Framework for Personal TV

André Claro[1], Paulo Rogério Pereira[2] and Luís Miguel Campos[3],

[1] Instituto Superior Técnico, Taguspark Campus, Av. Prof. Dr. Cavaco Silva, 2744-016 Porto Salvo, Portugal
[2] Inesc-ID, Instituto Superior Técnico, Technical University of Lisbon, Rua Alves Redol, 9. 1000-029 Lisboa, Portugal
[3] P.D.M.&F.C, Av. Conde Valbom n. 30, Piso 3, 1050-068 Lisboa, Portugal
andre.claro@ist.utl.pt, prbp@inesc.pt, luis.campos@pdmfc.com

Abstract. This paper proposes a study of the IPTV world, focusing on network and system architectures, video codecs, network protocols, services and quality assurance. Based on this study, a new framework for Personal TV was developed. This new system is designed primarily to provide new personalized services to the user. In the architecture of this framework there are three main elements: Clients, Aggregators and Producers. The most important element is the aggregator that provides all video contents from the producers to its clients. The client has the possibility of creating his own channel that is sent to the network through its aggregator, of creating customized channels that can be viewed by other clients, among other features. The architecture designed and developed is based on new and studied concepts. It was tested to prove its viability, to assess its performance and to draw conclusions about its scalability, based on functional tests, compatibility tests and performance tests.

Keywords: IPTV, IP, Television, Services, Personalization, Availability.

1 Introduction

Telecommunications companies are investing in ways to offer its customers triple-play services and some of them are starting to think about integration with mobile services. This means that there is a convergence at the network level for IP (Internet Protocol) to offer the different services. In an all-IP network, personalized and interactive services can be provided very easily. IPTV (Internet Protocol Television) is the technology responsible for this convergence and this technology will bring a variety of new business opportunities.

Based on the growing number of users of IPTV and also of VoIP (Voice over IP), a growing migration of existing technologies to IP technology is expected. Associated to this convergence, new services and new business opportunities may arise. IPTV emerges as the future of fixed or mobile television. Many of the carriers around the world are investing millions in technology in order to increase profit from their networks and to compete with rivals over providing a wider range of services. There are other companies which invest in Internet TV, which is based on the same concepts of IPTV, but with a best effort service.

The main goal of this paper is to investigate and study the protocols and concepts involved in IPTV with the aim of widening horizons and designing a Framework for Personal TV, e.g. personalized television services, where the user is a key element in the architecture.

The user can create its own personal channel and personalized channels. Channels generated by users can be distributed through the network as any other channel. The architecture is composed of three main elements: Clients, Aggregators and Producers. The aggregators are interconnected in a network of aggregators, with fault-tolerance and load-balancing, that distributes the TV contents for all clients. This architecture can operate on a private network using unicast or multicast, or on the Internet, where bandwidth is more limited and only unicast is available.

The developed architecture was tested, and results were analyzed to draw conclusions about the performance and functionality of the system.

2 State of the Art

The telecommunications companies are evolving their networks to all-IP networks. IPTV is one of the concepts and technologies used for this convergence. "IPTV" is the secure delivery of high-quality multichannel television, on-demand video and related multimedia contents, via a dedicated network, to a consumer electronics device such as a set-top box, a computer, or a portable device that is served by broadband IP access. This is in contrast to Internet TV (or Internet Video) – be it live TV or video-on-demand – brought to users via the open Internet on a best-effort basis, where multicast is not available [1][2][3].

2.1 Network and System Architectures

The ITU-T, through its study group on IPTV, presents a functional architecture based on existing technologies and concepts or on NGN (Next Generation Network). Fig. 1 shows the four functional domains [4].

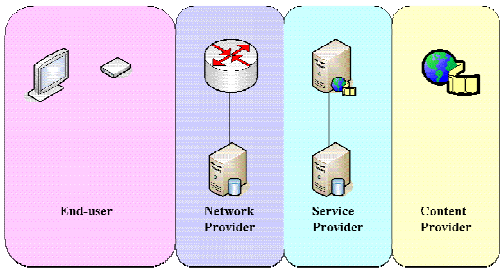

Fig. 1. Functional domains of IPTV according to ITU-T.

In an alternative model proposed by the BSF (Broadband Services Forum), an

IPTV system is based on the following elements: Video Head-end, Network Core, Network Access and the Home Network [5].

Content Distribution Networks (CDN) are a good way for IPTV operators to spread their services offer outside their networks and consequently expand the list of channels and services available. CDNs use "multicast" on the application layer, aiming independence from the network infrastructure. The architecture Prism (*Portal Infrastructure Streaming Media*) shown in Fig. 2 is an example of a CDN [6].

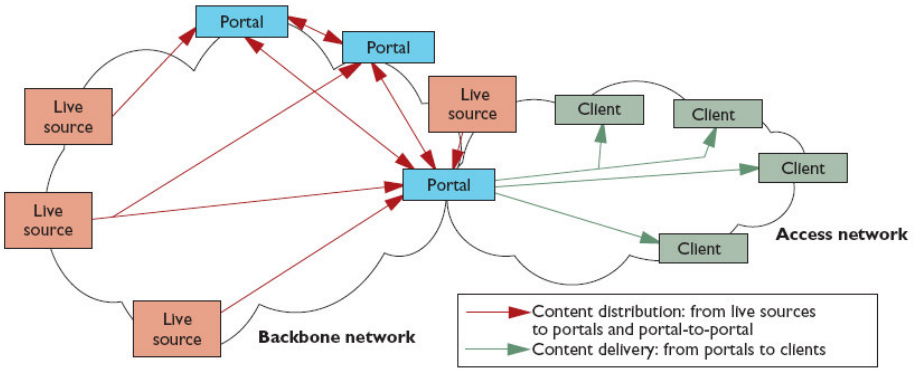

Fig. 2. Portal Infrastructure Streaming Media architecture.

