Escola Tècnica Superior d'Enginyeria
Departamento d'Arquitectura de Computadors
i Sistemas Operatius

P2P-VoD on Internet: Fault Tolerance and Control Architecture

Thesis submitted by Rodrigo Godoi under supervision of Porfidio Hernández Budé in fulfilment of the requirements for the degree of Doctor per the Universitat Autònoma de Barcelona.

Barcelona, July, 2009.

P2P-VoD on Internet: Fault Tolerance and Control Architecture

Thesis submitted by Rodrigo Godoi in fulfilment of the requirements for the degree of Doctor per the Universitat Autònoma de Barcelona. This work has been developed in the Computer Architecture and Operating Systems Department (CAOS) of the Universitat Autònoma de Barcelona, inside the program: PhD in Informatics, option A - Computer Architecture and Parallel Processing. The work was advised by Dr. Porfidio Hernández Budé.

Barcelona, July 2009

Advisor:

Dr. Porfidio Hernández Budé.

"My passionate sense of social justice and social responsibility has always contrasted oddly with my pronounced freedom from the need for direct contact with other human beings and human communities. I gang my own gait and have never belonged to my country, my home, my friends, or even my immediate family, with my whole heart; in the face of all these ties I have never lost an obstinate sense of detachment, of the need for solitude - a feeling which increases with the years. One is sharply conscious, yet without regret, of the limits to the possibility of mutual understanding and sympathy with one's fellow-creatures. Such a person no doubt loses something in the way of geniality and light-heartedness; on the other hand, he is largely independent of the opinions, habits, and judgments of his fellows and avoids the temptation to take his stand on such insecure foundations."

Albert Einstein in '*The World as I See It*'.

Acknowledgments

I would like to thank the Computer Architecture and Operating Systems (CAOS) department for giving me the opportunity to investigate at the *Universitat Autònoma de Barcelona*; thank you for providing a good environment, with all the conditions and resources necessary for the research.

To Dr. Porfidio Hernández Budé, thank you for the guidance and collaboration in the development of this work.

I express my gratitude to all members of the CAOS department, who helped in this journey in some way.

Especial thanks to Paula Gastal, for her unconditional support and infinite patience. Finally, I am very grateful to my family and friends, who love me just as I am.

Preface

A Video on Demand (VoD) system provides multimedia content to a set of clients in independent manner; users connect to the system, choose the content to view and start enjoying the service at any given moment. The video is down-streamed to the client, who receives, decodes and displays the content always expecting guaranteed Quality of Service (QoS) from the system. One of the main goals in designing VoD services is to support a great number of concurrent requests generated by geographically distributed clients; VoD systems must achieve a feasible large-scale and high-quality service with the lower costs and fewer deployment restrictions.

Recently, multimedia streaming distribution in the Internet presented a spectacular growing. The Internet is the most popular environment of connected users and is deployed throughout the world. Owing to the public and global scale features of Internet, it has become the most important environment to deploy large-scale Video on Demand service (LVoD).

Owing to the limitations of centralised server-client model, Peer-to-Peer (P2P) and multicast approaches are widely applied in the multimedia distribution to improve system scalability and performance by sharing resources. P2P is based in the free cooperation of equals in view of the performance of a common task; it takes advantage of available resources at the end host side (storage, content, bandwidth, power processing etc.). The multicast is a communication strategy where a sender has the capability to transmit information that can be received concurrently by a group of interested destinations. Nevertheless, P2P and multicast paradigms add new issues in the design of Internet VoD services. Peers are heterogeneous in their resources and act by their own free will, coming and leaving the system at any time; the lack or the change of data source provoked by peer faults strongly affects the QoS in VoD systems based in P2P and multicast techniques.

Fault tolerance has become a major issue in P2P-based VoD services in order to guarantee QoS. The fault tolerance mechanism is achieved through the exchange of control messages; moreover, the failure treatment is time limited for providing error absence and consequently maintaining the QoS. A good control scheme is needed and its design must be careful owing to the soft real-time restriction of multimedia service and the overhead imposed on the system.

This thesis presents a Fault Tolerance Scheme (FTS) that works by constructing a backup system in a distributed manner, based in own peers' capabilities. The FTS is designed to organise a small set of peers to store portions of the multimedia files statically in a buffer called the 'altruist buffer'. The clients that make up the distributed backup collaborate in system fault tolerance mechanism by reserving buffer space and upload bandwidth capacity; the selected peers form a Fault Tolerance Group (FTG).

Results show that the control mechanism has great impact over the system and demands a caution design; the proposed Fault Tolerance Scheme collaborates to reduce the overhead imposed on the system and is able to achieve low response times in dealing with failures; this improves user experience by reducing start-up delays and guarantees a better usage of buffer resources. The FTS also distributes the control tasks providing reliability and robustness to the VoD system.

Following, we present the organization of this thesis as well as a brief description of the content of each chapter.

Chapter 1

This chapter presents an introduction to VoD systems by describing their main characteristics, requirements and constraints. The Peer-to-Peer and Multicast paradigms are described presenting their classifications, protocols and topologies. The Internet features are exposed, considering the deployment of VoD service in such infrastructure and the current available multimedia applications. Relevant works in the area are evaluated showing how they deal with many issues of P2P and multicast usage in video systems. In the following, the control and fault tolerance mechanisms as well as their importance in the design of VoD systems are discussed. Finally, we summarise the observed topics and present the goal of this thesis.

Chapter 2

The Chapter 2 contains the definition of the system architecture considered feasible to the VoD deployment over the Internet. Here we also introduce and describe the PCM/MCDB and P2Cast service policies used in the development of this work. Subjects related to the Failure Management Process adopted as background in the study as well as the proposed Fault Tolerance Scheme finalize this chapter.

Chapter 3

This chapter holds the analytical model developed in this work for representing the Load and Time costs of the Failure Management Process. In the chapter's sessions equations related to detection, recovery and maintenance phases are presented and discussed; such equations comprehend VoD architectures applying IP Multicast and mesh-based P2P as well as tree-based P2P and Application Layer Multicast implementation. The chapter ends with the description of the expressions that govern the Fault Tolerance Scheme and the equations used to assess its performance.

Chapter 4

In this chapter we introduce the simulation environment for Load and Time costs evaluation, called the VoDSim. We present the simulation model characteristics, the probability functions adopted in the modelling process as well as some implementations features. The new functionalities added to the VoDSim during the development of this work and the analyses of the implementation are also showed here.

Chapter 5

Chapter 5 exposes the experimental evaluation carried out to assess the Load and Time costs as well as the proposed Fault Tolerance Scheme. We define the experimental environment, its settings and some parameters used in the analysis. Initially we compare results obtained through the analytical and simulation models in order to validate the developed expressions. In the following we verify the control relevance in the system, confirming its great importance. The performance of the Fault Tolerance Scheme is assessed by the behaviour of Load and Time costs in the Failure Management Process. Finally, the Fault Tolerance Scheme service performance is evaluated with respect to the construction time and capacity in handling simultaneous failures.

Chapter 6

This chapter presents the main conclusions obtained from this work and the open lines for future research that can be exploited in order to expand knowledge of designing VoD systems and improve presented solutions.

.

Contents

Chapter 2 – The Fault Tolerance Scheme for Internet P2P-VoD Service

Chapter 3 – Analytical Models for the Failure Management Process and Fault Tolerance Scheme

Chapter 4 – Simulation Environment for Load and Time Costs Evaluation

Chapter 5 – Experimental Evaluation

Chapter 6 – Conclusions

List of Figures

List of Tables

Chapter 1

Video on Demand Systems

1.1 Introduction

Video on Demand (VoD) systems are intended to provide multimedia content to a set of clients which may request service at any given moment. A VoD system is basically composed of three categories of elements: servers, clients and interconnection network. Multimedia content are downstreamed to clients, who receive, decode and display the video information, always expecting a continuous and synchronised visualisation during the streaming session. Some examples of VoD services currently offered on the market are Amazon video on demand [1], Telefonica Imagenio [2], ONO Video Club [3] and Broad & TV [4]; all of them are based on the server-client paradigm.

The design of a scalable VoD architecture is complex and hard to attain. Different areas of knowledge are related to system design such as data compression and cryptography, real-time operating systems, network communication protocols or even social studies to evaluate clients' behaviour.

In the last few years video streaming applications have awakened intense interest in academic and industrial areas. This tendency exists because VoD represents an attractive commercial service to be offered in dedicated environments which provide service to a restrict number of clients, and is much more interesting if extended to large-scale environments such as the Internet.

The distinct environments where VoD systems can be implemented show a drastic change in servers, clients and network characteristics. Dedicated environments have well-known resources, making the prediction of requirements quite simple. Therefore, the design and implementation of a VoD system in a dedicated environment tends not to be very complex. This kind of VoD system has obtained great success and can be observed in hotels, planes and schools.

The premises used in design and implementation of dedicated systems are not suitable, however, to provide VoD services in huge environments. A large-scale Video on Demand service (LVoD) has reference to systems which represent a great number of clients, who are

geographically distributed and ask for diverse multimedia contents. Clients can be dispersed over wide areas such as metropolitan regions, countries or even a continent. Therefore, the Internet has become the most important environment for deploying large-scale video services. The goal of the LVoD system is to provide multimedia content which always guarantees quality of service (QoS); owing to system dimension, however, many new constraints play important roles in its design, such as scalability (i.e., clients' requests and catalogue size) or elements diversity.

1.2 VoD Main Features

In this topic we describe the main characteristics of VoD systems. We present a brief description of multimedia contents as well as of the elements which compose the system. The service requirements are discussed with the service schemes and architectures used to system implementation.

1.2.1 Multimedia contents

Multimedia contents have a unique property that distinguishes them from traditional contents. Video content is formed by sequential images called frames. Furthermore, it combines various modalities of data (images, audio, multilingual text), multiple abstraction levels (raw data, features, semantics, metadata) and multiple resolutions (image thumbnails, video key-frames). These characteristics impose a high storage requirement on digital video content. Also, to provide a video session the images have to be displayed in correct order and respect the exact time of exposition.

The encoding of analogue video to digital form usually requires compression of some information to create usable digital files that can be stored, served and played. In general, video compression follows a standard for multimedia contents that encodes the content with a specific play rate. There are two main groups which define the video encoders: ITU (International Telecommunications Union) and ISO (International Standards Organization). The ITU-T group defines the H.26x video formats while the ISO group defines the formats which have emerged from committees of the Moving Pictures Experts Groups: MPEG-x. In order to reduce storage requirements, several compression techniques are proposed in the literature; here, we cite the examples of the MPEG-4 [5] and H.264 [6].

- ***MPEG-4***: The MPEG-4 standard is generally used for streaming media and CD distribution, video conversation and broadcast television. MPEG-4 incorporates many features of MPEG-1, MPEG-2 (designed for between 1.5 and 15 Mbits/sec.) and other related standards. The standard is still under development and is divided into several parts. It includes the concept of 'profiles' and 'levels', which allow a specific set of capabilities to be defined in a manner appropriate to a subset of applications. MPEG-4 is able to divide massive video files into pieces small enough to be sent over mobile networks. MPEG-4 offers important interactive features where the video functions are almost like a Web page but allow people to interact with the picture on the screen or to manipulate individual elements in real time. The standard also allows other types of content to be bundled into a file but these files require special software for playing. MPEG-4 would allow the interactivity of the video which may release the potential to do far more than just point and click at links on the screen. Individual elements of the video like a character, a ball in a sporting event, a rocket ship in a science-fiction epic, can exist in a separate layer from the rest of the video. This could allow viewers to interact with these elements somehow, even changing the direction of the story.

- ***H.264***: Is also known as MPEG-4 Part 10 or AVC (Advanced Video Coding). It is poised to become the next standard for format of convergence in the digital video industry regardless of the video playback platform. Big Internet players like Google/YouTube [7] or Apple iTunes [8] are all based on this format. The H.264 standard is jointly maintained with MPEG so that they have identical technical content. The intention behind the H.264/AVC project was to provide good video quality at substantially lower bit rates than previous standards. An additional goal was to provide enough flexibility to allow the standard to be applied to a wide variety of applications on a wide variety of networks and systems. The standard contains multi-picture and inter-picture prediction including features like the use of previously-encoded pictures as references in a more flexible way than in past standards, allowing up to 32 reference pictures to be used at times. This standard is more attractive for video network delivery and for delivery of high definition video. It is an open format with published specifications and is widely available for implementation.

In multimedia streaming services, the video consists of small data blocks. Each block has a numerical sequence number; receivers must use this identification to reorder blocks for playback correctly. Depending on the encode format, the storage requirement and concurrent service of videos can drastically change. The video characteristics demand a careful strategy for design of VoD systems.

1.2.2 VoD elements

There are three basic elements that compose a VoD system: server, clients and network; figure 1 illustrates such elements.

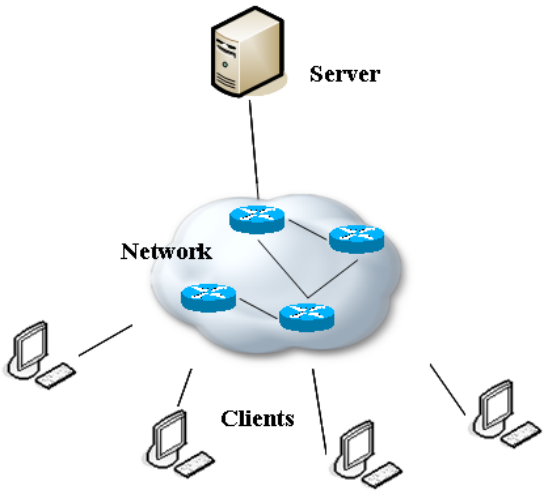


Figure 1. VoD system elements.

a) ***Server****:* The video server stores the multimedia catalogue and is designated to control the availability of resources to serve clients' requests with the required QoS. The server can be divided into sub-systems which play different roles in the service.

On the one hand, it is necessary to manage requests for service. The control interface is responsible for managing users' requests[o]. It must controlling server-client interactions in order to guarantee the system operation. This means that the control interface is responsible for verifying the availability of resources and then deciding if the request can be served without damage to other clients and also guarantee the QoS required by the new request. The application interface

manages the direct connection between a client and the server. It acts as the link between the client and the control interface. The application interface stores clients' information (resources, profile etc.) and data from the catalogue (availability, prices etc.).

On the other hand, there is the need to deal with the multimedia content. The storage interface responsible for storing and recovering multimedia information from the storage devices. The video data can be stored on a combination of magnetic or optic devices and must be retrieved in time to provide service. The storage capacity and service bandwidth are directly related to this server component because they determine the size of the catalogue supported and service restrictions. Figure 2 shows the basic design of a multimedia sever; the scheme illustrates the interaction of the processor sub-system with storage and communications sub-systems through the control, application and storage interfaces.

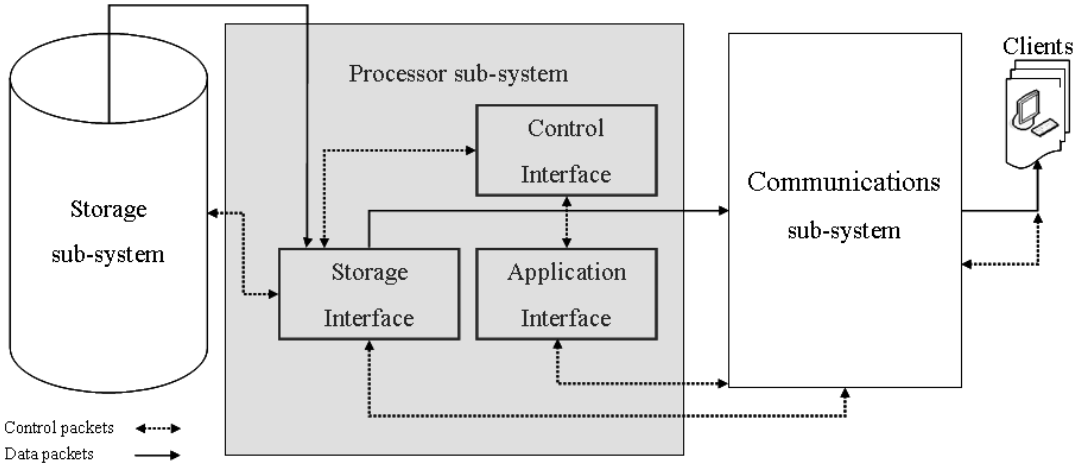


Figure 2. Multimedia server basic design.

b) ***Clients****:* The clients must receive, decode and display the video information always expecting a continuous and synchronised visualisation during the streaming session. The client unit can be either a general-purpose computer (Personal Computer - PC) or a dedicated embedded system commonly called Set-top-Box (STB). Both are able to provide customer requests for multimedia content and are made up of basic functions. The network interface receives the video information, which is stored on a region of memory used temporarily to

hold data, called a *buffer*. The buffer can be divided into many parts which play different roles in the VoD service. Its size can vary depending on the system design (can be expressed in minutes or MBytes). The distinct parts of the buffer can be used, for example, to prevent jitters and glitches owing to network latency or can allow the implementation of delivery techniques like multicast or peer-to-peer. A portion of the buffer is generally used to achieve smooth playback by storing data for further consumption; this portion is called a *cushion buffer*. The application of this strategy solves the problem of the short-time variable bandwidth because clients consume the information previously stored at the cushion, playing at the video rate. Nevertheless, the user experience is damaged because it is necessary to wait for a time in order to complete the reception of the cushion before starting the content visualisation. This retard is called *start-up delay*.

Another buffer portion is used by clients to cache video information from ongoing channels; this is the *delivery buffer*.

The multimedia content is stored in encoded format (e.g. MPEG-4) and is delivered to the client in a compressed way. To provide the visualisation of the content, it is necessary for the information to pass through a decompression step. This step is provided by the decoder, which demultiplexes the content into the corresponding audio and video parts and sends the information to the presentation stage. The presentation stage is responsible for synchronising the audio and the video streams. In case of any error it takes the appropriate actions (repeating the previous frame, for example) to solve the conflict.

c) ***Network****:* The last basic component of VoD systems is the network, which provides the interconnection of clients and servers by mean network elements (e.g. routers, cables). In addition to video transference, the network also includes another important function, the control signalling to service management. The network must be able to provide enough bandwidth for a designed video application. Control messages and video information flow through the network using a transport protocol.

The network can be generally divided in accordance with the area of action and bandwidth capacity. The backbone is responsible for the interconnection of networks. It has high bandwidth capacity and can cross long distances. The main

network interconnects servers distributed over a wide area. The local network interconnects the clients with the VoD system. This part of the network has only minor requirements in terms of bandwidth and comprises a small physical area.

1.2.3 VoD requirements

Building an efficient large-scale VoD system demands special attention to many aspects. An LVoD system has certain properties and particularities which must be respected in provision of video service. The most important requirements to be considered at the time of LVoD system design are the following.

***Storage and bandwidth resources**:* Multimedia are media with multiple content forms. They represent the convergence of text, images and audio into a single element. This diversity implies files of great size even if compression techniques are applied.

An attractive VoD service must be able to provide options to clients, which is translated as a catalogue of videos made up of a great amount of files. The natural characteristics of the multimedia content (big files) added to the volume of data generated by a considerable video catalogue lead to the need fir huge storage capacity; moreover, the bandwidth available at the network must be sufficient to guarantee concurrent video streams for many clients. One of the main challenges on designing VoD systems is to accomplish with storage and bandwidth requirements; strategies such as content placement policies or multicast communications (which will be discussed with more detail on next topics) are applied to achieve these goals.

***Scalability degree**:* There are two main parameters in VoD service which can grow in a relatively short time: the video catalogue size and the number of clients. Ideally, the system should support continuous expansion of the number of videos and service requests. The service capacity defines the maximum number of petitions that the system can support. A good system design must provide the service capacity to an initial set of clients and the necessary infrastructure in order to enable system expansion without much complexity.

A VoD system with a low degree of scalability may rapidly be overloaded, causing request rejection, since service constraints cannot be guaranteed. Moreover, high costs can be generated by the necessity to redesign the system in accordance with the new scenario, adding more elements.

There are many techniques used nowadays to guarantee scalability in VoD systems. The most commonly applied solution is to decentralise the service distributing the multimedia contents over a set of servers. Many questions emerge from this approach, such as the load balance or distribution criteria [9]. To deal with client growth effectively, resource-sharing strategies like multicast communications or peer-to-peer (P2P) collaborations are often used. Distributed architectures and resource-sharing strategies will be explained in more detail in the following chapters.

***Time constraints and Quality of Service**:* In order to guarantee a satisfactory visualisation of the multimedia content, it is not sufficient for the server to send the information to the user and for the user to receive it correctly; the reception of the data is time-limited. The time consumed in multimedia data operations must respect a threshold. This characteristic makes a VoD system a soft real-time application. A system is said to be real-time if the total correctness of an operation depends not only upon its logical correctness but also upon the time in which it is performed. The classical conception is that in a hard real-time or immediate real-time system, the completion of an operation after its deadline is considered useless (this may lead to a critical failure of the complete system). A soft real-time system on the other hand will tolerate such lateness, and may respond with decreased service quality (e.g. dropping frames while displaying the video).

The strict adherence to time constraints guarantees Quality of Service (QoS) to the clients. Providing acceptable QoS for all clients on the system is an important concept in any video service; it includes different aspects, like image quality, image loss frequency or audio and video synchronisation.

The usage of memory buffers helps to accomplish the soft real-time characteristic of the system. The information is stored at the buffer, allowing control of the amount of data necessary to perform a satisfactory service. In this way, a start-up delay is included in order to build a cushion of video information used to

guarantee a smooth playback; this technique is also referred as prefetching i.e., loading information ahead of time.

Despite the usage of prefetching, the data input and consumption rates must be balanced on video services; if the client consumes information faster than data arrive, the playback can be interrupted by the lack of data. In the case of the input data rate being bigger than the consumption speed, the buffer may overflow and information is dropped.

***Fault Tolerance**:* Large-scale systems are more prone to failures owing to their great number of components such as links, routers, switches, clients or proxies. On LVoD systems, a fault occurrence can affect the service to the clients. The system must guarantee reliability to the users; it is undesirable for clients to have the service interrupted owing to a failure occurrence.

A fault-tolerant system is defined by Jalot at [10] as a system which can mask the presence of faults by using redundancy. VoD systems present inherently a certain level of redundancy; the asynchronous nature of the requests and the usage of techniques such as distributed servers, multicast or peer-to-peer (P2P) add redundancy to the system. The soft real-time constraints on the delivery of video data demand a quick response in terms of failure management.

1.2.4 VoD service schemes

Providing video stream on demand over a network is a not trivial task. There are many aspects to consider; the multimedia information has a high bandwidth requirement and scalability is a key issue in the provision of service to a large number of clients. A popular content can attract a large number of viewers, who perform requests asynchronously. The service scheme defines the way information is sent from a source to the respective destination. There are three main types of service schemes: Unicast, Broadcast and Multicast.

Unicast: This is a point-to-point communication channel. This means that unicast establishes communication between a single sender and a single receiver over the network.

As shown by figure 3, unicast works by sending an independent flow of information to each one of the clients who request multimedia content.

The main advantage of this technique is the control mechanism. Each communication channel is independent, and thus the control is oriented to each client and there are no interactions between the channels. On the other hand, the great constraint of unicast usage is the bad scalability of the scheme. Each channel consumes server and network resources in a linear fashion, generating a rapid consumption of the resources predicted in system design. Unicast converts the server on a bottleneck for the VoD system.

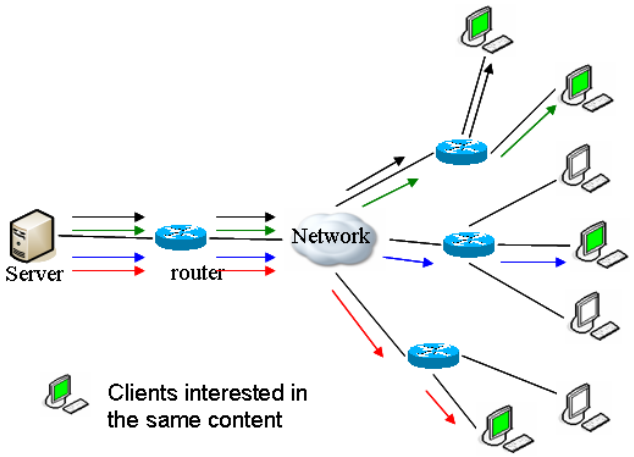


Figure 3. Unicast communication strategy.

Broadcast: This service scheme is based on the communication from a single sender to all the receivers over a network. In this scenario the video is broadcast on a dedicated channel within a predefined schedule. This approach can provide service to an unlimited number of requests for popular video content with a constant consumption of bandwidth. A great amount of resources is consumed pointlessly, however, if the popularity of the video is low. In addition, clients must wait until the scheduled time to be served. Figure 4 shows the broadcast communication scheme, where the information is distributed to every end host in the system.

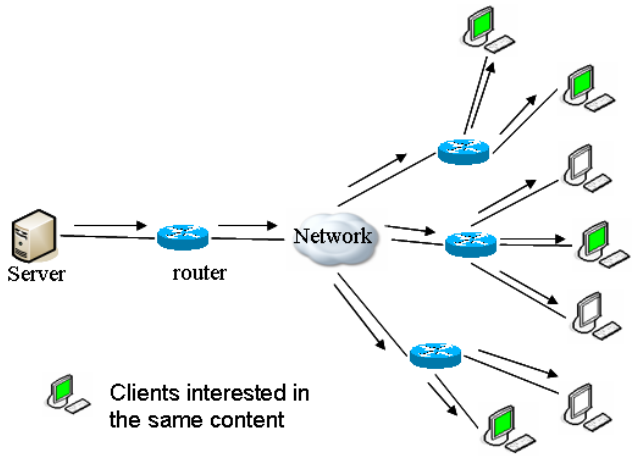


Figure 4. Broadcast communication strategy.

Multicast: The multicast technique consists of information flow from a source to a group of receivers who requested the same content. A number of requests for the same video are grouped and served by a single video stream. Therefore, the service scheme saves bandwidth by sharing video streams without wasting resources on non-requesting receivers. Multicast is shown in figure 5; it provides the best solution to scale large VoD systems.

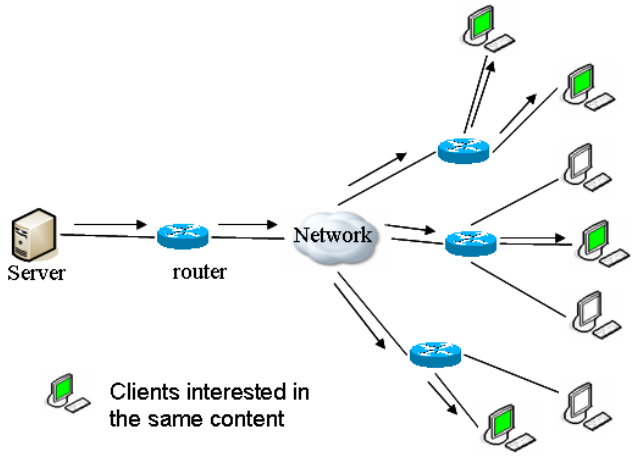


Figure 5. Multicast communication strategy.

There are many policies for grouping clients under a single video stream. In Batching [11], the video requests are delayed for a period of time. Thus, requests for the same video can be collected and grouped. The batch of requests is then served by a multicast video stream. The major drawback of this policy is that clients must wait for a batching interval until the video starts playing.

The Patching [12] strategy dynamically assigns clients to join ongoing multicast channels and patches the missing portion of video by applying unicast. Figure 6 represents the Patching service. Initially, the server creates the video channel; when a client requests a video of an ongoing channel, the server incorporates the new request in the existing channel and the missed initial portion of the content is sent to each new client through a unicast channel called Patch. The Patch goes from the time when the video begins streaming until the new client is joined to the system (B). This approach enables clients to be grouped for multicast service, saving bandwidth and also introducing zero start-up delay. On the other hand, this multicast policy increases the complexity at the client side, because users must be able to receive two simultaneous channels i.e. the main multicast stream and the patch stream. Clients must have a large enough buffer (of size of B or more) to store data of the main channel while receiving the initial portion of the video from the unicast channel.

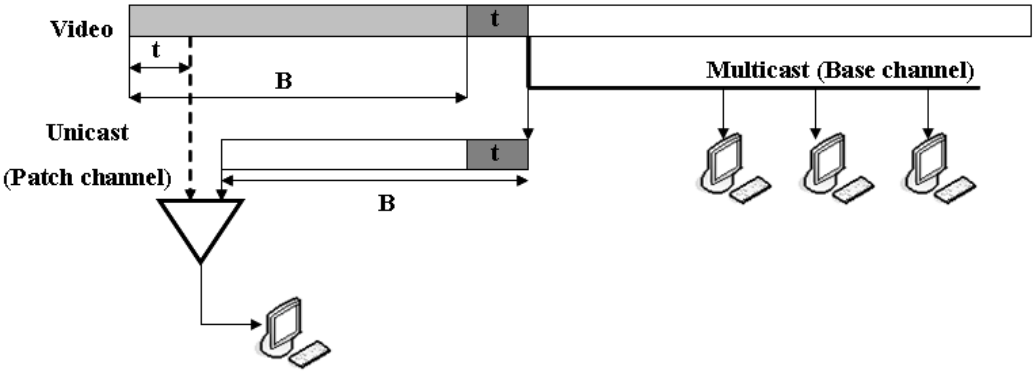


Figure 6. Patching technique for multicast implementation on video stream.

1.2.5 VoD architectures

The proposals for VoD architectures are many, from the traditional centralised client-server scheme to different distributed approaches. The solutions aim to provide a high performance and scalable video service. In the following, we present the main architectures applied to VoD systems.

Centralised architecture: In this architecture the clients are directly connected to the server through the network, as showed in figure 7. The service relies on the traditional client-server relation i.e. clients request multimedia content and are served by the server. The

server must have sufficient storage resources to keep the entire video catalogue and must have high output bandwidth in order to serve the petitions.

The major problem of this architecture lies in the poor scalability, since the service capacity is well defined by server limitations. A system expansion may lead to huge costs in resources increment. Moreover, the expansion of the architecture is limited by current technology, which means that the system could not grow indefinitely even though the costs do not represent an important restriction.

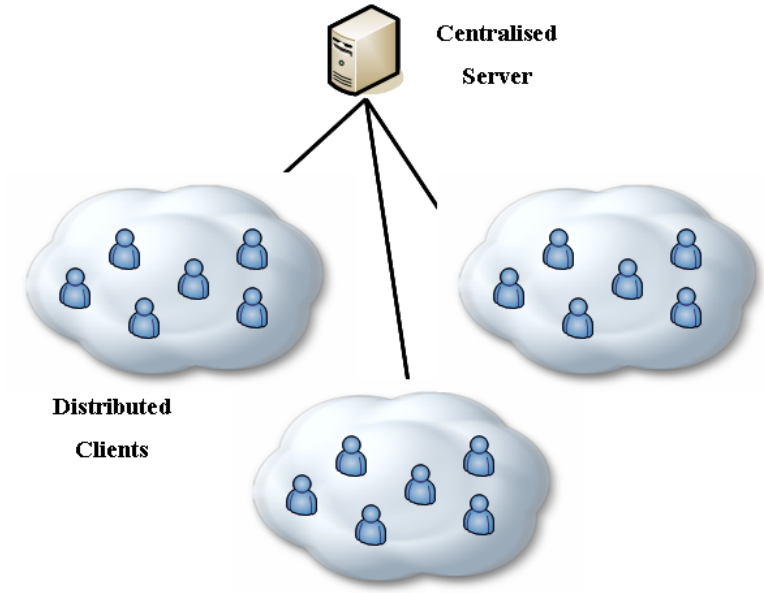


Figure 7. Centralised architecture

Distributed architectures: In order to address the scalability restriction of centralised topology, VoD systems were oriented to distribute the service through many servers. Diverse solutions are adopted to distribute video service.

The Content Delivery Network (CDN) is one of the solutions for decentralising the VoD service [13]. In a CDN-based solution, the video source server first pushes video content to a set of content delivery servers placed strategically at the network edges. Instead of downloading from the video source server, a client is normally directed to a nearby content delivery server to download the video. CDN effectively shortens the users' start-up delays, reduces the traffic imposed on the network, and serves more users as a whole. Nevertheless, the major challenge for CDN-based VoD service is still the scalability. A video session with good quality requires high bandwidth. The bandwidth provision, at video source servers or

local content delivery servers, must grow proportionally with the client population. The cost of maintaining a CDN is extremely high considering the massive CPU power requested to provide service to a considerable number of clients and the storage capacity to host all system catalogue at every local server [14].

An alternative adopted to reduce CDN costs is the full distribution of the contents over the servers placed on the network edges. There is no central server; contents follow a distribution strategy to be allocated to the local servers. These local servers are referred to as proxy-servers and store only the popular contents of a main catalogue. The contents with low popularity are distributed over the entire set of servers according to a placement policy (e.g. cache\mirror scheme [15]). If a client request cannot be met by the local video server, the request is redirected to adjacent video servers on the distributed architecture. Several proposals have been made for distributed video servers [16][17][18].

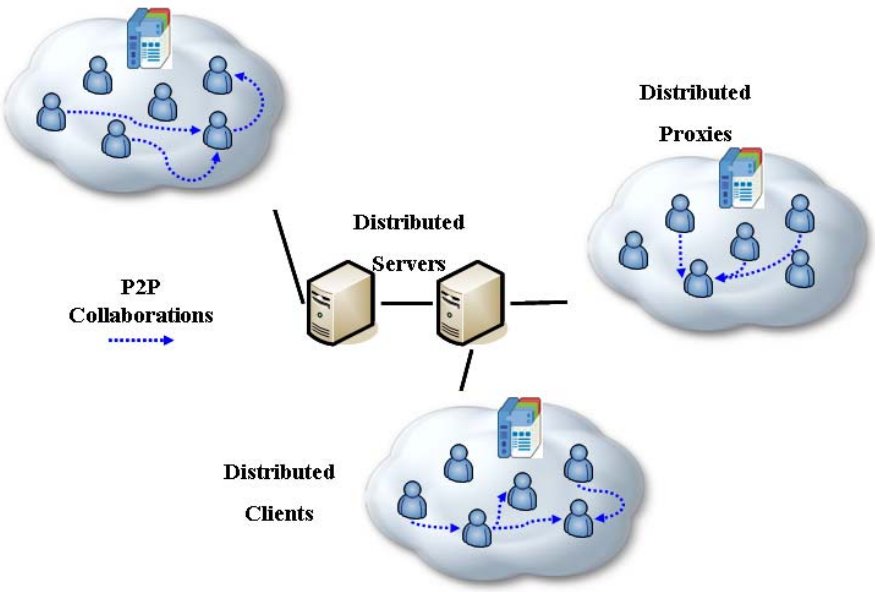


Figure 8. Distributed architecture with P2P collaborations.

The distributed proxy-server architecture reaches a good level of scalability by decentralising the service with local servers and applying popularity-based strategies to distribute contents. Aiming to increase the performance of VoD systems in order to achieve global environments such as the Internet, however, the peer-to-peer (P2P) mechanism has been applied. The basic design philosophy of P2P is to encourage users to act as both clients and servers, so-called peers. In a P2P network, a peer not only downloads data from the

network, but also uploads the downloaded data to other users in the network. The uploading bandwidth of end users is efficiently utilised to reduce the bandwidth burdens otherwise placed on the servers. Figure 8 shows an example of a decentralised architecture; distributed video servers combined with P2P paradigm can avoid the limitations of server-based architectures and provide a high degree of scalability and performance for LVoD systems.

Based on the P2P paradigm, fully decentralised systems for VoD service have been proposed [19] [20]. These systems assume the video catalogue is totally distributed over a pee-to-peer network, without content storage on any server. Nevertheless, this approach looks inadequate for LVoD implementation; the control of the distribution of legal content is much harder since peers must permanently store portions of the videos [21].

1.3 Pee-to-Peer Paradigm

Peer-to-Peer (P2P) is a specific form of relational dynamic, based on the assumed equipotency of its participants, organised through the free cooperation of equals in view of the performance of a common task, for the creation of a common good, with forms of decision-making and autonomy that are widely distributed throughout the network [22].

A P2P computer network uses diverse connectivity between participants in a network rather than conventional centralised resources where a relatively low number of servers provide the core value to a service or application. In computer science, the concept of P2P is increasingly evolving to an expanded usage as the relational dynamic active in distributed networks, i.e. not just computer-to-computer, but human-to-human. Associated with peer production are the concept of peer governance (referring to the manner in which peer production projects are managed) and peer property (referring to the new type of licences which recognise individual authorship but not exclusive property rights, such as the GNU General Public Licence [23] and the Creative Commons licences [24]).

According to [25], *"P2P is a class of applications that takes advantage of resources (storage, cycles, content, human presence) available at the edges of the Internet. Because accessing these decentralized resources means operating in an environment of unstable connectivity and unpredictable IP addresses, P2P nodes must operate outside the DNS system and have significant or total autonomy from central servers."*

Content delivery through P2P networks started to become popular in May 1999, when an 18-year-old college student, Shawn Fanning, built Napster [26] at the dormitory of Boston's

Northeastern University. One of Shawn's college roommates loved listening to MP3s and complained about the unreliable MP3 sites at that time. Music links were frequently dead and indices were often out of date. In response, Shawn developed a real-time system to help music lovers to find and exchange music files on the Internet. Shawn's program listed files which users were willing to share on a computer that others could access. Moreover, users of Shawn's program directly downloaded the music from other users' computers, bypassing the use of central servers. Shawn named the program Napster, which was his nickname, email alias and username in IRC rooms [27]. Along with Napster, other P2P applications to file sharing appeared, such as BitTorrent [28], FastTrack [29] and Gnutella [30].

Since users highly favour fast response in applications and shared contents are usually video or audio files, streaming was devised as an alternative mode of content delivery. Multimedia content's continuous streaming on the Internet generally comprisesTV channels, which can be transmitted via the P2P strategy; in this service each user is simultaneously downloading and uploading a video stream, thus helping to provide the service.

Currently, the P2P paradigm is a solution to provide additional resources to LVoD architectures, dealing with Internet constraints and service requirements. P2P is able to provide a high level of scalability for VoD systems by aggregating resources with low-cost interoperability; the whole is made greater by the sum of its parts.

The operation of any P2P content distribution system relies on a network of users (nodes) and connections (edges) between them. This network is formed on top of the underlying physical computer network; therefore it is referred as an 'overlay network'. Overlay networks can be distinguished in terms of their location scheme and topology. The topology of the overlay network and the location mechanism are crucial to the operation of the system because they have direct influence on messages and content distribution. They affect the system's fault tolerance, self-maintainability, performance and scalability [30] [33] [34].

1.3.1 Location scheme

The location schemes are responsible for indicating how the contents are distributed over the peer-to-peer network; these schemes can be categorised in terms of their structure; figure 9 illustrates the classification of P2P implementations. If the overlay network is created non-deterministically (ad hoc) as nodes and content are added, the location scheme is known as

unstructured; if the creation is based on specific rules, it is classified as *structured*. Although in their purest form P2P overlay networks are supposed to be totally decentralised, in practice this is not always true. Systems with various degrees of centralisation are encountered, whereby it is possible to identify three categories: *purely decentralised, partially decentralised* and *hybrid*.

