify the channel of each programme. Every 24 hours the EPG device runs TV GRAB DVB [31] application with the purpose of generating the *epg.xml* file. TV GRAB DVB uses a *channels.conf* to create the `channel id` tags. The channels.conf contains a list of channels and its frequencies. After this process, the EPG creates a file for each channel containing its programmes. It is achieved with the *xerces* library that has methods to read and parse a XML file.

The overall process of parsing the epg.xml and creating the new TV listings XML files can be synthesized in the below diagram.

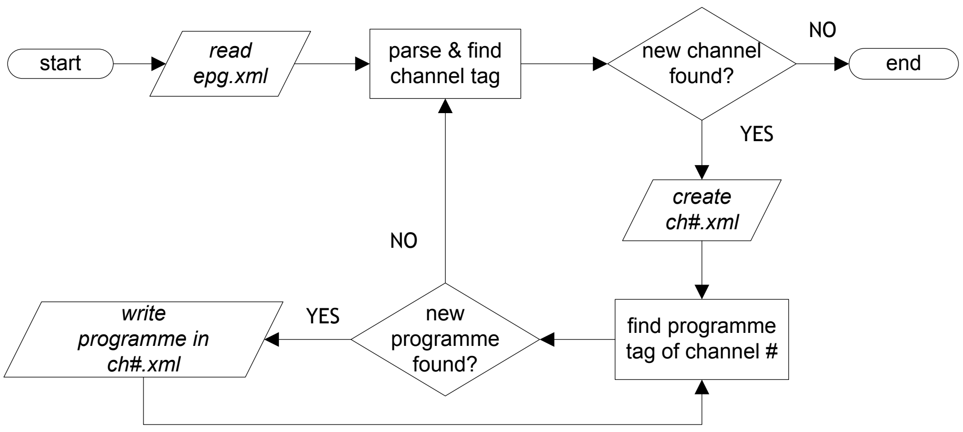


Figure 4.3- Fluxogram of the EPG operation.

The satellite HISPASAT 1C only broadcast two free to air channels supporting EPG, however it is enough to demonstrate the *Virtual PVR* recording features. This means that EPG only creates two files containing TV listings. The other files are empty because, as it can bee seen in the diagram above, the EPG application just writes on the file if a programme tag is found. Next there is a small sample of a file created by the EPG device.

```
- <channel>
  <name>TV5 Monde</name>

- <programme start="20090526120000" stop="20090526130000">
  <title lang="eng">TV5Monde, le journal</title>
  </programme>

- <programme start="20090526130000" stop="20090526143500">
  <title lang="eng">SuperGranny.com</title>
  </programme>

- <programme start="20090526143500" stop="20090526150000">
  <title lang="eng">L'autre émoi</title>
  </programme>
```

The Web Interface parses this data and displays it to the end user and then the recordings can be set.

4.3.4 - PVR

The PVR device launches the recordings sent by the Web Interface in a vlm file format. The main PVR folder is composed by two folders: *New* and *Old*. Every time a user gives an order to record a programme, the Web Interface creates a vlm file and sends it via FTP to the *New* folder. The PVR device checks the *New* folder every 5 minutes to confirm if there is any file. It is performed using some methods of the *dirent* library which helps directory traversing. If there is any file, the PVR assures that file transfer is completed by checking the modified date. If the difference between the actual time and the modified date is bigger than 60 seconds, it performs a fork. The child process makes a *system* call and launches VLC using the *dummy* interface and pointing to vlm file using the *--vlm-conf* parameter.

Recalling the section 2.3.3.1, VLM is capable of performing scheduled actions using the *schedule* script. The most used commands to perform the schedule action are *play* and *stop* associated with two date times. One to starts the schedule action and other to stops it. The schedule action performs these two actions over a media object. In this case, the media object is one of the TV channels streamed by the TV device.

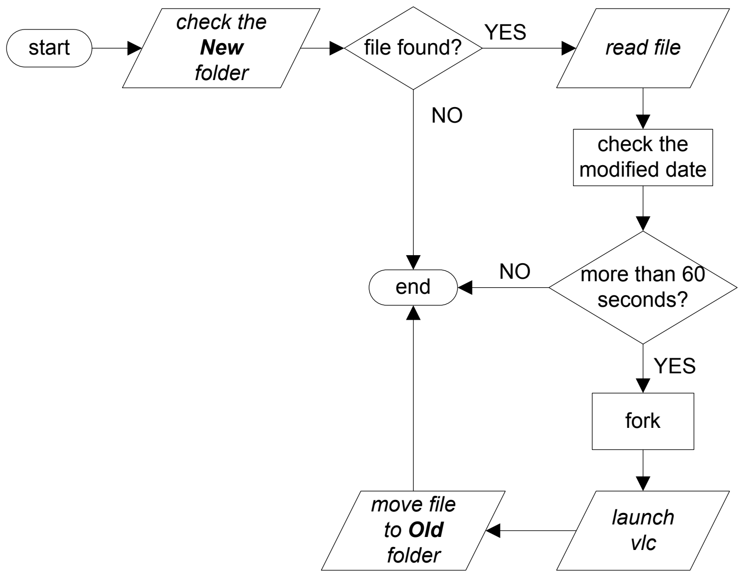


Figure 4.4- Fluxogram of the PVR operation.

It is important to remember that the schedule action is performed to record a stream so, it is necessary to specify the *output* of the media object streamed. The *Virtual PVR* stores the media recordings on the Media Server storage unit. As described before, the Media Server is mounted locally on a folder of the STB which means that the stream is stored in that folder.

Next there is an example of a vlm file created by the Web Interface.

```
new record broadcast enabled input rtsp://192.168.10.102:25000/ZON.tv output
#duplicate{dst=std{access=file,mux=ps,dst=/home/neurostv/rec/rec4.mpg}}

new start schedule enabled date 2009/5/23-18:46:00 append control record play
new end schedule enabled date 2009/5/23-18:46:09 append control record stop

new stop broadcast enabled input "vlc://quit"
new quit schedule enabled date 2009/5/23-18:46:09 append control stop play
```

First a Media object is created with the purpose of receiving the TV channel streaming and the *dst* defines where and the format to store it. As it can be seen, the *input* is the RTSP address of a specific *vod* session (*ZON.tv*). Next, the date time and the stop time of the streaming session are defined which represents the date time of the *play* order to receive the stream and the *stop* order to finish the streaming session. After performing the schedule actions explained above, VLC does not quit and keeps running although, not executing any command. Each vlm file received by PVR device means a process child that is forked and a VLC instance that is launched. It is necessary to implements a way of killing the child process and the respectively VLC instance. When VLC runs the command "`vlc://quit`" it kills the current process and performs an exit action. Defining a Media object which *input* is a `"vlc://quit"` solves the multiprocess problem but is necessary to define when to perform this action. A schedule action is defined to *play* the exit action at the same time date as the

TV streaming channel is over. In this way, when PVR device receives a vlm file, the VLC is launched and keeps waiting to the start date time to be reached. At the defined start time, the recording action begins and after the recording finish the VLC process is killed. After launching VLC the PVR moves the vlm file to the *Old* folder which is an historic of all recordings performed by the PVR device. The files present in the *Old* folder are announced via UPnP. This announcement allows the Web Interface to display the recordings that already have been completed and that ones which start time is not reached yet.