Peer-to-peer IPTV networks are another kind of architecture, more suitable to the public Internet. In peer-to-peer networks, client nodes are also servers so as to distribute the server load over all nodes in the network.

2.2 Video Codecs

Video encoding is a major factor required to offer IPTV, given the constraints at network level. H.264 is the video compression standard that is being adopted in new solutions for IPTV. It tends to replace MPEG-2 standard that is still widely used in some systems. VC-1 is also a valid alternative and competitor to H.264, by presenting a similar compression and image quality, amongst other advantages [7].

2.3 Network Protocols and IPTV Services

The transport of encoded video/audio in IP networks can be made based on several technologies. The main ones are: RTP (Real Time Protocol) [8] and MPEG-2 Transport Stream (TS) [9]. The use of native RTP has several advantages over MPEG-2 TS: bandwidth preservation, flexible audio/video stream selection, better resistance to errors, improved quality of service feedback through the use of RTCP and improved services [10].

IGMP is a multicast protocol used by IPTV, which allows minimizing packet

replication in the network when distributing the same contents to a group of destinations. Generally, IGMP version 2 or 3 is used [11][12].

Other protocols used by IPTV are: RTSP (Real Time Streaming Protocol) [13], a signaling protocol for VoD services; SAP (Session Announcement Protocol) [14], a multicast session announcement protocol; SIP (Session Initiation Protocol) [15], a signaling protocol used to control sessions between users; SDP (Session Description Protocol) [16], a protocol for describing the session characteristics, used in other protocols such as SAP, RTSP, etc.

Based on these protocols, the following services may be offered for IPTV [17]: Linear TV (audio, video and data), Linear TV with Trick Modes (pause, replay), Multi-View service, Time-shift TV, Pay Per View (PPV), Video/TV on Demand (VoD), Download Based Video Content Distribution Services (Push VOD), Content download service, Personal Video Recording (PVR) service (network or client-based), Interactive TV (iTV), Linear Broadcast Audio, Music on Demand (MoD) including Audio book, Learning (education for children, elementary, middle and high school students, languages and estate, etc.), Information (news, weather, traffic and advertisement etc.), Commerce (security, banking, stock, shopping, auction and ordered delivery, etc.), Telecommunication (e-mail, instant messaging, SMS, channel chatting, VoIP, Web, multiple), Entertainment (photo album, games, karaoke and blog, etc.).

2.4 Quality Assurance Service

In order to ensure the services described above, it is necessary to have the best architecture, the best video codecs and protocols for efficient data transport over the network.

There are many components to guarantee the quality of service (QoS). For the management of QoS, the following components are important: management of subscribers and content, management of the capacity of the network and systems, service level monitoring, image quality metrics, network monitoring and testing.

The concepts of QoS and quality of experience (QoE) are also important. QoE is a more complex concept compared to QoS which includes rates, delays, losses, latencies, etc. QoE includes the total end-to-end system effects and is measured based on user perception of the quality of the image. Fig. 3 shows the QoS and QoE domains [18].

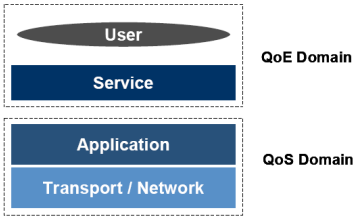

Fig. 3. QoS/QoE Domains.

3 Proposed Architecture

3.1 System Description

The designed solution is based on an end-to-end architecture, with the aim of providing new IPTV services. This system relies on network aggregators responsible for the distribution of audiovisual content to clients. This IPTV platform enables the management of multiple video sources by an operator. This architecture requires the existence of three entities, as shown in Fig. 4: Aggregators, Clients and Producers. It was built on the vision of ITU-T and of Content Distribution Networks (CDN).

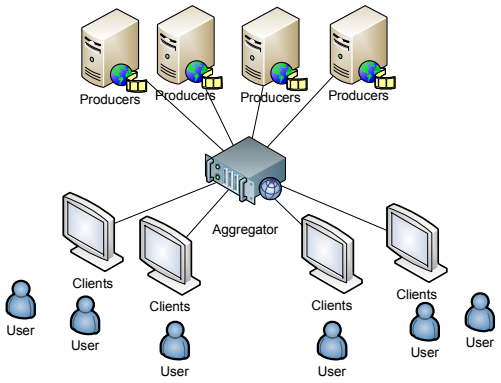

Fig. 4. Simple Architecture.

The aggregator is the most important element of the system, which in addition to distributing the content to other aggregators, offers the following new IPTV services to the client: Personal Channels and Personalized Channels. Personal Channels are composed of media generated by the user from a webcam or from a video file selection. Personalized Channels are channels produced by combining the channels available in the producers.

3.2 Aggregator Details

The Aggregator is composed by the main components shown in Fig. 5: Personal TV Management, Personal TV Streaming Server, Database, and Personal TV Administration Console. The figure also shows the types of interactions between the different components.