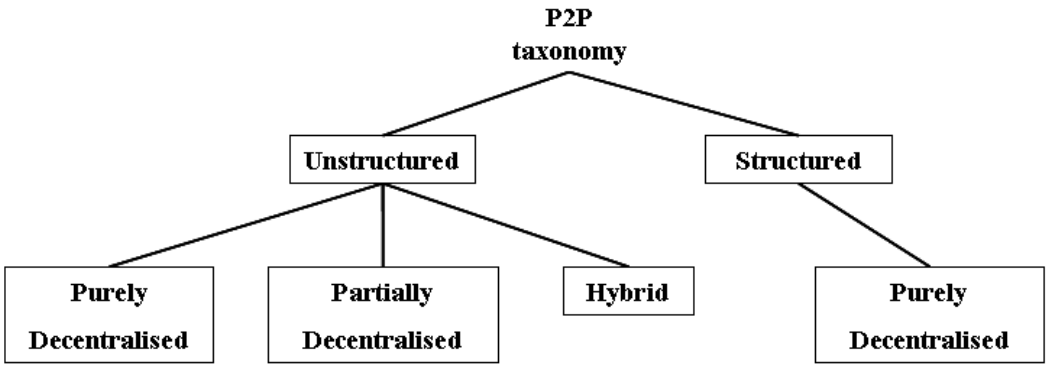


Figure 9. Classification of P2P implementations.

a) ***Unstructured***: In an unstructured network, content typically needs to be looked for by queries. Searching mechanisms range from brute force methods, such as flooding the network with propagating queries in a depth-first manner until the desired content is located, to more sophisticated and resource-preserving strategies that include the use of random walks and routing indices. The searching mechanisms employed in unstructured networks have obvious implications, particularly regarding matters of availability, scalability and persistence. Unstructured systems are generally more appropriate for accommodating highly-transient node populations. The search scheme can be implemented with distinct degrees of centralisation.

   1. ***Purely decentralised***: All nodes in the network perform exactly the same tasks, acting both as servers and clients; there is no central coordination of their activities, as illustrated on figure 10. A representative example of purely decentralised structure is Gnutella [30], which is used in applications such as LimeWire [37] or BearShare [38]; it presents an open architecture, a scalable and self-organising structure. The Gnutella architecture uses a flooding (or broadcast) mechanism to distribute *Ping* and *Query* messages:

each Gnutella node forwards the received messages to all of its neighbours. The response messages received are routed back along the opposite path through which the original request arrived. To limit the spread of messages through the network, each message header contains a time-to-live (TTL) field. At each hop the value of this field is decremented; so, when it reaches zero, the message is dropped. Scalability issues in purely decentralised systems arose from the fact that the TTL usage segments the network, imposing on each user a virtual 'horizon' beyond which their messages could not reach. On the other hand, when the TTL limit is removed the network becomes overwhelmed with messages.

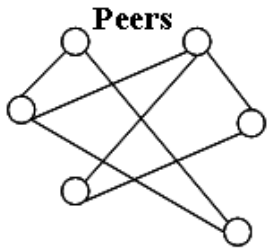


Figure 10. Purely decentralised unstructured P2P approach.

2. ***Partially decentralised***: The basis is the same as for purely decentralised systems but some of the nodes assume a more important role, acting as local central indices for files shared by local peers. Peers are automatically elected to become *supernodes* if they have sufficient bandwidth and processing power (although a configuration parameter may allow users to disable this feature). The way in which these supernodes are assigned varies between different systems. Supernodes index the files shared by peers connected to them and proxy search requests on behalf of these peers. All queries are therefore initially directed to supernodes. Figure 11 shows the partially decentralised approach; it is important to note that these supernodes do not constitute single points of failure for a peer-to-peer network, since they are dynamically assigned and, if they fail, the network will automatically take action to replace them with others. Two major advantages of partially centralised systems are: the discovery time is reduced in comparison with purely decentralised systems, while there still is no unique point of failure;

the inherent heterogeneity of P2P networks is exploited. In a purely decentralised network, all of the nodes will be equally loaded, regardless of their CPU power, bandwidth, or storage capabilities. In partially centralised systems, however, the supernodes handle a large portion of the entire load, while most of the other 'normal' nodes will be lightly loaded. FastTrack [29] is a partially decentralised protocol, which is applied by KaZaA [39].

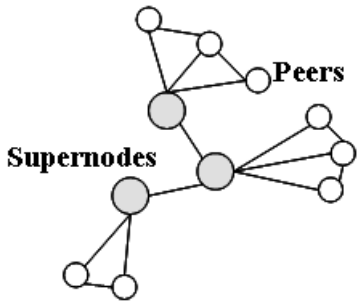


Figure 11. Partially decentralised unstructured P2P approach.

3. ***Hybrid*:** In these architectures, there is a central server facilitating the interaction between peers by maintaining directories of metadata, describing the shared content owned by the peer nodes. Although the end-to-end interaction and file exchanges may take place directly between two peer nodes, the central servers make the interaction easier by performing the lookups and identifying the nodes with the respective contents. Obviously, in these architectures, there is a single point of failure (the central server). This typically renders them inherently unscalable and vulnerable to censorship or technical failures; figure 12 represents the hybrid P2P approach. BitTorrent [28] is a widely-used sharing protocol based on a hybrid mechanism. The architecture consists of a central location called a tracker, which stores a *.torrent* file; the *.torrent* contains information about the content, its length, name, hashing information and the URL of a tracker. The tracker keeps track of all the peers who have the content, thus it can intermediate the connection among peers to download pieces of the file. BitTorrent downloads in a random or in a 'rarest-first' approach and uses *tit-for-tat* (peer responds with the same action that its other collaborating peer performed previously). The protocol is designed to achieve high content

availability and to discourage free-riders. Peers with high upload speed will probably also be able to download at a high speed, thus achieving high bandwidth utilisation. The download speed of a peer will be reduced if the upload speed has been limited.

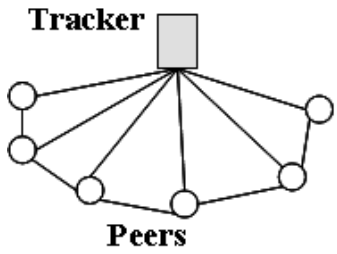


Figure 12. Hybrid unstructured P2P approach.

b) ***Structured***: These strategies have emerged mainly in an attempt to address the scalability issues that unstructured systems were originally faced with. In structured networks, the overlay topology is tightly controlled and contents (or pointers to them) are placed at precisely specified locations, making subsequent queries more efficient. These systems essentially provide a mapping between content (e.g. file identifier) and location (e.g. node address), in the form of a distributed routing table so the queries can be efficiently routed to the node with the desired content. The overlay network assigns keys to data items and organises its peers into a graph that maps each data key to a peer. This structured graph enables efficient discovery of data items using the given keys. Structured systems offer a scalable solution for exact-match queries, that is, queries where the exact identifier of the requested data object is known. In its simple form, however, this class of system does not support complex queries; another disadvantage of structured systems is the difficulty of maintaining the structure required for efficient routing of messages in the face of a very transient node population in which nodes are joining and leaving at a high rate.

Figure 13 illustrates the structured P2P approach, which is naturally decentralised. Such P2P systems use the Distributed Hash Table (DHT) as a substrate, in which data object (or value) location information is placed deterministically, at the peers with identifiers corresponding to the data object's unique key [32]. DHT-based systems consistently assign uniform random

NodeIDs to the set of peers in a large space of identifiers. Data objects are assigned unique identifiers called keys, chosen from the same identifier space. Each peer maintains a small routing table consisting of its neighbouring peers' NodeIDs and IP addresses. Lookup queries or message routing are forwarded across overlay paths to peers in a progressive manner, with the NodeIDs that are closer to the key in the identifier space. Typical examples of structured decentralised systems include Chord [40], CAN [41]and Kademlia [42] among others.

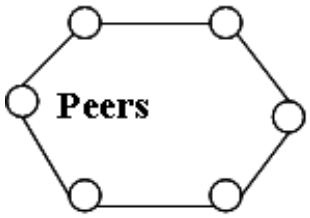


Figure 13. Decentralised structured P2P approach.

1.3.2 Overlay topology

The topology describes the way end hosts are connected i.e. the arrangement of the elements and the links between them. P2P video systems can be broadly classified into three categories based on the overlay network; they can form a *chain*, a *tree* or a *mesh* (figure 14).

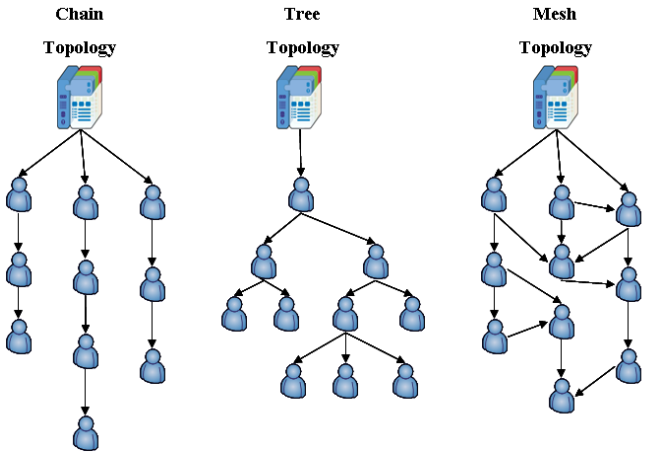


Figure 14. P2P overlay topologies for multimedia stream.

- ***Chain***: A chain consists of a series of connected end hosts. Owing to restrictions on network bandwidth and peer capabilities, a client can deliver information only to one other child and so on. This mechanism is presented in Chaining [45]; the pattern of data dissemination constitutes a chain. The idea is to 'pipeline' the information through the chain of nodes. A peer might leave the system at any time, unexpectedly. After a peer leaves, the chain is broken and all the descendants are disconnected from the source and cannot receive any data. To minimise the disruption, the chain needs to be recovered as soon as possible. A simple solution is to connect directly the two extremities created by the crashed node; it can, however, cause some loss of information in the transmission process.

- ***Tree***: The tree topology is similar to the chain; peers can, however, collaborate with more than one client. If the information flows from one peer to a set of nodes and from these nodes to others, the resultant topology is a tree. A tree can be viewed as a fusion of chains, constituting a more complex structure, but still with a hierarchical relation. Just as in the chain topology, a peer failure causes the disruption of the information flow through the tree, with the aggravation of reaching more clients on the tree levels beneath. The tree can be recovered by reassigning affected peers to the source and\or to other unaffected peers. Nevertheless, reassembly of the tree implies the recalculation of the topology by the source, which becomes a performance bottleneck; tree-based P2P systems may not recover fast enough to handle frequent peer churn. Another major drawback of the tree approach is that not all the leaf nodes contribute their resources. Since leaf nodes account for a large portion of peers in the system, this greatly degrades the collaboration efficiency.

- ***Mesh*****:** In the tree topology, a peer has only one parent and downloads all the content which is flowing through the distribution tree. This design introduces a single point of failure; if a peer's parent leaves, the peer, as well as its descendants, cannot receive data until it connects to another parent. In a mesh topology, a peer maintains peering relationship with multiple neighbouring peers; peers establish and terminate peering relationships dynamically. A peer may download\upload data from\to multiple neighbours simultaneously. If a peer's neighbour leaves, the peer can still download content from remaining neighbours. At the same time, the peer will search for new neighbours to keep a desired level of connectivity. The high peering degree

in mesh-based topologies makes them extremely robust against peer churn. The key concept of mesh topologies is the division of data into multiple blocks; this makes it possible to establish relations among diverse peers. The mesh topology can also be treated as a multi-tree based structure. The content source divides data into multiple blocks. Instead of one tree, multiple sub-trees are constructed, one for each data block; each peer joins all sub-trees to retrieve the blocks. Within each sub-tree, the corresponding block flows down level by level from the source to all the leaf nodes. A peer has different positions in different sub-trees. It might be positioned on an internal node in one sub-tree and on a leaf node in another sub-tree; peers resources are used to upload a block whenever it is placed on an internal node in some sub-tree.

1.4 Multicast Service Scheme

Multicast is a communication strategy where a sender has the capability to transmit information that can be received concurrently by a group of interested recipients. The mechanism is based on delivering the information over each link of a network only once, creating copies only when the links to the multiple destinations split. The sender(s) and receivers are assumed to be part of a group; a user can be a member of any number of multicast groups and the membership of a multicast group is dynamic, as the sender(s) and receivers can join or leave the group at any time. The multicast groups can be classified either as dense or sparse groups based on the distribution of the group members in the network.

In the age of multimedia and high-speed networks, multicast is one of the viable mechanisms by which the power of the Internet can be improved in an efficient manner [44] [46] [47] [48] [49].

1.4.1 Multicast approaches

Multicast communications can be applied on different ways, depending on the elements that run the routing and forwarding algorithms. In this section we present three approaches for multicast: IP Multicast, Overlay Multicast and Application Layer Multicast.

IP multicast: An IP multicast-enabled network provides end-to-end services in the network infrastructure (according to the OSI model) to allow any user to send information to an IP multicast address that any number of other end hosts widely dispersed can receive. The algorithms and protocols employed to achieve multicast communication are implemented by network routers, which are devices whose software and hardware are oriented to the tasks of routing and forwarding information. Figure 15 shows the IP Multicast functionality.

To receive multicast information, applications join the multicast group, which transparently generates a group membership report. This apparent simplicity is deceptive, however. Enabling multicast support in applications and protocols that can scale well in a heterogeneous network is a significant challenge. Specifically, sending constant bit rate data streams, reliable data delivery, security and managing many-to-many communications all require special consideration. Some solutions are available, but many of these services are still active research areas.

Currently multicast data communication is done by means of the User Datagram Protocol (UDP) [93]. This is to avoid the overhead of reliability and flow control associated with Transmission Control Protocol (TCP) [92].

IP Multicast can be viewed as a requirement, not an option, on the Internet in the near future. End-to-end global multicast service is not yet available, but the size of the multicast-enabled areas is growing. Recent advances provide IP multicast, making many local networks multicast-capable on today's Internet. Some examples of protocols used in IP Multicast implementation are the IGPM and PIM, which will be explained in more detail in the next topics.

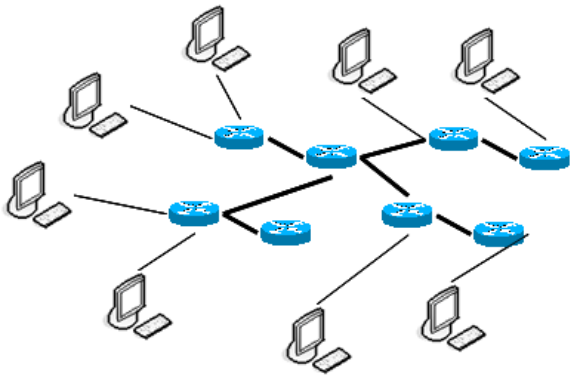


Figure 15. IP Multicast.

Overlay multicast: In order to provide global multicast even in an environment of heterogeneous networks, one solution is to use special nodes connected in an overlay network on top of the physical network. The implementation is based on an infrastructure of special nodes (proxies or dedicated servers) strategically distributed in the network; these nodes compose the overlay network and provide an efficient distribution of the information to a set of clients. A communication network is established through the dedicated elements, which are responsible for information routing and forwarding (figure 16). The advantage of the overlay multicast implementation lies in the functional dedication of the elements. Dedicated and better than normal users, the proxies are more trustworthy and robust; the proxies can be located strategically through the network, improving the overlay network efficiency. Nevertheless, the application of overlay networks presents some inconvenience, which restricts its deployment. Dedicated nodes are less adaptable to changes in the environment; their placement is static, and thus they demand manual rearrangement if necessary. There is also the cost added by the dedicated elements and communication infrastructure.

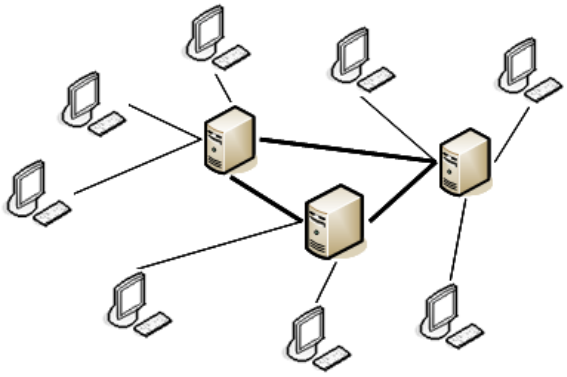


Figure 16. Overlay Multicast.

An example of overlay multicast implementation is OMNI, the Overlay Multicast Network Infrastructure [50], which offers an overlay architecture for media streaming applications. In OMNI, service providers deploy Multicast Service Nodes (MSNs) that run the routing and forwarding of information to a set of clients. The MSNs execute a distributed protocol to form a multicast data delivery backbone. The goal of OMNI is to improve i.e. minimise, the latencies for the entire client set. MSNs are given priorities

based on the population of clients; thus relative importance of the MSNs varies as clients join and leave the session.

Application Layer Multicast: Owing to the drawbacks presented by Overlay Multicast and the slow deployment of IP Multicast technology on the global Internet, an application layer solution has been adopted; this approach is referred to as Application Layer Multicast (ALM). In this multicast strategy, group membership, multicast tree construction (or some other delivery structure) and data forwarding are solely controlled by participating end hosts; thus, it does not require the support of intermediate nodes (such as routers or dedicated servers). The P2P approach has ALM premises. In ALM, groups of users form a communication network and data packets are relayed from one member to another as shown in figure 17.

In Application Layer Multicast, however, the lack of knowledge about underlying network topology usually results in performance penalty compared with IP Multicast, such as lower efficiency and longer end-to-end latency. Furthermore, it typically requires a large amount of control overhead to maintain group membership and multicast trees among the end users. NICE [51], ALMI [52] and ZigZag [53] are popular ALM protocols.

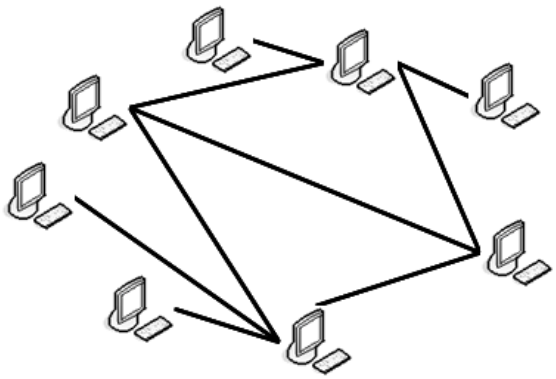


Figure 17. Application Layer Multicast.

NICE is a recursive acronym that stands for NICE is the Internet Cooperative Environment. This scalable application layer multicast protocol uses a hierarchical clustering approach to support a larger number of receivers. NICE was designed to provide architecture for low bandwidth soft real-time data stream applications such as real-time stock quotes and updates and Internet radio. It organises hosts in a hierarchy of

layers and each layer has several clusters of hosts. Each cluster has a leader to communicate with higher layers. On top of the hierarchy, NICE can build trees of different kinds. NICE allows nodes in a cluster to exchange periodic messages to maintain appropriate peer relationships. The cluster leader also exchanges messages to its higher layer members. The cluster leader is responsible for maintaining proper cluster size and thus applies a splitting or merging algorithm when necessary. Since node joining and leaving may result in changes, the cluster leader has methods for refinement of the structure.

1.4.2 Multicast routing algorithms

The data transmitted need to be transferred from the sender(s) to the receivers. The sender(s) and receivers are mostly end hosts. Intermediate nodes are the routers, which route\direct the data from the sender(s) to the receivers. A spanning tree (tree composed of all the vertices and some of the edges of a graph) has been considered one of the most efficient and viable mechanisms to perform the data transmission in such a scenario. Information is duplicated only when the tree branches and this ensures data communication is loop-free. An efficient multicast routing algorithm aims to build a Minimal Spanning Tree (tree that spans all the group members and minimises the total weight of the tree). The type of tree to be used depends on whether receivers are sparsely or densely distributed throughout the network. The main approaches used in multicast routing are source tree and shared tree algorithms.

Source tree: Source tree algorithms (also known as shortest path trees) build a separate tree for each source. Reverse Shortest Paths (RSP) connect each of the receivers to the source. RSP is constructed by using Reverse Path Forwarding (RPF) at the intermediate routers. The decision to forward traffic is based upon source address and not on destination address. It is implemented by using either a dedicated multicast routing table or alternatively the router's native unicast routing table.

When a multicast packet enters a router's interface it will look up the list of networks that are reachable via that input interface i.e. it checks the reverse path of the packet. If the router finds a matching routing entry for the source IP of the multicast packet, the RPF check passes and the packet is forwarded to all other interfaces that are participating

in multicast for this multicast group. If the RPF check fails the packet will be dropped. As a result the forwarding of the packet is decided on the basis of the reverse path of the packet rather than the forward path. This is efficient for high data rate sources; it provides minimal delay and exhibits lesser traffic concentration. The source tree, however, can stress the storage capability of the routers when the system presents a large number of groups, each with a large number of sources. Source trees consume more bandwidth for each individual multicast group.

Figure 18 illustrates the source tree implementation; multicast routing protocols that use source trees are DVMRP [58] and PIM-DM [56] for example.

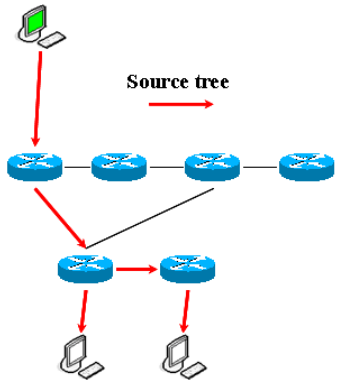


Figure 18. Source tree based IP Multicast.

Shared tree: Shared tree algorithms work with multiple sources, building a single tree to be used by all of them. The data communication in the tree can be one way or bi-directional. This is efficient for low rate sources and in the amount of state information that needs to be maintained at each router. It exhibits higher traffic concentration, however. Shared trees use a single location in the network called the Core or the Rendezvous Point (RP) to which all packets from the sources are sent and from which packets are sent to all receivers; therefore, the paths from certain receivers to the source may be longer, which may cause additional delay. This will be a disadvantage for delay-sensitive and high bandwidth applications; besides, the core is a potential bottleneck for data transmission. CBT [57] and PIM-SM [55] are examples of routing protocols making use of shared trees; figure 19 represents the shared tree IP Multicast implementation.

A shared tree offers more favourable scaling characteristics than all other multicast algorithms in terms of network state maintenance, bandwidth efficiency and protocol

overhead; it suffers, however, from the number of active sources. Routers between the source and the delivery tree do not incur any cost related to multicast as the packet is encapsulated and unicast to a core on the tree.

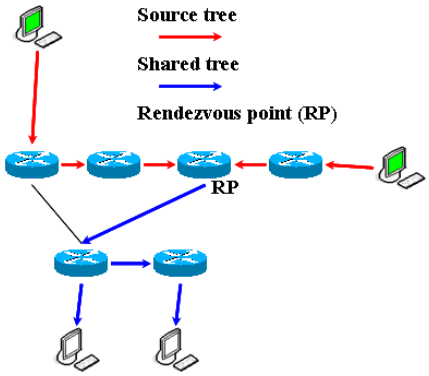


Figure 19. Shared tree based IP Multicast.

1.4.3 IP Multicast protocols

In the following, we describe three protocols used in IP Multicast implementation: the IGMP, which is responsible for groups management and the PIM (Sparse and Dense modes), which provide routing and forwarding of data. Figure 20 presents a classification of the IP Multicast routing protocols.

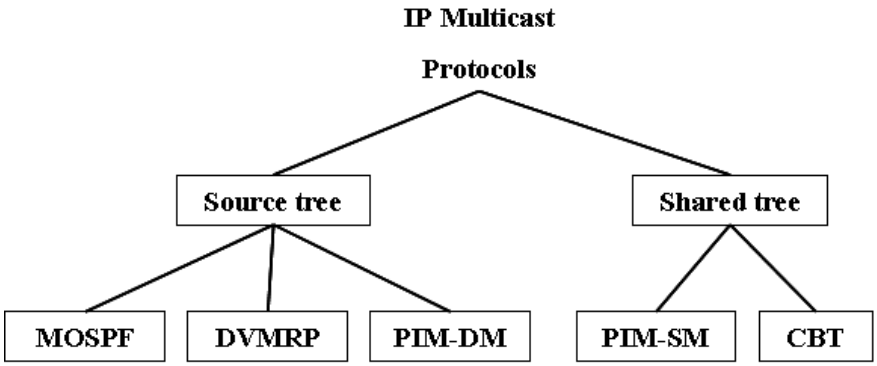


Figure 20. IP Multicast protocols classification.

IGMP: To receive multicast traffic, an interested user must tell its local router that it is interested in a particular multicast group address; the user accomplishes this task by

using the Internet Group Management Protocol (IGMP). IGMP is used by IP hosts and adjacent multicast routers to establish multicast group memberships [54].

The router periodically sends out an IGMP membership query to verify that at least one host on the sub-network is interested in receiving traffic directed to that group. When there is no reply to three consecutive IGMP membership queries, the router times out the group and disables the forwarding traffic directed toward that group. When a host decides to join a particular multicast group, it sends the request to the local multicast router. The local multicast router makes an entry for this group and propagates the information to other multicast routers to establish the multicast routes. Routers also listen to IGMP messages and periodically send out queries to discover which groups are active or inactive on a particular sub-network.

According to the group membership information learned from the IGMP, a router is able to determine which (if any) multicast traffic needs to be forwarded to each of its leaf sub-networks. Multicast routers use this information, in conjunction with a multicast routing protocol, to support IP multicasting across the Internet.

PIM-SM: Protocol Independent Multicast (PIM) is a family of multicast routing protocols that can provide one-to-many and many-to-many distribution of data over the Internet. The 'protocol-independent' label refers to the fact that PIM does not include its own topology discovery mechanism, but instead uses a routing table of the underlying unicast routing system.

PIM Sparse Mode (PIM-SM) is a protocol for efficiently routing to multicast groups that may span wide area inter-networks [55]. This protocol acts on sparse mode because it is suitable for groups where a very low percentage of the nodes (and their routers) will subscribe to the multicast session.

The protocol constructs a tree from each sender to the receivers in the multicast group; it uses explicit join messages to set up unidirectional shared distribution trees. In PIM-SM, a router is selected as the Rendezvous Point (RP) and all group communication takes place by sending the packets to it. Each of the sources in a PIM-SM multicast group sends their packets to the RP. Since it builds unidirectional shared tree, only the RP can forward data to the members. Intermediate nodes should forward the data only to the RP. The PIM-SM router with the highest IP address is the Designated Router (DR) for the subnet and is responsible for sending Prune\Join

messages to the RP. Information about RP is obtained by sending Bootstrap messages. The tree obtained is not necessarily optimal.

PIM-SM allows switching of the receiver connectivity to the tree from Shared tree path to Source tree path. When a group has numerous highly active sources, the bandwidth of the shared links may not be able to accommodate all the traffic.

PIM-DM: PIM-DM (Dense Mode) refers to the PIM protocol applied to a dense population [56]. PIM-DM is applied to groups where many of the nodes will subscribe to receive the multicast packets, so that most of the routers must receive and forward these packets (groups of a high density). PIM-DM floods packets everywhere and then prunes on the branches where there were no receivers. It uses source distribution trees and uses RPF checking to determine if a packet is to be forwarded.

The model of tree construction is a brute force method for delivering data to the receivers. This method would be efficient in certain deployments in which there are active receivers on every sub-network. The construction mechanism ,allows the multicast traffic to be spread throughout the network. Routers that have no downstream neighbours prune back the unwanted traffic; so, upon receiving a prune message, the router will modify its state so that it will not forward those packets that interface. If every interface on a router is pruned, the router will also be pruned. This process repeats itself within a certain frequency. Routers accumulate state information by receiving data streams through the flood and prune mechanism. These data streams contain the source and group information so that downstream routers can build up their multicast forwarding table.

1.5 Video on Demand and the Internet

The Internet is the most popular environment where clients are connected; it is a worldwide, publicly accessible network of interconnected computer networks that transmit data by packet switching using the Internet Protocol (IP) standard. The Internet can be described as ‘a network of networks’; it is made up of domestic, academic, business and governmental networks, which together carry various information and services such as electronic mail, online chat, file transfer and interlinked web pages as well as other documents of the World Wide Web. Figure 21 [59] gives an idea of the Internet scale, complexity and heterogeneity; it represents a partial map of the Internet based on data from

15 January 2005 found on *opte.org*. Each line is drawn between two nodes, representing two IP addresses. The length of the lines is indicative of the delay between those two nodes; lines are colour-coded according to their allocation.

Video on Demand represents an attractive commercial service to be offered on a public and global scale environment like the Internet. As VoD service over the Internet becomes more popular, the research has been intensified. The VoD service presents hard constraints and the Internet is a highly dynamic and heterogeneous environment; therefore, a range of issues have to be treated in order to obtain a quality service.

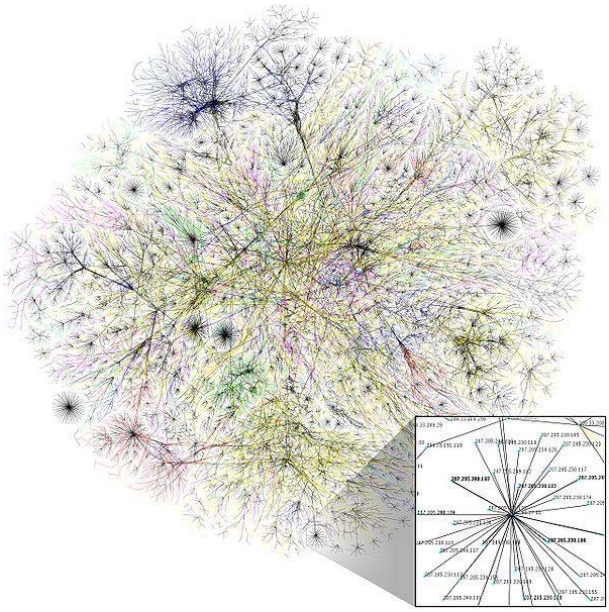


Figure 21. Internet topology representation.

1.5.1 Internet features and VoD deployment

Providing VoD on the Internet is not a single task owing to the particular features of such systems. The VoD service presents hard constraints; the system must be scalable and able to address real-time constraints to guarantee QoS during a video session. On the other hand, the Internet is a highly dynamic and heterogeneous environment; clients connected to the system through a network present distinct processing power, bandwidth, storage capacity and lifetimes.

Therefore, Internet characteristics must be taken into account at the time of VoD implementation. The most significant Internet features and its impact on VoD systems are described below.

Non-dedicated infrastructure and resources: The Internet is a non-dedicated and open environment. This means there is no resource reservation. Information flows through the network in a best-effort way. Therefore, it is impossible on the Internet to guarantee a fixed bandwidth on the VoD service in order to maintain the QoS. Clients must be able to deal with the variation in the reception data rate. Several codification formats for multimedia contents have been proposed to deal with the degradation of network bandwidth, although the usual solution to achieve a smooth playback is the use of buffers in client design. Moreover, the input bandwidth of a client connected to the Internet is generally smaller than a video play rate of good quality [76] [79]; the upload and download capacities are also asymmetric, with upload bandwidth smaller than the download. To deal with these issues, some video services assume a cushion buffer (as explained in client design). It is, however, a solution for short-time bandwidth fluctuations; in addition, the user experience is compromised owing to the service start-up delay. On the Internet, VoD applications have to constantly adapt to the available resources and share them with many other concurrent applications.

Highly heterogeneous environment: The Internet is extremely heterogeneous. This heterogeneity can be observed at hardware and software levels. The hardware's heterogeneity is present in the different technologies applied to network linking, such as optical fibre, copper cables or wireless connections. In addition, the Internet is made up of elements positioned on the path between the source and the destiny; these elements can be of various types, like repeaters, routers, switches or hubs, for example. This variability in the hardware directly impacts on system performance (e.g. data transfer rates and latencies). At the same time, even if hardware components are similar, the heterogeneity appears in the software. There is a huge variety of protocols which define the rules on information exchange. This set of communication protocols is known as the *Internet Protocol Suite*, which is made up of protocols of the Application Layer (DNS, RTP, RTSP, etc.), Transport Layer (TCP, UDP, etc.), Internet Layer (IPv6, IGPM, etc.) and Link Layer (ARP, OSPF, etc.). These heterogeneous features must be considered at the time of designing a VoD system and are crucial to guarantee scalability and QoS on the Internet.

Huge scale: The spread of the Internet surpasses the size of any other system or computer science infrastructure at worldwide level. This environment offers a service to millions of users geographically distributed throughout the world. The Internet provides the necessary infrastructure to deploy VoD systems on a global scale.

All the types of networks present on the Internet (e.g. LAN, WAN, MAN) are organised in Autonomous Systems (AS). The AS is a network domain with a well-defined routing policy that is under the control of one or more network operators. The organisation of the Internet in networks of distinct sizes governed by a set of ASs matches with the VoD feature of servers distributed in order to achieve a large-scale service. Video servers and proxy-servers shall be placed inside the Internet ASs, taking advantage of the Internet organization.

Security: The Internet is a wide-open and accessible environment. This means the information available on the system can be accessed by anyone. Moreover, by applying P2P, clients exchange information directly, without the verification of any entity. This is a great advantage in order to deploy LVoD systems, making the service accessible to a great number of users. The ease of access to the system, however, can represent a serious problem in terms of copyright and malicious users. A VoD service running over the Internet must guarantee the integrity of the content and respect copyright; it is important to adopt security mechanisms to prevent illegal copies, modification of the contents and unauthorised forwarding of videos.

1.5.2 Internet multimedia streaming

Multimedia streaming technology has been widely used on the Internet, enabling worldwide content distribution. The applications are various, such as news, education, training, entertainment or advertising. Video streaming on the Internet can be basically divided into two categories: real-time transmission of live events and streaming of on-demand contents.

In real-time events, clients are able to visualise the video starting from the moment that they join the system. This kind of system generally can be classified as ‘Internet TV’, which refers to the transmission of multimedia streams over IP networks. An Internet TV provider has no control over the final delivery and so broadcasts on a best-effort basis. Internet TV

rides on existing infrastructure including broadband, ADSL, Wi-Fi, cable and satellite, which makes it a valuable tool for a wide variety of service providers and content owners.

The majority of Internet TV systems currently available are implanted using the P2P strategy. These systems rely on peer-to-peer software applications designed to redistribute video streams in real-time on a P2P network; the distributed video streams are typically TV channels from all over the world. In a P2P-TV system, each user, while downloading a video stream, is simultaneously also uploading that stream to other users, thus contributing to the overall available bandwidth. The arriving streams are typically a few minutes time-delayed compared with the original sources. The video quality of the channels usually depends on how many users are watching; the quality is better if there are more users. The architecture of many P2P-TV networks can be thought of as real-time versions of BitTorrent [28]: if a user wishes to view a certain channel, the P2P-TV software contacts a 'tracker server' for that channel in order to obtain addresses of peers who distribute that channel; it then contacts these peers to receive the feed. The tracker records the user's address, so that it can be given to other users who wish to view the same channel. In effect, this creates an overlay network on top of the regular Internet for the distribution of real-time video content. The main drawbacks of these services are the lack of interactivity (users cannot choose to see an event from the beginning) and the non-guaranteed QoS. Each client is part of a chain of viewers where everyone can have a negative influence on the reliability of the stream (by having a low processing power or limited download\upload connection) [60].

There are many P2P-TV implementations available on today's Internet. Some examples of this kind of system are: PPStream [61], PPLive [62] and Justin.tv [64]. Justin.tv was founded in San Francisco by Justin Kan. It is a network of diverse channels providing a platform for 'lifecasting' and live video streaming of events online. The original Justin.tv was a single channel that evolved into the Justin.tv network of thousands of diverse channels. Wearing a webcam attached to a cap, Kan decided to start streaming continuously video and audio of his own day-by-day activities. In 2007, Justin.tv became a platform for more than 60 different channels, which were listed by popularity. In 2008, selectable categories were added for broadcasters, including: People & Lifecasting, Sports, Music & Radio, Animals, Entertainment and others. Currently, the international locations range from Australia, Brazil, the United Kingdom and France to the Netherlands, Sweden and Spain.

On the other hand, on-demand multimedia systems provide a service to clients who request contents at any given moment, starting the visualisation always at the beginning. Clients should be enabled to interact with the content through pause, rewind or fast-forward

operations. The most famous and popular on-demand video service on the Internet is YouTube [7]. YouTube is a user-generated content system, where millions of users are self-publishing consumers. Nevertheless, contents are delivered at low bit-rates and the system predominantly stores short contents. The content length can be reduced by two orders of magnitude and so can the production time; however, great films and five-minute clips are as different as symphonies and jingles [65] [66]. Figure 22 shows a snapshot of two clips of sports events where it is possible to verify the low quality on full screen exhibition.

We call Video on Demand (VoD) systems the on-demand services which provide big content visualisation. These types of applications handle high resource consumption and long-time sessions. There are some VoD systems running on the Internet, like Joost [67] or MegaVideo [68]. The easiest way to find sites providing online video content are the website aggregators such as OVGuide [69], SurfTheChannel [70] or DosPuntoCeroVision [71]. In these services, sites are submitted by users and selected on the basis of editorial review. Once a site is approved for inclusion, it is categorised according to its niche content. VoD services on the Internet, however, still present strong restrictions of QoS. It is common for a user to deal with long start-up delays, glitches, frozen-frames or even the removal of some content because of copyright infringement.

Figure 22. Snapshots of YouTube video streams.

1.5.3 Peer-to-Peer and multicast video services

Currently, there is a lot of ongoing research on multimedia streaming. Most of the systems make use of P2P and multicast paradigms; therefore, fault tolerance and control mechanisms are extremely important. In this section we assess some video services in order to observe their approaches and the manner in which they handle peer failures. The

evaluation ranges from VoD systems to video streaming services and analyses five parameters:

- The structure and topology of P2P implementation.
- The type of multicast approach (IP Multicast and\or ALM).
- Whether the system is designed for the Internet or not.
- If the system implements any fault tolerance mechanism.
- The existence of a control assessment considering the overhead on dynamic operation and the time constraint.

DirectStream: DirectStream [72] is made up of clients, content servers and an AMDirectory service. The content servers provide the same functionality as in the traditional client-server service model; they store contents in their repository and serve clients' requests so long as sufficient bandwidth is available. Clients in DirectStream act as both a traditional client and as a server. Clients cache a moving window of video and can serve other clients by forwarding the stream. A P2P overlay is formed among clients over which the video stream is forwarded. The AMDirectory provides a lookup service; it keeps track of all servers and clients participating in DirectStream and helps new clients to obtain the required service. The AMDirectory service employs application level multicast as the basis for scalable directory service. For each community sharing the same interest, an application level multicast group is created.

CoopNet: CoopNet [73] is a P2P architecture from the Microsoft laboratory. The system has a server that hosts content and (directly) serves it to clients. CoopNet is only invoked when the server is unable to handle the load imposed by clients. It is based on MDC (Multiple Description Coding) video codec. The streaming media content is divided into multiple sub-streams using MDC and each sub-stream is delivered to the requesting client via a different peer. This improves robustness and also helps balance the load amongst peers.

P2VoD: P2VoD [80] only assumes IP unicast at the network layer. Asynchronous requests of clients are handled by utilising the clients' resources. Each client in P2VoD has a variable-size FIFO buffer to cache the most recent content of the video stream it receives. Existing clients in P2VoD can forward the video stream to a new client as long

as they have enough output bandwidth and still hold the first block of the video file in the buffer. The design introduces the concept of generation and a caching scheme to deal with failures. The caching scheme allows a group of clients, arriving at the system at different times, to share the same video content; such a group forms a generation. In a generation clients have the same range of video blocks in their buffers. When a client (or member) in a generation leaves the system, any remaining member of that same generation (with sufficient bandwidth) can provide the video stream to the abandoned children. Control information exchanged between clients is limited by the number of members in a generation.