4.4 - UPnP Control Point

The UPnP Control point implemented is the bridge between Media Servers of the Home Network and the Web Interface, which is not a UPnP device or service. Once again *Platinum* provides the framework and tools to create a Control Point however the *Virtual PVR* Control Point is extended to function as the media items collector of the Home Network and provide this data to the Web Interface.

A new `PLT_CtrlPoint` is created broadcasting it services on UDP port 1900. It starts performing a search for other devices on the port 1900 aimed to 239.255.255.250 multicast address and to the 255.255.255.255 (network broadcast address) address for the case that some device does not support multicast. It announces itself as a "`upnp:rootdevice`" and it receives the discovery messages from the other Home Network devices. First it deals with the Media Servers by selecting the first Media Server found and getting all the media items from the Media Server root folder until the leaves. The Content Directory service allows the control point to make a browse through the media library available in the Media Server. It is done by checking if a resource is an item or a container. A container is a collection of other containers or items, i.e., like a folder. An item is a single entity or piece of content, i.e., like a file. When the Control Point realizes that the current resource is not a container, it defines it as a `PLT_MediaObject` provided by the AV Transport Service. At this point is possible to get its name and URI. The Control Point checks the MIME type of the item by comparing the URI extension and then it writes this information on a XML file.

The XML file is the Data Base of the system once it contains a list of all media items available at the moment.

The Control Point creates and edits the XML file as it is traversing the items of a Media Server.

Below there is an example of some tags of the description file:

```
<File>
      <Name>TV5 Monde</Name>
      <MediaServer>EPG</MediaServer>
      <URI>http://192.168.10.102:58976/content%3Fpath=TV5%20Monde.xml
   </URI>
      <MIME>unknown/unknown</MIME>
</File>
```

```
<File>
      <Name>Parlamento</Name>
      <MediaServer>TV</MediaServer>
      <URI>http://192.168.10.102:62425/content%3Fpath=Parlamento.asx</
URI>
      <MIME>tv/tv</MIME>
</File>


<File>
      <Name>Mysteries</Name>
      <MediaServer>[Iomega] TwonkyMedia</MediaServer>
      <URI>http://192.168.10.100:9000/disk/music/DLNA-PNMP3-OP11-
      FLAGS01700000/O1$11$207905893$2623825016.mp3</URI>
      <MIME>audio/mpeg</MIME>
</File>
```

Once traversed all the media items and containers of the first Media Server, the same is done to the other servers until all the resources are covered. The Control Point, in this way, provides the Web Interface with the URI of each media item available in the Home Network so the streaming becomes accessible to non-UPnP devices.

After this process the Control Point is not necessary anymore to the *Virtual PVR* system due to the streaming session being performed directly between the browser default media player and the Media Server. However, if a Media Server is switched off the Home Network Database needs to be refreshed and the same happens when a Media Server joins the Home Network. In order to avoid synchronization problems, the Control Point performs the discovery process every 30 seconds, refreshing the Home Networking Database.

There are some actions that need to be performed by the Control Point to allow the user to choose where to see the media content. The process consists on performing a *setMR* action to the Media Renderer chosen by the user to reproduce a media item. First the Control Points lists the available Media Renderers, associates a number to each one and creates a file with this information. The file provides the information necessary to the Web Interface to display the various outputs the end user can choose. The Web Interface sends the number of the selected Media Renderer to the Control Point through a TCP socket. When the user chooses the media item to play, the Web Interface sends back the URI of the item to the Control Point. The AV Transport Service methods are used by the Control Point to establish the connection between the Media Renderer selected and the Media Server.

4.5 - Interface

The *Virtual PVR* provides end user experience through a Web Interface that can be accessed using any Web browser. The Web Interface was developed using Codeigniter which is an open source application framework for building dynamic web portals with PHP language [36]. Benchmarks tests [37]show that Codeigniter is faster than other PHP frameworks, like Cake PHP, or Smarty, as it processes more requests per seconds and less time per request.

Codeigniter uses the Model-View-Controller development pattern, which separate business logic and data from presentation. In this way it is possible to change underlying business rules without affecting visual appearance and the opposite too.

The Web Interface has 5 major sections: Music, Video, Images, Live TV and Recorder. The data displayed on the Web Interface is provided by the Home Network Database which means, read and parsing the XML file generated by the Control Point. Each `<file>` tag represents an item to be displayed by one of the Web Interface sections. The Music, Video and Images sections display items based on MIME type. The TV and Recorder sections provide information announced by the TV, EPG and PVR devices. In this case the items are selected based on Media Server.

Virtual PVR offers streaming services (audio and video) that are media player and Operative System independents. Every time the user request an item to be streamed it means that the HTML embed tag is being processed using the item URI. The HTML embed tag puts the browser's default media player in the page to reproduce the item pointed by the URI. A Flash player could be used, however there are some devices that do not support Flash, such as iPhone and Android.

The Web Interface is also responsible to create the recording file in a vlm format, as described on 4.3.1.3 section. This is done using the channel name, start and stop date times. Then the file is uploaded to the *New* directory inside the main PVR directory.

4.6 - Conclusion

The diagram below illustrates the common sequence of requests and responses that leads to the stream of a media item to be displayed on the Web Interface.

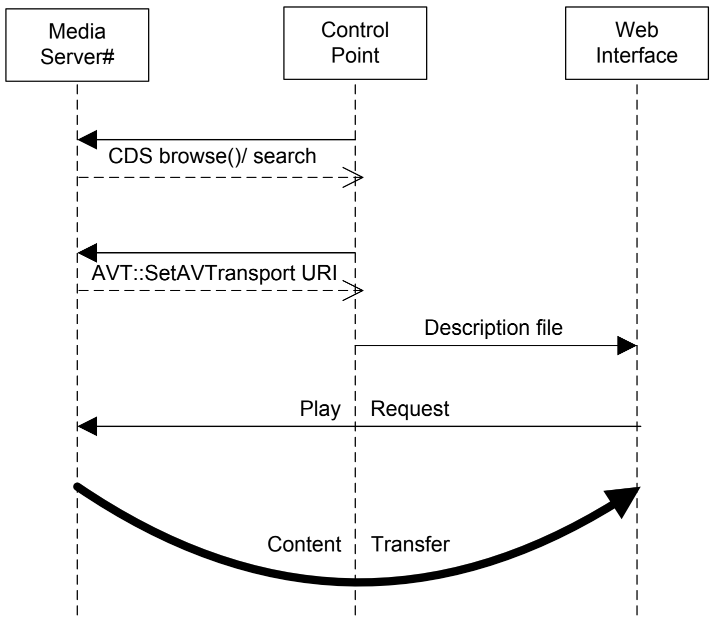


Figure 4.5- Web rendering process

The Web Interface is also responsible to control where to reproduce a media item. It is important to refer that even when the user chooses a Media Renderer as output, the file is reproduced on the Web Interface as well. It is possible because Media servers define a connection ID for each streaming session which means that an item being reproduced on a Media Renderer and on the browser's media player represent two different connections ID.