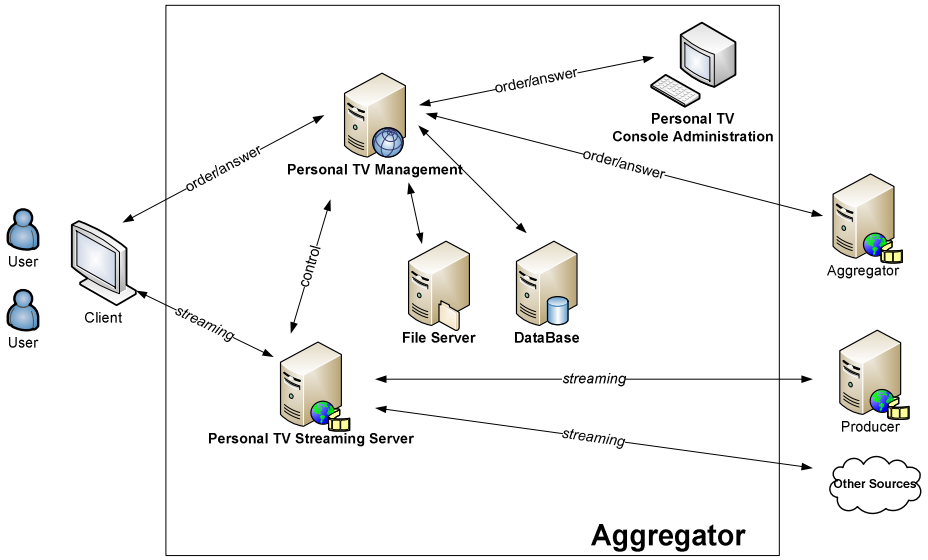

Fig. 5. Aggregator Details.

Personal TV Management

The Personal TV Management, which will be called "PTvM", is the main component of the aggregator, as shown in Fig. 6. It is responsible for all of its management and control. The main features are: communication with others aggregators; control of the streaming Server (PTvSS); communication with clients; management database; monitor applications; create/remove/change personalized channels; add/remove/ change producers; communication with billing systems and in interfacing with other systems.

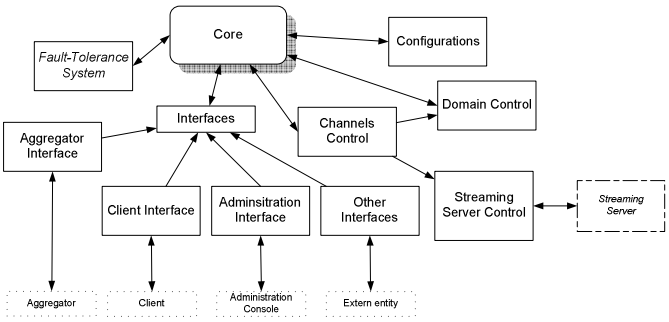

Fig. 6. PTvM Class Diagram.

Data Model

The domain shown in Fig. 7 represents the identities that are stored in a persistent database. A simple data model was drawn for the features developed.

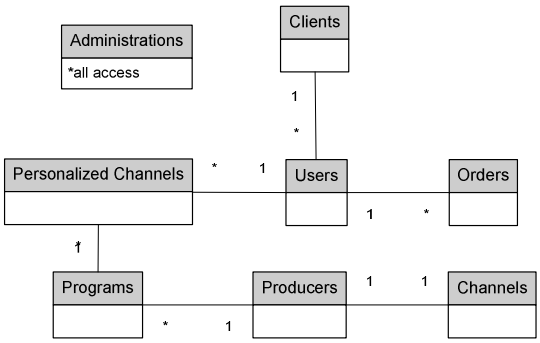

Fig. 7. PTvM Domain.

High Availability

To solve the problem of aggregator redundancy and load balancing, it was decided to introduce a High Availability solution in the architecture. In each geographical zone there are several Aggregators available. Instead of clients connecting directly to an Aggregator, they connect to a Load-Balancing Server, which forwards the requests to the Aggregator that is online and has higher available capacity.

Streaming Server

The Personal TV Streaming Server (PTvSS) is the server for content streams, which sends and receives content, from clients, producers and other aggregators. This element will support multiple protocols for transportation and various encodings, in order to achieve greater interoperability with other systems. It should also allow the transmission/reception in unicast and multicast.

3.3 Client

The Client is a simple entity, with a user interface and a video player, allowing the user to use all the functions provided by the aggregator to which it is connected. The Video Player allows the customer to send video streams to the network, working as a produced in order to offer the Personal TV service. The video streams generated may come from a webcam or from a selection of stored videos. Fig. 8 represents the client's class diagram.

3.4 Global Vision

This global system is shown in Fig. 9. It has aggregators as main entities, forming a network responsible for content distribution between producers and clients. Clients connect to an area of aggregators, through a load-balancing system that distributes clients through the aggregators of that area.

Producers only send streams of video to the aggregators. Producers get the flow of

video over any IP network, with or without of quality of service guarantees, or via satellite (DVB-S), or generate the flow from a file or DVD. The producer is a simple streaming server.

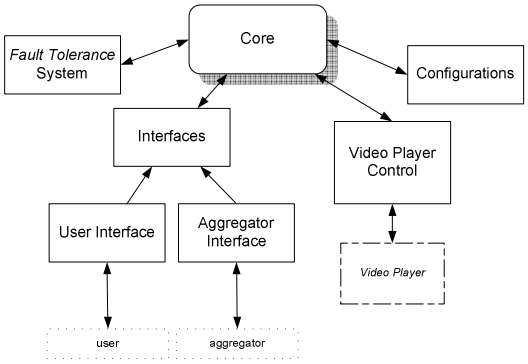

Fig. 8. Client Class Diagram.