DynaPeer: The DynaPeer [77] [78] [79] is not a server-less system; rather, it combines a server-based architecture with a P2P delivery scheme. DynaPeer system is compounded by three different entities: servers, clients/peers and Virtual Servers. Servers are responsible for global system management and accounting; they establish clients' collaboration process and have major responsibility for guaranteeing the QoS. System design considers a distributed video server architecture where the system catalogue is spread among them. Clients use system services, accessing video contents on demand. When clients collaborate with others providing requested services, they are called peers. The Virtual Servers are logical entities, which group a set of peers in order to facilitate their synchronisation and management to provide VoD services. The Virtual Servers are created by each video server only in the local network, grouping a set of peers, to allow peer collaborations. DynaPeer deals with heterogeneous network characteristics by aggregating peers' bandwidth; the system is based on one of two delivery policies (unicast and IP multicast), depending on the technology available on the ISP network.

PPLive: PPLive [62] [63] was created in Huazhong University of Science and Technology, China and is a well-known IPTV (Internet Protocol Television) service. The system streams live TV and video data through overlays of cooperative peers. The PPLive system has multiple channels, each of which forms its own overlay. Each channel streams either live audio-video feeds or movies according to a preset schedule. An end host may join any channel; it starts receiving the stream and relays feeds to other users. When an end-user starts the PPLive software, it joins the PPLive network and becomes a PPLive peer node. It then sends out a query message to the PPLive channel

server to obtain an updated channel list. Before a peer actually starts to watch a channel, it does not exchange data with other PPLive peers. After a peer selects one channel to watch, it sends out multiple query messages to some root servers to retrieve an online peer list for this channel. Peers are identified by their IP addresses and port numbers on the list. Upon receiving a peer list, the PPLive client sends out probes to peers on the list to find active peers for the channel of interest. Some active peers may also return their own peer lists, helping the initial peer to find more peers. Peers then share video blocks with each other.

P2Cast: P2Cast [81] is an architecture that uses a peer-to-peer approach to stream video cooperatively using patching [12], while relying only on unicast connections among peers. In P2Cast, each client acts as a server while it receives the video. The clients not only receive the requested stream, but also contribute to the overall VoD service by forwarding the stream to other clients and caching and serving the initial part of the stream. The P2Cast allow clients to forward the video data to more than one client, creating an ALM delivery tree. When a new client joins the session, it joins the base tree and begins receiving the ongoing base stream. Meanwhile, this new client must obtain the initial part of the video from the server or another client. This initial portion, known as the patch stream, starts from when the video begins streaming on the base tree to the time when the new client joins the system. P2Cast is based on a centralised server, which is the single contact point for client failure management. The fault-tolerance mechanism is based on the recursive reconstruction of the delivery tree. The recovery process is identical to a new client joining. That means that the server or peer with more available output bandwidth is selected to provide the stream.

P^{n}2P^n: This system considers distributed servers which store different parts of the catalogue [74] [75] [76]. The placement policy applied to the content distribution over servers is called cache and mirror [15]. The design considers local networks enabled with IP Multicast and works by exploiting clients' non-active resources in two ways. First, it allows clients to collaborate with the server in the delivery of initial portions of video (patch streams); and second, it provides collaboration groups, which are responsible for merging multicast channels that are separated by a time window. The peers of collaboration groups cache video information from an ongoing server multicast channel and bypass the stream to other clients.

BitToS: BitToS [19] is the acronym for BitTorrent Streaming, a protocol with the ability to support real-time streaming based on BitTorrent. It relies on mechanisms of piece (file blocks) selection, which is considered the only aspect that demands changing from the original BitTorrent protocol. BiToS is aware of the streaming order of the pieces, thus preferring pieces that will be played soon. The approach tries to reach a balance between downloading pieces in playing order, enabling smooth playback, and the rarest first order, enabling the use of parallel downloading of pieces. The mechanism consists of:

- Received Pieces: Contains all the downloaded pieces of the video stream that the peer has ever downloaded. The state of a piece can be Downloaded, Not-Downloaded or Missed. A piece has is Missed if it did not meet its deadline to be reproduced by the player.
- High Priority Set: Contains the pieces of the video stream that have not been Downloaded yet are not Missed and are close to being reproduced by the player. Thus, these pieces have higher priority in being requested over the rest of the pieces. A piece in this set can be in the following states: Not-Requested or Currently-Downloading.
- Remaining Pieces Set: Contains the pieces that have not been Downloaded, are not Missed and are not in the High Priority Set. The status of the pieces can be the same as the High Priority Set.
- .

Promise: PeeR-tO-peer MedIa StrEaming system, PROMISE [82], is a video streaming architecture based on structured strategy for location. PROMISE uses Pastry as the P2P substrate i.e. to discover the necessary information and active peers. The design of PROMISE is based on CollectCast [83]. CollectCast is a mechanism for selecting peers based on their topology interconnection information. The information about topology interconnection is obtained with network tomography [84] and avoids congestion over network links. Peer failures are detected in two ways: from the TCP control channel established between the receiver and each of the sending peers and if the rate coming from these peers is degraded. Once a failure is detected, the active set is adjusted by replacement of the failed peer with new one(s). The replacement of peers

applies a topology-aware selection, which avoids tearing down all the old connections and establishing new ones.

GloVE: Global Video Environment (GloVE) [85] is a prototype video system developed at Federal University of Rio de Janeiro (UFRJ), Brazil. The GloVE approach is based on CVC, Cooperative Video Caching [86]. CVC manages all the client buffers as a single global distributed memory that can be randomly accessed over the communication network. This feature provides scalable performance to conventional VoD servers. The system behaves like a P2P approach with a centralised metadata component that is responsible for monitoring the distributed video contents across the clients' local buffers and enabling their reuse by establishing cooperative streams between clients. GloVE was designed to operate over Ethernet-based communication networks, using switches to interconnect active clients themselves and the server to active clients; it adopts IP Multicast implementation.

Table 1 summarises the main aspects of the evaluated video services.

	Service	**P2P**	**Multicast**	**Internet**	**Fault Tolerance**	**Control Assessment**
DirectStream	VoD	unstructured tree	ALM	✓	✓	x
CoopNet	Streaming\ VoD	unstructured tree	ALM	x	✓	x
P2VoD	VoD	unstructured tree	ALM	x	✓	x
DynaPeer	VoD	unstructured tree	IP\ALM	✓	✓	x
PPLive	streaming	unstructured mesh	ALM	✓	x	x
P2Cast	VoD	unstructured tree	ALM	✓	✓	x
P^{n}2P^n	VoD	unstructured mesh	IP	x	✓	x
BitToS	VoD	unstructured mesh	ALM	✓	✓	x
Promise	streaming	structured mesh	ALM	✓	✓	x
GloVE	VoD	unstructured tree	IP\ALM	x	x	x

Table 1. Evaluation of P2P and Multicast multimedia stream architectures.

The assessed video services are of multimedia streaming and VoD sorts. The difference lies in client's interaction. Basically, in multimedia streaming users are able to visualise the content starting from the moment that they join the system and VoD systems provide service to clients who request contents at any given moment, always starting the visualisation at the beginning.

It is possible to observe that almost all of the video systems present unstructured P2P (overlay links are established arbitrarily). This is natural, since the connection between peers depends on the arrival time and the requested content. The topology assumed by the overlay network (mesh\tree) is related to peer communications, whether the content is striped into many parts or not. Hence, it leads to the presence of multiple or single information sources. Peers receiving and sending data from\to multiple peers result in a mesh topology; on the other hand, peers receiving information from a unique source and sending to many peers create a tree.

Systems without video servers such as Promise and BitToS are presently not viable for legal and reliable video service on the Internet. The content catalogue must be under control; totally distributed P2P approaches cannot guarantee content trustworthiness and availability (clients are able to modify, publish and remove files). Distributing videos (partially or entirely) to the Internet peers also means copyright protection is practically impossible.

Since the Internet is the most feasible infrastructure to provide VoD deployment at the global level, it is essential that its features are considered at the time the system is designed. Most of the analysed video services only consider unicast communications at network level; the multicast approach is implemented just at the application level (ALM). The Internet is constantly increasing in size and requirements; soon, IP Multicast will play a prominent role in the Internet scenario. Moreover, nowadays there are real solutions to multicast transmission on the Internet. One of the more representative solutions is the virtual networking MBONE (IP multicast Backbone) [87]. The MBONE architecture uses special network zones that physically support communications of type IP Multicast; it is able to interconnect these zones at worldwide level. MBONE defines routing protocols which allow exchange of information between routers through predefined tunnels. At one tunnel extremity, IP Multicast datagrams are 'encapsuled' in IP unicast datagrams and are sent through conventional routers. At the other extremity of the tunnel, the IP unicast datagrams are extracted and injected into the local network, emulating a global multicast network. Although MBONE is focused

initially on video-conference applications, the architecture is a generic solution for all the systems that need multicast on Internet.

As mentioned earlier, end hosts are heterogeneous in terms of resources and lifetimes. The variability in processing power or output bandwidth of a peer leads to different reception rates at the client side. Most designs do not consider this difference and adopt a peer output rate constant greater than or equal to the video play rate (e.g. P2VoD, P2Cast, GloVE); therefore, a client always receive the video stream in an 'instantaneous' playable time. It is clear this approach is not practicable to P2P-VoD systems on the Internet. Some approaches (e.g. P^n2P^n) assume a cushion buffer in order to deal with the temporal variability of the reception rate. The information is previously stored at the client buffer for further consumption; it enables a smooth playback when the input bandwidth fluctuates. The drawback of this solution is the degradation of user experience, as the user must wait for the cushion buffer to fill before starting to enjoy the service (start-up delay). DynaPeer applies the same strategy, but it also uses a scheme to aggregate peers' bandwidth in order to guarantee a client input rate equal to the video play rate. The mechanism groups collaborators and assigns video blocks proportionally to the peers' output capacity; thus the peers are synchronised to provide the information emulating a unique source.

The heterogeneity of peers' lifetimes refers to the unpredictable behaviour of users on P2P systems; it demands solutions in order to avoid service disruption by peer departure. This means that video services should present a strategy to tolerate peer failures. The natural method is to use the inherent redundancy of P2P video systems, i.e. the presence of multiple information sources. CoopNet, BitToS and Promise, for example, stripe contents into various blocks; in case of peer failure a substitute source with the desired video block is sought. DirectStream and DynaPeer attribute special functions to some peers (e.g. helpers, backup peers) in order to improve the search process. P2VoD and P2Cast look for replacement peers in different ways. P2VoD performs the search only inside the same generation of the failed peer and P2Cast sends multiple queries in a recursive recovery process. In P2Cast, when a client is disrupted by a failure, all clients belonging to the sub-tree rooted at this client are affected; only the head orphan in the tree is allowed to contact the server to perform recovery. If the process succeeds, the entire sub-tree is recovered. If it fails, the orphan client is rejected and its children will contact the server and restart the recovery process. The process is repeated until the leaf client is reached or recovery succeeds. The P^n2P^n system

coordinately switches peers during the service; one client each time acts as data source. This gives extra time for recovery, since the peer failure can occur when it is not the peer's turn to serve. In the worst case (collaborator leaves while serving), however, a peer with the same content must be found or the server assumes the service (if it has available resources).

All the presented fault tolerance mechanisms designed to deal with peer departures evaluate the control by the communication protocol complexity. The protocol complexity statically represents the messages interchanged to operate the failure process. Generally, current researches do not assess the control impact dynamically under a continuous process of peers joining and leaving. Nor do they consider underlying network influence when several changes of sources are performed. Finally, the deadline of the failure management process is not taken into account; the time efficiency of the control scheme, necessary to maintain the QoS, is not measured.

1.6 Control Importance and Fault Tolerance in Internet P2P-VoD Systems

The good experience of users on enjoying multimedia contents basically depends on the absence of errors during the visualisation. In this topic we discuss the control sub-system, which is very important on multimedia services, mainly when P2P and Multicast techniques are applied. In order to guarantee QoS on Internet P2P-VoD systems, we also discuss Fault Tolerance issues.

1.6.1 Control relevance

In order to improve the performance of VoD systems and achieve a large-scale service, P2P and multicast paradigms have been applied. P2P architectures allow the use of clients' available resources to decentralise the system load and to provide system scalability. A P2P system does not have the notion of clients or servers but only equal *peer* nodes that simultaneously function as both 'clients' and 'servers'. This model of network arrangement differs from the client-server model where communication is usually from and to a central server.

The use of P2P paradigms for video streaming on Internet leads to a range of new issues such as free-riders that reduce the efficiency of the system by not contributing or the

heterogeneity of clients [88]. On the Internet, users have different capabilities such as the processing power or bandwidth; clients' heterogeneity can lead to information loss in the delivery process. Clients also come and go freely, so the sudden departure of a peer can degrade the QoS owing to interruptions in the video information flow.

In the same way, the multicast communication decreases the network load by eliminating redundancy in the data transfer; the information flow from a source to a group of users who request the same content. Recent advances provide IP Multicast, making many local networks multicast-capable on today's Internet. The IP Multicast implementation only supports UDP transmissions. Nevertheless, in order to achieve global multicast, Application Layer Multicast (ALM) has been proposed. In ALM, groups of nodes form an overlay network and data packets are relayed from one member to another via unicast, which can be either TCP or UDP.

A node may suffer packet loss when a path is congested, when it fails or if the sender leaves the system. In IP Multicast, a node leaving or suddenly failing does not affect other nodes. If, however, the information source changes, network routers must rearrange the data distribution tree. Moreover, there is the feedback implosion issue; if every receiver reports the success or failure of the data transfer, the sender will be overwhelmed. In ALM, the departure or failure of a node leads to packet loss at all its descendants.

Clearly, high node dynamicity and huge amounts of nodes create more challenges in terms of failure management in VoD systems based on P2P and multicast. In sum, a VoD service on the Internet brings many challenges; moreover, the features of P2P and multicast paradigms add new issues to the system. Despite the availability of many solutions, the P2P and multicast usage make VoD services over the Internet an active research area with many challenging problems to be addressed. In the following, we mention a range of topics that demand special attention in order to achieve a feasible approach to Internet VoD services based on P2P and multicast techniques.

Content trustworthiness: With P2P-based video distribution, users have access to multimedia information not only from the server, but also from other users that have already handled the same data. The content provider no longer has control of the machines that distribute the videos; thus, it is necessary to consider the trustworthiness of the content. Copyright is a form of intellectual property which gives the creator of an original work exclusive rights in relation to that work, including its publication, distribution and adaptation. A P2P-VoD system must be able to respect the copyrights as well as guarantee

that clients have no opportunity to add fake versions of popular contents. To deal with such problems, P2P-based VoD systems need a way to identify the video provider and to prevent content modification.

Network status: The multimedia content's nature makes it highly sensitive to transmission over the Internet, which cannot guarantee the availability of resources. The network condition during a streaming session can be changed dramatically owing to concurrent services flowing through the Internet or even to the dynamic nature of P2P-VoD services. Therefore, it is important to monitor the network conditions regularly and balance the load injected into the system. The managing of network conditions is necessary for clever utilisation of available resources and to reduce the packet drop ratios.

Communications scheme: The service scheme defines the way information is sent from a source to the respective destination. The multicast approach is the most suitable solution to provide VoD service. The Internet, however, presents a range of different network infrastructures. The design of the system must take into account the level of the multicast implementation, i.e. whether the multicast is available at the network layer (IP Multicast) or at the application layer (ALM).

Peer heterogeneity: Peers are heterogeneous in their capabilities. Users are endowed with distinct hardware (i.e. microprocessors, memory, buses, etc.), presenting various processing powers. With respect to the bandwidth, the heterogeneity can be caused by different access networks connecting the peers, various types of network interface controllers or simply by difference in the willingness of the peers to contribute. The system must be prepared to face the different capabilities of each peer.

Peer dynamicity: Since the peers are end-user terminals, their behaviour remains unpredictable. Owing to the dynamic nature of P2P networks, they are free to join and leave the service at any time without making any prior notification to other nodes. Thus, dynamicity management is crucial for the smooth playback rate during the streaming sessions. To prevent service disruption owing to peer connection\disconnection, a robust and adaptive mechanism is necessary to manage such changes. The proposed mechanism must incorporate a recovery phase to tackle the sudden changes occurring in the environment.

When a sender peer leaves the system, it must be detected as early as possible and a replacement made from another source to perform a quality service.

1.6.2 Fault tolerance

One of the main challenges of VoD services is guaranteeing a good user experience, without any kind of error during the entire session. The maintenance of a certain level of system performance is referred to as Quality of Service (QoS).

P2P and multicast appear as important players to provide system scalability and higher performance, which are achieved by sharing resources. These paradigms, however, place new demands on video services design. Peers are free and thus can appear on or disappear from the system at any time. Multicast needs topology reconfiguration when a node departs, whether a change of source (IP Multicast) or node replacement in the distribution structure (ALM).

Normally, the terms error and fault are indiscriminately used; however, in computer science they have different meanings. *A fault (or failure) is a defect which compromises the behaviour of a system, making it deviate from that specified in the design. Hence, the consequence of a fault is an error*. Figure 23 illustrates the different aspects of fault tolerance (failures and errors) and shows the main solutions applied [89] [90].

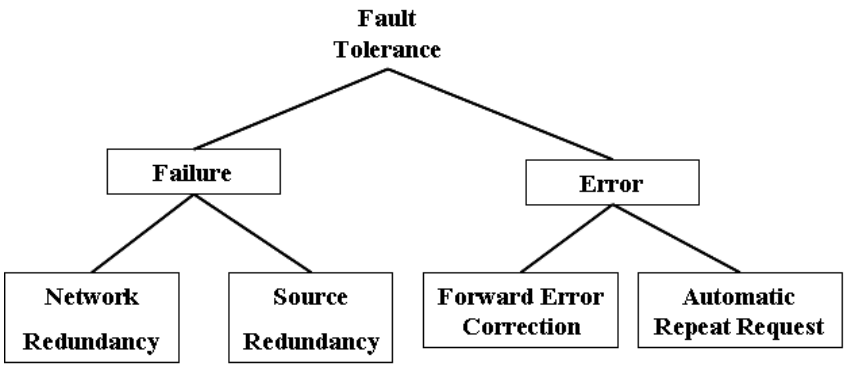


Figure 23. Fault tolerance aspects and solutions.

In VoD systems, errors are noticed by users in the content visualisation as jitters, frozen-frames, glitches or a lack of audio and video synchronisation. The errors can be solved, basically, by two strategies:

- ***Automatic repeat-request (ARQ)***: With ARQ, a lost data packet will be retransmitted by the sender. ARQ-based schemes consist of three parts:
  - Lost data detection: Loss can be detected by the receiver (gap-based loss detection or timeout) or by the sender (timeout).
  - Acknowledgment strategy: The receiver sends acknowledgments that either indicate which data have been received (positive ACKs) or which data are missing (negative ACKs referred as NACKs) and should be retransmitted.
  - Retransmission strategy: This determines which data are retransmitted in case of loss. The two best known retransmission strategies are the re-send of a well-defined block of information or a selective retransmission, which trade off simplicity of implementation and transmission efficiency.

  The mechanism basically works like this: the transmitter sends the data and also an error detection code, which the receiver uses to check for errors; hence the receiver requests retransmission of erroneous data through a NACK. In many cases, the request is implicit; the receiver sends an ACK of correctly received data and the transmitter re-sends anything not acknowledged within a reasonable period of time. ARQ strategy can overwhelm the source on the multicast transmission, since a considerable set of receivers can be enabled.

- ***Forward error correction (FEC)***: The idea of FEC is to get the transmission right first time. For this purpose, FEC transmits together with original data some redundant data, called parities, to allow reconstruction of lost packets at the receiver. The redundant data are derived from the original data using techniques from coding theory. The transmitter encodes the data with an error-correcting code (ECC) and sends the coded message. The receiver never sends any messages back to the transmitter. The receiver decodes what it receives and can use the redundant data to reconstruct the information in case of errors.

  The recovery of lost data by reconstruction at the receiver requires very little time, which makes FEC attractive for applications with real-time requirements. The FEC transmitter sends k data packets (defining a

transmission group) and adds additional h redundant parity packets. The ratio h/k is called over-code and indicates the amount of redundancy added. Unless the network drops more than h of the $h+k$ packets sent, the receiver can reconstruct the original k information packets.

Error correction strategies are very useful to guarantee a good user experience, since they recover information avoiding immediate loss of QoS. Nevertheless, they do not solve the problem. To achieve a robust and reliable service, it is necessary to treat the fault which caused the error.

As previously mentioned, fault tolerance consists of keeping the system working without errors by masking failures through the use of redundancy. There are two basic forms of redundancy on the VoD system: hardware and information. Owing to the distributed nature of LVoD systems, hardware redundancy is observed on multiple servers, networks and clients. Simultaneous and asynchronous clients' requests are responsible for information time redundancy, i.e. contents are flowing through the system to more than one destination at the same time and two video streams can be separated by a time window. These features might be applied to provide fault tolerance in VoD services.

The errors observed on video systems can be caused by three different factors: client, network or source failures. A client fault demands own client's attitude to find the origin of the problem (e.g. on the I/O interface); in other words, the system cannot do anything if the fault occurs inside the receiver element.

A network failure interrupts the information transmission, making communication between source and destination impossible, even though they work correctly. Network faults can be caused by link disruption, an element crash (router, switch, etc.) or path congestion. A classical solution for network failures is to provide redundancy by using alternative routes to deliver the information. Multiple paths are supplied through the network in order to guarantee the data delivery. The effectiveness of this solution depends on the network level where the failure occurred; it demands the redundancy of network hardware (cables, routers, etc.), which may not be available at the local level.

The source in P2P-VoD systems can be either a server or an end host. If the source of information fails, the only possibility of maintaining the service is to seek another source. In the case of server sources, when the server fails the solution is to migrate the service to another server which has the same content [96]. If the migration does not succeed, the service is interrupted. Distributed servers are the main strategy in LVoD deployment;

however, in order to guarantee scalability not every server hosts the same portion of the catalogue. Contents are distributed according to a placement policy; thus, finding a new server source may not be an easy task. Servers can be saturated serving popular contents or simply do not possess the required file. Fault tolerance can be achieved by adding dedicated backup servers, which provide the hardware and information redundancy and are able to cover failures of a range of servers. Nevertheless, this infrastructure redundancy requires an investment and represents extra costs to the content provider.

When the source is a peer, the service is subject to its presence on the system. Since peers are free to enter and leave at any moment, a peer departure is treated as a failure by the system. In this circumstance fault tolerance can be achieved by searching for new collaboration; another peer, who has the desired content, can replace the failed one. Peer search and replacement is not a trivial task, however, because it demands excellent peer communication and synchronisation.

In this work we focus on peer failures, since P2P and multicast have been largely used in video services over the Internet. Peer faults can be one of the most recurrent causes of errors in P2P-VoD systems. Moreover, changing the source peer can avoid errors at the client end owing to network problems or congestion.

Fault Tolerance on Internet video services still suffers from a lack of research. The control complexity added to the system by P2P and multicast strategies is strongly neglected in current VoD designs, which are more focused on the application level logic for data transmission. The VoD service imposes an important constraint on Fault Tolerance: the failure treatment is time-limited, i.e. the delay of the failure management process must guarantee error absence and consequently maintain the QoS. Monitoring peers means adding control to the system; the control mechanism is responsible for managing peer failures by detecting and recovering. The control scheme leads to an extra load flowing through the system (communication messages); this extra load can affect the whole system performance and is called control overhead.

A good control mechanism is of great importance in P2P-VoD systems, because it provides coordination and synchronisation among system elements and guarantees the reliability and accuracy of the service. Nevertheless, the control scheme demands a careful design owing to the soft real-time restriction and the overhead imposed on the system.

1.7 Goal of the Thesis

In this chapter we have presented an introduction to Video on Demand, specifying the main characteristics of the service as well as its elements. Owing to the nature of the service, VoD systems present strong requirements for storage capacity and bandwidth; such requirements demand a scalable design in order to provide a large-scale service able to handle many requests and a huge video catalogue. Moreover, time delays are of keen importance in a VoD service, which has soft real-time features. The information delivery process must respect a threshold in order to guarantee a smooth playback, without any kind of error and therefore maintaining the QoS to the clients. Achieving a large-scale VoD system in the first instance flounders on the infrastructure necessary for the service deployment.

In parallel, the Internet is constantly growing and, in the last few years, has become the main environment used to distribute audio and video contents. The Internet is the most popular environment for connected users; it is a publicly accessible environment of interconnected computer networks and is available throughout the world. VoD represents an attractive commercial service to be offered in a public and global environment like the Internet; hence, a lot of efforts are dedicated to research on VoD service deployment over the Internet.

The Internet is a highly dynamic and heterogeneous environment; it is made up of many types of networks and elements and works in a best-effort way. At the same time, the VoD service has many requirements and restrictions. Making the VoD service work in a satisfactory manner over the Internet is not a simple task. The Internet and VoD features must be taken into account at to the time the system is designed.

Many approaches have been proposed to provide Internet VoD applications [19] [72] [74] [79] [80] [81]. On the present proposals the multicast and P2P paradigms appear to be the most disseminated approaches to improve performance and scalability of systems by sharing resources. Nevertheless, the usage of these paradigms in Internet VoD systems invokes several new subjects such as: the multicast implementation level (network heterogeneity in the Internet environment), the impact of the Internet best-effort feature on P2P-VoD systems (no resource reservation on the network), legality and integrity of contents (legal commercial applications), the heterogeneity of end hosts (peer bandwidth and storage capabilities) or peer lifetime in the system (users connections and disconnections).

These issues demand in-depth investigation in order to provide a scalable, efficient, robust and reliable VoD service on the Internet. A strict control mechanism is necessary to deal with the service requirements and Internet characteristics. Current proposals are more orientated to the application level logic for multimedia streaming i.e., the efforts are almost totally focused on improving service performance (number of served requests) and scalability. The control complexity added to the system by P2P and multicast strategies, for example, is strongly neglected; the Internet video services still present a lack of research in this sense. A good control mechanism is crucial for the VoD service; the control sub-system is direct related to the service performance in terms of resources management and QoS.

The evaluation of the control impact over multimedia systems' performance normally is analysed only through the protocol complexity; it establishes the amount of communications necessary to perform tasks such as clients joining processes or topology reconfigurations. Nevertheless, the overhead of a protocol largely depends on the structure used and how it is maintained along the time. Furthermore, the VoD service is a soft real-time application, thus time constraints must be considered on the design of all system's mechanisms; time efficiency can be translated as system performance improvement in terms of lower start-up delay and better clients' buffer usage.

The system must be fault-tolerant, guaranteeing QoS to users even in a dynamic and heterogeneous environment such as the Internet VoD service based on P2P and multicast techniques. The control sub-system must provide a quick response when handling failures and, at the same time, the control overhead inserted by the failure management process must be as low as possible.

The goal of this thesis is to propose a Fault Tolerance Scheme to achieve a robust and reliable VoD service on the Internet. The Fault Tolerance Scheme is responsible for handling the frequent arrival and departure of peers and considering the impact of such mechanisms on the system i.e. the proposed Fault Tolerance Scheme is intended to enforce a low overhead level and solve failures efficiently. The solution treats the main issues arising from the application of P2P and multicast strategies to Internet VoD systems. The great target in designing a VoD system is to achieve a feasible large-scale and high-quality service with lower costs and fewer deployment constraints.

Chapter 2

The Fault Tolerance Scheme for Internet P2P-VoD Service

2.1 System Architecture

Since we have analysed VoD service requirements and Internet constraints in Chapter 1 as well as the P2P and multicast implementations and challenges, in the following we describe what we consider the most suitable architecture for the deployment of large-scale VoD services.

First of all, it is clear that the Internet is the ideal environment to deploy an LVoD service. The Internet is very popular, available worldwide and offers public access. Moreover, the system itself writes the rules for achieving global scale. Its structure is made up of networks with distinct capabilities, at different levels. These networks are organised under the control of one or more network operators (Autonomous System - AS).

The VoD service must take advantage of the Internet organisation when its infrastructure is being built. Distributed video servers must be strategically placed through an AS domain. In addition, proxy-servers might be used to put contents closer to e clients. The use of distributed video servers and proxies adds hierarchy to the system, balancing control and scalability. The servers host contents according to a cache\mirror placement policy [15]. In this policy, the most popular contents are replicated and cached on every server and the rest of the catalogue is distributed among them. This strategy is based on the Pareto principle (also known as the 80-20 rule); it states that, for many events, roughly 80% of the effects come from 20% of the causes [95]. Although it is necessary to conduct more research on users' behaviour, the cache\mirror policy cleverly applies the Pareto principle; thus 20% of the catalogue (popular videos) can be responsible for 80% of the petitions. The server only needs to cache this portion of the catalogue to serve most of the requests locally. The requests for non-popular contents are served locally or remotely depending on the site where the video was placed. The remote requests for less popular contents can be served through an overlay network built by the distributed servers. The distributed architecture of servers based on the Internet structure

and the content placement policy conjugate a feasible solution to accomplish the system storage requirement and to provide service scalability. Also, a server-less architecture, only based on P2P, is not viable to VoD service implementation, since it is very difficult to guarantee content legality and trustworthiness.

The architecture design must also be aware of the heterogeneity of the networks that make up the Internet. Many local networks in today's Internet are multicast-enabled; therefore, the IP Multicast communication must be used when available. Global IP multicast, however, is still hindered by many management and technical difficulties. This is because routers interconnecting these local multicast-capable networks, or so-called 'islands', are often either multicast-incapable or multicast-disabled. In this sense, the architecture can implement overlay techniques (overlay multicast and\or ALM) among the multicast islands. Appropriately applying IP Multicast at local level (when it is available), and overlay schemes on inter-networks communications, favours service scalability.

The P2P paradigm application improves system performance by decentralising the server load; peers' collaborations can run on any multicast approach (IP, overlay multicast or ALM). Connections between peers might be established at the AS level in order to concentrate the P2P traffic. The P2P paradigm depends on the availability of information. Highly popular videos mean a great range of end hosts that can be used on collaborations; conversely, contents with low popularity tend to be attended to by the server since, probably, few dispersal clients have made the same petition. This way, with popular contents replicated at the servers on the local networks, collaboration between end hosts can be well exploited at the AS level. Although peer collaborations' inter-networks (inter AS) are feasible, the gain may be limited; the complexity of dealing with large-scale P2P overlay topologies tends to be very high.

In addition, the system must deal with peers' heterogeneity (resources and lifetime) in order to respect the time constraint of the VoD service and thus maintain the QoS. To achieve this aim, end hosts must use their buffer, managing different portions to accomplish distinct tasks. The size and division of the buffer are defined in the client design; the information is first buffered by the client and then consumed at the video playback rate. Moreover, some strategy must be adopted to deal with low bandwidth availability; peers with bandwidth smaller than the video play rate must be enabled to collaborate. The usage of end hosts' buffers and bandwidth aggregation techniques are adopted to deal with peers' heterogeneity and the Internet best-effort nature. Also,

multimedia information stored at the buffer is a viable solution for implementing P2P collaborations in video services because it respects copyright and can guarantee that content is trustworthy. The entire video file is in charge of the content provider: end hosts handle only portions of multimedia files; the information stored on the buffer is always changing according to the visualisation time. When the end host turns off, the multimedia data disappear from its memory. Feasible system architecture for running a VoD service on a large Internet AS is illustrated by figure 24.

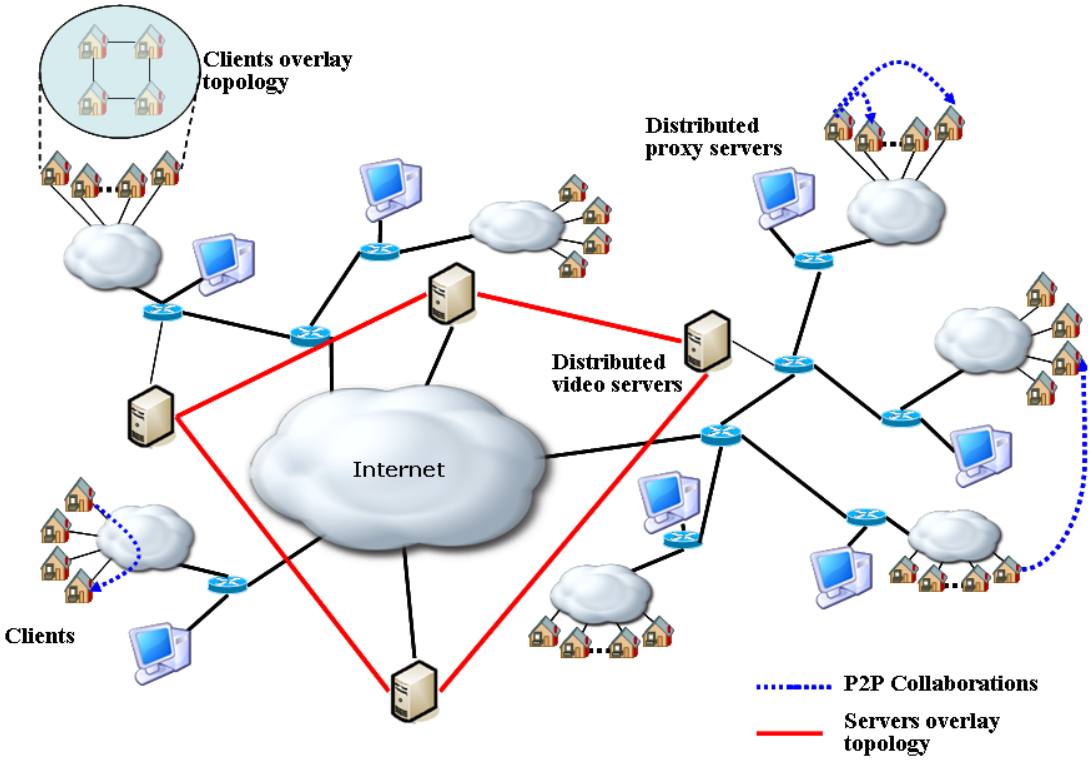


Figure 24. LVoD system architecture on an Internet AS.

A key feature of VoD system design is the service policy. The service policy is the strategy defined to provide multimedia contents to clients. The objective of this part of the work is not to evaluate complexity and performance of distinct service policies, nor to propose a new one. Much research is oriented to this end [79] [80] [82]. Here, we aim to define a feasible architecture for an Internet P2P-VoD system; the service policies can be diverse. Also, owing to the large scale of the system, each network level demands a solution in time to serve petitions. Inside IP Multicast islands, for example, a solution like P^n2P^n may be the most suitable; between islands, but still inside the AS, P2Cast or

P2VoD policies might represent a better solution. In the next topic we describe the service schemes adopted in our evaluations.

An efficient implementation of a P2P-VoD service over the Internet must be aware of the constant peer connections and disconnections. The focus of the work is to assess the dynamic involved in peer failures and in providing a robust and reliable fault tolerance scheme. Fault tolerance is one of the most important requirements of the VoD service on the Internet; it makes it possible to deal with unpredictable end host behaviour or even with network failures. Basically, the fault tolerance mechanism aims to change the data source (peer); the process of managing faults must aggregate low overheads on the system and respect the service time constraint.

Table 2 summarises the key features that we consider necessary to assess a feasible, reliable and large-scale multimedia service.

	Service	**P2P**	**Multicast**	**Internet**	**Fault Tolerance**	**Control Assessment**
System Architecture	VoD	unstructured	IP\ALM	✓	✓	✓

Table 2. System features.

2.2 PCM/MCDB and P2Cast Service Schemes

In VoD systems, the service scheme establishes how the multimedia information is delivered to clients. Aiming to achieve better performance, most of the current VoD delivery mechanisms apply P2P and multicast paradigms. As mentioned in Chapter 1, the main P2P topologies for video stream are mesh-based and tree-based; the multicast communications are generally implemented through IP Multicast or ALM (multicast at the application layer).

In this work, we assume the Internet as the environment of VoD service implementation. In this sense, we aim to analyse different levels of the system; we are interested in service schemes that can run on local networks with IP Multicast support as well as ALM delivery mechanisms that can be implemented between the local networks, i.e. at the AS level. The study is also oriented to observe the mesh and tree P2P topologies in order to evaluate their influence on the FMP. To accomplish this objective, we have selected for analysis two VoD service schemes, the PCM/MCDB [74] [75] and P2Cast [81].

The PCM/MCDB is the delivery mechanism of the P^nP^n VoD architecture; it is a service policy developed by our group in previous research. The P^nP^n architecture is designed for a large-scale VoD service; it considers distributed servers, P2P collaborations and multicast communication to achieve a high degree of scalability and performance. The PCM/MCDB delivery mechanism is designed to exploit P2P collaborations, acting on local networks with IP Multicast support. This scheme implements collaboration groups of peers organised in a mesh-based topology.

The P2Cast is an architecture proposed by Guo et al. to achieve scalability on Internet VoD service. It uses the P2P approach cooperatively to stream multimedia data while only relying on unicast connections among peers. The key idea of P2Cast is to have each client acting as data source while it receives the video. This strategy creates an ALM tree-based architecture, which can be applied at the Internet AS level.

Moreover, both service schemes adopts the patching [12] multicast strategy to group clients and serve petitions. Patching has proved to be an excellent technique for multicast implementation; this strategy dynamically assigns clients to join ongoing multicast channels and patches the missing portion of the video. It can be considered a good solution to multicast implementation on multimedia streaming services because it avoids start-up delay and relies only on buffer requirements at the client side.

Another feature was considered in the PCM/MCDB and P2Cast selection: the approach used to deal with peer failures. Both service schemes are enabled to use the different detection techniques (heartbeats and income-stream monitoring); also, each of them applies distinct recovery and maintenance strategies. In PCM/MCDB the local server maintains a list of the peers with their updated status. When a peer fails, the server indicates a substitute end host or assumes the service. The maintenance phase provides the updated information about peers and manages the multicast delivery trees at the network layer. The routers exchange messages to create, maintain or rearrange multicast groups; they apply protocols such as IGMP and PIM to multicast implementation.

The P2Cast scheme is also based on a centralised server, which is the single contact point for peer failure management. Failure management is based on the recursive reconstruction of the delivery tree. Subsequent queries are sent by the server to other peers, asking for their status and availability. If the process succeeds, the entire sub-tree is recovered. If it fails, the orphan client is rejected and its children will contact the server and restart the recovery process. The process is repeated until the leaf client is reached or recovery succeeds. This recovery strategy eliminates the need for the

maintenance stage to provide peers' information; however, its performance depends on the success of the queries. Routers' communications are established to create and maintain the unicast links of the ALM nodes. Routes are kept up to date dynamically, based on such system characteristics as bandwidth or distance. This way, routers must periodically exchange messages and process the calculations of the routing algorithms.

2.2.1 PCM/MCDB

The PCM/MCDB [74] [75] [76] is responsible for conducting the VoD service on the Pn2Pn architecture. It applies P2P collaborations and IP Multicast to improve system scalability and performance. PCM/MCDB uses patching as a multicast technique and builds collaboration groups of peers to alleviate server load. The patching strategy and the P2P collaboration model adopted by this service scheme are based on clients' buffer capacity. In a VoD service clients receive, decode and display the video information. In PCM/MCDB schemes, the client design includes buffers that temporarily cache information for three purposes:

1. A portion of the buffer is used to achieve smooth playback. This portion is called the cushion buffer. The size of this cushion buffer is a system design parameter; it is invariable and based mainly on the video format and the expected variations in network bandwidth.
2. Users cache video information from ongoing channels on the delivery buffer. This buffer is responsible for the implementation of the patching scheme; it caches multimedia data from the multicast base channel while the client consumes information from the patch channel. The size of the delivery buffer changes according to the delivery policy.
3. Any part of the client buffer that is not used for the previous two purposes is used for P2P collaboration. This portion of the buffer caches video information for sending to another client and is called the collaborative buffer.