The next diagram describes the process of selecting a Media Renderer to reproduce a given media item.

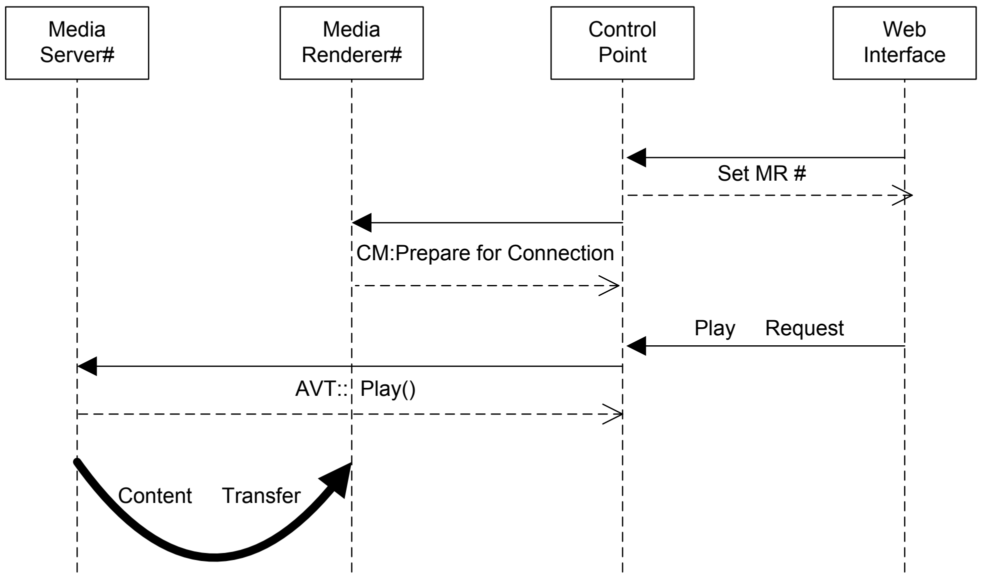


Figure 4.6- Remote Rendering

The Web Interface requests a Media Renderer to be set. The Control Point is responsible for this process. After Media Renderer become prepared for the connection the Control Point waits the Web Interface (user) to choose the item available on a given Media server. Then the Media Renderer and the Media Server proceed to the content transfer.

Chapter 5

Evaluation and Validation

5.1 - Overview

In order to evaluate the *Virtual PVR* prototype a set of lab experiments were performed. It is important to test the functionalities of each developed device or module, such as the TV, EPG, PVR and Control Point. The UPnP devices exchange discovery and configuration messages through the Home Network. These messages are going to be captured and examined to better comprehend if they follow the UPnP specifications. The user accesses the system through the Web Interface. The Web Interface screenshots display some of the main features of the developed prototype. The Web Interface is still under development with the purpose of being more intuitive and friendly.

5.2 - TV tuning and streaming

As described on Section 4.3.2, the TV devices start by tuning the channels present on the *vlm.conf* file. The VLC is launched and checks if there is any anomaly on the DVB-S parameters. After this process, it creates the *vod* sessions for each channel. Next, we can see a sample of the VLC output messages visible during the initial tuning.

```
[00000489] [Media: ZON.tv] dvb access debug: Opening device
/dev/dvb/adapter0/frontend0

[00000489] [Media: ZON.tv] dvb access debug: Frontend Info:
[00000489] [Media: ZON.tv] dvb access debug:   name = ST STV0299 DVB-S
[00000489] [Media: ZON.tv] dvb access debug:   type = QPSK (DVB-S)
[00000489] [Media: ZON.tv] dvb access debug:   frequency_min = 950000 (kHz)
[00000489] [Media: ZON.tv] dvb access debug:   frequency_max = 2150000 (kHz)
[00000489] [Media: ZON.tv] dvb access debug:   frequency_stepsize = 125
[00000489] [Media: ZON.tv] dvb access debug:   frequency_tolerance = 0
[00000489] [Media: ZON.tv] dvb access debug:   symbol_rate_min = 1000000 (kHz)
[00000489] [Media: ZON.tv] dvb access debug:   symbol_rate_max = 45000000
(kHz)
```

```
[00000489] [Media: ZON.tv] dvb access debug: trying to tune the frontend...
[00000489] [Media: ZON.tv] dvb access debug: frequency 11771000 is in Ku-band
[00000489] [Media: ZON.tv] dvb access debug: using inversion=2
[00000489] [Media: ZON.tv] dvb access debug: using fec=5
[00000489] [Media: ZON.tv] dvb access debug: using voltage=13
[00000489] [Media: ZON.tv] dvb access debug: using tone=1

[00000501] vod_rtsp vod server debug: created RTSP url: /ZON.tv
[00000501] vod_rtsp vod server debug: media has 2 declared ES
[00000501] vod_rtsp vod server debug:   - ES mpga (/ZON.tv/trackID=0)
[00000501] vod_rtsp vod server debug:   - ES mpgv (/ZON.tv/trackID=1)
```

The example above refers to the VLM media object *ZON.tv* and it is possible to verify the Technisat Skystar2 capabilities such as type, frequency and symbol rate range. After capturing the device capabilities, VLC starts to tune the TV channel using the DVB-S parameters defined in the *vlm.conf* file. It recognizes the channel frequency as in the Ku band and once the channel is successfully tuned, the RTSP URL */ZON.tv* is created as well as, the audio and video elementary streams. This means that the RTSP server is ready to start streaming the channel. In this case, it is just necessary to specify the RTSP URL as *rtsp://192.168.10.102:25000/ZON.tv* .

5.3 - Home Network message flow

The *Virtual PVR* UPnP Control Point is responsible for searching devices and services available in the Home Network. The test scenario has 4 UPnP devices: TV, EPG, PVR and Iomega Home Network device with a built in TwonkyMedia UPnP Server. The Wireshark packet sniffer was used to capture the messages exchanged between the Home Network UPnP devices. The examples below were captured using the *follow UDP stream* and *follow TCP stream*, functionalities embedded in Wireshark. The examples refer to the Control Point and the EPG devices. Both share the STB IP address, *192.168.10.102* .

5.3.1 - Discovery

Control Point request:

```
M-SEARCH * HTTP/1.1
MX: 5
ST: upnp:rootdevice
MAN: "ssdp:discover"
User-Agent: Platinum/0.4.7
Host: 239.255.255.250:1900
Content-Length: 0
```

The message above is the initial multicast SSDP discover message sent by the Control Point in order to discover new devices and services on the Home Network.