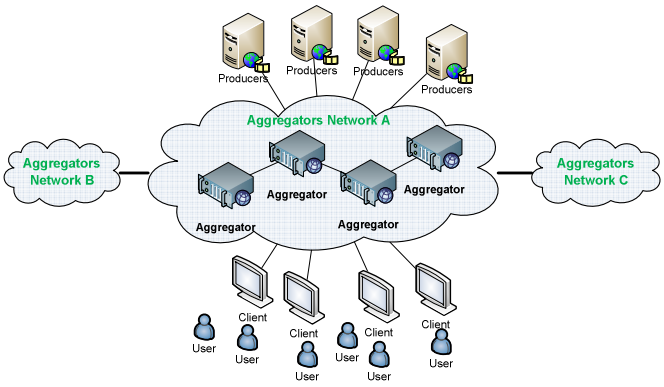

Fig. 9. Global IPTV Architecture.

4 Application Analysis

The following streaming servers were analyzed: Adobe Flash Media Streaming Server 3 [19], Darwin Streaming Server [20], Feng [21], Helix DNA Server [22], Live 555 Media Server [23], MPEG4IP [24], VLC [25] and Windows Media Server [26].

Based on the analysis carried out on those applications, it was decided to use VLC as a server for streaming and also as a client in the designed architecture referred in the previous section. This choice was due to several factors, including: the flexibility of being able to make changes as the application evolves; the lack of licensing costs;

compatibility with various platforms namely Linux, a free operating system. Another aspect that was considered was the number of updates and fixes that the development team regularly does, allowing constant improvements and increased functionality of the solutions implemented around them. This program has a free software license based on General Public License (GPL).

5 Implementation

The prototype developed is based on free applications. The majority of applications are based on GPL, which greatly facilitates its use and does not require any license. The development was carried out with the support of the Linux operating system, Ubuntu. It is a multi-platform solution and it is designed to run on other systems such as Windows and Mac OS X.

5.1 Aggregator

In the Aggregator, the main element is the Personal TV Management, which is responsible for the management and control of the entire IPTV system. This element controls the streaming server, the database and a file server. There is also a management console (Personal TV Console Administration), which enables remote management and administration of the aggregator. Fig. 10 shows the aggregator illustrating the connections between the various elements and the protocols used for each one.

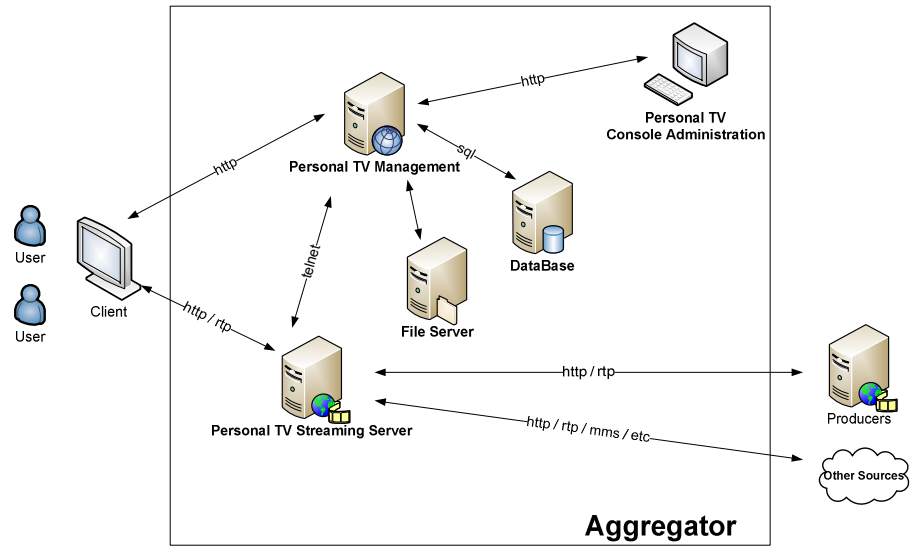

Fig. 10. Aggregator Details.

Personal TV Management (PTvM)

This Application Server manages everything that happens in the aggregator.

The following applications and libraries were used to develop the PTvM: Java (programming language), MySQL (database management), JWSDP [28] (Java Web Services Developer Pack, web services library), Hibernate [28] (database connection library), Apache Commons Net [29] (telnet library).

High Availability

The High Availability solution is implemented based on the LVS (Linux Virtual Server) [31] and the HAProxy [30] proxy, as shown in Fig. 11. Based on the LVS, the following applications were used: keepalived [28] and Heartbeat [29].

The keepalived application is responsible for the setup of virtual addresses. The Heartbeat application is responsible for verifying the status of the other load-balancing server. In case of failure of the primary server (master), the secondary server (slave) takes the lead, and sets the virtual address, through keepalived. The HAProxy redirects the requests for aggregators that are in operation and with lower loads.

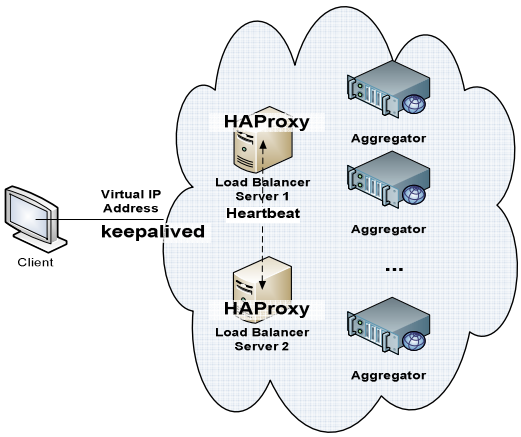

Fig. 11. High Availability Solution.

If there is any problem with the aggregator to which the client is connected, the load-balancing server forwards requests to another server that is available. This process is transparent to the Client.