PCM/MCDB design is based on two policies: Patch Collaboration Manager (PCM) and Multicast Channel Distributed Branching (MCDB).

The PCM mechanism applies the patching technique to provide service and has the aim of creating multicast channels to serve groups of clients. Clients receive video information from multicast (base channel) and unicast (patch channel). The multicast channels are created by the server, whereas the unicast channels could be created either by the server or the clients. Figure 25 shows the PCM scheme.

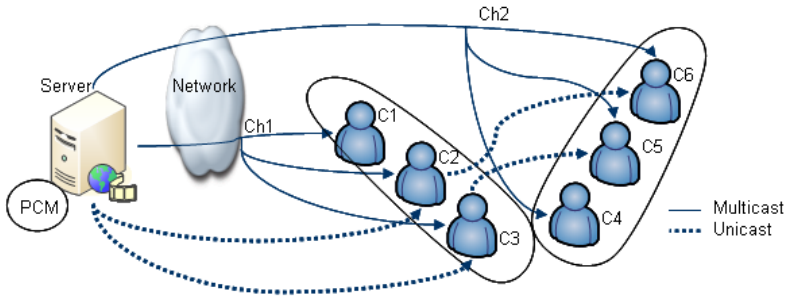


Figure 25. PCM delivery policy of P^n2P^n architecture.

Complementing PCM, the objective of MCDB is to eliminate multicast channels so as to reduce server load [74]. The policy replaces an ongoing multicast channel with a local multicast channel. A group of collaborative end hosts (peers) are organised in a mesh and are synchronised to form a Distributed Collaborative Buffer. The clients in this group use their buffers to cache video blocks from another multicasting channel. The cached blocks are delivered by the peers in order to generate the local multicast channel. Figure 26 illustrates the MCDB mechanism; meanwhile, figure 27 shows how peers combine their resources to form the distributed collaborative buffer.

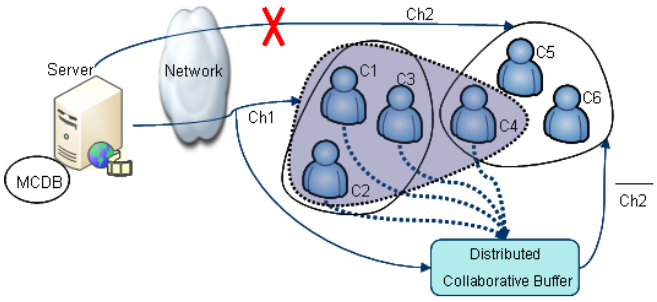


Figure 26. MCDB collaboration policy of P^n2P^n architecture.

PCM/MCDB has distributed detection; the recovery and maintenance phases are centralised. Each peer of a collaboration group exchanges messages with its neighbours; in the streaming process one peer each time acts as data source and the delivery tree is built at the network level.

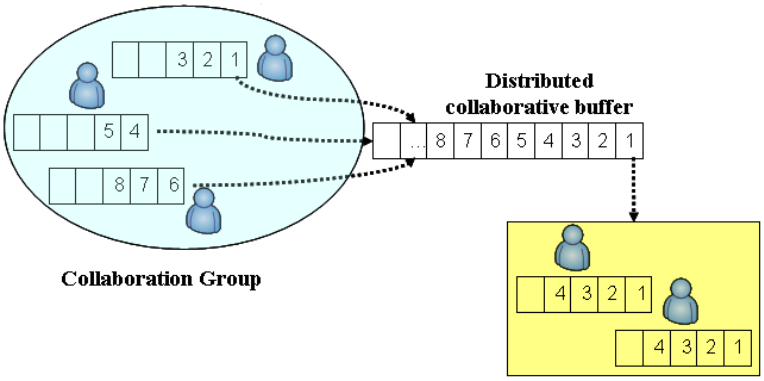


Figure 27. P2P collaboration group.

In the MCDB P2P delivery process, a collaborative client can be in one of four states: (1) Peer is delivering video information. (2) A peer could be waiting to start the delivery process. (3) A peer in the group caches the video information from another multicast channel. (4) Peers could be waiting to start the caching process. The collaborators in a PCM scheme are always in state 1. In figure 28, C1 is delivering video information, C2 is completely buffered and waiting to start delivery, C3 is caching and C4 is waiting to start caching. Once C1 finishes, C2 must start the delivery, C3 will be full buffered, C4 will start caching and, finally, C1 will await caching. Clients in a collaboration group interchange messages in order to synchronise their actions, thus guaranteeing the correct information flow. All clients assume a specific state each time, forming a cycle. Peers may fail in different states and thus distinct actions are necessary.

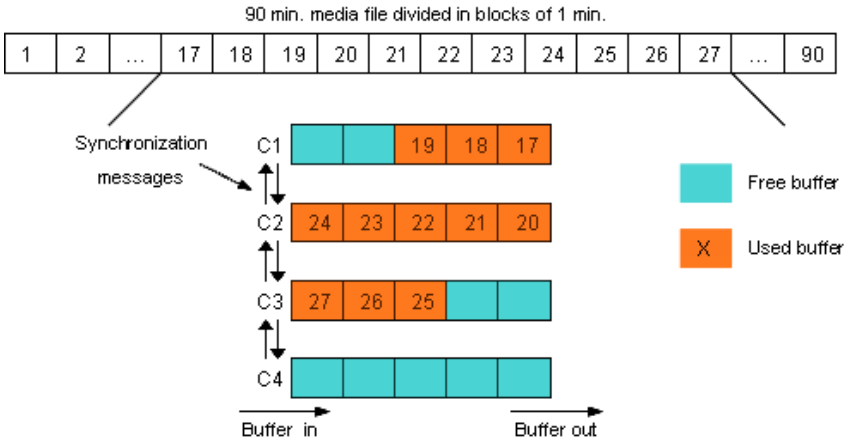


Figure 28. Buffering states on the collaboration process.

State 1 Delivering: The replacement peer must have the same collaborative buffer capacity. If the new collaborator does not have the current video information, the server must take over the delivery process temporarily. Since the detection mechanism needs a period of time to trigger the recovery, clients of the disrupted channel will not receive any information before the recovery process. The QoS is guaranteed in this case by the cushion buffer built through the start-up delay and the patching process. For example, if C1 in figure 28 fails, and the recovery process is unable to find a new collaborator that has blocks 17, 18 and 19, the server must send the respective blocks; after that, the cycle can be re-established.

State 2 Waiting for delivery: The recovery action for this state is similar to the case for state 1. The new collaborator or server, however, has to send all the video blocks that were cached by the failed peer.

State 3 Caching: The new collaborator must continue with the caching process of the failed client. The video blocks that were already cached by the failed client must be delivered by the new collaborator or server. For example, in figure 28, C3 fails and the new collaborator will cache blocks 28 and 29. Blocks 25, 26 and 27 will be sent by the server or the new collaborator (if it has such video blocks).

State 4 Waiting for caching: The failed client has no useful information in its buffer, so the recovery process only needs to find a new collaborator with a free buffer to replace it.

When a peer fails in any case, the information source must be changed; a new source implies on the delivery tree rearrangement by the routers enabled with IP Multicast protocols.

2.2.2 P2Cast

The P2Cast [81] architecture is intended to achieve scalability in Internet VoD services. The service scheme relies on multiple distribution trees built for clients according to their arrival time. In P2Cast, clients not only receive the requested stream, but also contribute to the VoD service by forwarding the stream to other clients and caching as well as serving the initial part of the stream.

P2Cast is described by a single server and clients, divided into sessions, combining patch and base streams to provide service. Associated with P2Cast is a threshold T. Clients that arrive within that threshold form a session. The server and clients belonging to the same session form an ALM tree, called the base tree. The entire video flows over the base tree and is denoted as the base stream. When a new client joins the session, s/he joins the base tree and begins receiving the ongoing base stream. Meanwhile, this new client must obtain the initial part of the video from the server or another client. This initial portion, known as the patch stream, goes from the time when the video begins streaming on the base tree to the time when the new client joins the system.

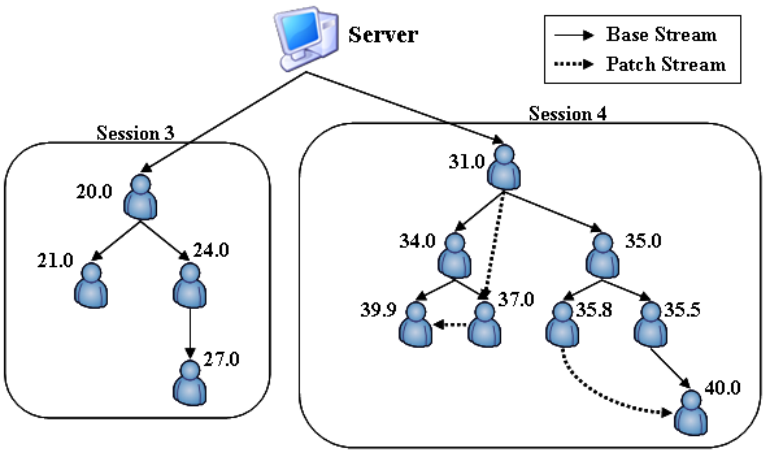


Figure 29. P2Cast service scheme example.

Figure 29 illustrates how P2Cast builds ALM trees and sessions to provide video service. The example shows a snapshot of the system at time 40. Two sessions are represented (3 and 4), which start at time 20.0 and 31.0, respectively. The threshold considered is T = 10. Clients' arrival times are marked beside each one. A solid arrow is used to represent the parent-child relationship in the base tree; and the patch server-client relationship is represented by a non-continuous arrow. The server and the clients in a session form an ALM tree to deliver the base stream. In this example, all clients in session 3 are receiving only the base stream, while three clients in session 4 are still in the process of receiving a patch stream. Clients belonging to different sessions are independent of each other.

The centralised server is the single contact point for peer failure management. The fault tolerance mechanism is based on the recursive reconstruction of the delivery tree. When a client is disrupted by a failure, all clients belonging to the sub-tree rooted at this client are affected. Only the head orphan in the tree is allowed to contact the server to perform recovery. The recovery process is identical to a new client joining, i.e. the server or peer with more available output bandwidth is selected to provide the stream. If the process succeeds, the entire sub-tree is recovered. If it fails, the orphan client is rejected and its children will contact the server and restart the recovery process. The process is repeated until the leaf client is reached or recovery succeeds.

2.3 Control Mechanism and the Failure Management Process

In VoD systems peers may warn the server before leaving or depart unexpectedly. Peers who have already finalised their sessions are prone to terminate the collaboration; thus they are able to inform about their departure before leave. Unexpected failures occur suddenly. Users can decide to download or receive other information from the network compromising their available resources. Furthermore, an end host can crash or simply turn off, communication links can break down or power supply can be interrupted. These kinds of failures can occur at any moment during clients' visualisation. In any case of failure, the system must keep control and take decisions to keep the service active with necessary QoS.

An effective control mechanism is of great importance in P2P-VoD systems. The control scheme is responsible for providing coordination and synchronisation among

system elements (e.g. servers, peers, routers); moreover, it must guarantee robustness and reliability of the service. Therefore, the control scheme requires a careful design owing to the soft real-time constraint of video service and the overhead imposed on the system. We define two metrics to indicate the performance of the control mechanism: Load and Time costs [105].

- ***Load cost***: This metric represents the control overhead imposed on the system for managing peer failures; i.e. we focus on the evaluation of the volume of control messages that flows through the system on failure management processes. The load cost expresses impact over the system better than protocol complexity, as most of the P2P\ALM protocols use simpler overlay construction techniques. In addition, dynamic control information is more interesting because the overhead of a protocol largely depends on the overlay structure used and how it is maintained along the time

- ***Time cost***: The Time cost metric is meant to express the time consumed by solving peer failures. Measuring this cost is extremely important for maintaining the QoS in VoD services A limited time spent on treating peer failures is able to guarantee continuous video playback, avoiding streaming disruptions from the lack of data source. Obviously, a set of parameters must be taken into account to provide QoS. However, the Time cost of the failure management process indicates the control efficiency; reducing this cost helps to improve system performance trough a lower start-up delay and better clients' buffer usage.

Since we have defined our assessment metrics, in the following we systematise peers' Failure Management Process (FMP). When a peer departs, it stops sending video information; thus the QoS can be degenerated. Orphan clients must find other sources for the video streaming before the user notices any kind of errors. The FMP tries to represent the dynamic involved in peer failures. The FMP is based on three components: detection, recovery and maintenance. Figure 30 shows the FMP and its components; we specify the FMP as background to evaluate Load and Time costs.

Owing to the QoS requirements presented by the system, the FMP must be executed respecting a deadline. This deadline is basically determined by the amount of data buffered at each orphan client; the buffered information guarantees the playback without errors while the search and replacement process is performed.

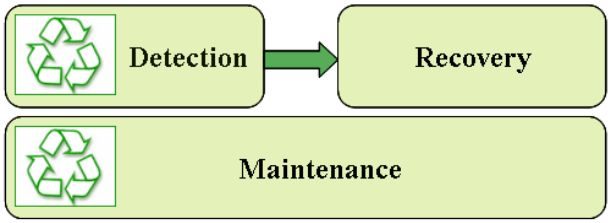


Figure 30. The Failure Management Process

2.3.1 Detection

The first step in dealing with faults is to detect the absence of a peer. The detection consists of monitoring the peers' connections. Basically, there are two ways to detect peer failures.

- ***Source monitoring***: This strategy assumes a control element (controller) that exchanges messages with every collaborator (data source). This scheme is known as heartbeat strategy and consists of the periodic send of control messages from the data source to the controller; these messages mean that a peer is alive in the system and acting on collaborative functions. The heartbeat mechanism is widely applied by the community [80] [94]. The detection is simple: when a heartbeat message is not received by the controller, a query message is sent to the possible failed peer. If the peer does not answer after a specified number of attempts, it is assumed to have failed. Therefore, a recovery phase must be triggered. Depending on the frequency of sending heartbeat messages this strategy can generate excessive traffic over network and compromise the system's delivery capacity. On the one hand, a small time interval between messages supposes more network usage; on the other hand, large intervals can be inadequate for fine detection and QoS maintenance.

- ***Client monitoring***: In this approach the client periodically examines the buffered video blocks and calculates the transmission speed. This strategy is named client buffer monitoring. The relation of received and consumed data on a client's buffer is used in order to evaluate the quality of the streaming session. In this method, each client establishes a time interval to monitor the volume of information received. If the input data rate is lower than a predefined threshold, the client triggers a wait-counter from this moment. The wait-counter is necessary to avoid false alarms, since the network can present short time fluctuations. When the wait time finishes and the transmission speed is still under the threshold, the recovery is triggered. This approach relies on the removal of heartbeat messages from every peer to the controller; thus, the check period can be shorter than the time interval between heartbeats because it does not increase the control load. This detection scheme, however, is more prone to invoke unnecessary recovery (clients with poor resources can request recovery even when the system works well) and is more expensive at the client side.

2.3.2 Recovery

The failure detection triggers the recovery phase. The objective on recovering is to take corrective actions in order to avoid loss of QoS. This stage is responsible for finding an adequate peer to substitute the crashed one. The substitution implies that the new collaborator must contribute satisfactory resources (e.g. content, bandwidth). The recovery can be based on two strategies:

- The system maintains the updated status of the peers on a specific element (controller). Thus, in case of failure the controller is able to indicate directly a substitute peer because it previously knows the set of candidates. This strategy for failure recovery demands a mechanism to provide the updated peers' information. The maintenance stage accomplishes this purpose.

- The second strategy used to find a substitute consists of sending subsequent queries to peers over the system, asking for peer status information and availability. The queries can be performed by a centralised controller or

even by the own orphan clients. This strategy eliminates the need for the maintenance stage; however, it diminishes the accuracy on the recovery and aggregates more control overhead owing to the multiple queries.

If the search for a new peer does not succeed, the next recourse for handling failures is the server. The video server must be consulted to assume the service if it still has sufficient resources; the server is frequently the last opportunity before the system reneges on a client petition.

2.3.3 Maintenance

This stage is necessary when recovery indicates replacement based on a previously known range of peers. The end hosts must send information messages, within a certain frequency, to the controller. This element must know peers' status with precision; whether they are available or not and all the resource information needed to compose a set of possible collaborators.

Beyond the application level, the communications mechanism adopted at the network layer has direct impact on system performance. The routers are elements responsible for routing and forwarding data through the network, establishing the communication link between source and destination. For this purpose, they implement algorithms and protocols that provide the necessary functionalities to build links. In IP Multicast, routers communicate to build, maintain and rearrange the distribution trees. If the data source changes (peer failure), the distribution tree also changes and thus the tree needs to be reconfigured [111]. In ALM, building the overlay topology to connect peers relies on numerous unicast channels at the network. Multiple unicast routes strongly affect the links state, which has influence on the number of messages interchanged by routers [91] [99]. Router communications are directly responsible for maintaining the VoD service and can be relevant to the system load; thus we believe the network level must be considered in the evaluation of the control mechanism.

2.4 The Proposal of Fault Tolerance Mechanism

The QoS of P2P-VoD services on the Internet strongly depends on performance and efficiency to handle the free arrival and departure of peers.

The Failure Management Process is handled of many ways on current VoD service proposals. The most common way to handle peer failures is using the redundancy available on P2P video systems i.e. the presence of multiple information sources (peers). Some architectures (e.g. CoopNet, BitToS or Promise) divide contents into various blocks; in case of peer failure the system looks for a substitute source with the desired video block; special functionalities can be attributed to some peers in order improve the search process (e.g. backup peers on DirectStream). Another approach to deal with peer failures is to send subsequent queries for replacement. P2VoD, for example, performs the search only inside the same generation of the failed peer and P2Cast sends multiple queries in a recursive recovery process. The PCM/MCDB service policy coordinately switches peers during the service; one client each time acts as video source; the failure can occurs when the peer is not in its turn to serve, thus the system has extra time to recover. If a collaborator leaves the system when it is serving (worst case) an available peer with the same content must be found or the server assumes the service while having sufficient resources.

Current fault-tolerance mechanisms, designed to deal with peers departures on video streaming services, evaluate the control demanded to handle such failures by the communication protocol complexity. The protocol complexity statically represents the messages interchanged to operate the failure process. Generally, the control impact is not assessed dynamically, under a continuous process of peers joining and leaving. Moreover, they do not consider underlying network influence when several changes of sources are performed. Basically, the approaches consider the detection and recovery stages; the maintenance phase overlaps with the previous two stages and hence it is generally not included in current fault tolerance designs and evaluations.

The temporal redundancy of information that is present in the system owing to content popularity and asynchronous requests is commonly used to provide fault tolerance when peers suddenly depart. Nevertheless, the approaches do not assess the efficiency of the solution. The redundant information exhibited by the system is constantly changing in accordance with the clients' visualisation time; therefore a substitute peer is searched opportunistically, depending on the part of the content that it

is consuming (and has buffered) and its available resources. Finally, the deadline of the failure management process is not taken into account; the time efficiency of the control scheme, necessary to maintain the QoS, is not measured in any proposal.

The Fault Tolerance on P2P video streaming services is not systematised; each architecture proposes a solution without a deep analysis. The skill in handling peer failures can be analysed under three aspects:

- If the fault tolerance scheme is able to guarantee an acceptable level of success on the search and replacement of a peer;
- The overhead imposed on the system by the control mechanism established to provide the failure management process;
- The efficiency of the mechanism in dealing with the service time constraint.

These points are extremely important because they determine the level of QoS that the system is able to offer. Initially, we evaluate the available solutions that treat the connection and disconnection of peers. The analysis showed us that the main challenge is efficient handling of the data redundancy to guarantee fault tolerance. The sliding window of information at clients' buffers demands a lot of coordination and synchronisation to replace a failed peer. The arrival time of the peers and its resources (buffer capacity and bandwidth) play a determining role to establish collaborations, adding difficulties to peer replacement.

Our proposal is based on efficiently handling the asynchronous redundancy present in the system in order to achieve a robust and reliable Fault Tolerance Scheme. We use the time redundancy to create data replication. The replication is applied in order to improve reliability and accessibility; data replication consists of the same information stored on multiple devices. It can be achieved by aggregating a dedicated infrastructure of backup servers; they should be able to provide hardware and data redundancy. Nevertheless, this solution represents extra costs for service deployment and maintenance. The proposal aims to guarantee success on the replacement of a failed peer; the solution must enforce low control overhead to the system and be time efficient to provide QoS to the clients.

The following topics describe the Fault Tolerance Scheme proposal exposing its elements, features, functions and premises. The characterisation of the proposal by mean

of mathematical models and algorithms as well as the expressions used to measure Load and Time costs are showed in chapter 3.

2.4.1 The Fault Tolerance Service Scheme

The aim of our *Fault Tolerance Scheme (named FTS)* is to build a distributed backup system based on peers' capabilities. The FTS is designed to organise a small set of peers to store portions of the multimedia files statically. The distributed backup system takes advantage of peers' resources; it also guarantees content legality issues, because end hosts hold information only on their volatile memory.

The FTS assumes that clients have a portion of buffer reserved for the VoD service; this portion is used for different purposes. Beyond the usual division of the buffer in client design (e.g. cushion, delivery or collaboration buffers) we assume a new fraction to be used on fault tolerance. This part of the end hosts' buffer is responsible for caching a well-defined video window while the client is online. When a client starts enjoying a video, the multimedia file is continuously flowing through its buffer. The idea is that end hosts reserve a portion of the buffer and store some video blocks, i.e. peers are designated to cache a portion of the content that they have requested in the buffer for future collaboration. This buffer portion is named the 'altruist' buffer; the video blocks statically stored in the individual clients must compose the whole multimedia file. Figure 31 shows the new buffer division in the clients of the VoD service.

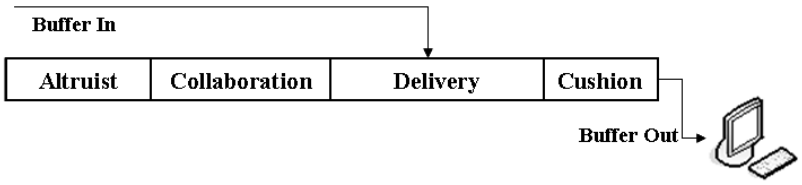


Figure 31. Clients buffer division on the Fault Tolerance Scheme.

The set of clients that made up the distributed backup is selected primarily on the basis of a peer's intention to reserve some resources (basically, buffer space and output bandwidth) in order to provide the fault tolerance service. Since the relational dynamic of P2P is based on the cooperation of equals with the aim of reaching a common goal, we believe that some peers might make their resources available. Once peers

demonstrate their interest in collaboration, the feasibility of certain parameters is considered. The selection criteria must take into account the amount of available resources of each peer, i.e. the clients with the largest buffers and more output bandwidth have priority to participate in the fault tolerance scheme. The selected peers form the *Fault Tolerance Group (FTG)*; the FTG maintains a full copy of a specific content and is able to deal with a limited number of simultaneous failures.

2.4.2 The Manager Node

The selection, coordination and integrity of a FTG are the responsibility of a special entity. We define a super node named *Manager Node (MN)* to run the FTG organisation. The Manager Node is responsible for monitoring the peers who are serving the same content; it also designates and controls the members of the FTG. Each MN is in charge of one FTG. Basically, we insert a control level in the traditional server-client hierarchy. The super nodes distribute the control of peers throughout the own end hosts; this added level in the control architecture provides scalability for the system by alleviating the load at the server. An overview of the proposed Fault Tolerance Scheme can be observed in figure 32.

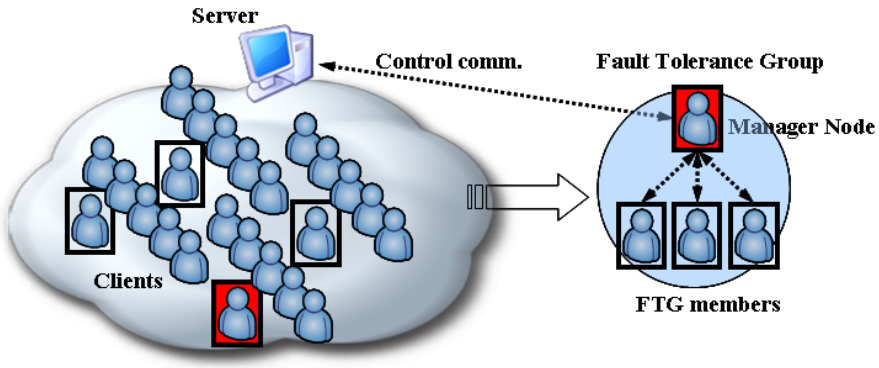


Figure 32. The Fault Tolerance Scheme.

The MN is the first collaborator designated to perform the construction of an FTG. The MNs establish relations with other peers and report to the server system status. MNs also communicate with the server to prevent their own failures. If the MN fails, there is a pre-established order of peers that can assume the position. The FTS process organises the FTG members according to their processing power and bandwidth; peers with high

capabilities are preferred as MNs. If a normal member of an FTG fails the MN must replace this peer in order to maintain the distributed copy integrity. The MN maintains the list of peers who have expressed interest in composing the FTS; it contacts one of these standby members to try to define a substitute.

2.4.3 Building Fault Tolerance Groups

The backup copies of the contents (i.e. FTGs) are generated according to the request rate of each video. Highly popular contents rely more on P2P service, and thus more FTGs are created. The FTS is also based on the Pareto principle (like content distribution at the server level, 80% of petitions ask for 20% of the catalogue). The P2P and multicast approaches exploit the common interest of clients, increasing performance by sharing resources; if the common interest of users is low, then the P2P and multicast benefits will be low, too. Therefore, the FTS considers the following strategy to handle peer failures:

- Less popular contents generate low P2P and multicast action in the systems, and thus the service tends to be sustained by the server. The server is a more reliable element than end hosts; therefore, its failure probability is much lower than that of the peers. In the case of P2P, collaborations are established even when contents present modest popularity, and the server must handle the failure management process by assuming the service itself. Owing to the volume of clients in such a situation, it is feasible to let the server handle the requests and the FMP.

- Highly popular videos lead to better benefits when P2P and multicast paradigms are applied. These strategies decentralise the service, making use of the common interests of users. The service is inclined to be in charge of peers, alleviating the load at the server. In this scenario, when a peer fails, it is viable to replace it with another one; if there is no success in the search for a new collaborator then the server can be asked to assume the service. Nevertheless, the server can be a bottleneck in dealing with all peer failures, especially depending on the frequency with which peers join and leave the system. Therefore, the implementation of the distributed backups (FTGs),

monitored by the MNs, provides scalability to the fault tolerance scheme. The creation of the FTGs is proportional to the popularity of the content, i.e. to the number of P2P streams for each video.

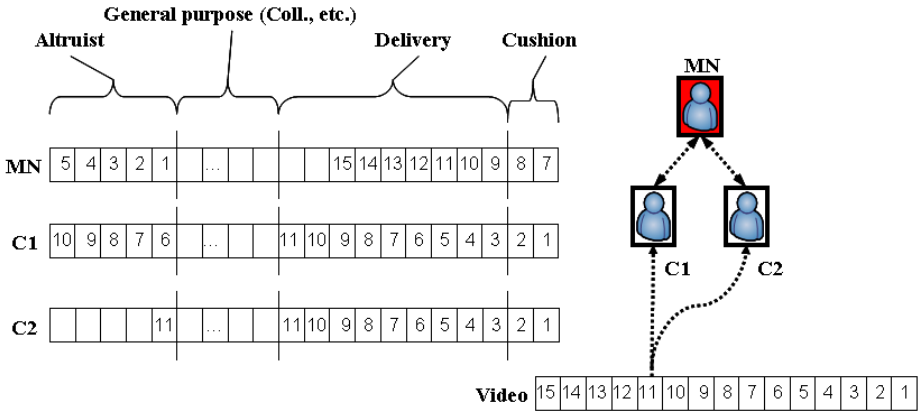


Figure 33. Fault Tolerance Group formation process.

The content replication rate (number of FTGs) is regulated by the fault tolerance service performance established in FTS design. This parameter represents a fraction of the number of streams served by peers. The design of the FTS must guarantee that the total collaboration capacity of all the FTGs can offer the specified fault tolerance performance. The size of the FTG is determined by the number of peers necessary to cache a whole video file; moreover, it is necessary to provide sufficient bandwidth for at least one backup channel. These and other issues of the FTS design are explained in more detail in Chapter 3.

Figure 33 shows an example of an FTG with three members: the MN, C1 and C2. The buffer division of each end host is defined and a portion is reserved for fault tolerance purpose (altruist buffer). The MN has already finished the download process and owns all the blocks necessary to terminate the visualisation (from 7 to 15). C1 and C2 started the visualisation at the same time; they have their buffers full and are still downloading multimedia data. Early packets will be disposed of to allow the storage of the final portion of the video. The altruist buffer of each FTG member is responsible for storing five video blocks; thus, the whole group maintains a distributed copy of the fifteen-minute video. The MN has started the group, and thus it is responsible for the five initial blocks; C1 and C2 arrived together and are in charge of the other ten blocks. C1 already has its altruist buffer complete with blocks from 6 to 10; meanwhile, C2 is

still in the process of caching the final portion (blocks from 11 to 15) and then finishing the FTG formation process.

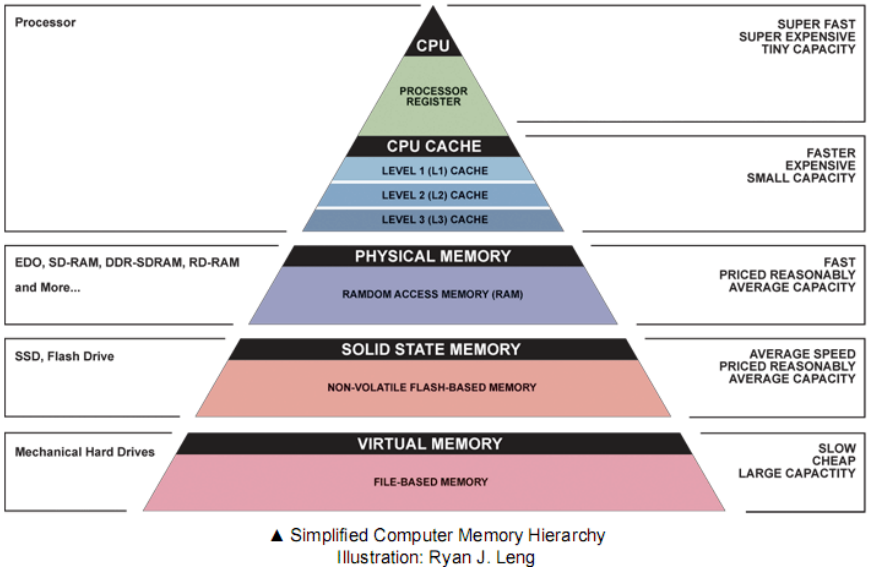


Figure 34. Storage solutions and features.[1]

The FTS demand an altruistic spirit of peers, and it is a premise of the P2P paradigm; in addition, there are many researches on collaboration incentive policies [97]. Technically, the design needs some peer resource reservations. In order to have the service capacity to handle failures, the FTS has to know the available buffer capacity and bandwidth of each element of the FTGs. Nowadays, buffer capacity is not a constraint factor owing to the technical evolution in memory hardware and its relatively low cost. Figures 34 and 35 show that the usage of end host buffers provides a solution with reasonable speed, cost and capacity. Conversely, bandwidth is still a tough constraint on the Internet, mainly on video service, which handles large files and is a soft real-time application; thus, the FTS must support low bandwidth contributions.

[1] Figure extracted from the article '*The secrets of PC Memory*'. Available at: www.bit-tech.net; accessed on May 2009.

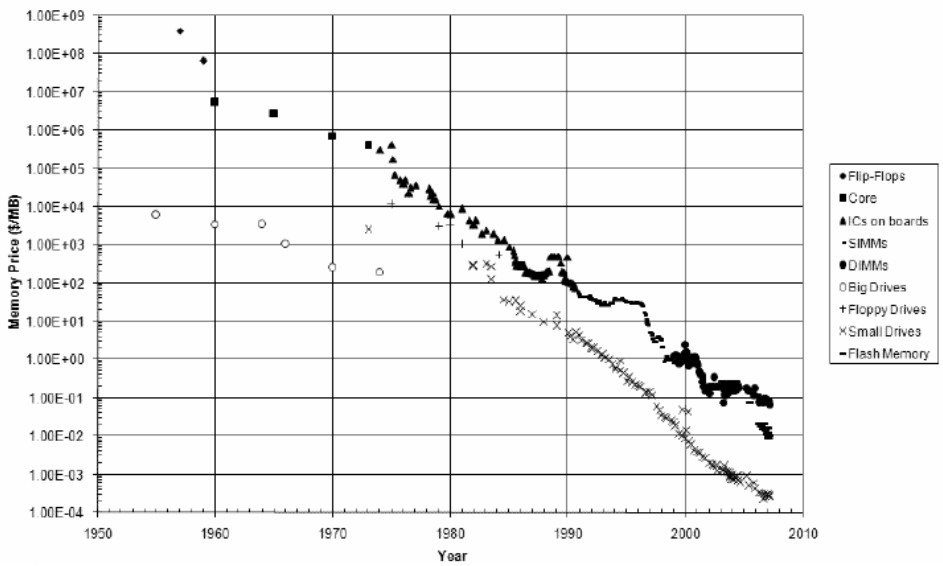


Figure 35. Historical cost of computer memory and storage.[2]

Finally, the proposed FTS represents a distributed solution to manage the redundant information of the system. The approach does not need retransmission of data to build replicas or new dedicated elements. The backup copies are constructed by the own peers during the visualisation of a video and are proportional to the content popularity. The fault tolerance method is intended to be independent of the service policy and feasible for diverse VoD architectures. Moreover, the FTS is able to achieve a predefined level of success when peers fail; the number of simultaneous failures that the system can support and the necessary resources for this are defined by parameter design that can be tuned according to the service policy and client's behaviour. The application of the proposed solution reduces the control overhead by avoiding multiple queries and the maintenance phase; also, it can provide low delay in the failure management process.

[2] Figure extracted from the article '*Definition, History, Usage and Future of Computer Data Storage*'. Available at: www.organdi.net; accessed on May 2009.

Chapter 3

Analytical Models for the Failure Management Process and Fault Tolerance Scheme

3.1 Introduction

In this chapter we present the analytical models developed for two main purposes:

1. To assess the impact of the failure management process of peers. These models are intended to express the costs involved in dealing with end host faults; the metrics adopted to represent the costs are the communication overhead generated by the process (Load cost) and the time spent on solving failures (Time cost). The expressions are also used to measure the performance of the proposed Fault Tolerance Scheme.

2. To define the rules that govern the Fault Tolerance Scheme construction and maintenance. We specify expressions to define how the Fault Tolerance Groups are built (i.e. number of members per group and replication rate of back-up copies) as well as the amount of simultaneous failures supported by the scheme.

3.2 Cost Models of the Failure Management Process

In case of peer failure, the system must be enabled to handle the situation in order to maintain the service running with QoS. This task is not trivial and demands an effective control mechanism; the control scheme is responsible for providing the coordination and synchronisation of servers, end hosts and routers. At the same time, the control scheme must guarantee robustness and reliability to the service. Therefore, the control mechanism design needs special attention; it must be able to deal with the soft real-time constraint of video service and to generate low overheads for the system. We define Load and Time costs as metrics to indicate the performance of the control mechanism;

these costs are based on the stages of the failure management process: detection, recovery and maintenance.

We evaluate the Load and Time costs of two VoD service schemes: PCM/MCDB [77] [78] [76] and P2Cast [81]. These policies were selected to represent different approaches to multimedia delivery; both present key features of large-scale VoD deployment. Next we explain how they work and the important characteristics that they have.

By analysing the costs of the FMP on the PCM/MCDB and P2Cast we can confirm the relevance of the control mechanism of Internet VoD systems based on P2P and multicast paradigms. Aiming to achieve better performance in handling peer failures, we propose a Fault Tolerance Scheme. The FTS is intended to manage the natural data redundancy of the system to provide robustness and reliability. The performance of the proposal is also evaluated by the Load and Time metrics.

3.2.1 Modelling process

Models are based on the Failure Management Process (FMP). The FMP is intended to represent the necessary steps in handling peer failures; these steps are detection, recovery and maintenance.

Detection has the objective of monitoring peers; it notices when a collaborator leaves the system. Basically, detection can be implemented by monitoring sources (peers) or by the own clients. The detection of a fault automatically initiates the recovery phase. This stage must take corrective measures to maintain the QoS; it searches for and indicates a suitable substitute for the failed peer. The maintenance step overlaps with the two previous steps. The FMP demands this phase in order to keep peer status up to date for future collaborations. Maintenance also represents the weight of the underlying network in the failure management process.

The stages of the FMP are used as background to define the Load and Time costs. The Load cost represents the control overhead imposed on the system in managing peer failures and the Time cost of the failure management process indicates the control efficiency. Assessing and tuning these costs combine to improve VoD service performance.

3.2.2 Parameters and variables

All the parameters and variables used in the Load and Time costs models are shown in table 3.

C_d	Network load injected by the detection phase.
C_r	Network load injected by the recovery phase.
C_m	Network load injected by the maintenance phase.
C_t	Time cost of the FMP.
N_{cc}	Number of peers (collaborators' clients).
N_C	Total number of active clients.
N_H	Number of routers in a multicast group.
O_f	Number of clients that trigger a recovery process (orphans).
G_M	Number of multicast groups.
H	Number of routers in the network.
k	Average connectivity degree of a router.
l	Latency at a network edge.
W	Wait time interval.
p_e	Probability of finding a collaborator.
f_{HB}	Heartbeat message frequency.
f_e	Clients' failure frequency.
f_{CI}	Clients update messages frequency.
f_{TI}	Routers status messages frequency.
β	Messages required for the detection protocol.
σ	Messages required for peer communications in the recovery protocol.
ρ	Messages required for subsequent queries in the recovery protocol.
γ	Messages required between routers in the recovery protocol.
φ	Messages required for multicast group membership.
ω	Messages required for the clients' maintenance protocol.
α	Messages required between routers for the maintenance protocol.

Table 3. Parameters used in the modelling process of the Load and Time costs.

3.3 PCM/MCDB Service Policy

The PCM/MCDB is the delivery mechanism of the P^n2P^n VoD architecture; the scheme design is based on two policies: Patch Collaboration Manager (PCM) and Multicast Channel Distributed Branching (MCDB). The PCM/MCDB considers distributed servers, P2P collaborations and Multicast communication to achieve a high degree of scalability and performance; it is designed to exploit P2P collaborations, acting on local networks enabled with IP Multicast. The scheme implements collaboration groups of peers organized on a mesh-based topology.