EPG responses:

```
NOTIFY * HTTP/1.1
Host: 239.255.255.250:1900
LOCATION: http://192.168.10.102:50469/DeviceDescription.xml
NTS: ssdp:alive
CACHE-CONTROL: max-age=1800
SERVER: UPnP/1.0, Platinum UPnP SDK/0.4.7
USN: uuid:72356701-61b9-11de-8a39-0800200c9a66::upnp:rootdevice
NT: upnp:rootdevice
Content-Length: 0
```

```
NOTIFY * HTTP/1.1
Host: 239.255.255.250:1900
LOCATION: http://192.168.10.102:50469/DeviceDescription.xml
NTS: ssdp:alive
CACHE-CONTROL: max-age=1800
SERVER: UPnP/1.0, Platinum UPnP SDK/0.4.7
USN: uuid:72356701-61b9-11de-8a39-0800200c9a66::urn:schemas-upnp-
org:device:MediaServer:1
NT: urn:schemas-upnp-org:device:MediaServer:1
Content-Length: 0
```

```
NOTIFY * HTTP/1.1
Host: 239.255.255.250:1900
LOCATION: http://192.168.10.102:50469/DeviceDescription.xml
NTS: ssdp:alive
CACHE-CONTROL: max-age=1800
SERVER: UPnP/1.0, Platinum UPnP SDK/0.4.7
USN: uuid:72356701-61b9-11de-8a39-0800200c9a66::urn:schemas-upnp-
org:service:ContentDirectory:1
NT: urn:schemas-upnp-org:service:ContentDirectory:1
Content-Length: 0
```

```
NOTIFY * HTTP/1.1
Host: 239.255.255.250:1900
LOCATION: http://192.168.10.102:50469/DeviceDescription.xml
NTS: ssdp:alive
CACHE-CONTROL: max-age=1800
SERVER: UPnP/1.0, Platinum UPnP SDK/0.4.7
USN: uuid:72356701-61b9-11de-8a39-0800200c9a66::urn:schemas-upnp-
org:service:ConnectionManager:1
NT: urn:schemas-upnp-org:service:ConnectionManager:1
Content-Length: 0
```

The EPG device responds with SSDP messages announcing it as UPnP root device and announcing its services: Content Directory and Connection Manager. The location of its device description file is also announced. It is possible to verify that both Control Point and EPG were implemented using the *Platinum* library, as described in Section 2.3.2.

5.3.2 - Description

The next step is the *Description* of the devices and services announced on the *Discovery* messages. The Control Point uses the device description URL announced before to perform a HTTP GET request of the XML file.

Control Point request:

```
GET /DeviceDescription.xml HTTP/1.0
Connection: close
User-Agent: Neptune/1.0.1
Host: 192.168.10.102:58778
Content-Length: 0
```

EPG response:

```
HTTP/1.1 200 OK
Server: Platinum/0.4.7
Content-Length: 1702
Content-Type: text/xml
Connection: close
User-Agent: Platinum/0.4.7

<root xmlns="urn:schemas-upnp-org:device-1-0" xmlns:dlna="urn:schemas-dlna-
org:device-1-0">

(...)


</root>
```

Control Point request:

```
GET /ContentDirectory/72356701-61b9-11de-8a39-0800200c9a66/scpd.xml HTTP/1.0
Connection: close
User-Agent: Neptune/1.0.1
Host: 192.168.10.102:58778
Content-Length: 0
```

EPG response:

```
HTTP/1.1 200 OK
Server: Platinum/0.4.7
Connection: close
User-Agent: Platinum/0.4.7
Content-Length: 6779
Content-Type: text/xml

<scpd xmlns="urn:schemas-upnp-org:service-1-0">
(...)
</scpd>
```

Control Point request:

```
GET /ConnectionManager/72356701-61b9-11de-8a39-0800200c9a66/scpd.xml HTTP/1.0
Connection: close
User-Agent: Neptune/1.0.1
Host: 192.168.10.102:58778
Content-Length: 0
```

EPG response:

```
HTTP/1.1 200 OK
Server: Platinum/0.4.7
Connection: close
User-Agent: Platinum/0.4.7
Content-Length: 4371
Content-Type: text/xml

<scpd xmlns="urn:schemas-upnp-org:service-1-0">


(...)

</scpd>
```

The Control point issues the request of the device description, Content Directory and Connection Manager description files. Accessing these files allows the Control Point to know in detail the commands and actions that can be performed as well as, the parameters for each action.

5.3.3 - Control

The Control Point knows the services that EPG offers so it starts sending Control messages with the objective of listing all the media items announced by the Content Directory service of the EPG device. For each media item the Control Point sends a HTTP POST request and the EPG responds with a message expressed in XML using SOAP protocol, as defined on UPnP guidelines.

Control Point request:

```
POST /ContentDirectory/72356701-61b9-11de-8a39-0800200c9a66/control.xml
HTTP/1.0
Content-Type: text/xml; charset="utf-8"
SOAPAction: "urn:schemas-upnp-org:service:ContentDirectory:1#Browse"
Connection: close
User-Agent: Neptune/1.0.1
Host: 192.168.10.102:58778
Content-Length: 475


<s:Envelope s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
```

```
  <s:Body>
    <u:Browse xmlns:u="urn:schemas-upnp-org:service:ContentDirectory:1">
      <ObjectID>0</ObjectID>
      <BrowseFlag>BrowseDirectChildren</BrowseFlag>
      <Filter>*</Filter>
      <StartingIndex>0</StartingIndex>
      <RequestedCount>1024</RequestedCount>
      <SortCriteria></SortCriteria>
    </u:Browse>
  </s:Body>
</s:Envelope>
```

EPG response:

```
HTTP/1.1 200 OK
Server: Platinum/0.4.7
Content-Type: text/xml; charset="utf-8"
Ext:
Connection: close
User-Agent: Platinum/0.4.7
Content-Length: 1694

<s:Envelope s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <u:BrowseResponse xmlns:u="urn:schemas-upnp-
org:service:ContentDirectory:1">
      <Result

       (...)

       </Result>
      <NumberReturned>3</NumberReturned>
      <TotalMatches>3</TotalMatches>
      <UpdateID>1</UpdateID>
    </u:BrowseResponse>
  </s:Body>
</s:Envelope>
```

5.3.4 - Eventing

After browsing for all media items available in the EPG Media Server, the Control Point sends unicast eventing messages to the EPG device. It represents a subscription for updates of the names and values present in the Content Directory or Connection Manager service. The EPG device responds to confirm the subscription. From now on, if a media item is deleted or a new one is added to the EPG Media Server, the Control Point will receive an event message in order to refresh the media items list.