5.2 Client

The client was also developed based on the Java programming language and the VLC video player. A user interface based on the command line was developed. The JWSDP library to program web-services communication was used to assure the communication with the aggregators. There is a fault-tolerance system for quick recovery in case of an error on the server. For personal channels, many configuration

parameters can be changed, for example: codec, rate and send mode (unicast or multicast).

5.3 Communication Diagrams

In this section, some of the most important concepts of the implementation of communication between components of the system are presented. There are several diagrams representing the communication between the components of the architecture: Client, Aggregator (PTvM and Streaming Server).

The communication between a Client and its Aggregator, to request the channel list and to select a channel to be viewed, is represented in Fig. 12.

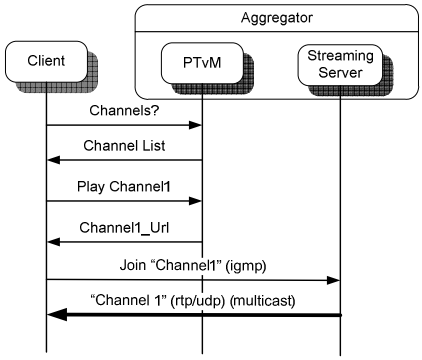

Fig. 12. Request of the Channel List and joining a Multicast Channel.

Fig. 13 and Fig. 14 represent the creation of a personal channel and a personalized channel, respectively. When creating a personal channel, the client sends a request to the aggregator and it asks the client for the video stream. In the personalized channel creation, the client sends new channel information and programs of the new channel to the aggregator. In both cases, after the channel configuration on the Aggregator, the channel information is propagated to the other aggregators.

6 Tests and Evaluation

Tests were designed to demonstrate the main features of the implemented solution, and assess system performance. The tests are divided into three types: functional tests, compatibility tests and performance tests.

The collection of data from tests was done with Cacti [34] (software for monitoring computer networks and systems), TOP (Linux application) and a software module developed for the PTvM.

Tests were conducted in a laboratory using twelve computers, connected to an Ethernet switch (Cisco Catalyst 2950) at 100 Mbps. The computers used had a 3 GHz

processor with 1 GB of RAM (Random Access Memory) and Ubuntu Linux operating system. They were prepared to serve as clients and aggregators.

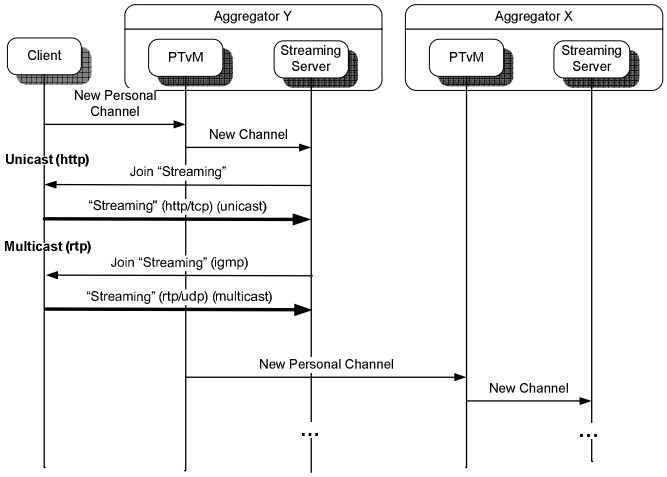

Fig. 13. New Personal Channel.

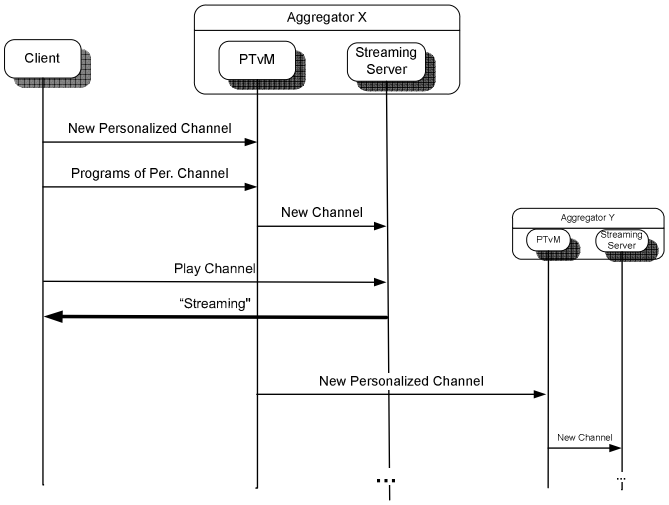

Fig. 14. New Personalized Channel.

6.1 Functional Tests

First, the features provided by Aggregators to Clients were tested: view a channel, create a personal channel, create a personalized channel, etc. These tests were based

on a simple architecture, with a client and an aggregator that is linked to several producers, as shown in Fig. 15.

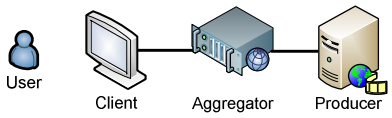

Fig. 15. Functional Tests Scenario.

Based on the tests carried out, all the features have been well implemented, both on the client side and on the server side. In the section on performance tests, the performance of each function will be analyzed.

6.2 Compatibility Tests

All implemented features are compatible with Linux, Windows and Mac OS X operating systems, and only some parameter changes in streaming server execution are needed.

6.3 Performance Tests

Function Execution Time

The performance of the implemented features was tested by measuring their execution times. No concurrent function execution was tested. The results shown in Table 1 indicate that the time to open/change a channel takes on average 1 second. A personal channel takes on average 4.6 seconds to be available to be viewed by another client. For a personalized channel by category, the creation delay is between 4.1 and 4.35 seconds, with 95% confidence.

Table 1. Function Execution Times without Competition (seconds).