3.3.1 Load cost model

The development of the models is based on the detection, recovery and maintenance stages. The scheme assumes every client inside a collaboration group exchanges heartbeat messages with two neighbours. The detection cost (C_d) defined by expression (1) represents the peers (N_{cc}) of each collaboration group (G_C), exchanging β messages at a frequency f_{HB}.

$$C_d = f_{HB} \cdot \beta \cdot \left[\sum_{i=1}^{G_C} N_{cc\_g(i)} \right] \qquad (1)$$

Every failure triggers a recovery process, establishing contact between server and orphan clients. We consider that failures occur on an f_e frequency and each process generates σ messages for each orphan O_f in order to trigger the recovery. Also, since the source of the streaming changes with the recovery, the routers (N_H) of the multicast group generated by the new data source (G_M) exchange γ messages in order to rearrange the distribution tree; the clients, who were attended by the failed peer, must now attach to the new group through φ messages (membership requests and replies). Equation (2) expresses the cost of the recovery stage (C_r).

$$C_r = f_{e.} \cdot \left(\sigma \cdot O_f + [\gamma \cdot N_{H_{G_M}} + \varphi \cdot N_{C_{G_M}}] \right) \quad (2)$$

The maintenance stage cost (C_m) is represented by equation (3). In the first part, all clients available for collaboration (N_C) send ω status messages at a frequency f_{CI} to update the information at the server. The second part of equation (3) describes the routers (N_H) of each group (G_M) exchanging α messages at a frequency f_{TI} in order to maintain the multicast distribution tree.

$$C_m = f_{CI} \cdot \omega \cdot N_C + f_{TI} \cdot \sum_{i=1}^{G_M} \alpha \cdot N_{H\,g(i)} \qquad (3)$$

3.3.2 Time cost model

In the PCM/MCDB scheme a central server keeps track of every collaborator in the system. The detection works with heartbeat messages, but nevertheless a non-received heartbeat does not necessarily mean a client failure, since the problem could be in another system element like the network. The orphan client of a failed peer triggers a counter from the moment that it notices the first non-received heartbeat and starts sending query messages. After W cycles of the counter, the recovery process is triggered. Each counter cycle has the same length of heartbeat cycle ($1/f_{HB}$). This means the detection scheme must wait for W sequential non-received heartbeat messages before starting the recovery. The first part of equation (4) represents the detection time cost.

$$C_t = \frac{W}{f_{HB}} + \left(\sigma \cdot l \cdot \frac{\ln(H)}{\ln(k)} \right) \qquad (4)$$

The second part of equation (4) describes the time cost of the recovery stage. The server knows the status of every collaborator in the system (information provided by the maintenance phase), thus the recovery cost expresses the time consumption in the communication between server, orphan client and new collaborator. The cost is defined by the number of messages (σ) that cross a path of size $ln(H)/ln(k)$, where each network segment presents latency l. The path size means the number of routers that make up each route. Our model takes into account the small-world effect, which is observed in many kinds of networks, including computer networks e.g. the Internet [98]. The small-world concept in simple terms describes the fact that despite their often large size, in most networks there is a relatively short path between any two nodes. The distance between two nodes is defined as the number of edges along the shortest path connecting them. The number of routers in the network is represented by H and k is the mean connectivity of a router (average number of edges connected at a node).

3.4 P2Cast Service Policy

The P2Cast is an architecture proposed to achieve scalability in an Internet VoD service. It uses the P2P approach to cooperatively stream multimedia data while only relying on unicast connections among peers. The key idea of P2Cast is to have each client acting as data source while it receives the video; thus, this strategy creates an ALM tree-based architecture with peers to distribute multimedia information.

3.4.1 Load cost model

In order to establish the same detection basis for both models, we assume the P2Cast scheme with the heartbeat strategy. Expression (5) defines the detection load cost for P2Cast, where all collaborator peers send heartbeat messages to the server. The total number of collaborators (N_C) sends β messages at a frequency f_{HB}.

$$C_d = f_{HB} \cdot \beta \cdot N_{cc} \qquad (5)$$

Equation (6) expresses the cost of the recovery stage. The frequency of client failures is f_e and each orphan O_f exchanges σ messages with the server triggering the recovery. In order to find a new collaborator, ρ messages are exchanged between the server and peers in an ALM session. Since the recovery process is recursive, the number of searches increases the load cost. Thus, the number of messages is inversely proportional to the probability of success in the collaboration search, p_e.

$$C_r = f_{e.} \cdot \left(\sigma \cdot O_f + \frac{\rho}{p_e} \right) \qquad (6)$$

The support stage of the FMP is the route maintenance i.e. router communications. This cost is described by expression (7). Routes are kept up to date dynamically, based on such system characteristics as bandwidth or distance. Consequently, routers must periodically exchange messages and process the calculations of the routing algorithms. Every router

sends α messages on a f_{TI} frequency. The total volume of messages also depends on the number of routers in the system (H).

$$C_m = f_{TI.} \cdot \alpha \cdot H \qquad (7)$$

3.4.2 Time cost model

The time cost of P2Cast scheme is expressed by equation (8). The detection cost is represented just as it is in PCM/MCDB (W/f_{HB}). P2Cast presents a single conceptual difference: the server monitors the heartbeats instead of neighbour peers.

The time cost of the recovery stage on P2Cast is basically governed by two actions that are responsible for time consumption: search and communications.

$$C_t = \frac{W}{f_{HB}} + \frac{1}{p_e} \cdot \left(\sigma \cdot l \cdot \frac{\ln(H)}{\ln(k)} \right) \qquad (8)$$

The time spent with communication also represents the message exchange between server, orphan client and new collaborator. The representation is the same as used in PCM/MCDB, which depends on the number of messages (σ), the path size ($ln(H)/ln(k)$), and the network latency (l). The main difference lies in the recursive nature of the recovery process. This can lead to quite a number of searches, making the time cost increase, so the total communication time is inversely proportional to the success probability p_e.

3.5 The Proposed Fault Tolerance Scheme

.

A server-based solution may be able to handle all the delivery tasks of the VoD service; distributed servers\proxies are necessary, however, to achieve system scalability. The control mechanism assumes a more important role in VoD service when the P2P and multicast strategies are used to improve system performance; they bring new behaviour patterns and demand more attention to implement the system.

Owing to the increasing weight of the control mechanism, the decentralisation of the control is also necessary to achieve system scalability. By distributing control, it is possible to save server resources i.e. the server becomes free from handling the entire load that the

FMP represents. A simple queue model is enough to show the extra load that fault tolerance process can represent to servers (equation 9). The model expresses the system average response time to answer a request (t_r). The μ represents the rate of petitions that the server is able to handle and λ is the average rate of video requests arriving at the system; the ε is the average rate of recovery petitions. When the server is responsible for handling delivery and every failure process, the time to attend any kind of request is increased; this makes the system performance worse.

$$t_r = \frac{1}{\mu - \pi} \;;\; \pi = \lambda + \varepsilon \qquad (9)$$

The proposed Fault Tolerance Scheme (FTS) has the aim of providing robustness and reliability to an Internet P2P-VoD service in a distributed way. By establishing a control hierarchy, the system keeps the scalability feature. The mechanism builds a distributed backup system by using the own peers' capabilities. The design is based on the organisation of a small set of peers to store portions of the multimedia files on their buffers statically. Together, the parts of the content distributed over peers must constitute the total video file. In this way, these peers form the Fault Tolerance Group (FTG); this group must be able to provide at least one backup channel. The FTGs are generated according to the request rate of each video and to design parameters.

In this section we present the models that govern the construction and operation of the FTS and its elements. For convenience, all parameters used in the analysis are defined in table 4.

N_{CFTG}	Number of peers in a Fault Tolerance Group.
N_{FTG}	Number of Fault Tolerance Groups.
N_{MN}	Number of Manager Nodes in the system.
L	Total video length.
d	Buffer space for collaboration in the FTS.
V_{pr}	Video playback rate.
bw_o	Output bandwidth available for collaboration in the FTS (upload capacity).
CC	Collaboration Capacity of a FTG
s_f	Simultaneous failures.
$FTSP$	Fault Tolerance Scheme service performance.
$N1$	Number of peers demanded to build a FTG owing to bandwidth restriction.
$N2$	Number of peers demanded to build a FTG owing to buffer capacity restriction.
Q	The quotient of the division of N2 by N1.
r	The remainder of the division of N2 by N1.

Table 4. Parameters used in the modelling process of the Fault Tolerance Scheme.

3.5.1 Formation law

The rules that control the establishment of the FTS are based on input information provided in the system design. The FTS needs to know the expectations about the service i.e. the load that the Fault Tolerance Scheme has to support. This load is translated as the number of simultaneous faults (s_f) that the mechanism can handle. In a P2P-VoD service, end hosts can frequently fail; faults may affect peers in different playback points or even at the same exhibition range. In a first consideration we assume simultaneous faults by failed peers who were serving the same content at the same playback point. This is the worst case in terms of peer failures because the FTS must have enough resources to attend more than one petition at the same time. This situation leads to two possibilities. On the one hand, if the local service relies on IP Multicast, a unique FTG can serve as source for all the orphans; the network layer is responsible for duplicating the information in the data distribution. On the other hand (critical case), if the service is based on ALM, each orphan requests the same resources that the failed parent was providing. It means that the FTG has to guarantee sufficient output bandwidth to serve more than one backup channel at the same time. The solution to this issue is twofold: either peers that form the FTG have sufficient output bandwidth to provide the necessary number of backup channels or the FTS must guarantee the resources by the presence of multiple FTGs. The output bandwidth (upload capacity) is still a highly restrictive factor as regards the Internet end hosts, and thus the trivial solution to the problem is replicating the FTGs. Moreover, the construction of the FTG requires information about the amount of resources that each peer is able to provide for the fault tolerance service (buffer space and upload load bandwidth capacity). Hence, the design of the FTS assumes three input parameters to define the service:

- The simultaneous failures that the FTS must support;
- The range of output bandwidth that peers must reserve for the fault tolerance service [bw_{min} ; bw_{max}]
- The range of altruist buffer that peers must provide for the FTS [d_{min} ; d_{max}]

With these input parameters we can orientate the FTS, i.e. the creation of the FTGs. It provides control of the number of peers that make up each group and the number of groups necessary to accomplish the required performance (number of simultaneous assistances).

In the following we will describe the FTS construction based on two cases: a) when the output bandwidth of peers is greater than the video play rate and b) when the available output bandwidth is lower than the playback rate. The first case is the more trivial, since the service has no bandwidth restriction; we will analyse this case because most of the proposals do not take into account the bandwidth limitation in their designs [80] [81].

In the Internet environment, generally, the upload rate is reduced and smaller than the multimedia play rate. Considering no network constraints, the data income rate (output bandwidth of the source peer – bw_o) and the data consumption speed (playback rate - V_{pr}) must be balanced at the client. If these rates are unbalanced, the user experience is damaged owing to buffer overflow ($bw_o >> V_{pr}$) or lack of data ($bw_o < V_{pr}$); the most common situation in Internet VoD service is the second one.

The solution to balancing output bandwidth and playback rate when $bw_o < V_{pr}$ is to aggregate peers' resources in order to emulate a unique information source. This approach is adopted in many researches [77] [79] [100]. The principle of this strategy is to divide video blocks in proportion to the peer output capacity. The blocks are thus stored on different peers that combine their resources to guarantee an input rate at the client equal to the playback. Figure 36 illustrates an example of resource aggregation when $bw_o < V_{pr}$. The FTG members C1, C2, C3 and MN have no restrictions on the altruist buffer and are enabled to collaborate with 200 kb/s, 300 kb/s, 500 kb/s and 500 kb/s respectively; the playback rate is of 1500 kb/s. The four peers store different parts and amounts of data in order cooperatively to provide a video stream. Let us take the video block *A*; this block is divided in portions proportionate to the output capacity of each peer (bw_o/V_{pr}). Peer C1, in this example, will store a portion A^* of block *A* corresponding to $A \cdot 2/15$; hence, each FTG member stores a portion of each video block *A*, *B*, *C*, until the content is totally cached. Owing to their different capabilities the total amount of information stored is also proportional to the peer upload bandwidth, i.e. the MN will store $L \cdot 5/15$ and so on (where *L* is the total video length).

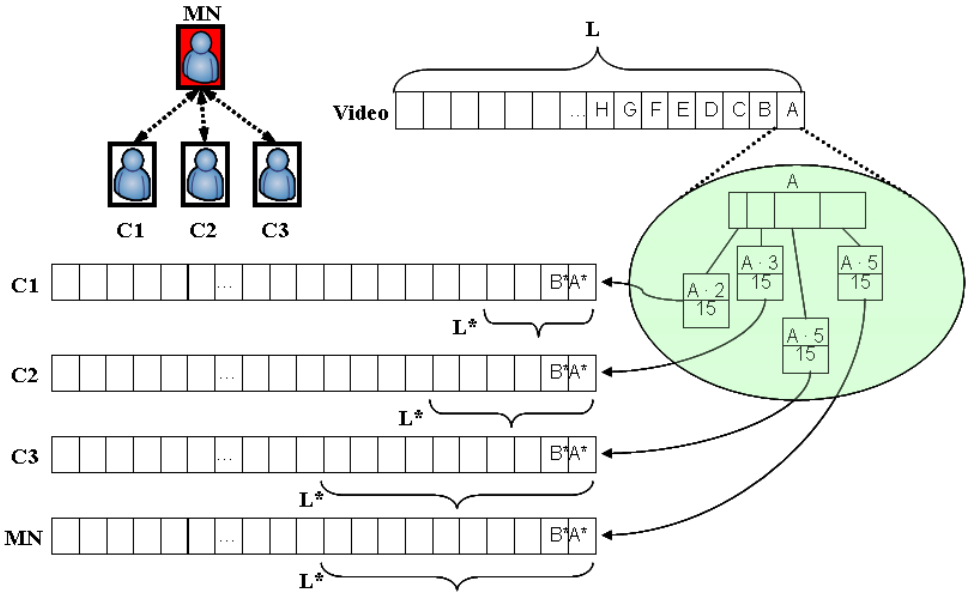


Figure 36. Resource aggregation strategy.

a. Peers' bandwidth greater than playback rate ($bw \geq V_{pr}$)

In this topic we define the fashion how groups are built; this refers to determining the number of clients that make up each FTG and the number of groups demanded by the system. Peers make available different quantities of bandwidth and buffer space to the fault tolerance service; therefore the number of peers to create a group varies according to collaborators' capabilities. We consider that the resources are bounded by minimal and maximum values of buffer ($d = [d_{\min}; d_{\max}]$) and output bandwidth ($bw_o = [bw_{\min}; bw_{\max}]$).

In the present case the bw_{min} reserved by peers is sufficient to provide at least one video stream, and thus there is no need to coordinate end hosts in order to aggregate their capacities. The bandwidth is not a restriction in terms of building the group. The FTS only need to deal with heterogeneity in storage space (d); the group must be built to compose a full copy of a video v. Considering the total length of the video as L, the sum of all peers' buffers must be equal to L (equation 10).

$$L = \sum_{i=1}^{N_{C_{FTG}}} d_i \qquad (10)$$

The number of members of an FTG (N_{CFTG}) is defined by the buffer capacity of each peer. If the storage space of the peers were uniform, the N_{CFTG} would be the simple relation L/d. The amount of buffer destined by each end host varies, however, between d_{min} and d_{max}. Hence, we define the range of the number of clients as expressed in equation (11).

$$\frac{L}{d_{\max}} \leq N_{CFTG} \leq \frac{L}{d_{\min}} \rightarrow \left(N_{CFTG_{\min}} \leq N_{CFTG} \leq N_{CFTG_{\max}}\right) \quad (11)$$

The final number of peers in an FTG will be inside the defined range; the exact number of members depends, however, on the available resources of each peer. In order to determine the number of video streams that the scheme can provide, we define the Collaboration Capacity (*CC*) of the FTG (expression 12). The *CC* represents the number of multimedia streams that the FTG can provide simultaneously at the playback rate (V_{pr}); it is bounded by the peer with minimum availability of output bandwidth.

$$CC = \frac{MIN(bw_o)}{V_{pr}} \qquad (12)$$

The last parameter used to define the rate of replication of FTGs is the Fault Tolerance Service Performance (*FTSP*). This parameter determines the number of simultaneous failures (s_f) that the FTS can deal with. The FTSP is a project parameter, established by the system designer. Nevertheless, this parameter must not be arbitrarily established; it should be based on the number of clients present in the system and on their distribution (clients/video session); the FTSP is expressed as a percentage value. Besides, the number of simultaneous failures must consider the content popularity, i.e. the number of streams of a video *v* that are being provided by peers ($P2P\_steams_v$). This avoids wasting resources on video sessions where P2P does not have a special participation; resources are dedicated to sessions that have more P2P interaction and thus are more vulnerable to peer failures. This way of defining the service capacity of the FTS follows the 80-20 rule and allows tuning of the replication rate; equation (13) represents the number of simultaneous failures (s_f) defined on the FTS design. The minimum value of the s_f is one backup channel.

$$s_f = FTSP\% \cdot P2P\_streams_v \qquad (13)$$

Finally, the replication rate is governed by the s_f and CC parameters. Algorithm 1 represents the dynamic of FTS establishment. While the Collaboration Capacity provided by all FTGs is lower than the pre-defined number of simultaneous failures, the scheme must add peers to FTGs or create new groups. When the condition is satisfied, ($\sum CC = s_{f_{(bw)}}$) the construction is done and the operation stops. This process is repeated periodically, on an assessment frequency f_a, in order to update the FTS status. The periodical process provides the dynamic tuning of the service proportional to the system state. If the number of P2P streams becomes lower for any reason, the CC can be reconfigured and some peers are enabled to leave the FTS. The FTS must first run when the system starts the service; the number of simultaneous failures supported is an input parameter; thus the system should concentrate its efforts on reaching the specifications. Algorithm 1 describes the creation of FTGs only according to buffer restriction.

```
While (∑CC < s_f(bw))

    If (∑d < L)
    then
    Add Collaborator to FTG

    If (∑d = L)
    then
    New FTG.
```

Algorithm 1. Construction of the Fault Tolerance Groups without bandwidth restriction.

b. Peers' bandwidth lower than playback rate ($bw < V_{pr}$)

The previous case is the most common hypothesis assumed in current VoD approaches. Nevertheless, it does not reflect the behaviour observed in the Internet environment; generally, peers' upload bandwidth is lower than the download rate and insufficient to feed the video play rate.

In this section we define the establishment of the FTS considering the low bandwidth peers' feature. The number of clients that composes each FTG and the volume of FTGs necessary to reach the designed performance (s_f) are governed in the same way as in the previous approach ($bw_o \geq V_{pr}$). End hosts are heterogeneous and make different bandwidth and buffer space available to the fault tolerance service. The number of peers needed to build an FTG varies according to the available resources. The FTS determine a range of buffer ($d = [d_{\min}; d_{\max}]$) and output bandwidth ($bw_o = [bw_{\min}; bw_{\max}]$) in which the resources can oscillate.

Differently from the previous case, in the present situation the range of reserved *bw* is insufficient to provide a video stream by itself; hence it is mandatory to coordinate peers and aggregate their resources. The bandwidth represents a constraint for building the FTG; thus the FTS needs to handle both parameters (*bw* and *d*) to build the groups. The FTGs must form a full copy of a video *v* and at the same time guarantee service bandwidth to provide a video channel. These constraints are represented in expression 14. The sum of all peers' buffers must be equal to *L* and the total output bandwidth must guarantee the playback rate.

$$L = \sum_{i=1}^{N_{CFTG}} d_i \text{ and } V_{pr} = \sum_{i=1}^{N_{CFTG}} bw_{oi} \qquad (14)$$

Now, the number of members of a FTG (N_{CFTG}) will depend on the buffer and bandwidth capabilities of each peer. The amount of resources destined by each end host varies between $[d_{\min}; d_{\max}]$ and $[bw_{\min}; bw_{\max}]$. Therefore, the range of the number of peers on a FTG ($[N_{CFTG_{\min}}; N_{CFTG_{\max}}]$) can be defined as showed in (15).

$N_{CFTG_{\min}}$:

$$N_1 = \frac{V_{pr}}{bw_{\max}} \;;\; N_2 = \frac{L}{d_{\max}}$$

if:

$$N_1 > N_2 \rightarrow N_{CFTG_{\min}} = N_1$$

if:

$$N_1 < N_2 \begin{cases} {}^{N_2}\!/_{N_1} = (Q, r) \\ r = 0 \rightarrow N_{CFTG_{\min}} = N_2 \\ r \neq 0 \rightarrow N_{CFTG_{\min}} = N_1 \cdot (Q+1) \end{cases}$$

$N_{CFTG_{\max}}$: (15)

$$N_1 = \frac{V_{pr}}{bw_{\min}} \;;\; N_2 = \frac{L}{d_{\min}}$$

if:

$$N_1 > N_2 \rightarrow N_{CFTG_{\max}} = N_1$$

if:

$$N_1 < N_2 \begin{cases} {}^{N_2}\!/_{N_1} = (Q, r) \\ r = 0 \rightarrow N_{CFTG_{\max}} = N_2 \\ r \neq 0 \rightarrow N_{CFTG_{\max}} = N_1 \cdot (Q+1) \end{cases}$$

$$N_{CFTG_{\min}} \leq N_{CFTG} \leq N_{CFTG\ \max}$$

The N_{CFTG} varies inside the defined range $[N_{CFTG_{\min}}; N_{CFTG\ \max}]$. The Collaboration Capacity (*CC*) of the FTG in this case is defined by the sum of all members' output capacities (equation 16). The *CC* (number of video streams that the FTG can provide at the rate V_{pr}) is limited to one. The construction of an FTG now relies on achieving the storage of the full video (*L*) and sufficient bandwidth for one stream.

$$CC = \frac{\sum bw_o}{V_{pr}} \qquad (16)$$

The parameter that orientates the replication of FTGs is the number of simultaneous failures to be supported by the scheme. In the case of $bw_o \geq V_{pr}$, each fault tolerance group is able to handle more than one failure process at the same time ($CC > 1$); the collaboration capacity is bounded by the peer with less available output bandwidth. As consequence of the limited collaboration capacity of each FTG in the present case ($CC = 1$), reaching s_f depends exclusively on the number of groups (N_{FTG}).

The s_f and CC parameters will determine the N_{FTG}. The algorithm 2 is similar to 1, but now considers the storage space and bandwidth restrictions. While the total CC is lower than the designed s_f, the scheme must keep adding peers or creating FTGs. As in the previous case, the process is repeated periodically, on an assessment frequency f_a, in order to update the FTS status.

```
While (∑CC < s_f(bw))

    If (∑d < L) ∨ (CC < 1)
    then
    Add Collaborator to FTG.

    If (∑d = L) ∘ (CC = 1)
    then
    New FTG.
```

Algorithm 2. Construction of the Fault Tolerance Groups with bandwidth restriction.

3.5.2 Complexity of the Fault Tolerance Scheme

The entire control scheme relies on messages exchange to establish communications between the elements of the system. The complexity of the FTS is defined by the communications protocol necessary to establish and maintain the service i.e., messages exchange between server and peers. In this section we briefly describe a standard client join process as well as the FTS complexity to build and keep running the Fault Tolerance Groups.

Client join: We assume a standard join process for clients connecting to the system. Most of video approaches rely on the server to establish the service; the server can act as

data source or even as a tracker to intermediate peers connections as well as remote server service [80] [81] [83] [85].

Figure 37 illustrates the communications established during the join process. When a client joins the system, it first sends a message to the server asking for service and informing about its conditions (IP address, availability for collaboration, resources etc.); the server then answers, indicating the result of the request (I). The first option is to serve the petition through P2P collaboration; if there is any session with a peer in a condition to provide service to the joining client, the server indicates the peer that will act as data source and then the new client and the peer must establish communication (II). If there is no possibility of P2P collaborations and the server still has sufficient resources it assumes the service (III). If the server has not got the content, the service must be provided remotely. The server contacts the source of the content (server of other area) and establishes the overlay connection, forwarding the data from the remote server to the client (IV). The joining process generates a control load necessary for the service establishment; therefore, the overhead of the joining process is directly proportional to the request rate and the type of source (i.e. peer, local server or remote server).

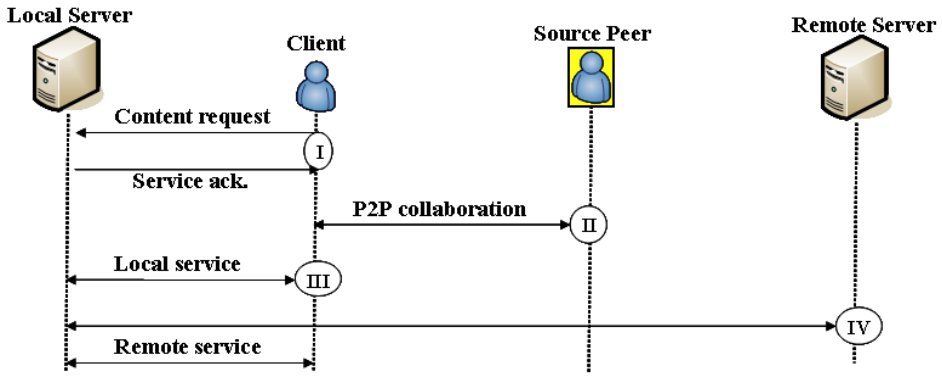


Figure 37. Clients' join process.

Establishment and maintenance of the FTS: The building process of the Fault Tolerance Scheme runs in conjunction with the VoD service joining process and is showed in figure 38. First, a client that joins to the system through any service policy informs about its reserved resources for collaboration matters. While the joining process determines which source will serve the client request, the FTS process defines whether this client is able to participate in the fault tolerance service or not (figure 38, I). If the application of the client in the FTS is feasible, then the server can indicate the following attitudes:

- If there is an FTG under construction which is demanding more collaborators, the client is redirected to contact the respective MN to get instructions in order to build its backup portion and join to this FTG (II);

- If there is no open FTG for the requested video or the existent FTGs are not able to reach the specified performance (s_f), the peer is assigned to start a new FTG and is made the MN of the group (III).

- If the service performance restriction is satisfied, the FTS is done; momentary, there is no need to add end hosts in FTGs neither to create new FTGs. The peer is put on standby state in the FTS (IV).

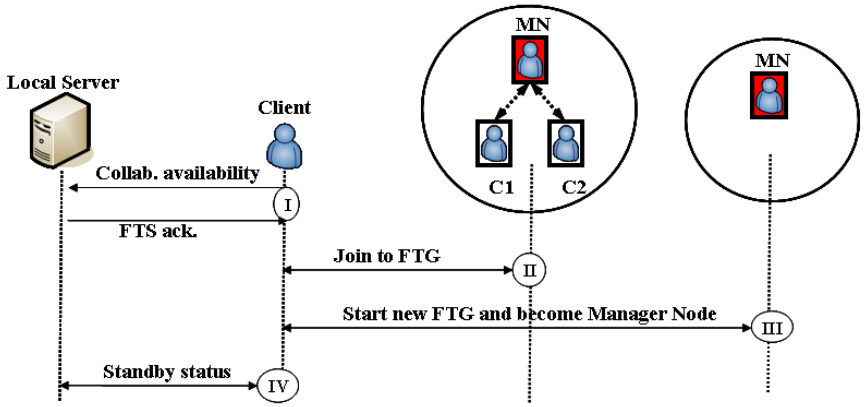


Figure 38. Peer joining process in the Fault Tolerance Scheme.

This process is triggered whenever a client joins the system and informs about its willingness to compose the fault tolerance service. Also, the FTS updates its status according to the number of opened P2P streams for a video *v*; the reconfiguration process is triggered on an assessment frequency f_a. This reconfiguration process is also responsible for evaluating the FTGs in order to verify the conditions of the members.

Peers which make up the FTGs are known by the server and are organised according to their capabilities; the server establishes an order of succession which is applied to determine the most suitable peer to assume the MN position in case of its failure. Figure 39 shows the FTGs maintenance; when an MN fails, the server establishes contact with the peer with the

best capabilities in the FTG (two messages) and it becomes an MN; next, the new MN communicates with every member of the FTG and starts to search for a peer to store the video portion left by the previous MN (I). The complexity of these communications is shown in expression (17).

$$f(N_{CFTG}) = 2 + (N_{CFTG} - 1) \rightarrow O(N_{CFTG}) \quad (17)$$

In the same way, when a standard member of an FTG fails, the MN of the group is responsible for finding another peer to store the same video portion to compose the full backup copy. Thus, the MN contacts the standby peers which were available for collaboration but were not used because the FTS was already completed (II). A peer visualising the content on a point before that where the failed client had stored is preferred. If it is not possible or the lack of information strongly damages the FTS performance (i.e. the FTS cannot wait for the new peer to reach the playback range of the failed peer), the server must send the respective video block to the new FTG member.

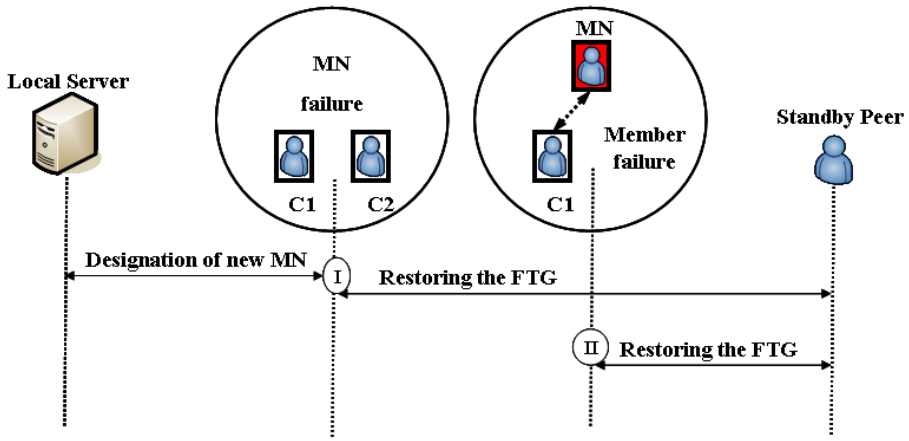


Figure 39. Fault Tolerance Groups maintenance.

3.5.3 PCM/MCDB with the Fault Tolerance Scheme

By applying the Fault Tolerance Scheme to PCM/MCDB, the expressions for Load cost change. The detection now is carried out by the Manager Nodes of the FTGs, which monitor all the peers. In addition, the Manager Nodes exchange messages with the server in order to prevent their own failures and update server information ($2 \cdot N_{MN}$).

Each FTG has its own MN; the MNs of distinct FTGs (but of the same video *v*) share the responsibility for monitoring servant peers. The distribution of the peers which are under the responsibility of an MN is defined by a location strategy [76]; peers are designated to establish communications with the closest MN. At the joining process, the server, which retains information about every FTGs and MNs, informs the client about the designated MN.

The recovery step presents no changes with the application of the FTS; the only difference is that the service, after a peer failure, is now provided by the FTG instead of a peer indicated by the server.

The most significant change lies in the maintenance phase. There is no further need for status messages by peers in order to keep updated information and guarantee the success of the recovery. The FTG automatically provides a substitute peer in case of failures. Equations (18) and (19) describe the new expressions for PCM/MCDB detection and maintenance, respectively.

$$C_d = f_{HB} \cdot \beta \cdot \left[\sum_{i=1}^{G_C} N_{cc\_g(i)} + (2 \cdot N_{MN}) \right] \quad (18)$$

$$C_m = f_{TI} \cdot \sum_{i=1}^{G_M} \alpha \cdot N_{H\,g(i)} \quad (19)$$

The FTS does not cause any impact on the Time cost model of the PCM/MCDB scheme. The detection and recovery phases are the same for treating peer failures.

3.5.4 P2Cast with the Fault Tolerance Scheme

The Load cost for the P2Cast scheme presents a more significant change when FTG is used. The detection relies on the Manager Nodes and messages are exchanged with the server ($2 \cdot N_{MN}$), as in the PCM/MCDB approach. The expression (20) presents the new P2Cast detection Load cost.

$$C_d = f_{HB} \cdot \beta \cdot \left[(N_{cc}) + (2 \cdot N_{MN}) \right] \quad (20)$$

The most important point in this case is the removal of the recursive recovery process of the ALM tree provided by the usage of the Fault Tolerance Scheme. The service after a failure is now guaranteed by the FTG without the need to ask other peers. The maintenance step has no changes and is the same as defined by (7); it consists of routers communications. Equation (21) describes the recovery cost of the P2cast with the FTG approach.

$$C_r = f_{e.} \cdot \sigma \cdot O_f \qquad (21)$$

With the application of the FTS, the model that describes the time cost for PCM/MCDB and P2Cast schemes becomes the same. Equation (22) represents the Time cost, which depends on the heartbeat frequency (f_{HB}), the number of messages for recovery (σ), the path size (*ln(H)/ln(k)*), and the network latency (*l*). The FTG provides the replacement of a failed peer and few messages are necessary to perform the recovery process. Therefore, the success probability in the search for a new collaborator (p_e) disappears from the P2Cast model presented in (8).

$$C_t = \frac{W}{f_{HB}} + \left(\sigma \cdot l \cdot \frac{\ln(H)}{\ln(k)} \right) \qquad (22)$$

3.5.5 The Fault Tolerance Scheme with client buffer monitoring

In order to extend our analyses and the performance behaviour of the Fault Tolerance Scheme, we assess another detection approach for the FMP; we evaluate the technique of client buffer monitoring. In this method, each client establishes a time interval to monitor the volume of information at its buffer and calculates the transmission speed. The relation of received and consumed data on client's buffer is used in order to evaluate the quality of the streaming session. If the input rate is lower than a predefined threshold, recovery must be triggered.

The main consequence of this approach for the Load cost of the FMP is the removal of heartbeat messages for all peers. The only elements monitored by heartbeats are the Manager Nodes of each FTG. PCM/MCDB and P2Cast present the same Load cost for detection. This cost is defined by equation (23), where the Manager Nodes (N_{MN}) of each FTG exchange β messages with the server at a frequency f_{HB}. Both service schemes have no alteration at the recovery and maintenance expressions.

$$C_d = f_{HB} \cdot \beta \cdot N_{MN} \qquad (23)$$

The buffer monitoring also changes the expression for the Time cost. The detection now is based on check periods, where clients verify the quality of the input stream; nevertheless, the network can present short time fluctuations. In order to avoid unnecessary recovery processes the orphan client triggers a wait counter from the moment that it notices the input stream is under the threshold.

$$C_t = (\tau + W) + \left(\sigma \cdot l \cdot \frac{\ln(H)}{\ln(k)} \right) \qquad (24)$$

The new expression for time cost is given by equation (24), where τ is time interval to check the rate of the video stream and W is the counter waiting time. The second part of the equation represents the recovery stage and has no modifications.

Chapter 4

Simulation Environment for Load and Time Costs Evaluation

4.1 Introduction

Traditionally, mathematical language is widely applied to describe a system; it is used in many areas of knowledge such as natural sciences, engineering, economics, sociology and political science. Eykhoff [101] defined a mathematical model as '*a representation of the essential aspects of an existing system (or a system to be constructed) which presents knowledge of that system in usable form*'.

Modelling systems via a mathematical model attempts to find analytical solutions to problems and thereby enable the prediction of some aspects of the behaviour of the system from a set of parameters and initial conditions. Since the early decades of the twentieth century computational models have been used to expand the potential of systems analysis. A computational model is a computer program, or network of computers, that attempts to simulate an abstract model of a particular system. Computer simulations have become a useful part of mathematical modelling of many systems. While computer simulations might use some algorithms from purely mathematical models, computers can provide a more dynamic and scalable analysis, revealing performance limits or defects during long running times.

In order to check the behaviour of the mathematical models developed to represent the costs of the Failure Management Process, we use a simulator of video on demand systems called VoDSim [15][76][106]. The VoDSim was developed at the Computer Architecture and Operating Systems Department of the *Universitat Autònoma de Barcelona* (UAB) to provide a flexible test bed for studying and evaluating the performance of LVoD systems. The simulator was developed with the object-oriented C++ programming language for execution in a Linux-based environment and allows the configuration and tuning of multiple parameters.

The VoDSim is based on a discrete event-driven model. A discrete event simulation manages events in time; in this type of simulation, the simulator maintains a queue of

events sorted by the simulated time they should occur. The simulator reads the queue and triggers new events as each event is processed. It is not important to execute the simulation in real time, since the operation of a system is represented as a chronological sequence of events. Each event occurs at an instant in time and marks a change of state in the system. This feature provides great flexibility to the VoDSim; it is capable of simulating all the steps followed by requests from the clients to servers passing through the network infrastructure during entire video sessions.

The simulation tool also includes algorithms that define the service policies used in the content requests. In this work, we add the P2Cast service scheme to the VoDSim in order to evaluate peer failures. The P2Cast was described in Chapter 2; it is an architecture that uses the P2P approach to implement the ALM through a tree-based topology. This service mechanism is proposed to achieve scalability on VoD service and can be applied inside the Internet ASs. In addition to the P2Cast service scheme we aggregate the functionalities of peer failures and recovery, which were not present in the previous simulator implementations; some methods were added to the VoDSim in order to instrument parameters, enabling the Load and Time metrics calculation. In the following we describe the main features of the VoDSim as well as the characteristics of FMP implementation.

4.2 VoDSim

In this topic we describe the main features of the simulation tool, VoDSim. We present the design structure, the simulation model, the main simulation events, the client request life cycle, the simulator classes and, finally, the simulation settings.

4.1.1 Design structure

The simulation tool must clearly define the system which will be represented. In the design of VoDSim, the model used to describe the VoD system is represented by a hierarchical structure of software components that translate the different entities of the simulated system. The design is divided into two blocks: the system model and the client behaviour model; both blocks interact with the module of event management. The system model defines aspects of the structure and the logic involved in the

implementation: the structure defines the physical elements while the logic is responsible for resource management. The client behaviour model determines parameters such as the arrival rate, available resources (bandwidth and buffer) and collaboration willingness.

The VoD system architecture adopted is configured by a set of servers, interconnected through a network, providing service to distributed clients. On the VoDSim, the server can be specified as centralised or distributed proxies. The server is composed of several modules such as the admission control, resource management and delivery policies. The resource management policies, for example, are responsible for controlling the network bandwidth, server storage and service bandwidth, monitoring their usage during the simulation. The network is modelled by the interconnection topology and the implemented routing scheme. The topology will define the interconnection between VoD servers and plays an important role in system efficiency to serve remote petitions; the routing scheme defines if IP Multicast is enabled or the channels are transmitted via unicast.

The system and client behaviour models interact and are responsible for defining the system load. This load is described by the total number of clients and servers, their respective resources and distribution throughout the system as well as the network and content characteristics. Figure 40 shows the VoDSim structure.

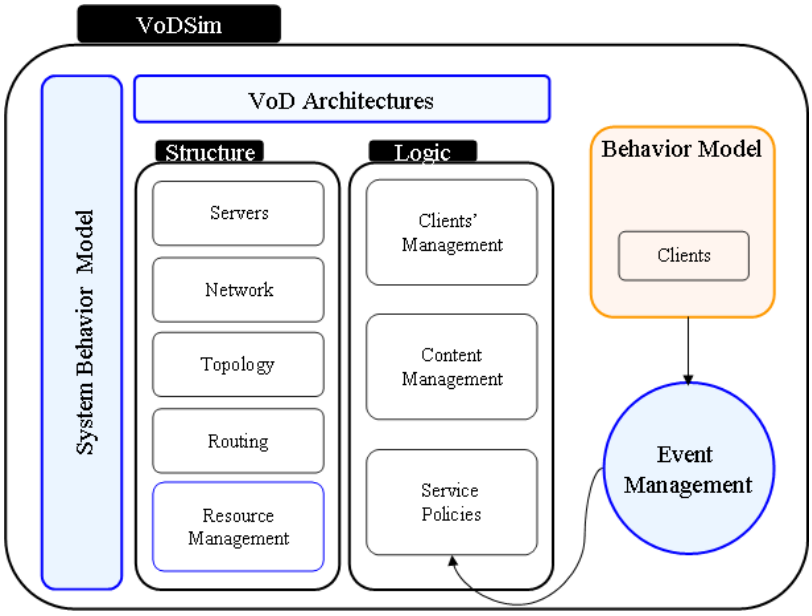


Figure 40. Design structure of VoDSim.