Control Point subscription issue for Content Directory service:

```
SUBSCRIBE /ContentDirectory/72356701-61b9-11de-8a39-0800200c9a66/event.xml
HTTP/1.0
NT: upnp:event
CALLBACK: <http://192.168.10.102:56931/72356701-61b9-11de-8a39-
0800200c9a66/urn:upnp-org:serviceId:CDS_1-0>
TIMEOUT: Second-1800
Connection: close
User-Agent: Neptune/1.0.1
Host: 192.168.10.102:58778
Content-Length: 0
```

EPG device accepts the subscription:

```
HTTP/1.1 200 OK
Server: Platinum/0.4.7
SID: WSSYKIHDYQYUERMEXIT
TIMEOUT: Second-1800
Connection: close
User-Agent: Platinum/0.4.7
Content-Length: 0
```

Control Point subscription issue for Connection Manager service:

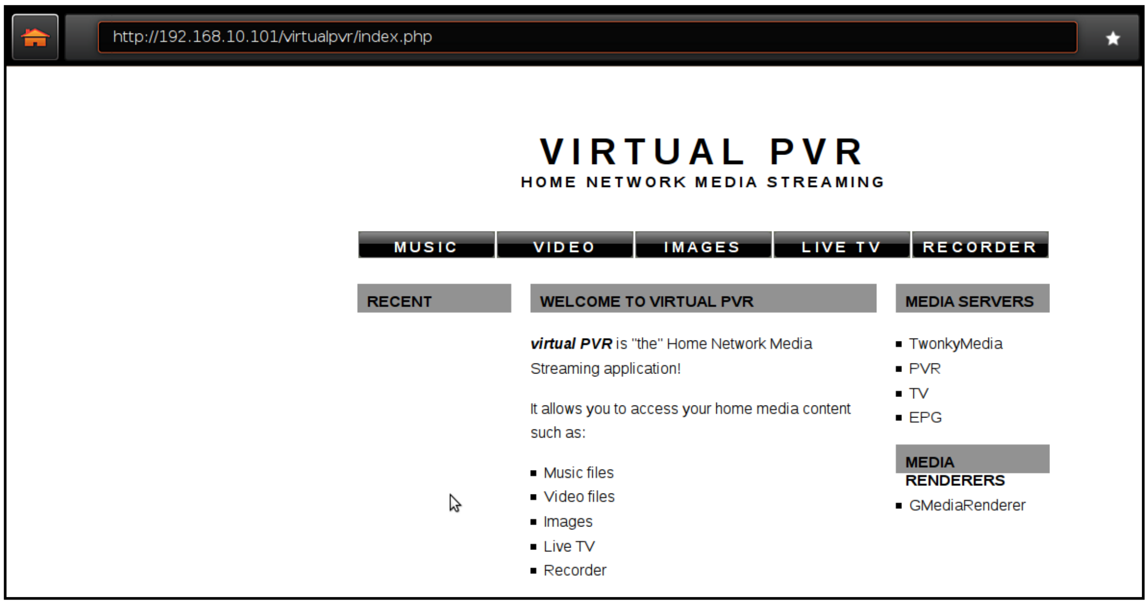
```
SUBSCRIBE /ConnectionManager/72356701-61b9-11de-8a39-0800200c9a66/event.xml
HTTP/1.0
NT: upnp:event
CALLBACK: <http://192.168.10.102:56931/72356701-61b9-11de-8a39-
0800200c9a66/urn:upnp-org:serviceId:CMGR_1-0>
TIMEOUT: Second-1800
Connection: close
User-Agent: Neptune/1.0.1
Host: 192.168.10.102:58778
Content-Length: 0
```

EPG device accepts the subscription:

```
HTTP/1.1 200 OK
Server: Platinum/0.4.7
SID: HIHFDDIOCLXPEDJBWKD
TIMEOUT: Second-1800
Connection: close
User-Agent: Platinum/0.4.7
Content-Length: 0
```

5.4 - Interface

The end user accesses the *Virtual PVR* web interface and a welcome page is displayed. As described in Section 4.5, the user can choose between 5 sections: Music, Video, Images, Live TV and Recorder.



http://192.168.10.101/virtualpvr/index.php

VIRTUAL PVR
HOME NETWORK MEDIA STREAMING

MUSIC | VIDEO | IMAGES | LIVE TV | RECORDER

RECENT

WELCOME TO VIRTUAL PVR

virtual PVR is "the" Home Network Media Streaming application!

It allows you to access your home media content such as:

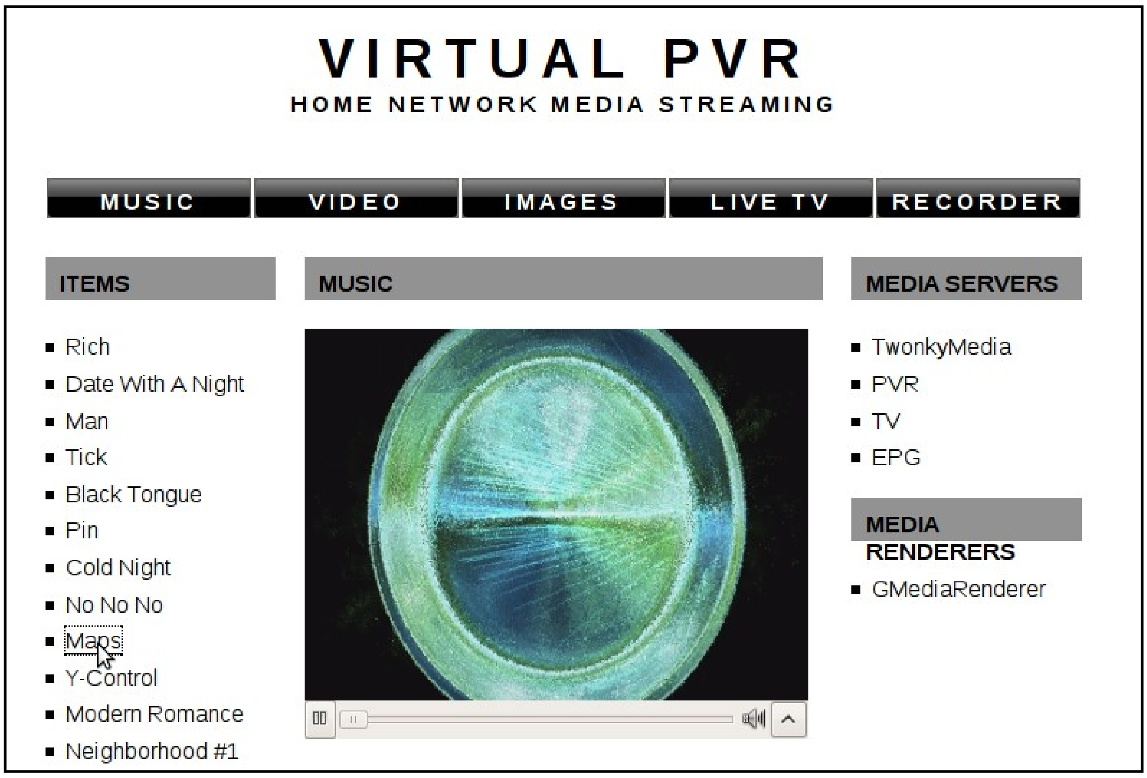
- Music files
- Video files
- Images
- Live TV
- Recorder

MEDIA SERVERS

- TwonkyMedia
- PVR
- TV
- EPG

MEDIA RENDERERS

- GMediaRenderer



VIRTUAL PVR
HOME NETWORK MEDIA STREAMING

MUSIC | VIDEO | IMAGES | LIVE TV | RECORDER

ITEMS

- Rich
- Date With A Night
- Man
- Tick
- Black Tongue
- Pin
- Cold Night
- No No No
- Maps
- Y-Control
- Modern Romance
- Neighborhood #1