Function	Average	Standard Deviation
Open/Change Channel	1.030	0.177
Create Personal Channel	4.660	0.178
Create Personalized Channel	3.210	0.202
Create Personalized Channel by Category	4.210	0.228

Client Performance

The performance of the client in terms of CPU processing, memory used and network speeds was tested. The Client starts by receiving a video stream (at 2 Mbps) and after

sends a video stream (at 1 Mbps) for the network (Personal Channel). Fig. 16, Fig. 17 and Table 2 show the test results.

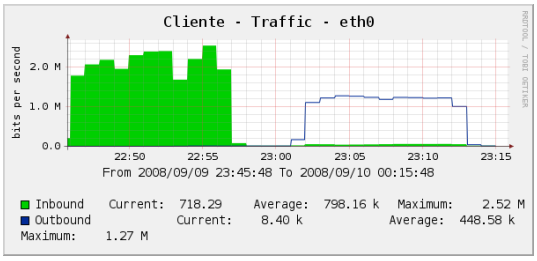

Fig. 16. Client Network Traffic.

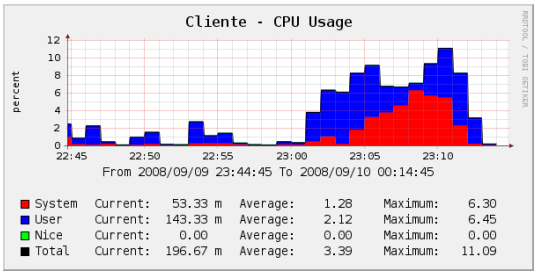

Fig. 17. Client CPU Usage.

Table 2. Client Application Used Memory (MBytes).

	Average	**Standard Deviation**
Receive Channel	20.19	1.53
Send Channel	22.08	0.71

It can be observed that the client requires a minimum computational load when it is receiving or sending a stream. Using a computer or embedded system with fewer capabilities, the developed client solution would continue to function properly without affecting the quality of the user's viewing experience. Even with increased resolution (2 or 4 Mbps), the increase of processing is not relevant.

Aggregator Performance

The performance of the Aggregator was tested with different clients in two parts: unicast streams and multicast streams to the clients. Fig. 18 to Fig. 21 show the test results. The aggregator sends a 2 Mbps video stream encoded with H.264. The memory used by the aggregator applications is presented in Table 3.

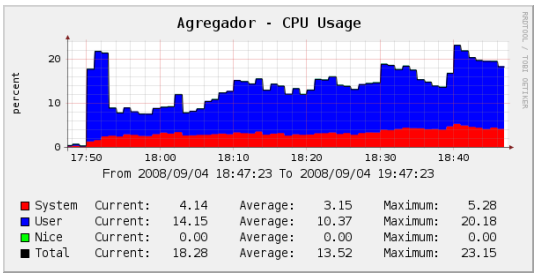

Fig. 18. Aggregator CPU Usage (unicast).

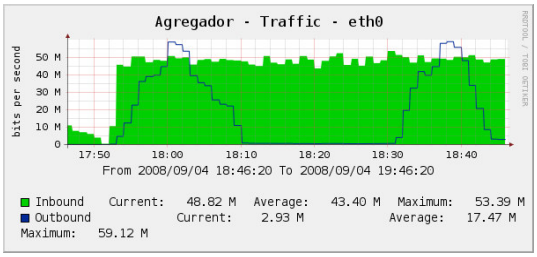

Fig. 19. Aggregator Network Traffic (unicast).

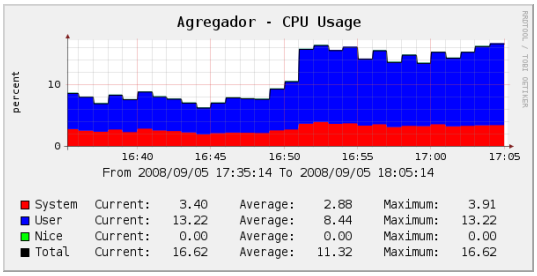

Fig. 20. Aggregator CPU Usage (multicast).

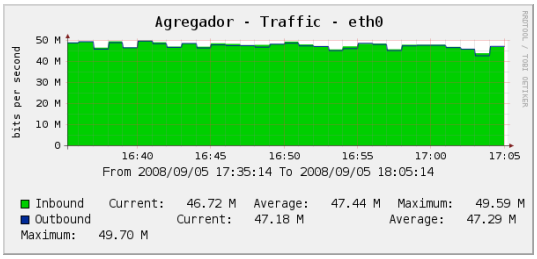

Fig. 21. Aggregator Network Traffic (multicast).

Table 3. Aggregator memory usage (MBytes).

	Clients	Average	Standard Deviation
Multicast	10	163.210	1.901
	20	167.029	1.131
Unicast	10	272.540	4.219
	20	274.695	4.658

Based on the results, it is seen that when the aggregator sends unicast streams, it needs more average processing than while sending multicast. The average speed of incoming traffic is identical in both tests because the Aggregator is connected to the same number of producers.

The processing increases with the number of producers linked to the customer as each corresponds to a channel stream being received. When the solution is working on multicast, less memory is used compared to the unicast case. This probably happens because there is a single socket for multicast, with a single thread, while in unicast there is a socket and a thread for each client.

Aggregator Communication

The communication time between Aggregators was measured. For this test, the personal channel functionality was used and the time necessary for a channel to be available on another aggregator was measured.