4.2.2 Simulation model

Modelling a system leads to different levels of detail; this issue is handled by defining how the system is abstracted in order to represent a real environment while resulting in a feasible implementation. The abstraction is the process or result of generalisation by reducing the information of a concept or phenomenon in order to retain only information which is relevant for a particular purpose; typically, when more detail is added to the representation, more complexity is incorporated in the implementation of the model and the simulation running time is incremented. The level of detail must be balanced to provide a low degree of complexity for the simulation model; this adjustment can be achieved by abstracting modules of the real system that have low impact on the target of the analyses.

The VoDSim simplifies some components of the system to reach a feasible and reliable implementation. In the following we describe abstractions assumed in the simulation model.

Service Capacity: The aspects that are not related to the VoD service and those that do not interfere with the VoDSim goals were not considered; e.g. server disk bandwidth, buffer management or disk schedulers. The server features are summarised in the service capacity. The service capacity defines the server storage capacity and the output bandwidth; these features are sufficient to model the capacity of servers in managing requests. The output bandwidth of servers and the video play rate define the maximum number of simultaneous streams supported by the service.

Network: The network complexities of dropped packets, congestion control, protocols etc., were suppressed and limited to the network bandwidth concept. The simulation model supposes the network infrastructure is enabled to guarantee the mechanisms of real-time protocols; consequently, the VoDSim assumes that the requests arrive at servers and are answered without errors. If the network bandwidth is completely used the requests are reneged without reaching the server. The routing model is simplified by adopting the shortest path between two elements of the system; thus, the path with fewer hops is always preferred. This abstraction does not consider traffic balance policies possibly implemented by routers.

Multimedia Content: The content model follows the constant bit rate (CBR) codification. The CBR encoding means that the rate at which a codec's output data should be consumed is constant. CBR is useful for streaming multimedia content in limited capacity channels since it is the maximum bit rate that matters, not the average, so CBR would be used to take advantage of all of the channel capacity; this assumption facilitates the simulation process.

In real systems the size of multimedia contents can vary. In order to provide different video sizes the simulator accepts, as input parameter, the specification of the video size in seconds. The standard value adopted in simulations is a 90-minute content.

The video format, compression rate and image quality will define the service requirements; the storage capacity and the bandwidth necessary to serve a client petition depend on these factors. The VoDSim allows us to model different video formats through the play rate specification. The standard video play rate assumed is 1.5 Mb/s, which represents a MPEG-1 compression [102]; this pattern is feasible for use in an Internet environment.

Request rate: This parameter specifies the volume of client requests generated by the system in each access point. To model this parameter, VoDSim implements a Poisson process [103]. The homogeneous Poisson process is characterised by a rate parameter λ, also known as the number of events or arrivals that occur per unit time; it represents the client petition rate. The number of events in time interval $[t,(t+\tau)]$ follows a Poisson distribution with associated parameter $\lambda \cdot \tau$; this relation is given as: $P[N(t+\tau)-N(t)] = \frac{e^{-\lambda\tau}(\lambda\tau)^k}{k!}$, where $[N(t+\tau)-N(t)]$ describes the number of events in the time interval $[t,(t+\tau)]$.

Contents popularity: It specifies how often multimedia contents are requested. A Zipf [104] distribution is used to model this parameter; Zipf's law applied to VoD service states that the probability of a video i being requested by a client is: $p(i) = \frac{1}{i^z \cdot \sum_{j=1}^{N} \frac{1}{j^z}}$, where N is the catalogue size and z is the skew factor responsible for adjusting the distribution. The skew factor adopted as standard is z = 0.729; this value makes the distribution following the Pareto principle (80-20 rule) [95]. The skew

factor is parameterised, however, and can be established by the user, changing video popularities and thus providing different workloads.

4.1.3 Simulation events

As previously mentioned, the VoDSim is built from the event-driven simulation model. Events represent state changes in the system during the simulation process. In the following we summarise the main events present on the VoDSim.

User Request (VoDUserRequestEvent): This event represents service petitions made by clients on the VoD system; it can be understood as a client request to the server for one multimedia content. The necessary parameters to model client petitions are: the client identifier (id), the simulation arrival time, the server id (request destination), the movie id and the service policy responsible for handling the request.

Renege Event (VoDRequestRenegeEvent): This event models the cancellation of a client request at its own discretion. This situation occurs when the reneging time, defined by the client, expires; this means that the client does not want to wait any longer to receive the multimedia information.

Refuse Event (VoDRefusedRequestEvent): By processing this event, the system refuses a client petition. This situation occurs when there are no more available resources in the system to serve a client, i.e. server or network bandwidths as well as the impossibility of P2P collaborations.

Acknowledge Event (VoDAckResponseEvent): This event is generated when the system accepts a client request. It allows updating of the system status and the scheduling of the subsequent events in order to serve the petition. The acknowledge event generates the necessary information to the client to start the video visualisation (server source, channel id etc.)

Sending Event (VoDDataSendEvent): It allows modelling of the server scheduler functionality. This event represents the server transmission of a portion of video to a

system element. The information associated with this event contains the source and destination as well as transmission channel identifier. It is responsible for scheduling the event of the entire sequence of video block transmissions.

End Sending Event (VoDEndDataSendEvent): In this event a transmission is finalised; this means that the client has received a video block.

End Petition Event (VoDEndUserRequestEvent): In order to have full control of resource availability, the simulator must know the state of the elements (server, network and clients). This event guarantees the resource release and allows the update of system status, i.e. this event defines when a client leaves the system, making service resources free.

4.1.4 Petition life cycle

The service starts by a client requesting a specific multimedia content from the system catalogue through a play command. The VoDUserRequestEvent is started to notify the server that a client has a petition. At same time that the client makes the request, the wait counter for petition attention is started; it determines the maximum delay allowed by a client to be served.

When the designated server receives the request its admission control is started. Depending on the service policy and the set of active channels for the requested content, the admission control will evaluate the necessary resources to serve the petition. If all required resources are available, the respective reservation is done and the service begins. If the requested content is not available in the server, the petition is redirected to another server through the network topology to be served remotely. If the content is not available at any server or there are no available resources to serve the petition, the client is informed that its request could not be served through the generation of a VoDRequestRefusedEvent. The VoDRequestRenegeEvent is generated when the start-up delay is greater than the client waiting time; it occurs, for example, when the server has the content but at a determined moment there were no available resources.

When a request is accepted by the system, the admission control policy starts the service module, sending confirmation to the client. This confirmation is triggered in the

VoDAckResponseEvent. Hence, the scheduler is commissioned to create or update the transmission channel assigned to the request. The next step is to begin the sending of the data blocks; this task is carried out by the VoDDataSendEvent. When the necessary time to data transmission is reached the client receives the video blocks from the server. Each data block reception relies on the update of client buffer information and the visualisation pointer; this will allow it to know when the video execution is finished. When the visualisation pointer reaches the end of the video, the client informs the server that the entire playback is done by using a VoDEndUserRequestEvent; the request finalisation relies on release of resources and statistics updates.

Figure 41 shows the steps followed during a client request in the VoDSim; the main modules and events involved in the process are indicated.

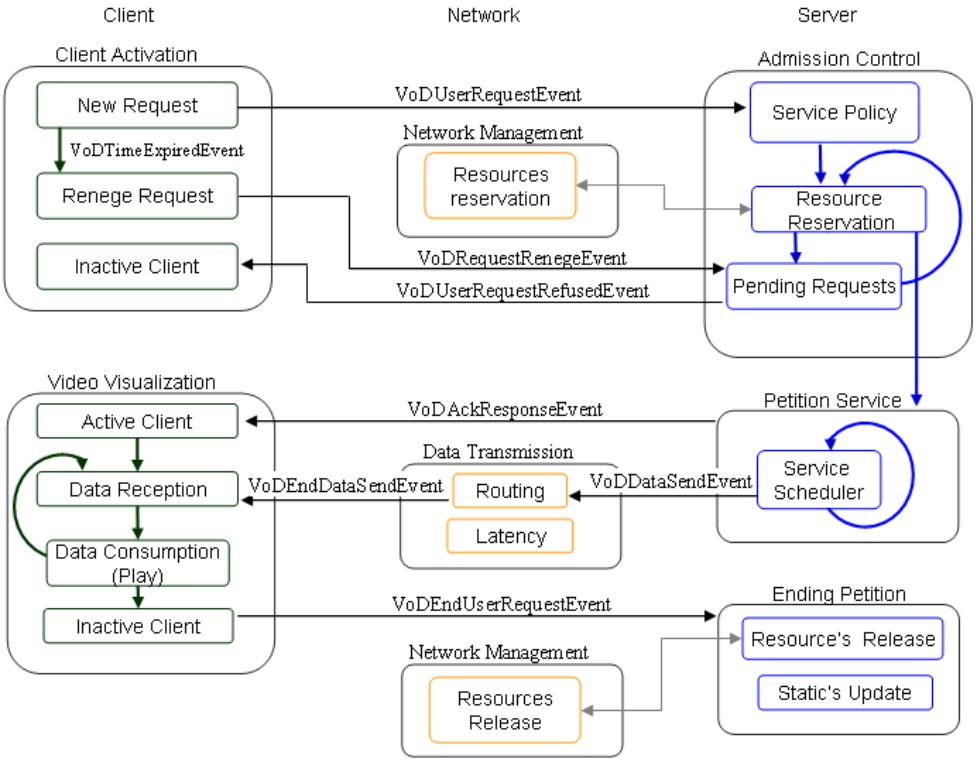


Figure 41. Client petition cycle.

4.2.5 Simulator classes

The VoDSim is based on object-oriented programming; this programming paradigm uses objects and their interactions to design applications. The object-oriented programming technique may include features such as information hiding, data abstraction, encapsulation, modularity or polymorphism. One of the main components of programmes developed using object-oriented techniques is the Class. Each class defines the abstract characteristics of a thing (object), including its features (attributes, fields or properties) and behaviours (methods, operations, etc). Classes provide modularity and structure in an object-oriented computer program.

The simulator is made up of a set of classes which represent many system elements, i.e. different classes were developed to implement the modules of the simulator. VoDSimulator is the main class. The main function of this class is to interconnect other classes providing the simulation start-up. It is responsible for reading the configuration file and generates the statistics and trace files; this class also supports event management (creation, execution and releasing). The other classes are implemented to model each one of the VoD components. The VoDSystem class groups the system architectures and the active users; therefore, the main functionalities of user management are implemented in this class.

Following the same line of thought, other classes implement elements such as servers, network, transmission channels or service policies. For example, different types of servers are managed into different classes. VoDServer implements all the functionalities related to server bandwidth resource; this class is also responsible for the algorithms related to the requests (start playback, stop transmission, service cancellation, etc). The VoDProxy is a sub-class of VoDServer and implements the standard server functionalities, as well as the policies necessary to handle the stored files (Cache\Mirror placement policy). The VoDChannel implements the transmission channel between two or more system components supporting the unicast or multicast communication techniques. The VoDChannelP2P is a sub-class of VoDChannel with the capability to communicate with end users.

In this work we added new functionalities to the VoDSim simulator; the VoDMulticastP2CastServicePolicy and VoDChannelALM were aggregated to run the P2Cast service scheme and the peer failures. More details about the classes and VoDSim

implementation can be found in [15] [76] [79] [106] [107]. Figure 42 illustrates some of the classes implemented by the simulation tool.

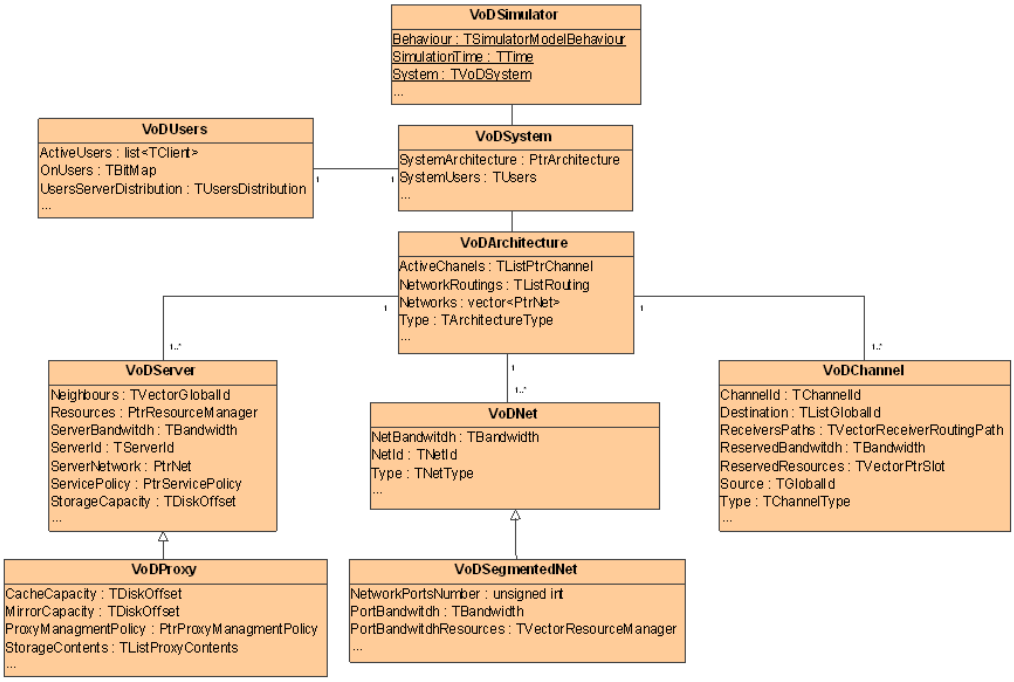


Figure 42. Sample classes of the VoDSim implementation.

4.1.6 Simulation setting

Finally, the configuration of the simulation process is done by the user, who must specify some system parameters. Basically, the input information can be divided into three categories:

Test type: By configuring some parameters the user is able to choose the type of experimentation desired. He\she can define the number of simulations to be executed, the type of output or the end condition of the simulation.

System characteristics: In this configuration step, all the parameters that affect the system architecture are defined. The set of parameters allows the definition of the architecture, resources management policies, service policies, system components (network, servers, etc); moreover, the specific configuration of each one of the components is available, i.e. network and server bandwidths, storage capacity etc.

Workload: The parameters defined by the system workload configure the system behaviour model (catalogue distribution, size, etc.) and the client behaviour in the simulation (arrival rate, content popularity etc.).

4.3 P2Cast Service Scheme

P2Cast uses the P2P approach to stream multimedia information cooperatively; it provides scalability to the VoD service. In P2Cast the clients act as data source while they receive the content; hence, this scheme creates an ALM tree-based architecture to provide video transmission through peers. Clients are assumed to have sufficient upload bandwidth to feed more than one video stream; at the network layer the approach only relies on unicast connections among peers. In the following we describe the main technical features of P2Cast used in the implementation as well as its service performance.

4.3.1 Join algorithm

P2Cast is based on distributing the video stream through a P2P tree topology. In the construction of the overlay topology the Application Layer Multicast tree is created directly (without construction of the underlying mesh); peers need only perform the data forwarding function. The arrival or departure of clients triggers the reconstruction of the tree. As mentioned in Chapter 2, the P2Cast applies the patching approach to group clients, and thus it presents two types of streams: patch and base (the base stream flows through the base tree).

A new client arriving at the system contacts the server to establish the base tree joining process. The VoD service requires a minimum amount of available bandwidth from a parent peer to serve a child client; thus, the server measures the available bandwidth from itself to the new client and decides whether this client can be its child or not. If the server admits the new client, this client joins the base tree and receives the base stream from the server. Otherwise, the server redirects the new client to one of its child clients (candidates). The candidate client makes its own decision about the admission of this new client; if the candidate does not serve the new client, it is

redirected to its child node. The process continues recursively until the client successfully joins the base tree or is rejected by insufficiency of resources.

Clients which arrive after the beginning of the session will need to get the initial portion of the video; a patch server is responsible for serving this portion to a new client. The first client in a session receives the entire video (base stream) from the server; all other clients will miss the initial part of the video and will require a patch. Since the server stores the entire video, it can always be a patch server as long as it has sufficient bandwidth; a peer that arrives earlier and has sufficient bandwidth can also, however, be a patch server.

The algorithm used during the join process is referred to as the Best Fit (BF); it is in charge of building the base tree and selecting the patch server. In the following we describe the BF algorithm used in the implementation of P2Cast in the VoDSim.

4.3.2 Best Fit algorithm

The following procedure is executed by the requesting client.

(a) Step 1. The requesting client N contacts a candidate parent P (starting with the server).

(b) Step 2. P estimates the bandwidth from P to N, B(P, N). Meanwhile, it sends message to all of its direct children in the base tree, denoted as C(P), asking them to measure their respective bandwidth to the requesting client.

(c) Step 3. P collects the bandwidth measurement from its children, and identifies the child client $C_{max}(P)$ that has more bandwidth available to N, i.e. C_{max} = MAX{B(C, N)}. Depending on the measurement reported back to P, there are two alternatives: ***(1)*** candidate P has the greatest available bandwidth to the requesting client N (i.e. $B(P, N) > B(C_{max}, N)$); or ***(2)*** one of the children has more output bandwidth volume to N ($B(P, N) \leq B(C_{max}, N)$).

(1) If $B(P, N) > B(C_{max}, N)$, then P has is enabled to support at least one stream. If N only looks for the base stream, it can join the base tree using P as its parent. If a

patch stream is required and P arrives earlier than the requesting client, then P becomes the patch server. If both a base stream and a patch are required, the patch has the priority over the base stream. If P can serve the patch, it will become N's patch server. If P has sufficient leftover bandwidth to serve the base stream, N joins the base tree with P as parent. If P cannot completely handle N's request, a candidate peer is forwarded to N; thus, N contacts the indicated candidate and starts from Step 1 again.

(2) If $B(P, N) \leq B(C_{max}, N)$, then N is redirected to C_{max} and starts from Step 1.

4.3.3 Failure management

Owing to the ALM tree topology implemented by P2Cast, the departure of a client (or link\path failure as well as available bandwidth fluctuation) can disrupt the delivery of the base or patch streams. P2Cast provides forms of failure recovery to both; owing to the short life of the patch compared with the base stream, however, the base recovery is focused on the analysis.

If the parent of a client *A* departs early, it is disrupted from the tree; all clients belonging to the sub-tree rooted at this client (*A*) can be affected by the failure. To prevent the server from receiving a large number of recovery requests, P2Cast only allows client *A* to contact the server and perform recovery. The recovery process is identical to the joining process; if another peer on the session is able to assume the service, the recovery attempt succeeds and the entire sub-tree is recovered. If it fails, client *A* is rejected. Then, the children of client *A* (orphan clients) will contact the server and start the recovery process representing their own sub-trees. This process continues recursively. Originally, the detection on P2Cast is performed by peers constantly monitoring the incoming traffic; when the incoming traffic's quality degrades to a certain degree, the recovery process is triggered.

4.3.4 P2Cast implementation

The VoDSim simulator already had unicast transmission implemented as well as patching and batching multicast service strategies. In this work, we extended the

functionalities of the VoDSim through the implementation of P2Cast service policy, which is based on the use of ALM distribution trees.

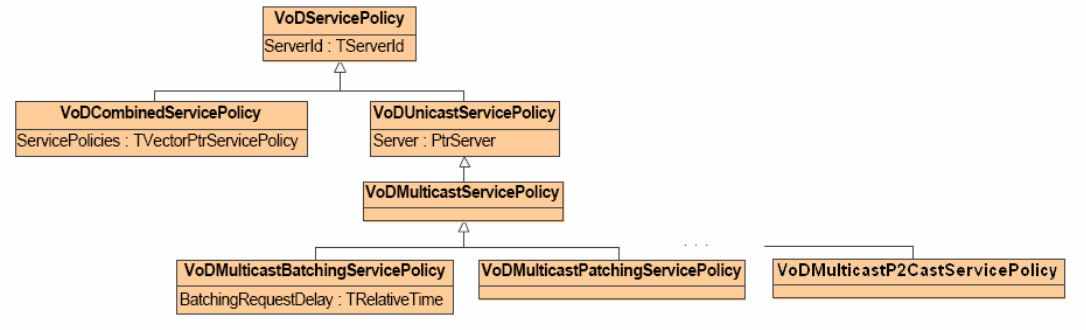


Figure 43. P2Cast service scheme incorporated to VoDSim classes.

The aggregation of P2Cast demanded the design of a new class derived from the VoDChannel, so-called VoDChannelALM class. It is responsible for creating the overlay network of clients, thus establishing the ALM sessions. The main class incorporated in the simulator is named VoDMulticastP2CastServicePolicy; this class carries out the implementation of the P2Cast and is responsible for running the Best Fit algorithm and all necessary steps when serving a client petition (establishment of connections between clients, opening and closing of ALM sessions etc.). Figure 43 shows a scheme locating the implemented VoDMulticastP2CastServicePolicy class on the VoDSim; next we describe the main functionalities that made up the implementation.

FindChannelALM: This method is responsible for verifying if some ongoing ALM session is able to host a joining client. FindChannelALM checks if the arrival time of the client request is lower than the pre-established threshold; it also verifies the sessions which are playing the same content required by the client. If an ALM channel fulfils the requirements, it will be shown as ready to receive the new client.

SearchCollaboration: The Best Fit algorithm runs by this method; it means that the SearchCollaboration compares the available bandwidth between the parent candidates (server or peers) and the new client. At the end of the process, the method selects the most feasible element to serve the base and patch streams.

CreateNewALMSession: When there are no ongoing ALM sessions enabled to host the joining client, this method is invoked. If the server has sufficient resources, a new ALM session is opened and the joining client is the first member; following petitions can be added onto this session when arriving within the established threshold. It should be noted that the threshold is defined in this method on the creation of the ALM session.

PlayMovie: This is the basic method of the implementation since it receives the request of a new client on the P2Cast service scheme. All the calls to any other methods of the implementation start on the PlayMovie; it receives the return values of each method and then takes the necessary actions. The new client can be added to an ongoing ALM session, a new session may be created or the request can be dropped owing to lack of resources.

4.3.5 P2Cast service performance

Aiming to assess the implementation of the P2Cast service policy, in this topic we evaluate the behaviour of the server load when the request rate and catalogue size vary. The unicast and multicast (patching scheme) are used as background to analyse the P2Cast performance. The server load metric can reflect the benefits provided by the P2P and multicast strategies; it is defined as the average load generated to serve the petitions during the simulation process. Table 5 shows the simulation settings.

Parameter	Value	Parameter	Value
Catalogue size	[10;200]	Server storage capacity	800GB
Playback rate	1.5 Mb/s	Video length	90 minutes
Peers upload bandwidth	3.0 Mb/s	Peers buffer capacity	5 minutes (56MB)
Server bandwidth	400 Mb/s	Request rate	[2;70] requests/min.

Table 5. Experimental settings for P2Cast simulation.

Request rate evaluation: We analyse the effect of the request rate over the server load assuming that clients do not depart during the entire simulation. Figures 44 and 45 show the behaviour of the server load; to serve 60 requests per minute, the patching scheme demands almost 1300 server channels. P2Cast reduces the server load by 42%

compared with the patching strategy (1300 channels vs. 750 channels); when compared with pure unicast, the reduction of the server load provided by P2Cast reaches 86% (5500 channels vs. 750 channels). P2Cast demonstrates its scalability feature by using clients' resources to distribute the server load. Nevertheless, its application can generate extra loads at the network level, since the overlay topology of clients is built through unicast connections.

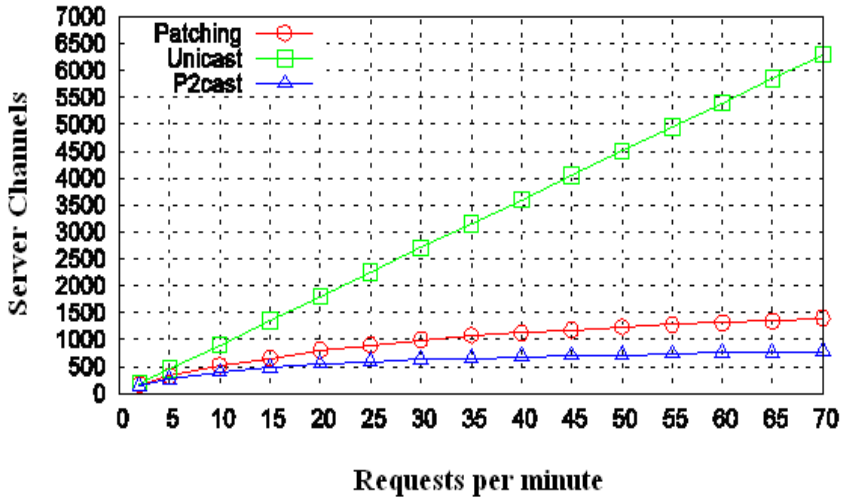


Figure 44. P2Cast performance evaluation: Arrival rate x Server load.

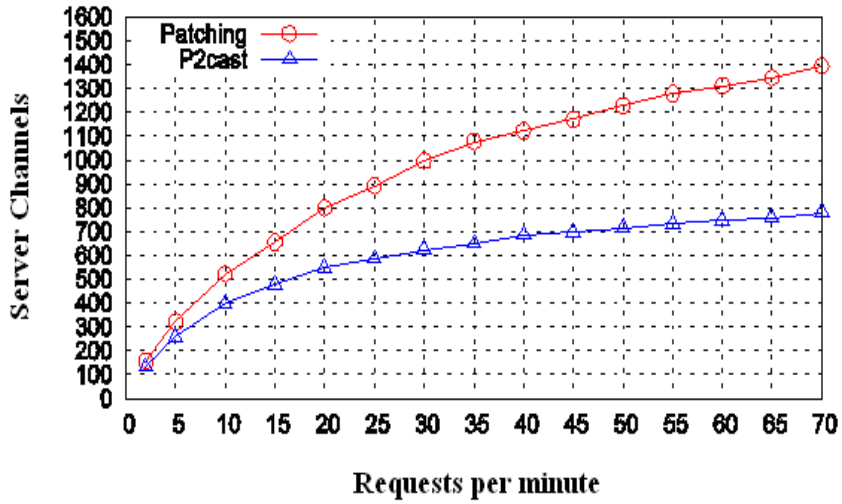


Figure 45. Arrival rate x Server load (Patching x P2Cast detail).

Catalogue size: By analysing the server load behaviour when the catalogue size varies it is possible to verify the P2Cast scalability feature; as assumed in the previous case, clients do not depart during the entire simulation. The request rate is now fixed at 30 requests per minute. Figure 46 shows the results; compared with patching, P2Cast can decrease the server bandwidth consumption by 33% (160 videos: 1200 channels vs. 800 channels). The unicast service scheme presents a constant behaviour while the catalogue size grows; this service strategy is independent of the catalogue characteristics (size, popularity etc.). At the end of the simulation, the number of unicast channels served is directly proportional to the request rate and the simulation time. The patching and P2Cast schemes rely on sharing resources (at network level and application level, respectively); therefore, content features will influence their behaviours because the number and size of video sessions will vary according to content popularity, availability etc. When P2Cast is compared with unicast with a catalogue of 160 videos, the server load is reduced by 70% (2700 channels vs. 800 channels).

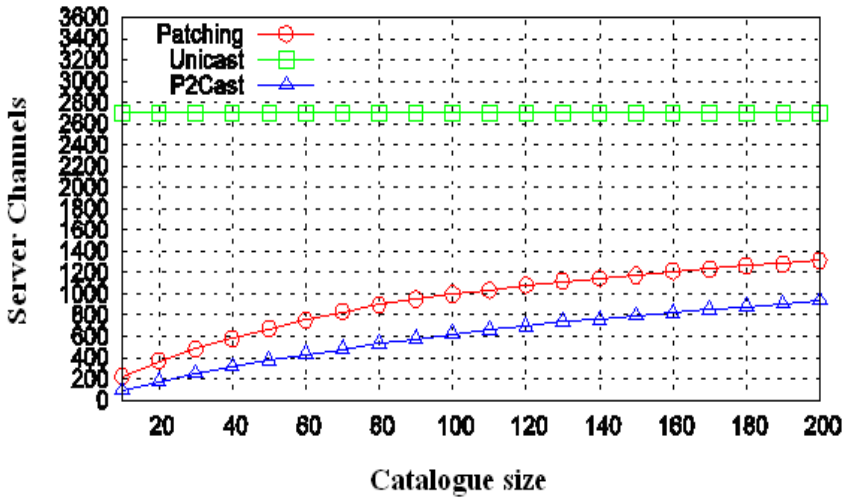


Figure 46. P2Cast performance evaluation: Catalogue size x Server load.

4.4 Peer Failures

The P2P strategy consists of the usage of end hosts resources in order to collaborate on service tasks. Since the peers are end users, they are free to join or leave the system at any instant. The frequencies of clients arriving or leaving the system in the simulation environment are defined by probability functions.

The VoDSim uses the Poisson distribution [103], which is widely deployed in video systems simulations, to model the clients' arrival process. It assumes that the inter-arrival time of client requests is $1/\lambda$, where λ is the request rate per unit time.

In this work we add to the VoDSim the peers departure functionality. The Weibull distribution [108] is selected to model the process of peer failures. This distribution is one of the widely used lifetime distributions in engineering; it is an exponential-like distribution often used in reliability testing and failure analysis.

The Weibull cumulative distribution function (CDF) is given by ($1 - e^{-(x/\beta)^{\alpha}}$), where α is the scale parameter and β is the shape parameter; it represents the probability of a peer to fail at a time x or less during a video session. The complementary cumulative distribution function (CCDF) is represented by $e^{-(x/\beta)^{\alpha}}$ and reflects the lifetime of a peer in the system; peers are more prone to fail in long video sessions.

On [109] the Weibull distribution functions are used to fit measured peers lifetime; they also verify the average peer lifetime. The measurements show that for different applications, the peer lifetime CCDFs follow a Weibull distribution but do not have the same average connection time.

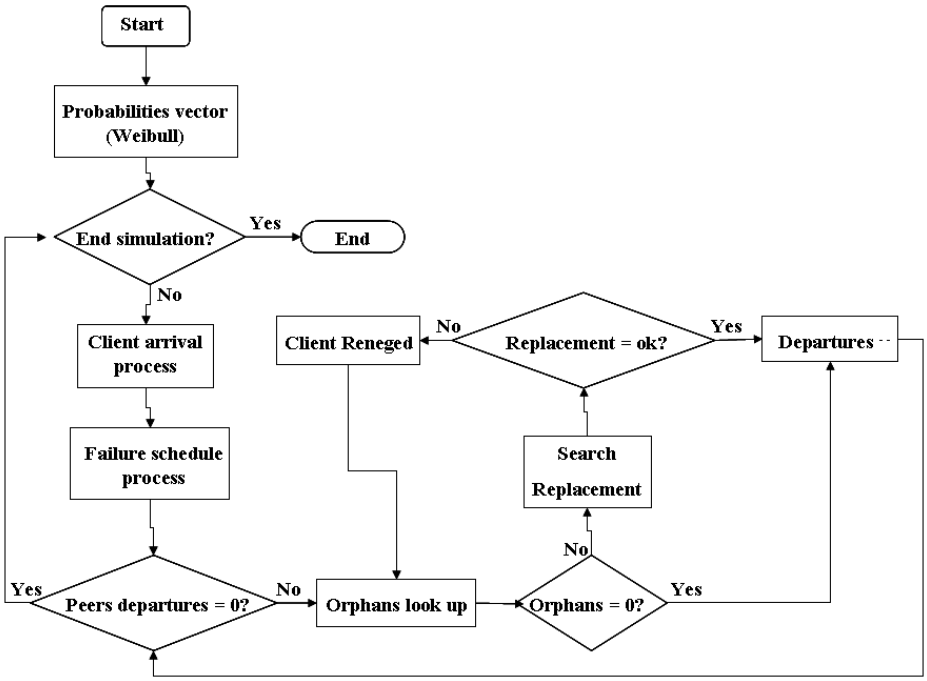


Figure 47. Flow chart of the peer failure schedule procedure.

Figure 47 shows how peer failures are scheduled. The VoDSim implementation of peer departures is oriented to a session of 90 minutes. At the beginning of the simulation process the values of the Weibull CDF for the range [0;90] minutes are calculated and stored at a vector. Thus, with each new request arrival, the system verifies the need to schedule peer failures. The method responsible for scheduling departures first looks at the simulation time, then retrieves the respective failure probability from the vector. The probability is multiplied by the total number of clients present in the system at that moment; if the result is equal to or greater than one, failures are scheduled. When the number of peers that will leave the system is defined, the clients are chosen according to their range of arrival time. The next step is to verify the orphans of the elected failed clients and try to find a substitute. The method looks at each orphan and seeks a replacement peer, in the same session, enabled to assume the service. If this operation does not succeed, the orphan is scheduled to leave the system; thus, the algorithm has to repeat the process, verifying the orphans left by the previous client. The process is then repeated for each client scheduled to depart at the beginning.

At each recovery process, the Load and Time costs are measured and stored on global variables. The Load cost is calculated considering the simulation time and the messages involved in the process; detection and maintenance phases only depend on the simulation time while the recovery cost is added at each search for replacement. The Load cost is updated until the simulation reaches the end. The Time cost counting starts when a peer is elected to be disrupted by a failure; it uses latency and hop count information provided in each recovery process to calculate the time spent on this stage. The detection portion of the cost is defined by the monitoring interval ($1/f_{HB}$) and is considered just once, at the beginning of the search process; each time an orphan is rejected and leaves other orphans on the system, the Time cost is incremented. The counting finishes when the recovery succeeds or the entire sub-tree rooted at the first disrupted client is removed. The assessment of peer failures in the simulation environment is presented in Chapter 5, allied to the evaluation of the Failure Management Process and the Fault Tolerance Scheme.

Chapter 5

Experimental Evaluation

5.1 Introduction

In this chapter we perform the evaluation of the Load and Time costs as well as the assessment of the proposed Fault Tolerance Scheme.

We start defining the environment adopted to perform the evaluations. We assume a transit-stub model generated by GT-ITM [110] to represent the Internet topology. In the same topic we expose the considerations related to the communications protocols and IP Multicast as well as some expressions used during the analyses.

The models based on the Failure Management Process are validated through simulation using the VoDSim; the P2cast policy is adopted in the validation. Moreover, in the experimental evaluation we confirm the importance of the control in P2P-based video streaming services; the overhead imposed on the system and the time constraint are extremely significant in the design of Internet P2P-VoD services.

In the following, we analyse the behaviour of both costs applied to PCM/MCDB [74] [75] [76] and to P2cast [81]. The proposed Fault Tolerance Scheme (FTS) is applied to the selected service schemes. The behaviour of the Load cost in the PCM/MCDB and P2cast schemes is evaluated under five different parameters:

Number of clients in the system: This parameter allows evaluation of the scalability of the FMP as the quantity of active clients in the system grows.

Number of multicast groups: A multicast group includes clients who are visualising the same video and who arrived inside the time interval necessary to join the multicast channel. Groups can be formed by either ALM sessions or IP Multicast channels.

***Multicast group size*:** The size of multicast groups varies according to clients' behaviour. The number of multicast groups and their size indicate the content popularity as well as the arrival pattern of clients.

Failure Frequency: End host faults periodically happen. Client failures can be described by a probability distribution (Weibull); also, measurements on real IPTV systems indicate that clients are connected for at least five minutes[109]. When a peer failure occurs the recovery process is triggered in order to find a substitute collaborator. This parameter allows the evaluation of the Load cost according to the rate of peer failures.

Heartbeat Frequency: Heartbeat messages are one of the main ways to monitor peers. The rate of heartbeat message exchange strongly affects the FMP; high monitoring frequencies may create great overheads.

The Time cost behaviour is observed with respect to three parameters:

Network Latency: The messages exchanged during the recovery will determine the time consumed at this stage. The latency at the network edges plays an important role in substituting for failed peers.

Network dimension: The network size is defined by the number of routes that make up such a network. This parameter allows evaluation of the influence of the underlying network infrastructure in the FMP.

Heartbeat Frequency: As in the Load cost analysis, the frequency of exchange of heartbeat messages has great impact on the time performance of the FMP. The monitoring frequency represents an important part of the time consumed in handling peer failures.

Finally, the service performance of the FTS is analysed. On the basis of the formation law defined in Chapter 3, we assess the time necessary for the FTS establishment and the impact of the number of simultaneous failures supported on the VoD system.

5.2 Experimental Environment Definition

The first step in the experimental evaluation is the definition of the system environment and the network topology. In this work, the performance evaluation is carried out by

adopting a router topology generated by GT-ITM [110]. The GT-ITM tool supports the study of large internetworks through scalable and realistic models. The Internet research comprises a wide range of internetworking problems (routing, resource reservation, administration etc.); the study of protocols, algorithms and policies to address such problems often involves analysis using an abstraction of the system. The aim of GT-ITM is to provide a tool for Internet environment investigation; generally, it is more efficient to assess solutions using analytical models or simulation to avoid the waste of efforts and resources.

The network topology used in the experimental evaluation presents two levels: one transit-domain and six stub-domains. A stub-domain carries only traffic that originates or terminates in the domain. Transit-domains do not have this restriction. The purpose of transit-domains is to interconnect stub-domains efficiently; without them, every pair of stub-domains would need to be directly connected to each other. Stub-domains generally correspond to campus networks or other collections of interconnected local area networks (LANs), while transit-domains are almost always metropolitan or wide area networks (MANS or WANs).

In the adopted topology, the six stub-domains are made up of 54 routers, where each one has associated local networks with limited capacity to connect clients; each router can lead to the connection of a limited number of clients. The transit-domain comprises three routers that have no clients directly associated; the video server is connected to one of the transit routers. The topology is presented in figure 48 and its average connectivity is $k = 3$ (mean number of edges connected at a router).

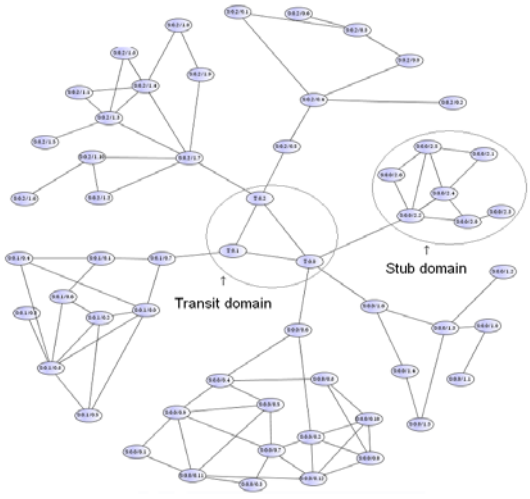


Figure 48. Network topology generated through GT-ITM.

The routers are networking devices enabled with software and hardware developed to provide the routing and forwarding of information. To accomplish with these tasks, routers run specific algorithms and protocols. In the experiments we assume routers running IP Multicast through IGMP [54] and PIM-SM [55] protocols. The IGMP is responsible for group creation and management and PIM-SM is the protocol in charge of the data distribution tree. The IGMP and the PIM-SM are two of the most applied protocols in IP Multicast deployment.

In the unicast implementation we admit OSPF (Open Shortest Path First) protocol [91]. OSPF is a dynamic routing protocol for use in IP networks to establish point-to-point communications. Specifically, it is a link-state routing protocol operating within an Autonomous System (every node constructs a map of the connectivity of the network, in the form of a graph, showing which nodes are connected to which other nodes; each node then independently calculates the best next hop from it to every possible destination in the network forming the node's routing table). The routers' messages (IP Multicast and unicast) involved in the FMP are described in table 6.