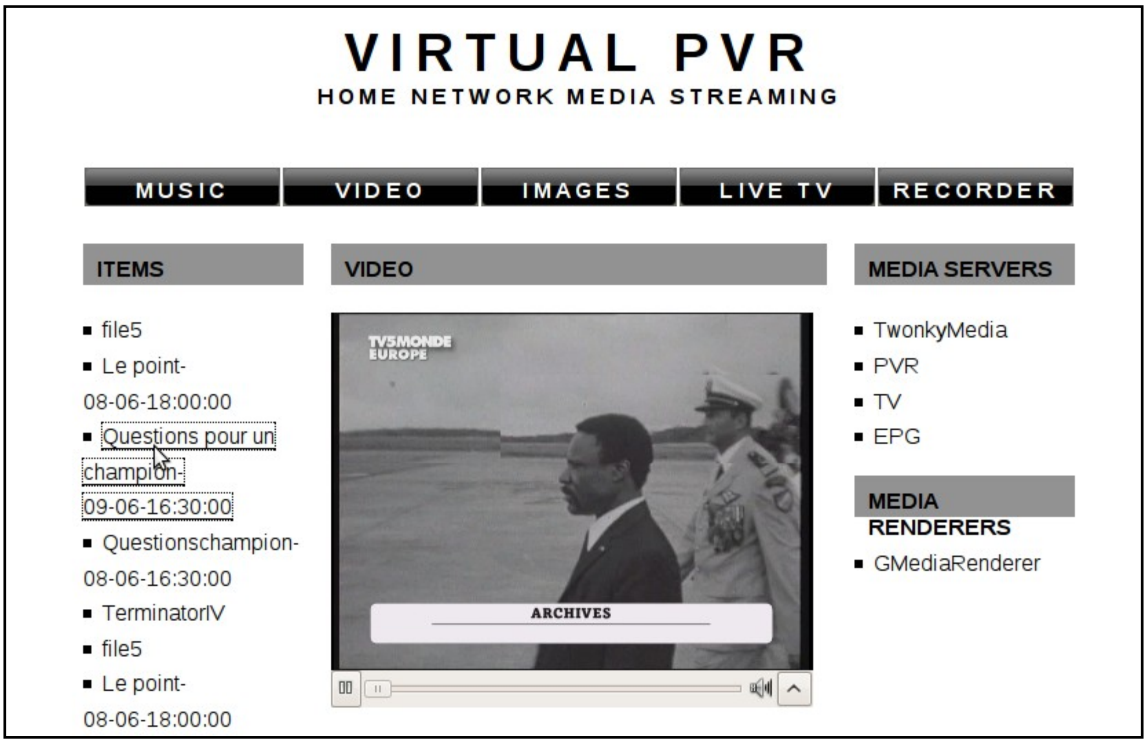
MUSIC

MEDIA SERVERS

- TwonkyMedia
- PVR
- TV
- EPG

MEDIA RENDERERS

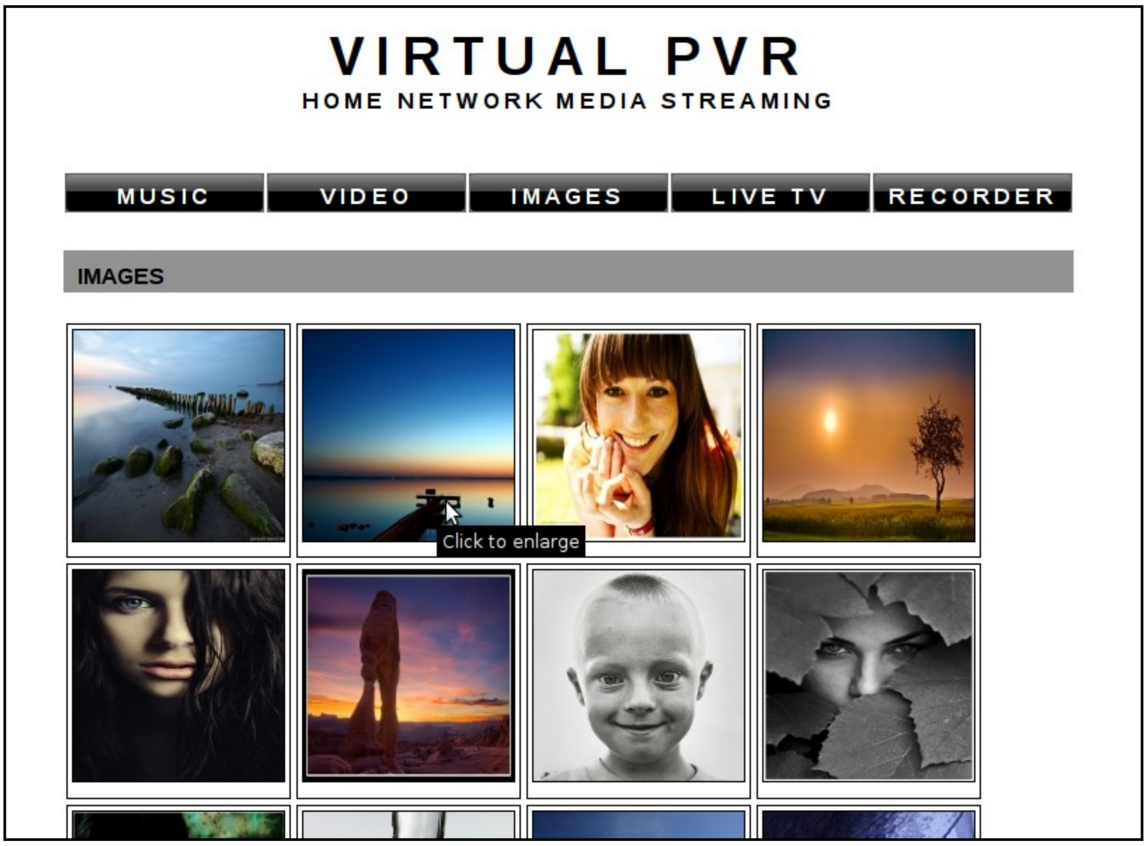
- GMediaRenderer

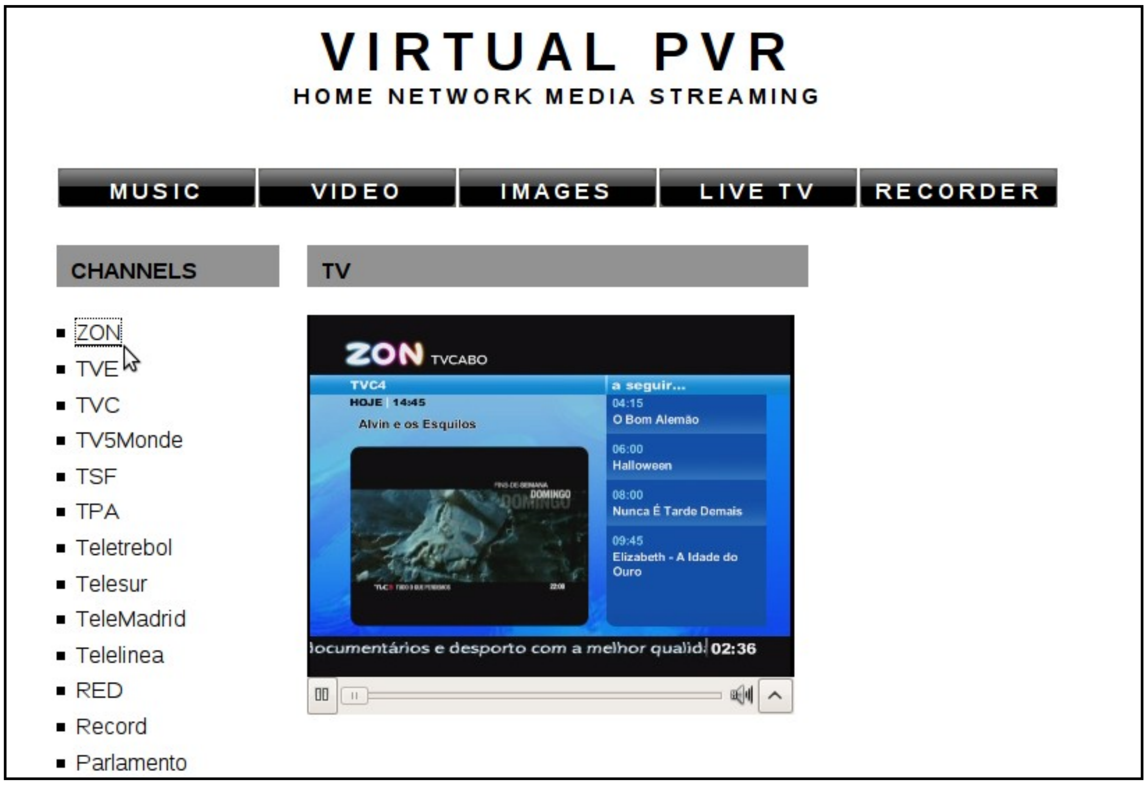


- file5
- Le point-08-06-18:00:00
- Questions pour un champion-09-06-16:30:00
- Questionschampion-08-06-16:30:00
- TerminatorIV
- file5
- Le point-08-06-18:00:00

- TwonkyMedia
- PVR
- TV
- EPG

- GMediaRenderer





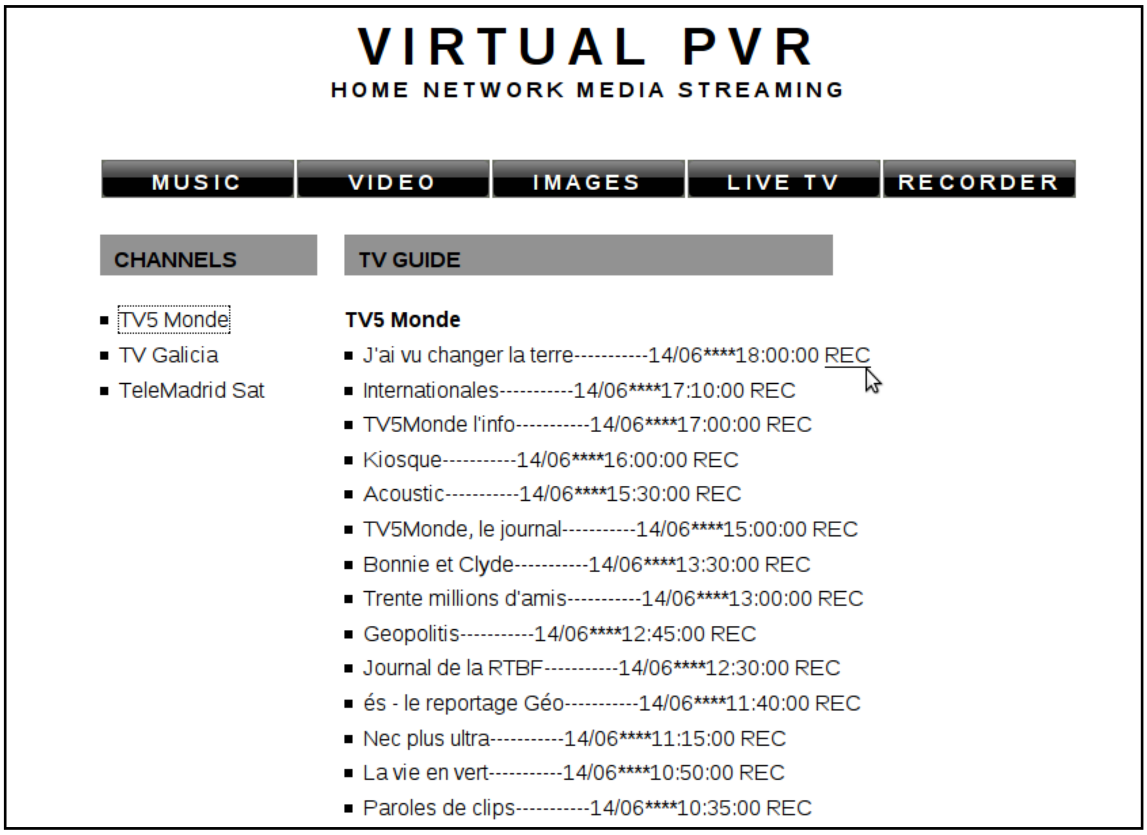
VIRTUAL PVR

HOME NETWORK MEDIA STREAMING

CHANNELS

- ZON
- TVE
- TVC
- TV5Monde
- TSF
- TPA
- Teletrebol
- Telesur
- TeleMadrid
- Telelinea
- RED
- Record
- Parlamento

TV



VIRTUAL PVR

HOME NETWORK MEDIA STREAMING

CHANNELS

- TV5 Monde
- TV Galicia
- TeleMadrid Sat

TV GUIDE

TV5 Monde

- J'ai vu changer la terre----------14/06****18:00:00 REC
- Internationales----------14/06****17:10:00 REC
- TV5Monde l'info----------14/06****17:00:00 REC
- Kiosque----------14/06****16:00:00 REC
- Acoustic----------14/06****15:30:00 REC
- TV5Monde, le journal----------14/06****15:00:00 REC
- Bonnie et Clyde----------14/06****13:30:00 REC
- Trente millions d'amis----------14/06****13:00:00 REC
- Geopolitis----------14/06****12:45:00 REC
- Journal de la RTBF----------14/06****12:30:00 REC
- és - le reportage Géo----------14/06****11:40:00 REC
- Nec plus ultra----------14/06****11:15:00 REC
- La vie en vert----------14/06****10:50:00 REC
- Paroles de clips----------14/06****10:35:00 REC

The Music section displays all the items with audio MIME type, available in the Home Network. The user is free to choose which item wants to play and the default media player starts the stream session.

The Video section shows the video items available in the Home Network. Once again the user is free to choose which video wants to see and the stream session starts.

The Image section is quite different from the sections described before. As it is not necessary to use a media player to receive the stream the web interface creates a thumbnail for each image, using the GD library. If the user wants to see the image in a real size just needs to click on the image. The slideshow mode is also available and was implemented using *Javascript* language.