Based on the results, one aggregator takes on average 0.6 seconds to receive information of a new aggregator channel and to provide that channel to clients. With ten aggregators in series, the last aggregator takes on average 4.93 seconds with a variance of 0.012 seconds to receive and provide the channel contents to its clients. With aggregators in parallel, the time to propagate a new channel increases slightly with the number of aggregators connected and the increase is mainly due to the increase in CPU processing of the aggregators that send the information.

Failure Recovery

Client recovery in response to aggregator failure was also tested. In case of failure of an aggregator, its clients should quickly connect to another aggregator in the same zone and continue receiving the video streams. First, a network with only two aggregators, in which clients were all connected to the same aggregator was tested. Afterwards, a network with 6 aggregators was tested. Both topologies are shown in Fig. 22.

The results are shown in Table 4. It was observed that the average recovery time is about 1.5 seconds, when the number of client varies between 1 and 10, and the number of aggregators available is between 1 and 5.

The results are satisfactory, if such problems do not happen frequently, because 1.5 seconds without television service is a long time.

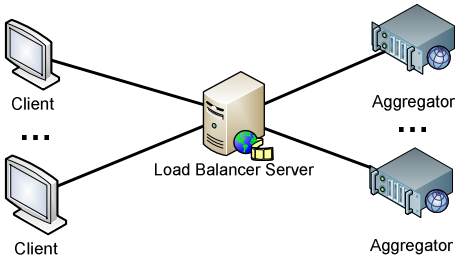

Fig. 22. Failure Recovery Test Scenario.

Table 4. Failure Recovery Times (seconds).

Aggregators at start (after)	Clients	Average	Standard Deviation
2 (1)	**1**	**1.37**	0.183
	10	**1.81**	0.3
6 (5)	**1**	**1.32**	0.132
	10	**1.48**	0.148

Scalability of the Solution

The system CPU processing at the aggregator depends more on the number of producers than on the number of clients connected. Fig. 23 shows an estimate of CPU processing versus the number of producers. The increase is linear.

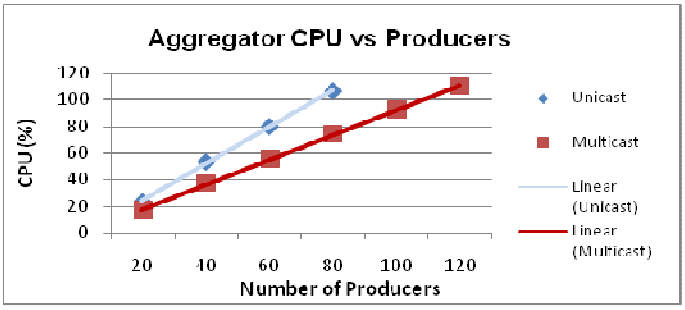

Fig. 23. Estimate of CPU vs Producers.

From Fig. 23 it can be observed that the maximum number of producers connected in unicast may be just over 75 per aggregator for the type of computer used (3 GHz). In multicast, that number is higher: around 110 producers.

The aggregator memory necessary is mainly related to the number of clients. Thus we can conclude that in multicast the memory occupation is lower than for unicast. Fig. 24 shows that the memory used grows approximately exponentially with the

number of clients.

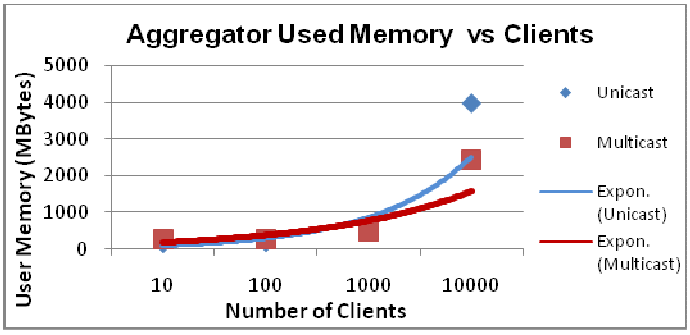

Fig. 24. Estimate of Used Memory vs Clients.

It can be concluded that the network scalability depends on the method used by the aggregator to send the video streams to the clients. If the method is unicast, the number of users is limited by the maximum speed of the network. With multicast, the number of channels transmitted depends on the number of channels in the aggregator and does not depend on the number of clients.

Tests Conclusions

The number of clients supported by an aggregator depends on the server hardware where the aggregator is operating. But the network's bandwidth also limits the scalability. The use of multicast can solve part of this problem if the number of channels is not very high.

In order to distribute the CPU processing for the developed solution, the following steps may be done: put PTvM and the Streaming Server on different computers and also increase the number of computers for each of these applications (both redundancy and performance of the system are increased).

7 Conclusion

This paper presents a study of the concepts and technologies used in IPTV: architectures, network protocols, video codecs and others concepts. The IPTV area is having a large expansion and development. A Framework for Personal TV was developed, based on an architecture that assumes the existence of three entities: Clients, Aggregators and Producers. This architecture is designed to be scalable, fault-tolerant and provide customized audiovisual content and new services to clients.

A prototype system for IPTV was designed, implemented and tested, with the aim of providing the client the following features: display of real-time channels, creating custom channels and personal channels. This opens a door to new services implementations. The prototype is designed based on the study of IPTV architectures,

especially the view of ITU-T and CDN.

Much of the system implementation was performed in the aggregator, which has the following features: serving clients with the best quality and performance, communication with other aggregators to share information and video content, user statistics and others features. The developed solution can run on a private network with guaranteed quality of service or on the Internet. In this case, it is always limited by the speed of the network and of its connection to the user. The absence of guarantees of quality of service can also restrict the flows and impose limitations.