	Description	Stage
IP Multicast PIM-SM	**Hello** - sent periodically on each PIM-enabled interface, with a destination address for all the PIM routers multicast group.	*Maintenance*
	Join/Prune - consists of a list of groups and a list of Joined and Pruned sources for each group in order to build the distribution tree.	*Recovery \ Maintenance*
IP Multicast IGMP	**Create Group Request** - requests the creation of a new transient host group.	*Recovery*
	General Query - periodically solicits the group membership information.	*Recovery*
	Host Membership Report - message to the group address for each group to which a host desires to belong.	*Recovery*
Unicast OSPF	**Hello** - used to perform neighbour discovery, continually sent to notice when connectivity has failed.	*Maintenance*
	Link State Advertisement - communicates the router's local routing topology to all other local routers in the same OSPF area.	*Maintenance*

Table 6. Messages of IP Multicast and Unicast protocols.

On the basis of the network protocols adopted and the communication between peers and server, we define the messages involved in the FMP; the number of messages in the detection, recovery and maintenance phases is shown in table 7. For the sake of simplicity, in the maintenance phase, we assume the frequencies of client update messages (f_{CI}) and router status messages (f_{TI}) as one every fifteen seconds [111] [112] [113] [114]. The number of detection counter cycles W is fixed at three. For all evaluations we vary one parameter at a time while attributing fixed values to the others; therefore, Load and Time costs will be modified according to the variation of the parameter under analysis describing the cost behaviour.

Settings changes are described in each experiment; the available bandwidth and buffer capacity of peers generally refer to the resources reserved for the Fault Tolerance Scheme. When the FTS is not under evaluation these resources are peers' service resources.

Parameter	Messages	Description	Stage
β	1	1 message from a peer to inform that it is alive (heartbeat).	*Detection*
σ	4	1 message triggers the recovery process; 1 message to inform the orphan about the new source; 2 messages for the handshaking between orphan and new peer.	*Recovery*
ρ	2	1 message to query a peer about its status; 1 message for peer answer.	*Recovery*
φ	2	IGMP: group membership and reply.	*Recovery*
γ	1	PIM-SM: Join/Prune	*Recovery*
ω	2	1 update message with peer status; 1 acknowledgement.	*Maintenance*
α	2	OSPF: 1 Hello + 1 LSA	*Maintenance*
α	2	PIM-SM: 1 Hello + 1 Join/Prune	*Maintenance*

Table 7. Messages used in the Failure Management Process.

In the first part of the experimental evaluation, we validate the FMP model; analytical results are compared with simulation results. We use a range of 800 simulations to obtain the simulated costs. The curves are drawn with the average simulated values and present the standard deviation expressed in equation (25). Table 8 shows the parameters used in the expression of the experimental evaluation.

$$s = \sqrt{\frac{(Z - M)^2}{N}} \qquad (25)$$

Z	Simulated value
M	Average simulated cost.
N	Number of simulation samples.
s	Standard deviation.
Δ	Percentage gain\loss in the FTS application.
C_{SP}	Cost of the FMP for a service policy.
C_{SP_FTS}	Cost of the FMP for a service policy applying the proposed FTS.
Δ_w	Percentage of control traffic compared with the server video traffic.
$Tr_{control}$	Control traffic generated by the FMP.
Tr_{server_video}	Server video traffic.

Table 8. Parameters used in the expressions of the experimental evaluation.

The proposed FTS is evaluated by comparing the costs of the FMP in PCM/MCDB and P2Cast. Equation 26 represents the difference in the costs (Δ) when service schemes use the FTS and without it. If the application of the FTS generates a reduction in the cost, the value of Δ is negative and reflects the gain in the FTS usage (lower costs). On the other hand, if the FTS approach adds cost to the FMP of the service scheme, the value is positive and represents a loss in the system performance.

$$\Delta = \frac{(C_{SP\_FTS} - C_{SP.})}{C_{SP}} \cdot 100 \qquad (26)$$

The expression described in equation 27 is used to represent the weight of the control scheme in the system performance. It reflects the percentage of the control traffic compared with the video traffic generated by the server.

$$\Delta_w = \frac{Tr_{control}}{Tr_{server\_video}} \cdot 100 \qquad (27)$$

5.3 Analytical Model Validation Using VoDSim

In this section we compare the cost patterns described by the analytical models with simulation results. The objective is to verify if the behaviours generated by the expressions are valid in a more dynamic situation. Load and Time cost models of the Failure

Management Process (detection, recovery and maintenance) are evaluated through the VoDSim; the P2cast service scheme is adopted in the validation.

5.3.1 Load cost

The Load cost is based on the volume of messages exchanged during the FMP; it represents the control overhead imposed on the system in dealing with peer failures. The configuration parameters in the simulated experiments are presented in table 9.

Parameter	Value
Request rate	10 - 60 requests/minute.
Client's output bandwidth	3000 kb/s.
Client's buffer	113MB (10 min. of video).
Video catalogue	1 video.
Video length	90 minutes.
Video play rate	1500 kb/s.
P2Cast threshold	10% of video length (540 sec.).

Table 9. Simulation settings.

Number of clients in the system: Figure 49 shows the behaviour of the P2Cast Load cost observed by the application of the analytical model and simulation. Results show that the Load cost increases as the number of clients grows. This behaviour was expected owing to the greater probability of failures when there are more peers in the system, demanding high activity of the FMP. When the system reaches something in the order of thousands of clients, the curves present a more significant difference and the simulated results have a bigger standard deviation. The difference presented between analytical and simulated results can be because of the loss of precision of the analytical models when the interactivity of systems' elements is high (thousand peers); simulation provides more dynamic and realistic scenarios while in the analytical models only the parameter that is being discussed is varied. The greater standard deviation in the simulated results could be generated by the mentioned dynamicity provided by the simulator; when there are many peers on the system, there are many failed management processes and thus the range of the costs can be greater owing to the various possible conditions on treating faults (e.g. latency, path size, tree length). Nevertheless, simulated and analytical results present the same behaviour pattern, which supports the developed models and enables their usage as evaluation tools of the FMP performance.

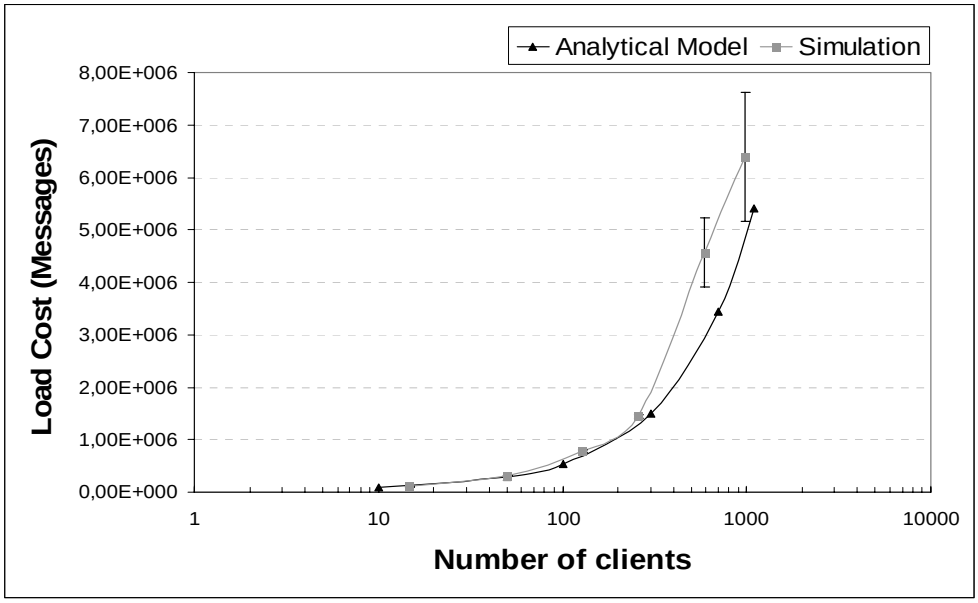


Figure 49. Load cost validation: Simulation x Analytical model.

5.3.2 Time cost

The Time cost metric is meant to express the time consumed in the FMP, i.e. how long the detection and recovery take when a peer fails. According to the analytical model developed (equation 8), the detection stage as well as the network latency and size contributes to the total time cost. In the P2Cast service scheme, however, the determining matter in time consumption during the FMP is the recursive recovery. When a recovery process succeeds, the ALM distribution tree is recovered; if it fails, the orphan client is rejected and its children restart the recovery process. In this scenario, the simulation settings are the same as those presented in table 9.

Success probability in the search for a replacement peer: In P2Cast each ALM session is made up of various end hosts. The search for a new peer is performed only inside the session where the fault has occurred; thus, the probability of success in the process of querying peers for collaboration depends on the number of end hosts in such ALM sessions. Small sessions present few clients to be queried, resulting in a reduced number of messages and less time consumption. In figure 50 it is possible to observe the behaviour of the Time cost by means of the analytical model and simulation results. The curves show that the Time

cost diminishes as the probability of finding a new peer increases; high probabilities mean a reduced number of queries necessary to find the replacement peer and thus a low Time cost, bounded by the detection phase. Again results present variability at the curves' extreme, when the system size is greater. Low success probability in the search for collaboration generally occurs in big sessions which present a great range of clients to be consulted: the variations can be caused for the reasons presented in the Load cost evaluation; simulation exploits the dynamic of the service better than the analytical models, and relies on the observation of one parameter each time. Also, the variation in the standard deviation observed at low success probabilities is directly related to the range of distinct conditions created during the simulation process (latency, path size etc.); this variability is made evident as the number of queries for recovery is greater. Figure 50, however, shows the curves generated through simulation and analytical models with the same behaviour pattern; it corroborates the validity of the models and their application in the assessment of the FMP.

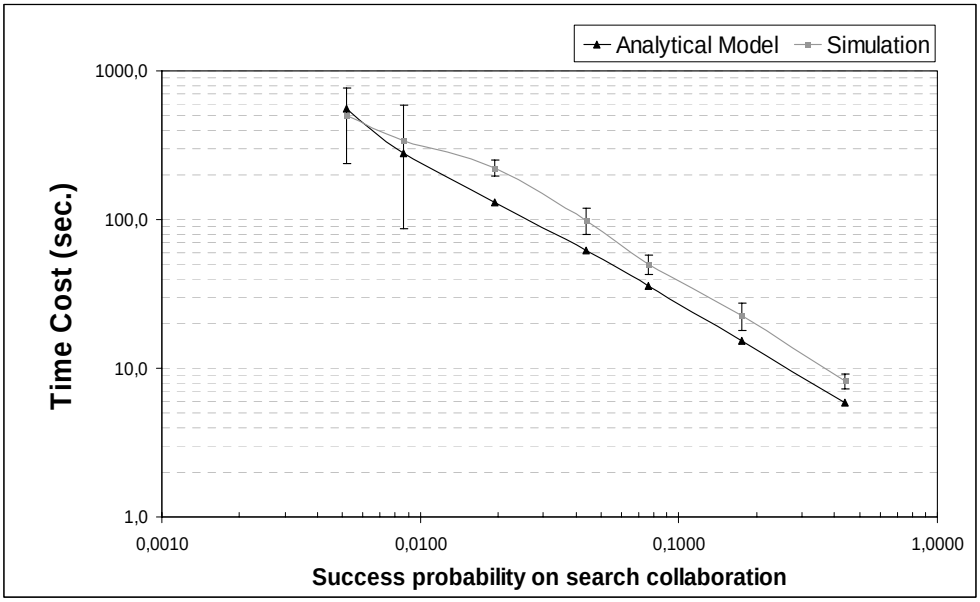


Figure 50. Time cost validation: Simulation x Analytical model.

The simulation tool still provides flexibility for some evaluations of P2Cast and its ALM delivery tree topology that the analytical models do not. Analysing figures 51 and 52, we can observe that systems with more clients can present higher Time costs. By varying the clients' arrival rate it is possible to evaluate the three different scenarios presented in figure 51: 100, 500 and 1000 clients. A high arrival rate means more clients in the system and more clients

sharing the same P2Cast session, which can be translated as larger ALM trees and greater time consumption in the recovery phase. The Time cost is directly related to the number of peers that make up a distribution tree because of the recursive nature of the recovery process and the querying approach applied to the search for a new collaborator.

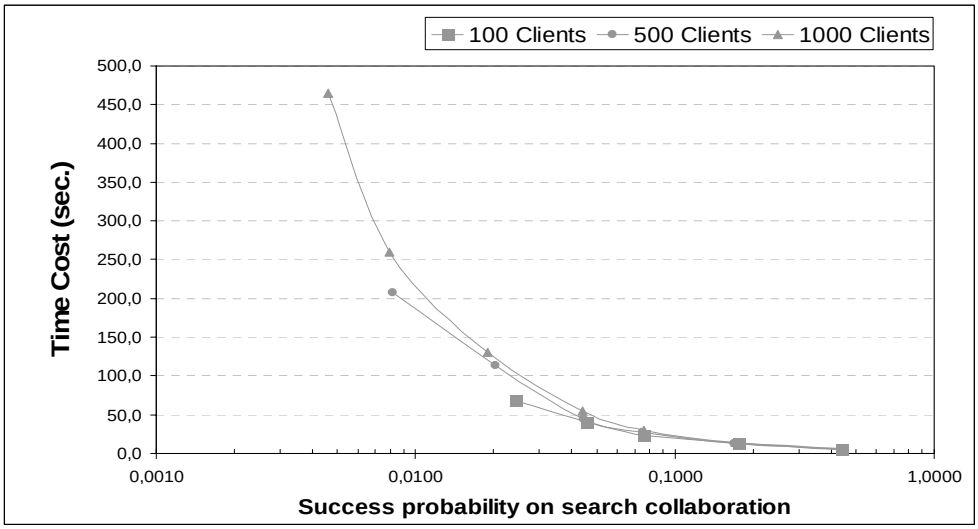


Figure 51. Simulated results for Time cost in systems with 100, 500 and 1000 clients.

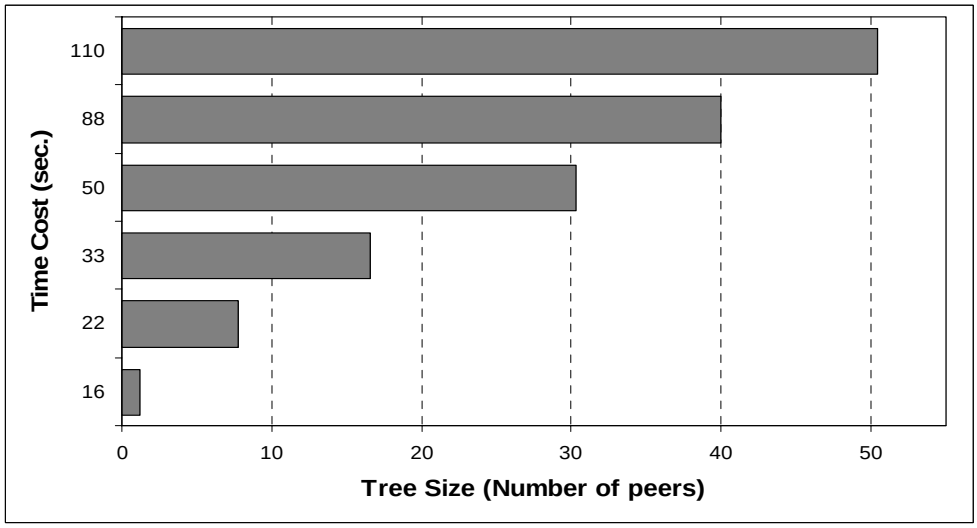


Figure 52. Influence of ALM tree size over Time cost.

5.4 Control vs. Multimedia Traffic

In this section we aim to express the importance of the control mechanism; it is responsible for the coordination and synchronisation of the service. P2P and multicast paradigms bring scalability and improve performance of VoD systems; they demand more control, however, to achieve the goals for which they were proposed. The P2P-VoD service over the Internet must be designed carefully, taking into account the control mechanism; the design must make the extra load that the control scheme can represent as low as possible.

Our study is focused on the evaluation of the number of control messages that flow through the system during the FMP. The overhead of a protocol largely depends on the overlay structure used and how it is maintained along the time; thus, the information about the control Load cost of the FMP is very useful.

Messages can be of various sizes on the system owing to their different natures. Messages carry information and play different roles; heartbeat messages, for example, generally do not have a great amount of data because they are used simply to inform that an element is alive. On the other hand, clients' status messages or Link State Advertisement (on the OSPF routing protocol) can be heavier because the data carried by these messages depend on the information transported. Clients' status messages can contain information about peer bandwidth, list of related peers, the content and visualising time; the Link State Advertisement can transport graph information and its size depends on the number of routers in the network area.

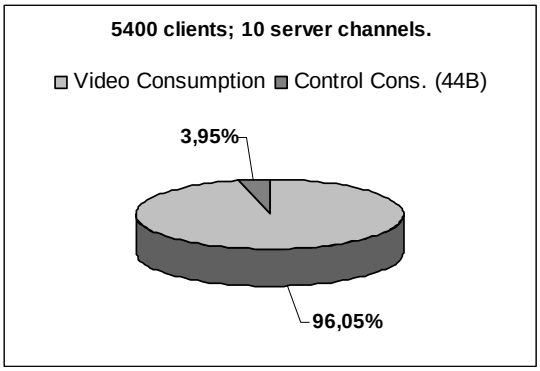
In computer science messages are basically made up of two parts. The portion of the message that carries the data is called the body. The initial part of a message follows a clear and unambiguous specification or format and is referred to as the header. The header is supplemental information placed at the beginning of a block of data being transmitted; it carries information such as the sender's and the recipient's IP addresses or the protocol governing the format. The header of TCP packets can vary from 20B to 60B [92] and the UDP header has a theoretical minimum size of 8B [93], without space for data. Nevertheless, generally no TCP or UDP messages are generated without at least some data. Similarly, messages of routing protocols can vary from a few bytes to more than 100B, depending on the protocol, type of message and network size [114]. Hence, in order to consider the variability in the messages size we assume a range of values. First, we consider messages of 44B, 92B, 128B and 256B. In the other experiments we adopt a range of messages from 44B to 128B.

Each video session is provided by a multicast channel (IP Multicast or ALM); these sessions represent the communication channels provided by the server with a constant bit rate of 1.5Mb/s. Clients are grouped into the video sessions according to their arrival time in the system. A high number of sessions served by the server can be translated as low P2P and multicast usage. Fewer server sessions mean performance improvement by exploiting P2P collaborations and multicast communications, although, the system increases the control demand since the service is more dependent on the peers. Table 10 summarises the experimental settings.

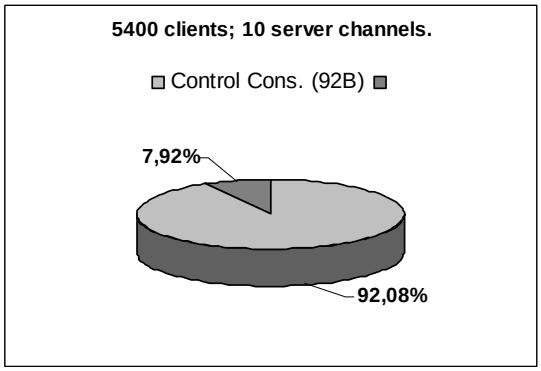
Parameter	Value
Number of clients	5400
Server video channels	10
Messages size	44B, 92B, 128B and 256B.
Service policy	PCM/MCDB and P2Cast.
Video length	90 minutes.
Video play rate	1500 kb/s.

Table 10. Experimental settings.

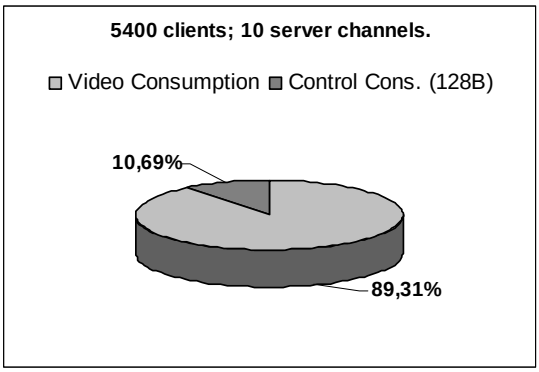
In figures 53 and 54 we can see the augmentation on the weight of the control load (Δ_w). Considering the range of message size from 44B to 256B, the Δ_w varies from almost 4% to 19% with the PCM/MCDB scheme and from 6% to 27% in P2Cast. These results point to the growing control importance; according to the experiment it is completely feasible that the control traffic represents more than 10% of the video traffic.



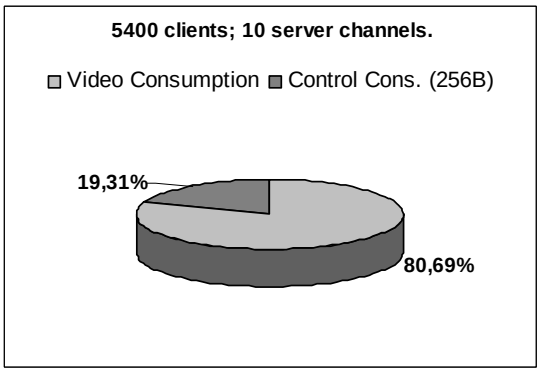
a) Control messages of 44 MB.



b) Control messages of 92 MB.

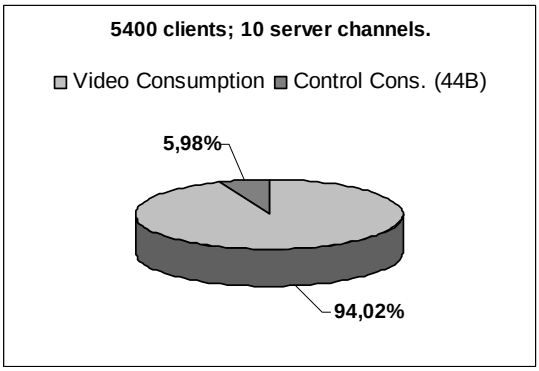


c) Control messages of 128 MB.

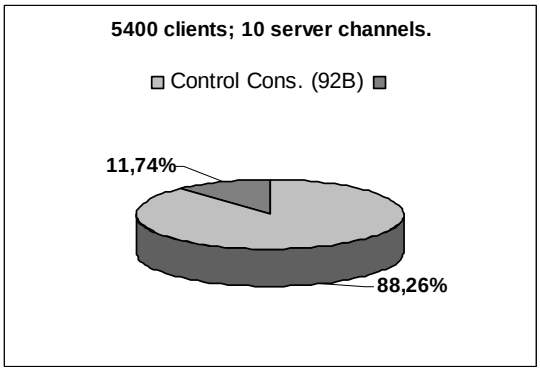


d) Control messages of 256 MB.

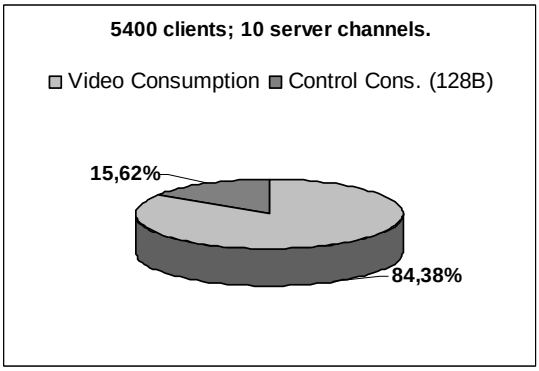
Figure 53. Control consumption in the PCM/MCDB Failure Management Process.



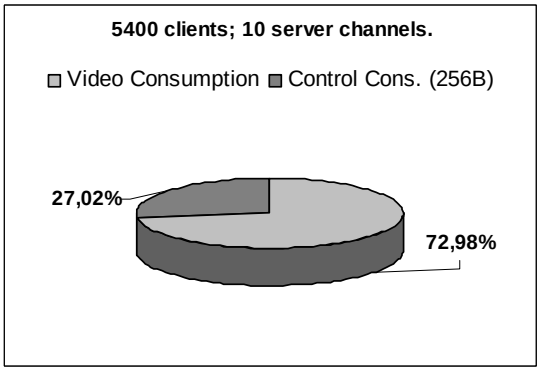
a) Control messages of 44 MB.



b) Control messages of 92 MB.



c) Control messages of 128 MB.



d) Control messages of 256 MB.

Figure 54. Control consumption in the P2Cast Failure Management Process.

Aiming to evaluate the importance of the control when the server and P2P interaction vary, in the next experiment we assess two different scenarios; the server is made responsible for fifteen and five video sessions. Analysing figure 55, we can observe the control load behaviour for PCM/MCDB and P2Cast when the system has 2700 and 5400 clients. When the number of server channels is reduced the P2P action increases, maintaining the service throughout the system. The results show that server resources consumption is lower when the number of channels decreases, as expected. The Load cost is practically the same when the system size is constant; the control, however, is much more representative when the service is more dependent on the peers. Also, the control traffic increases when the system becomes larger (5400 clients).

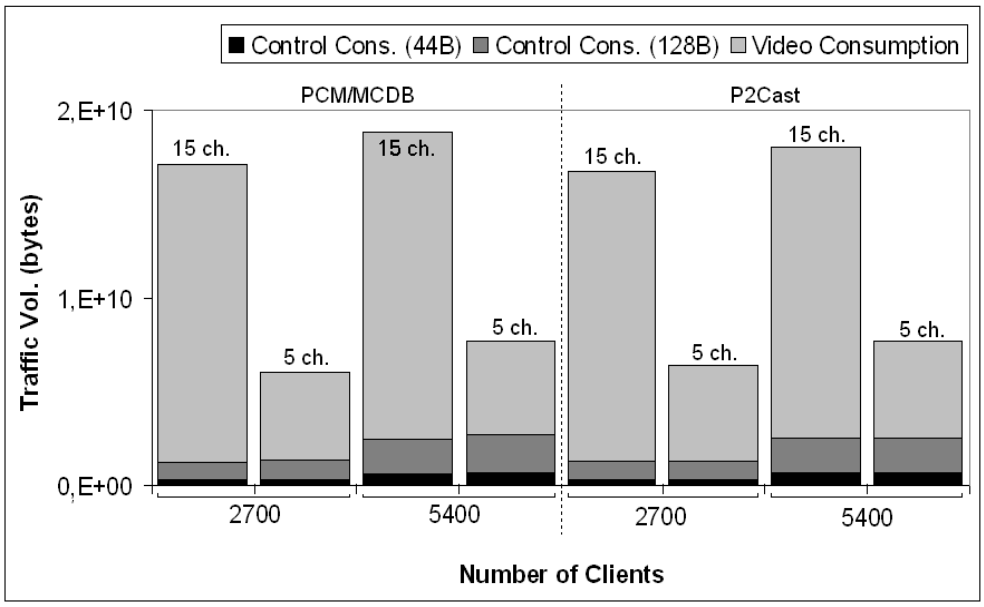


Figure 55. Control traffic x Video traffic for PCM/MCDB and P2Cast.

Table 11 presents the percentage of control traffic compared with the server video traffic (∆w). When we observe the system with 5400 clients, we see that the Δ_w can reach values between 13% (for 44B messages) and 39% (for 128B messages) for PCM/MCDB and 13% (for 44B messages) and 37% (for 128B messages) for P2Cast. The analysis confirms that the control load is more representative as P2P assumes more participation in the system. As fewer video channels are opened, the system saves server resources. However, service depends more on P2P collaborations, and thus increasing control relevance.

Service Scheme	Number of clients	Server video channels	Δw for messages of 44B	Δw for messages of 128B
PCM/MCDB	2700	15	2,09%	6,09%
		5	6,79%	19,75%
	5400	15	4,09%	11,91%
		5	13,29%	38,67%
P2Cast	2700	15	2,13%	6,18%
		5	6,39%	18,59%
	5400	15	4,24%	12,33%
		5	12,74%	37,07%

Table 11. Control traffic importance.

In the following we verify the control importance through simulated results presented by the P2Cast service scheme. P2Cast groups clients in ALM sessions according to their arrival time in the system. The parameter that defines the session size is the threshold (*T*). A small *T* restricts the number of clients in a session; P2P action is limited and the server needs to provide more communication channels to guarantee the service. While a larger *T* improves performance in the sense of saving server resources by exploiting P2P collaborations, the system becomes more control demanding, since the service relies on the clients. In this experiment, we consider fixed thresholds of 1%, 5% and 10% of the video size. Three different system sizes are observed: of the order of tens (79), hundreds (485) and thousands (1287) of clients.

By analysing figure 56, we can see that video consumption decreases when *T* increases; clients can be grouped into larger sessions, saving server resources. Nevertheless, control traffic increases when the system becomes larger. When we observe a number of clients in the order of tens, the control load has a low impact on the system, representing less than 1% when compared with video traffic. When an order of thousands of clients is observed, however, control can reach between 10% (for 44B messages) and 28% (for 128B messages) if we compare it with server data consumption. Table 12 summarises the simulated Δ_w presented in P2Cast analysis.

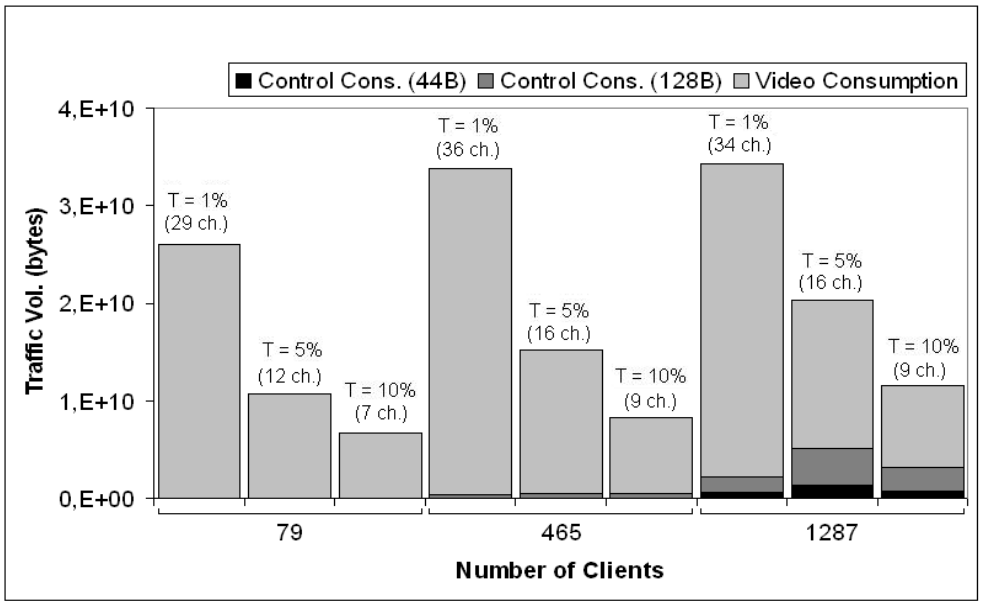


Figure 56. Control traffic x Video traffic: simulated results for P2Cast.

Number of clients	Session threshold (nº. of channels)	Δw for messages of 44B	Δw for messages of 128B
79	1% (29)	0,09%	0,25%
	5% (12)	0,21%	0,62%
	10% (7)	0,34%	0,99%
495	1% (36)	0,35%	1,00%
	5% (16)	0,91%	2,63%
	10% (9)	1,57%	4,58%
1287	1% (34)	1,80%	5,25%
	5% (16)	8,60%	25,02%
	10% (9)	9,75%	28,37%

Table 12. Control traffic importance: simulated results.

The results show that the volume of control load during a video session can reach considerable values. Large-scale systems can present an overhead of around 10% compared with the multimedia data generated by the server (in the best case, i.e. smaller message sizes); in the worst case (bigger message sizes) the overhead can consume the same amount of resources as 25% of the server video traffic (or more). The range of resource consumption makes clear the importance of the control scheme in P2P-VoD service on the Internet. Distributing the service through the peers improves system performance and scalability but, at the same time, demands a well-designed control mechanism to achieve the planned service with the desired level of QoS.

5.5 The Fault Tolerance Scheme Analysis

Previous topics show the importance of the control mechanism in the P2P-VoD service over the Internet. The Load and Time costs are two metrics which reflect the performance of the system on the clients' Failure Management Process. In this work we consider two service schemes in order to assess the Load and Time behaviours: PCM/MCDB and P2Cast. These service policies present different design characteristics such as multicast implementation level, P2P approach usage and fault tolerance scheme; thus, they were selected to serve as background for the evaluation. In order to improve the performance of the FMP, we apply the proposed FTS and analyse the behaviour of the Load and Time costs in both service schemes.

5.5.1 Load cost

The Load cost represents the dynamic of control messages necessary to implement the FMP; it strongly depends on the communication protocols adopted, i.e. type of message and send frequency. Basically, these parameters will define the amount of messages necessary to run the fault tolerance mechanism. The load imposed on the system to run the FMP represents an overhead; the volume of control information that flows through the system must be kept small in order to avoid heavy resource consumption by the control sub-system.

Aiming to evaluate the Load cost of the PCM/MCDB and P2Cast service policies as well as the proposed Fault Tolerance Scheme, we analyse five different aspects that govern the behaviour of this metric. We consider the number of clients in the system, the number of multicast groups, the size of multicast groups, the client failure frequency and the frequency of heartbeat messages. The behaviour of the FMP control load varies depending on the policy and the weight that each parameter has in the process. P2Cast generally is more expensive (if compared with PCM/MCDB) in terms of Load cost owing to the ALM implementation, which consumes more resources at the network layer.

Number of clients in the system: Scalability is one of the most important requirements in the Internet P2P-VoD service. The number of clients present in the system allows evaluation of the Load cost of the FMP as the system grows. Table 13 shows the settings assumed in this experiment.

Parameter	Value
Number of clients	[27 ; 5400]
Number of multicast groups	11
Heartbeat frequency	60 msg./min.
Failure frequency	1 fault each 5 minutes.
Peers available bandwidth[3]	1500 kb/s
Playback rate	1500 kb/s
Buffer capacity[3]	113MB (10 min.)
Video length	90 minutes.

Table 13. Experimental settings for the Load and Time cost evaluations.

Figure 57 shows that the Load cost grows when the number of clients in the system rises. The PCM/MCDB presents a lower cost compared with P2Cast owing to the different P2P and multicast implementations. PCM/MCDB takes advantage of the multicast capability at the network layer (IP Multicast) and relies less on P2P collaboration. The packet diffusion from a source to many destinations is in charge of the routers; the collaboration groups (P2P) are composed of a few end hosts in order to provide the bypass in the multicast channels. On the other hand, P2Cast is totally based on P2P to provide the ALM distribution tree. Thus, multiple unicast channels flow at the network layer in order to build the overlay topology. Moreover, P2Cast does not present a maintenance stage to keep updated status of the peers on the system; the recovery is based on subsequent queries for searching for a replacement peer: meanwhile, in PCM/MCDB the server periodically (f_{CI}) receives status messages from the peers; thus, in the recovery phase the server is able to inform directly about the substitute peer. These differences between the service schemes are reflected in distinctive volumes of heartbeats or topology maintenance messages.

[3] Peers resources applied exclusively in the FTS.

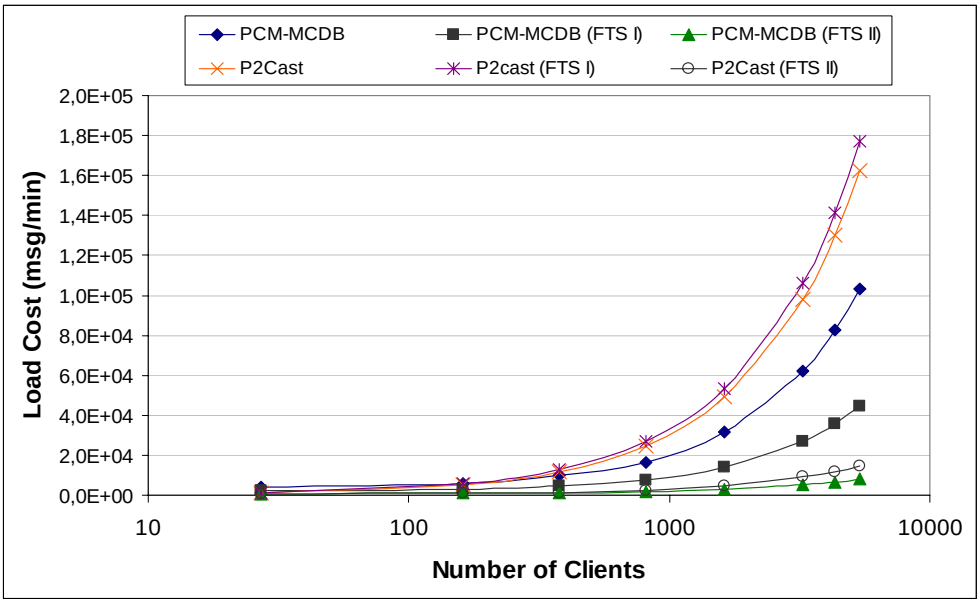


Figure 57. Load cost: evaluation of the number of clients in the system.

With the intention of organising the redundancy present on the system and reducing the costs of the FMP we propose the Fault Tolerance Scheme. The FTS is based on a set of peers caching distributed parts of the content while they are online. We apply the FTS to the PCM/MCDB and P2Cast service policies and observe the behaviour of the Load cost; in figure 57 the curves with the label (FTS-I) represent the Load cost for each service scheme when the FTS is applied. It is clear that the FTS can reduce PCM/MCDB Load cost; the elimination of the status messages by available peers in the maintenance stage is the strongest reason for the reduction. In P2Cast, the usage of FTS-I generally represents a low increment on the Load cost; the communication between Manager Nodes and server is the major cause of this increment.

The Load cost generated by the usage of the FTS in both schemes is compared with the cost without the proposed fault tolerance mechanism. The difference (increment or decrement) in the Load cost is showed in table 14. The PCM/MCDB presents a reduction of 52% when the FTS-I is applied and P2Cast a low increment of 8%; these values are the arithmetic mean of Δs for all the points along the curves.

	Average Load cost difference (Δ)
P2Cast x PCM/MCDB	28.3%
FTS I x PCM/MCDB	-52.2%
FTS II x PCM/MCDB	-87.7%
FTS I x P2Cast	8.2%
FTS II x P2Cast	-85.2%

Table 14. Load cost performance evaluation: number of clients.

In order to evaluate the Failure Management Process we change the detection strategy; now we asses the Fault Tolerance Scheme with the detection technique of client buffer monitoring. In this method, each client establishes a time interval to monitor the volume of information received. If the input rate is lower than a predefined threshold, the recovery must be triggered. This approach is referred to as FTS-II. The main consequence of this approach for the Load cost is the removal of heartbeat messages; unique elements monitored by heartbeats are the Manager Nodes of each FTG. In figure 57 it is possible to observe that the buffer monitoring strategy reduces the Load cost when applied to IP Multicast/mesh-based (PCM/MCDB) or ALM/tree-based (P2Cast) schemes. In the evaluations using the FTS-II approach, the Load cost decreases 88% for PCM/MCDB and 85% for P2Cast, on average. This detection scheme, however, is more prone to trigger recovery process erroneously (clients with poor resources can request recovery even when the source and network are running well) and is more expensive at the client side.

Number of multicast groups: The multicast paradigm is used on VoD service to improve its performance; clients who arrive close together and are interested in the same content are grouped to share system resources. In our study we consider the groups as ALM sessions or IP Multicast channels. In this experiment we aim to analyse the behaviour of the Load cost for a range of multicast groups on the system. The number of multicast groups is directly related to clients' behaviour. Few groups concentrating on many clients means popular content and a hot spot on requests; few groups with few end hosts indicates dispersal interest of clients in terms of time and catalogue; many groups of few clients indicates popular content with distributed request pattern and finally, many groups with many clients indicates high popular contents are being requested at any time; it can indicate that the system is reaching a saturation point.

We assume the number of clients constant and equal to 5400; the number of multicast groups varies from 40 to 700; hence the number of clients per group varies with the number of groups. A small number of multicast groups mean many clients at each group, while a great number of multicast channels implies on less clients per group. All the other parameters are the same as shown in table 13.