The Live TV section allows the user to see a TV channel in real time. The STB prototype has only one TV tuner which means that only one user can use this service. The user can choose which channel to see and switch the channel.

The Recorder section displays the TV listings of the channels that support EPG. The user is able to choose which programmes want to record. Once recorded the programme becomes available to see on the Video section.

5.5 - Conclusion

In this Chapter we tested the functionalities of the *Virtual PVR* prototype through a set of lab experiments. The Section 5.1 illustrates the initial tuning of a TV device that launches the VLC streaming server using the parameters defined on a *vlm.conf* file. The UPnP devices present in the Home Network exchange messages that can be captured using the Wireshark. The messages captured, as described on Section 5.2, demonstrate the correct operation of the UPnP devices developed during this project. In fact, the message flow is equal to the message flow described on the UPnP guidelines (see Section 2.1.2). The Web Interface allows the end user to stream music, video, and image items. It also allows the user to see the broadcast TV channels and set up the recordings for later viewing.

Chapter 6

Conclusion

6.1 - Overview

The result of this thesis is a proof-of-concept prototype of the *Virtual PVR* concept. The first part of the work consisted on studying the existing Service Discovery and Remote Management standards, Set-top-box operation and Media Centers available in the market. This study was important to realize what are the potentialities and limitations of the existing solutions. Next it was necessary to define an integrated solution to the *Virtual PVR* system. The specification will allow future developments of the system and integration with other modules.

The existing PVR systems allow users to record the TV programs in digital format in a hard disk. Nowadays, information is shared among digital devices however, the existing PVR systems are solutions restricted to the STB hard disk. This means that if someone is viewing a TV show and other user wants to view a recorded show, it is not possible. The developed *Virtual PVR* prototype takes advantage of UPnP and DLNA capabilities to stream the recorded show to an external disk available at the moment. It is also possible to render the contents in any UPnP display available in the Home Network. The developed system allows users to stream music, video and image items available in different devices without having to perform a specific configuration. The great advantage of the system is that it can be accessed seamlessly and remotely through well-established web interfaces which means, that the user only needs to access a terminal with a WWW browser to interact with the system. Nowadays there are a wide range of devices with a WWW browser available in the Home Network like TVs, game consoles, laptops, smartphones, etc. However the system has some limitations that escape from the scope of this project. The Live TV service and PVR are conditioned by the existence of only one TV tuner. This means that, only one user is able to see a live TV channel and if a recording is set up to the same time, the user will not be able to watch a different channel. Other limitation resides on the security of the solution. The proof-of-concept prototype of the *Virtual PVR* concept is fully based on UPnP devices and services. The UPnP purpose is to allow service discovery and configuration. It is not concerned to be a secure remote management protocol and therefore it is a mechanism to be used within a secured Home Network.

6.2 - Future Work

The solution, specified in Chapter 3, is composed by UPnP devices and services, a Web Interface and CWMP modules. The integration of the *Virtual PVR* prototype with the CWMP modules would enables the Telco to provide new services. Developing a root framework for CWMP would be a longstanding work which represents itself a Master Thesis. Nowadays, social networks are a common reality. The *Virtual PVR* system could provide rating and recommendation services as well as Instant Messaging services which could allow users to share and comment TV shows, the favorite movie or the new music album that was released yesterday.

References

[1] Allegro Software Development Corporation, "A UPnP / DLNA Overview," 2006.

[2] UPnP Forum, "UPnP™ Device Architecture 1.1," 2008.

[3] John Ritchie, "UPnP AV Architecture:1," UPnP Forum, 2002.

[4] Digital Living Network Alliance, "DLNA Overview and Vision Whitepaper," 2007.

[5] DSLHome-Technical Working Group, "TR-069 CPE WAN Management Protocol," Broadband Forum, 2004.

[6] DSL Home-Technical Working Group, "TR-111 Applying TR-069 to Remote Management of Home Networking Devices," Broadband Forum, 2005.

[7] DSL Home Technical Working Group, "TR-140 - TR-069 Data Model for Storage Service Enabled Devices," Broadband Forum, 2007.

[8] DSL Home-Technical Working Group, "TR-135 Data Model for a TR-069 Enabled STB," Broadband Forum, 2007.

[9] Home Gateway Initiative, "Home Gateway Technical Requirements: Residential Profile," 2008.

[10] Home Gateway Initiative, "HGI guideline paper - Remote Access," 2008.

[11] Alexander Sirotkin, "Hacking Embedded Linux Based Home Appliances," Metalink Broadband, 2007.

[12] G. O'DRISCOLL, *The Essential Guide to Digital Set Top Boxes and Interactive TV*.: Prentice Hall, 2006.

[13] B.Sundareshan, "Digital Set Top Box (STB) - Open Architecture/Interoperability Issues," BECIL,.

[14] (2009, January) Windows Embedded. [Online]. http://www.microsoft.com/windowsembedded/en-us/products/wexpe/default.mspx

[15] (2009, January) Linux Online. [Online]. http://www.linux.org/

[16] (2009, January) Apple Mac OS X Leopard. [Online]. http://www.apple.com/macosx/technology

[17] (2009, January) Android Official Website. [Online]. http://code.google.com/android/

[18] (2009, January) ITV Dictionary. [Online]. http://www.itvdictionary.com/set-top_box.html

[19] (2009, January) MythTv Forum. [Online].

https://help.ubuntu.com/community/MythTV

[20] (2009, January) XBMC. [Online]. http://xbmc.org/

[21] (2009, January) My Media System. [Online]. http://mymediasystem.org/

[22] (2009, January) Freevo Home Theater Platform. [Online]. http://freevo.sourceforge.net

[23] (2009, January) Neuros Technology Wiki. [Online]. http://wiki.neurostechnology.com

[24] (2009, January) Boxee Project. [Online]. http://www.boxee.tv/

[25] (2009, January) Dream Multimedia. [Online]. http://www.dream-multimedia-tv.de/

[26] (2009, January) Scientific Atlanta. [Online]. http://www.sciatl.com/

[27] Giovani Spagnolo, "FLOSS Media Centers State of the Art," Telematics Freedom Foundation, 2008.

[28] (2009, January) VideoLAN - VLC media player. [Online]. http://videolan.org

[29] (2009, January) MPlayer. [Online]. http://www.mplayerhq.hu

[30] (2009, May) Linux TV. [Online]. www.linuxtv.org

[31] (2009, May) TV Grab DVB. [Online]. http://darkskiez.co.uk/index.php?page=tv_grab_dvb

[32] Managing TV with XMLTV. [Online]. http://www.onlamp.com/pub/a/onlamp/2005/12/08/xmltv.html

[33] (2009, January) FFmpeg Project. [Online]. http://www.ffmpeg.org/

[34] (2009, January) Video Disk Recorder. [Online]. http://www.cadsoft.de/vdr/

[35] (2009, June) LyngSat. [Online]. www.lyngsat-com

[36] Ellislab Network. (2009, May) CodeIgniter. [Online]. http://codeigniter.com

[37] (2009, June) Start Up Web 2.0. [Online]. http://www.sellersrank.com/php/cakephp-codeigniter-benchmark/