Based on the prototype testing, it is concluded that the system responded as expected. With the machines used (CPU at 3GHz), customers get times in the order of 1 second to open or change channels and the times of recovery observed in a client in the event of an aggregator failure are between 1 and 2 seconds. To create a personalized or personal channel, a time between 3 and 5 seconds is needed until the channel becomes available. It was found that an aggregator can support 75 producers (each corresponding to an IPTV channel) in unicast, or 110 producers in multicast, because the processing and memory required in unicast is higher than in multicast. As multicast is not supported by all networks, unicast should be used for sending video streams. This method makes the scalability of the solution, and the number of users more limited. Client applications consume at most only 12% of CPU time (at 3GHz) and 25 MB of RAM. The time for sharing information between aggregators was found to be less than 1 second for a new channel to be available in neighboring aggregators. The solution can scale to any number of channels by the use of a network of aggregators by zone. The network of aggregators can be connected to other area zones to share resources between them.

Future Work

As future works, a list of activities expected to strengthen the idealized concepts and the features of the system is presented. There is a need to integrate the solution developed with a cache system. Other features needed are: graphical interface for the client and a secure architecture to support the framework developed by using secure communication channels for communication between the elements of the system.

The solution can also be integrated with other systems, such as: DRM / CA (Digital Right Management / Conditional Access System), AAA (Authentication, Authorization and Accounting), BSS (Business Support System), CRM (Customer Relationship Management), Order Management (management applications), Network Inventory/Service Activation. Integration with other systems can be the base for offering new services such as: VoIP, video conferencing, customized advertising to each user and others services.

The possible evolution of a solution based on open source software for a professional solution based on a commercial streaming server could also be a topic for future work. Naturally, the concept of the designed architecture should be kept: Clients, Aggregator and Producer.

IPTV should be considered seriously because with this concept and technology, television will evolve more over the next five years than it evolved in the last twenty

years.

References

1. William Cooper and Graham Lovelace, “IPTV Guide - Delivering audio and video over broadband”. Informitv, 2006.
2. IPTV-news.com. IPTV News, The Magazine for all your ipTV information, 1st edition. November 2007. http://www.iptv-news.com/
3. Robin Good. IPTV vs. Internet Television: Key Differences. April 4, 2005. http://www.masternewmedia.org/
4. Focus Group. FG IPTV-DOC-0115 - Working Document: IPTV Architecture. ITU-T, 31 July 2007.
5. BSF. IPTV Explained - Part 1 in a BSF Series. http://www.broadbandservicesforum.org/
6. Cranor, Charles, Green, Matthew and Kalmanek, Chuck. Enhanced Streaming Service in a Content Distribution Network. IEEE Internet Computing, July 2001. pp. 66-75.
7. Sunna, Paola. AVC/H.264 - An Advanced Video Coding System for SD and HD broadcasting. EBU Technical, 2005.
8. Schulzrinne H.; Casner S. RFC 3550 - RTP: A Transport Protocol for Real-Time Applications. IETF. July 2003.
9. DVB-ETSI. ETSI TS 102 034: Transport of MPEG 2 Transport Stream (TS) Based DVB Services over IP Based Networks. ETSI, March 2007.
10. ISMA. Planning the Future of IPTV with ISMA. http://www.isma.tv/technology/white-papers/ISMA-IPTV_whitepaper_V11_2006-09-14.pdf
11. Juniper Networks. Introduction to IGMP for IPTV Networks. October 2007. http://www.juniper.net/solutions/literature/white_papers/200188.pdf
12. Fenner, W. RFC 2236 - Internet Group Management Protocol, Version 2. IETF, November 1997.
13. Schulzrinne, H. and Rao, A. RFC 2326 - Real Time Streaming Protocol (RTSP). IETF, April 1998.
14. Handley, M. and Perkins, C. RFC 2974 - Session Announcement Protocol. IETF, October 2000.
15. Rosenberg, J. and Schulzrinne, H. RFC 3261 - SIP: Session Initiation Protocol. IETF, June 2002.
16. Handley, M. and Jacobson, V. RFC 4566 - SDP: Session Description Protocol. IETF, July 2006.
17. IPTV Focus Group. FG IPTV-DOC-0116 Working Document: IPTV Service Scenarios. ITU-T, July 2007.
18. Kishigami, Jay. The Role of QoE on IPTV Services. Ninth IEEE International Symposium on Multimedia, 2007.
19. Flash Media Streaming Server. Adobe. http://www.adobe.com/products/flashmediastreaming/
20. Darwin. Darwin Streaming Server. http://dss.macosforge.org/
21. Feng. the RTSP/RTP streaming server. http://www.lscube.org/projects/feng
22. Hellix. Hellix Community. https://helixcommunity.org/
23. Live555. LIVE555 Media Server. http://www.live555.com/
24. MPEG4IP. MPEG4IP: Open Source, Open Standards, Open Streaming. http://www.mpeg4ip.net/
25. VLC. VideoLanClient. http://www.videolan.org/vlc/
26. Windows Media Server. Microsoft WMS 9. http://www.microsoft.com/windows/windowsmedia/forpros/server/server.aspx

27.Java Web Services Developer Pack. http://java.sun.com/webservices/downloads/previous/
28.Hibernate. http://www.hibernate.org/
29.Apache Commons Net. http://commons.apache.org/net/
30.LVS. Linux Virtual Server. http://www.linuxvirtualserver.org/
31.HAProxy. HAProxy - The Reliable, High Performance TCP/HTTP Load Balancer. http://haproxy.1wt.eu/
32.KeepAlived. http://www.keepalived.org/
33.Heartbeat. http://linux-ha.org/
34.Cacti. Cacti - The Complete RRDTool-based Graphing Solution. http://www.cacti.net/