In figure 58 it is possible to see that the cost in P2Cast is constant. This behaviour occurs owing to the multicast implementation (ALM). In the developed models, the amount of ALM sessions opened does not affect the Load cost; the unicast connections at the network layer are considered the same for ten groups of 540 clients or 500 groups of eleven users, for example. The PCM/MCDB relies on IP Multicast, and thus the number of multicast sessions has great influence on the Load cost. A few multicast channels generate more messages flowing through the system because each group demands maintenance and reconstruction phases by the network routers.

The usage of the FTS-I and II reduce the Load cost when the number of multicast groups grows, for both service schemes; table 15 shows the comparisons of the Load costs.

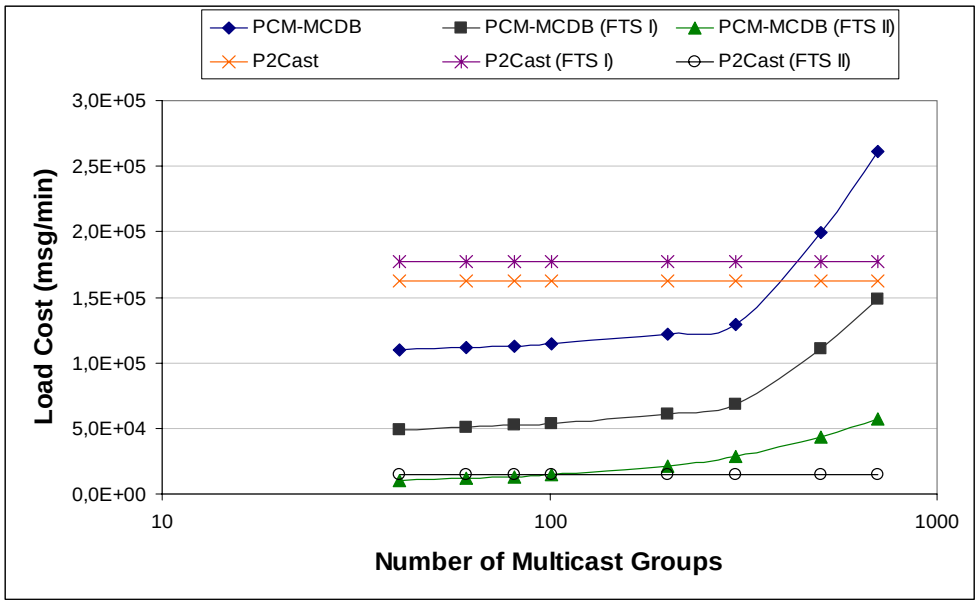


Figure 58. Load cost: evaluation of the number of multicast groups.

	Average Load cost difference (Δ)
P2Cast x PCM/MCDB	6.2%
FTS I x PCM/MCDB	-48.5%
FTS II x PCM/MCDB	-83.2%
FTS I x P2Cast	8.5%
FTS II x P2Cast	-85.2%

Table 15. Load cost performance evaluation: number of multicast groups.

Multicast group size: The size of a multicast group refers to the number of clients grouped at the same multicast session\channel. The multicast group size is the relation of the two previous analysed parameters, the number of clients and multicast groups. The number of multicast groups and their size indicate the content popularity as well as the arrival pattern of clients.

We evaluate this parameter under two heads: considering the number of clients constant, thus varying the multicast group size according to the number of channels and fixing the number of channels; hence, the number of clients per group is directly proportional to the number of clients in the system.

Figure 59 and table 16 refer to the analyses when the number of clients is constant (5400). We can observe that when the size of groups grows the Load cost decreases; this behaviour is in accordance with the pattern presented in figure 58 because when the group size increases the number of groups diminishes when the total number of clients is constant ($N_C = ct.$). In this case, small groups represent clients distributed through many multicast sessions, and thus the Load cost is greater for PCM/MCDB, which implements IP Multicast. In P2Cast, the number of members in an ALM session has theoretical influence on the number of exchanged messages; in this case, however, these messages represent small impact over the total volume and cannot be noticed through the curves. The application of the FTS largely reduces the Load cost under the variation of the multicast group size.

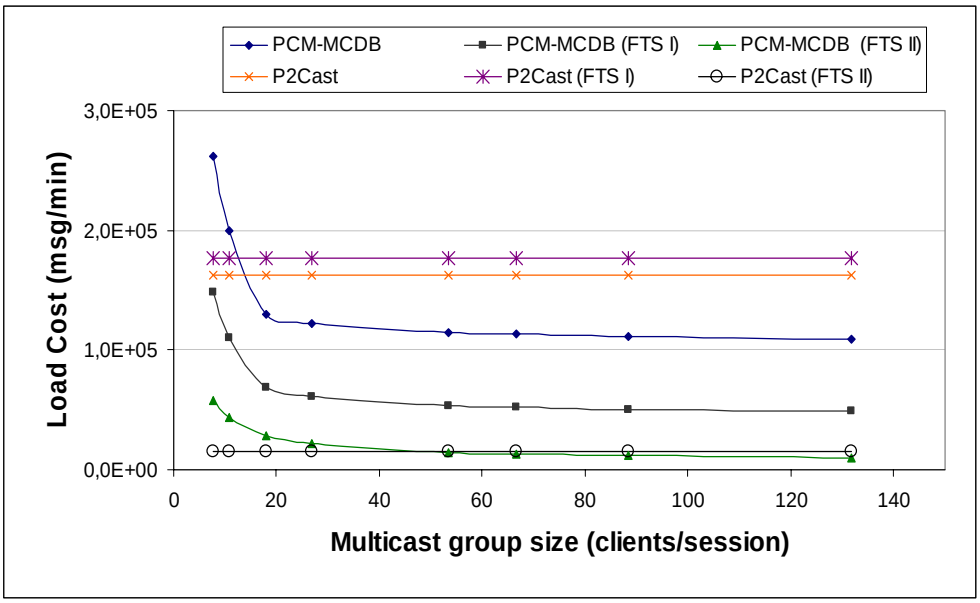


Figure 59. Load cost: evaluation of the multicast group size (N_C=*ct.*).

	Average Load cost difference (Δ)
P2Cast x PCM/MCDB	22.7%
FTS I x PCM/MCDB	-50.2%
FTS II x PCM/MCDB	-84.0%
FTS I x P2Cast	8.9%
FTS II x P2Cast	-90.9%

Table 16. Load cost performance evaluation: multicast group size (N_C=*ct.*).

The second scenario in this analysis considers the number of multicast channels constant and equal to 40 (G_M = *ct.*); therefore the group size is directly proportional to the number of clients in the system. It is possible to verify that the curves present the same tendency of the results showed in figure 57. In this graphic we can identify the relation of the multicast group size and the Load cost for the ALM approach; P2Cast presents lower cost than PCM/MCDB for small groups (few clients in this case), but when the number of clients per group grows the cost of the P2Cast surpasses PCM/MCDB. Table 17 summarises the average Load cost difference (Δ) for the curves of figure 60.

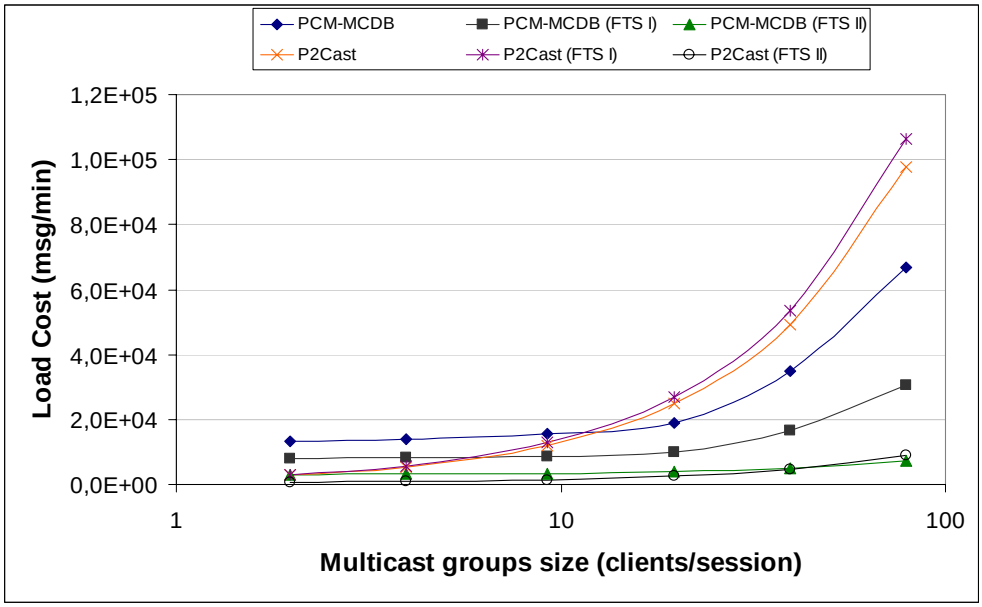


Figure 60. Load cost: evaluation of the multicast group size (G_M=*ct.*).

	Average Load cost difference (Δ)
P2Cast x PCM/MCDB	6.2%
FTS I x PCM/MCDB	-60.3%
FTS II x PCM/MCDB	-85.5%
FTS I x P2Cast	8.5%
FTS II x P2Cast	-87.5%

Table 17. Load cost performance evaluation: multicast group size(G_M=*ct.*).

Failure Frequency: The Load cost of the FMP strongly depends on the occurrence of faults. Peers arrive and depart from the system with certain frequency [109]; the recovery process is triggered when a peer leaves the system in order to find a substitute collaborator.

Figure 61 allows the evaluation of the Load cost according to the rate of peer failures. In PCM/MCDB, the cost grows when the failure frequency is high. The application of FTS I and II decrease the Load cost at 50% and 75% respectively (table 18); the behaviour is the same, however, since the IP Multicast demands control messages for the distribution tree every time the source changes. The P2Cast service scheme also presents an increment in the Load cost when the failure frequency rises; the search process for collaboration is responsible for this behaviour. When the FTS is used, however, the recovery process changes and the search for a new peer and the recursive process disappear (i.e. the removal

of "$1/p_e$" in the model). The Load cost presents a reduction when the failure frequency is high; the average Δ is -1% for FTS-I and -92% for FTS-II.

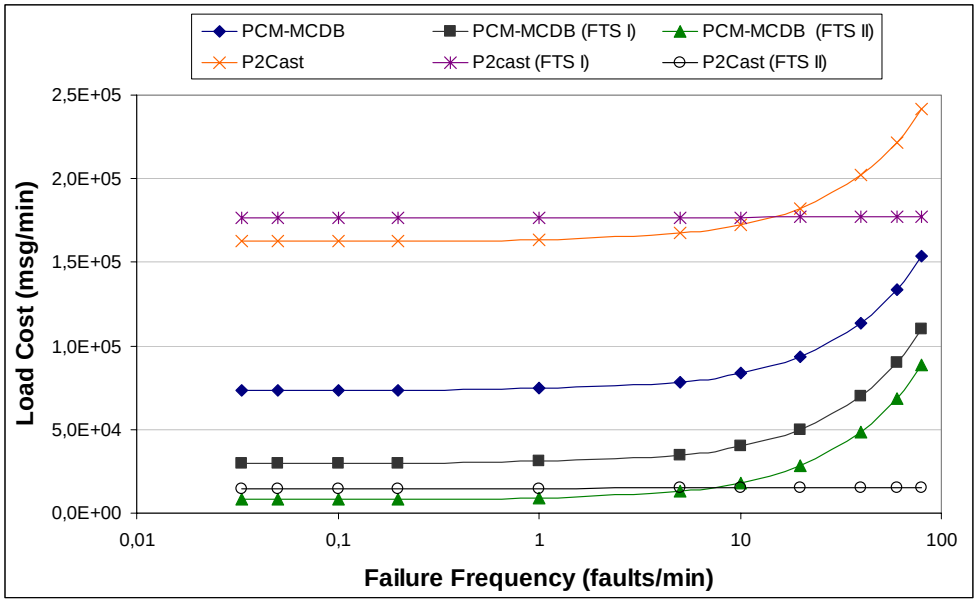


Figure 61. Load cost: evaluation of the failure frequency.

	Average Load cost difference (Δ)
P2Cast x PCM/MCDB	101.7%
FTS I x PCM/MCDB	-49.9%
FTS II x PCM/MCDB	-75.0%
FTS I x P2Cast	-0.9%
FTS II x P2Cast	-91.6%

Table 18. Load cost performance evaluation: failure frequency.

Heartbeat Frequency: Monitoring peers through heartbeat messages strongly affects the FMP; high heartbeat frequencies improve detection time, but may create much control overhead. A frequency of 0.2 messages per minute represents one message every five minutes, while ten messages per minute means one heartbeat every six seconds.

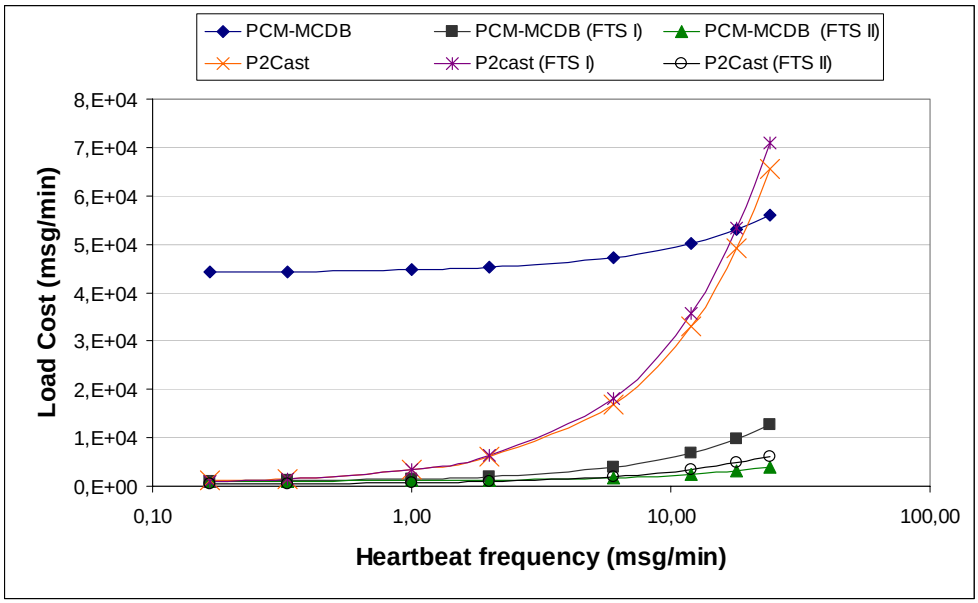


Figure 62. Load cost: evaluation of the heartbeat frequency.

	Average Load cost difference (Δ)
P2Cast x PCM/MCDB	-57.7%
FTS I x PCM/MCDB	-90.6%
FTS II x PCM/MCDB	-96.1%
FTS I x P2Cast	2.1%
FTS II x P2Cast	-80.4%

Table 19. Load cost performance evaluation: heartbeat frequency.

PCM/MCDB sends heartbeat messages for two neighbours while in P2Cast each collaborator sends one heartbeat to the server; P2P interaction, however, is much more representative in P2Cast (to build the ALM tree) than in PCM/MCDB, where few peers make up the collaboration groups. Hence, when the heartbeat frequency is high the Load cost of the P2Cast is heavier than that presented by PCM/MCDB, where the detection messages have less weight in the total cost. Figure 62 shows that the Load cost increases as the heartbeat interval is lower (high frequencies). Nevertheless, the FTS reduces the Load cost in the majority of the cases (table 19).

5.5.2 Time cost

The soft real-time feature of the VoD service makes the time a key piece of system evaluation; the time constraint defines the QoS level provided by the service. The Time cost metric of the FMP represents the time consumed in dealing with peer failures. Like the Load cost, this metric is closely related to the communication protocols. The protocols define the dynamic involved in detecting and recovering failures. The time cost allows the evaluation of the effectiveness of the Failure Management Process. A reliable FMP must be able to respect the time constraints.

In this topic we analyse three different aspects that have influence on the behaviour of the Time cost. In the analyses this metric is applied in the PCM/MCDB and P2Cast service policies and with regard to the proposed Fault Tolerance Scheme. We consider the network latency, its dimension (number of routers) and the frequency of heartbeat messages. The FTS does not have influence on network parameters, and thus the usage of the proposed Fault Tolerance Scheme does not change the Time cost behaviour in latency and network size evaluations; the FTS, however, removes the need for subsequent queries in the search for a new peer.

Network latency: The first parameter to be assessed is the network latency. Figure 63 shows the Time cost behaviour according to the latency variation from 200μs to 100ms [115]. Observing the results, we see that the latency is much more important for systems which rely on constant communication for finding a new peer, like P2cast. Network latency has less impact over time cost when communications to treat a client failure are limited. PCM/MCDB limits the number of messages exchanged during the recovery process, because the peers' status is already known through the maintenance phase.

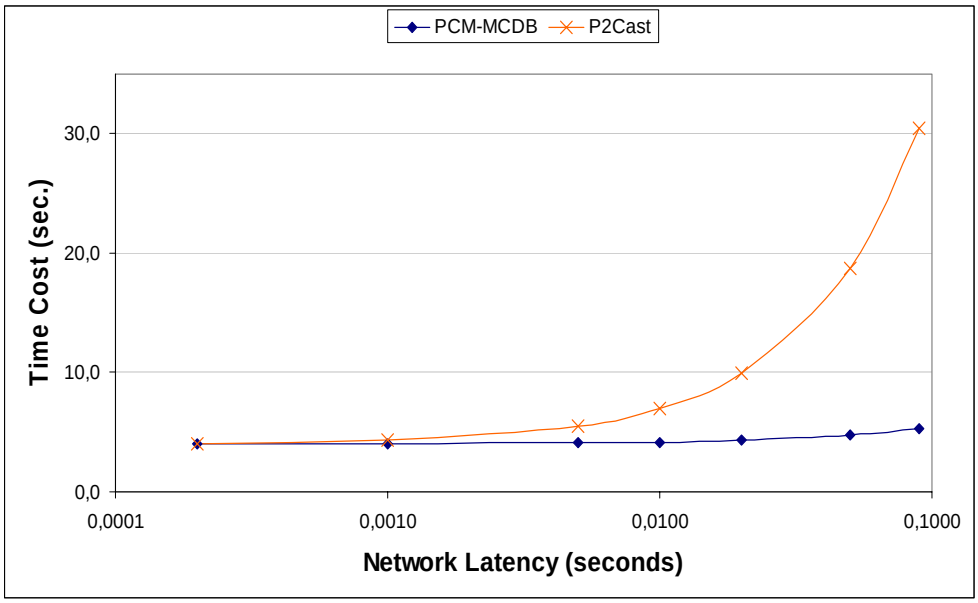


Figure 63. Time cost: evaluation of the network latency.

Network dimension: Following the same vein as the previous evaluation, the number of routers (network size) strongly affects the Time cost behaviour of FMPs that rely on much data exchange at real time, i.e. approaches which need to query information during the process of solving a failure. Figure 64 shows the Time cost of PCM/MCDB and P2Cast service policies; the network topology starts with three routers and reaches 57.

The benefit provided by the usage of the proposed Fault Tolerance Scheme considering network characteristics (latency and size) relies on the fact that the FTS limits the communication during the process of looking for a new peer. The location of the information is already known (the FTG); thus, the FTS brings the curve of P2Cast to the same behaviour as that presented by PCM/MCDB.

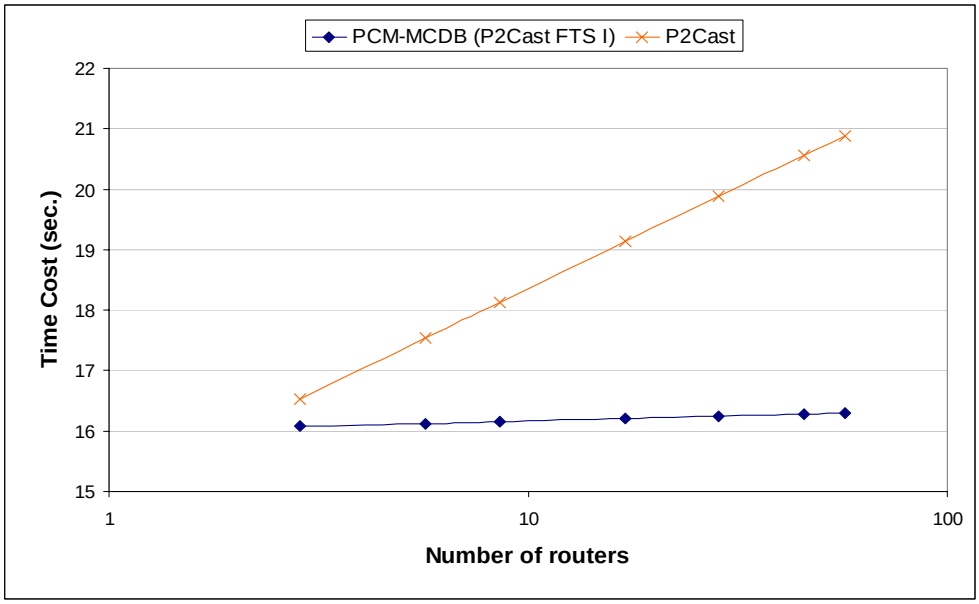


Figure 64. Time cost: evaluation of the network size.

Heartbeat frequency: The heartbeat frequency has great impact on the Time cost. In figure 65 we can observe that higher monitoring frequencies reduce the time cost. The difference between PCM/MCDB and P2Cast relies on the recursive recovery.

P2Cast depends on query messages to find a new collaborator; moreover, the search for recovery can be longer for ALM sessions with more clients inside (the process finishes only when the leaf client is reached). We can also notice that the cushion buffer (e.g. 60 seconds) may not be sufficient to avoid glitches because, in some cases, the detection and recovery process can consume more time than the cushion buffer can guarantee.

Figure 65 shows that the application of FTS-I eliminates the need for the recursive recovery on P2Cast. As previously mentioned, the back-up copy provided by the FTG guarantees the replacement of a failed peer and limits the amount of recovery messages. Thus, the Time cost for PCM/MCDB and P2Cast FTS-I becomes the same.

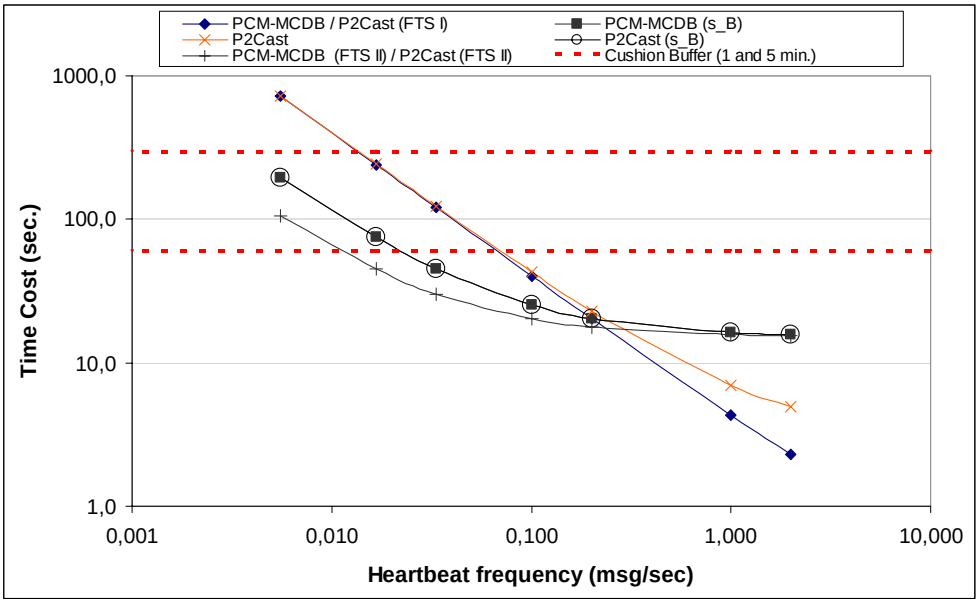


Figure 65. Time cost: evaluation of the heartbeat frequency.

Aiming to achieve a better time response on the FTS-I, we try one change in the Time cost model. Initially, the counter cycle had the same length of the heartbeat cycle ($1/f_{HB}$), and thus the detection scheme waits for W sequential non-received heartbeat messages before starting the recovery. Now we fix the wait counter W, making the period to trigger the recovery constant, as presented in equation (28).

$$C_t = \left(\frac{1}{f_{HB}} + W\right) + \left(\sigma \cdot l \cdot \frac{\ln(H)}{\ln(k)}\right) \qquad (28)$$

In figure 65 for higher f_{HB}, the new strategy (referred to by the label s_B) causes an increment on Time cost compared with the approach defined to PCM/MCDB and P2Cast (FTS-I). It occurs because W is responsible to limit the minimum value that the cost can reach. By applying the new mechanism, however, we see that the Time cost reduces when the heartbeat rate is small. A fixed W diminishes the range of the Time cost.

The buffer monitoring mechanism used as alternative for detection also changes the expression for the Time cost. The detection phase depends on the verification of the quality of the input stream by clients. Nevertheless, the network can present short time fluctuations; therefore, the client triggers a wait counter from the moment that it notices the input stream

is under the threshold before starting the recovery process. The check period (τ) can be shorter than $1/f_{HB}$ because it does not rely on load cost increment. Figure 65 shows at the curve tagged with FTS-II that the Time cost can be reduced by applying $\tau = 1/(2 \cdot f_{HB})$. This detection scheme relies on clients, however, and thus end hosts with low resources can pollute the system with requests for recovery even when the system works well; also, high monitoring frequencies can demand much processing power from the clients.

Parameter	Value
Playback rate	1500 kb/s
Cushion buffer	11MB (1 min.) and 56 MB (5 min.)
Video length	90 minutes.
Peers total buffer capacity	1GB
Peers upload bandwidth (ADSL2+)	1000 kb/s

Table 20. Experimental settings for the evaluation of the Fault Tolerance Scheme establishment time.

Considering the system characteristics like described in table 20, it is possible to show the benefits provided by a low Time cost. We assume the input rate at a client equal the playback rate in order to avoid buffer overflow or lack of data. In this case, two peers combining their resources are necessary to provide the video stream (750 kb/s + 750 kb/s). Considering a cushion buffer of five minutes of video (56MB) to guarantee QoS during network variations or peer failures, the time spent with the cushion construction is the same i.e. five minutes (input rate equal to video play rate); hence, the user will have a five-minute start-up delay before beginning to enjoy the content. Nevertheless, if the FTS is able to guarantee a Time cost lower than five minutes, the start-up delay can be reduced and the buffer usage is improved. The curve of the FTS-II in figure 65, shows that the scheme can guarantee a Time cost lower than 60 seconds for a great range of f_{HB}. Reducing the cushion buffer to one minute of video, the buffer capacity demanded is 11Mb and the start-up delay is one minute (for the system with the same conditions).

Results make clear the trade-off between load and time costs. A high detection rate provides lower response times in failure processes, but increases the Load cost. In the same way, the maintenance stage inserts a considerable amount of messages at the system, but guarantees few messages at the recovery phase, improving the response time. The proposed

Fault Tolerance Scheme reduces the number of messages and time response by organising and managing back-up copies in Fault Tolerance Groups.

5.6 The Fault Tolerance Scheme Service Performance

In this section we analyse the Fault Tolerance Scheme not from the Load and Time cost perspective, but considering its service performance. We evaluate the time necessary for the system to get the full distributed copy by using the natural redundancy present in the system and the number of peers necessary to guarantee the FTSP defined at service design.

The time consumed in the construction of the FTGs indicates the transient interval needed by the groups to reach a steady state. The number of peers necessary to handle a determined amount of simultaneous failures reflects the feasibility of the Fault Tolerance Scheme for such condition.

5.6.1 Formation time of a complete FTG.

The time consumed in the construction of the FTGs represents a transient phase. Once the transient interval is overcome the groups reach a steady state. In the steady state, the FTG is enabled to act on failures at any point in the content visualisation. This state can be perturbed by failures in the FTG members; the reconstruction of the full backup copy, however, consumes much less time than the transient stage.

One of the basic premises of the FTS is the use of the natural redundancy of the VoD service for building the FTGs. The portion of the video stored by a peer at the altruist buffer is obtained from a request made by this end host; a peer that requested a video v will store a parcel of this content proportional to its resources (buffer and bandwidth). Hence, the transient phase of the FTG construction depends on the time interval between the arrivals of FTG members, the order of arrival, the buffer capacity and the available bandwidth.

For the sake of simplicity, in this analysis we consider the buffer capacity of peers uniform and sufficient to store more than half of the multimedia file; as presented in Chapter 2, peers' memory resource is not the crucial constraint to Internet VoD service nowadays. Also, we assume the set of peers needed to compose an FTG available at the same time (the beginning of the FTG composition); this scenario reflects clients' behaviour for very popular

contents during service rush hours. When the members of a FTG present different arrival times, this interval increases the transient phase.

Expression 29 represents the time necessary to store an amount S (kb) of a multimedia file depending on the relation of the video play rate (V_{pr}) and the peer input bandwidth (bw). Figure 66 shows the transient and steady states for the construction of an FTG; three different bw availabilities are considered. Assuming a 90-minute video ($L = 5400$ *sec.)* and the input bw equal to the playback rate, the construction of the full distributed copy consumes the same time as the content visualisation; low input bandwidths make the transient phase larger. We consider as mandatory the resource aggregation to the VoD service over the Internet when peers have low bw for collaboration, so the input rate is always equal to or greater than the playback rate ($bw \geq V_{pr}$); considering all peers necessary to form an FTG available at the same time and sufficient buffer capacity, the transient interval must be equal to or lower than the exhibition length ($t \leq L$).

$$S = L \cdot V_{pr} \text{ and } t = \frac{L \cdot V_{pr}}{bw} \qquad (29)$$

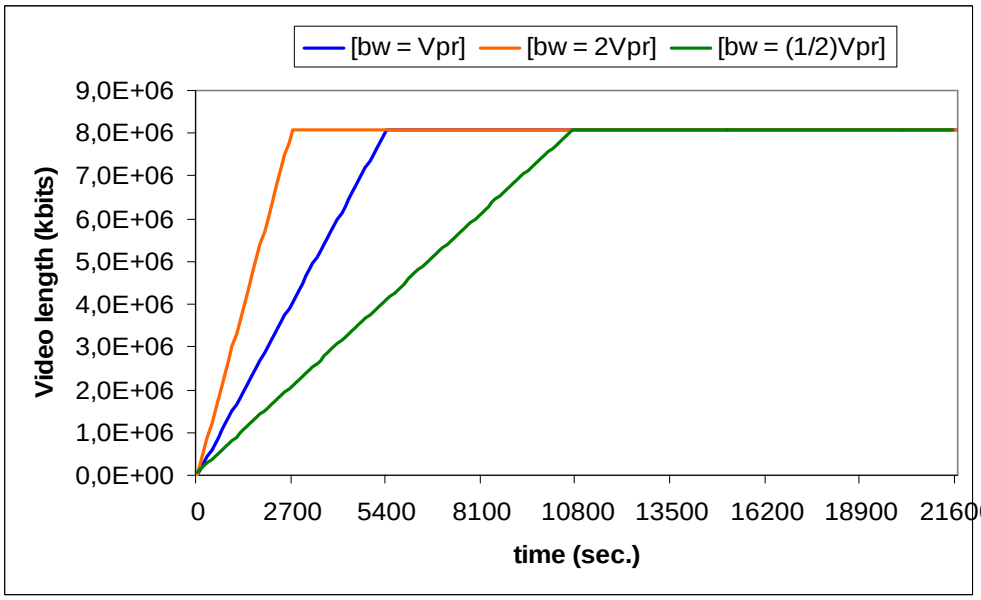


Figure 66. Formation time of Fault Tolerance Groups.

5.6.2 FTS service performance: simultaneous failures and the FTSP

The Fault Tolerance Service Performance (*FTSP*) is used to define the rate of replication of the FTGs; the FTSP is a design parameter that determines the number of simultaneous failures that the FTS can handle. The number of simultaneous failures supported by the scheme is not arbitrarily established: it takes into account the number of streams of a video *v* that rely on peers collaborations; therefore, the FTS follows the Pareto principle and allows tuning of the number of FTGs according to the P2P interaction.

The FTGs are made up of peers who make their resources available for the fault tolerance service. The number of peers necessary to handle a specific amount of simultaneous failures reflects the feasibility of the Fault Tolerance Scheme for such condition. In this topic we evaluate the FTS according to different FTSP levels and system conditions. Table 21 summarises the parameters used in the analyses.

Parameter	Value
Number of clients	10800
Video channels with P2P collaboration	1000 and 100.
Bandwidth range	300, 750, 1500, 3000 and 6000 kb/s.
Buffer capacity	338MB (30 min.) and 102MB (9 min.).
Video length	90 minutes.
Video play rate	1500 kb/s.

Table 21. Experimental settings for the Fault Tolerance Scheme service performance evaluation

In the first part of this experiment, we consider end hosts with large buffer capacity for the fault tolerance service (338 MB); therefore, three peers have enough storage resources to support the entire multimedia file (1GB). Clients' output bandwidth varies through the whole range, from 300kb/s to 6Mb/s. Moreover, the 10800 end hosts are dispersed throughout 1000 multicast sessions; this distribution results in small sessions with multicast groups of ten or eleven peers. This scenario can occur in long time evaluations of popular contents.

In this setting, the figure 67 shows the percentage of the total amount of clients necessary to guarantee different performance levels of the FTS. We can observe the collaboration demanded to provide backup copies for 10%, 20%, 50% and 100% of the video sessions. Results show that low bandwidth capacity relies on more peers collaborating to achieve the designed performance. High bandwidth availability drastically improves the

performance of the FTS by reducing the number of groups and peers; such situations, however, are not feasible in today's Internet environment, where an ADSL end host with an average upload rate of 6Mb/s is not yet viable. The FTSP of 100% in this case means that the FTS must be able to serve 1000 clients simultaneously and these clients are in the same range of visualisation (1-30, 31-60 or 61-90 minutes).

When peers have poor collaborative bandwidth, sustaining a copy for each video channel can be very expensive, demanding more than 45% of the total end hosts to compose the FTGs. This situation worsens if we observe figure 68; in this scenario clients have less buffer capacity (102 MB). Considering the same range of *FTSP* (10%, 20%, 50% and 100%) and available bandwidth (300kb/s to 6Mb/s), the most representative constraint is the storage space; now the low bandwidth is not the main restriction. The number of clients in the FTGs is defined by the buffer capacity; regardless of whether clients have 300 kb/s, 750 kb/s or 1500 kb/s, the number of clients necessary to build the group is the same (ten). When the FTSP is 100% the amount of peers necessary to reach such a performance level is more than 90% of the total number of clients in the system. This demonstrates that the FTSP can lead to an unfeasible scenario for the Fault Tolerance Scheme, depending on the clients' distribution (i.e., relation clients/group, number of groups and number of clients) and their available resources.

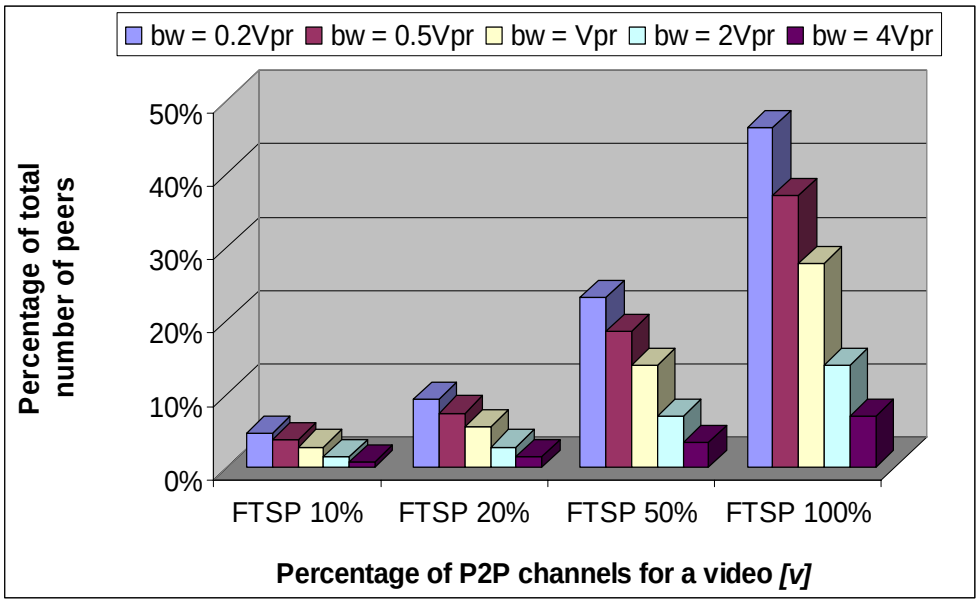


Figure 67. Fault Tolerance Scheme service performance evaluation with 1000 video sessions and peers with 338 MB of buffer capacity.

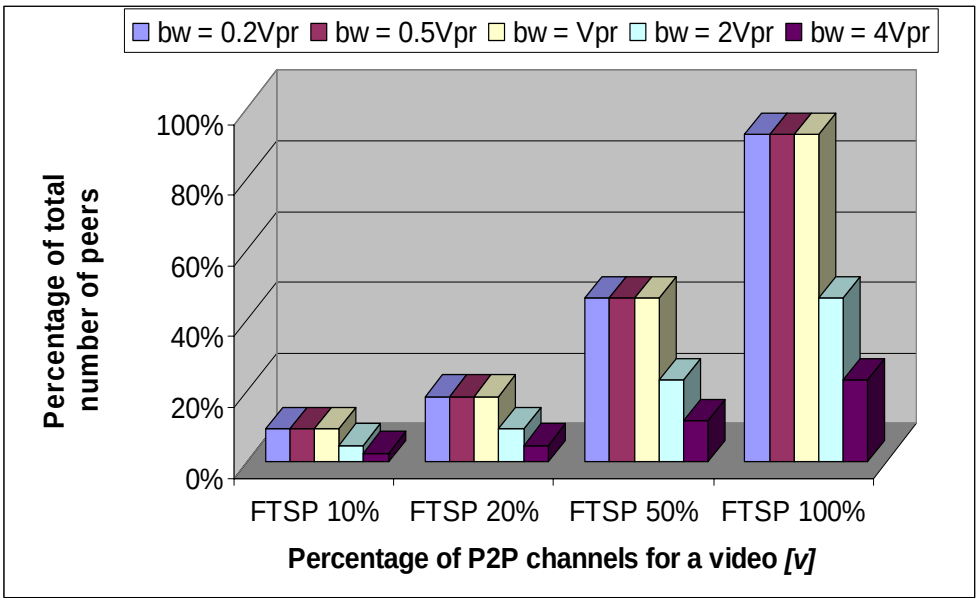


Figure 68. Fault Tolerance Scheme service performance evaluation with 1000 video sessions and peers with 102 MB of buffer capacity.

Figures 69 and 70 basically represent the same previous settings; in these cases, however, end hosts are distributed over 100 multicast groups, resulting in 108 clients per group on the average. This behaviour can be caused by hot spots of popular videos; thus clients can be condensed into a few large groups. In this case the FTSP of 100% is feasible because the number of channels is lower than in the previous case, i.e. now the FTS must handle simultaneously 100 failure processes at the same visualisation range (1-9, 10-18, 19-27, 28-36, …, 82-90 minutes). Despite the number of channels being lower, the total number of clients on the system is the same (10800); therefore, hot spot scenarios demand more copies of the same stream because groups are larger. Considering this, we assume the FTSP range of 100%, 200%, 400% and 600%. In the worst case (buffer capacity of nine minutes and 300kb/s of bandwidth), the FTS demands the collaboration of more than 55% of the end hosts.

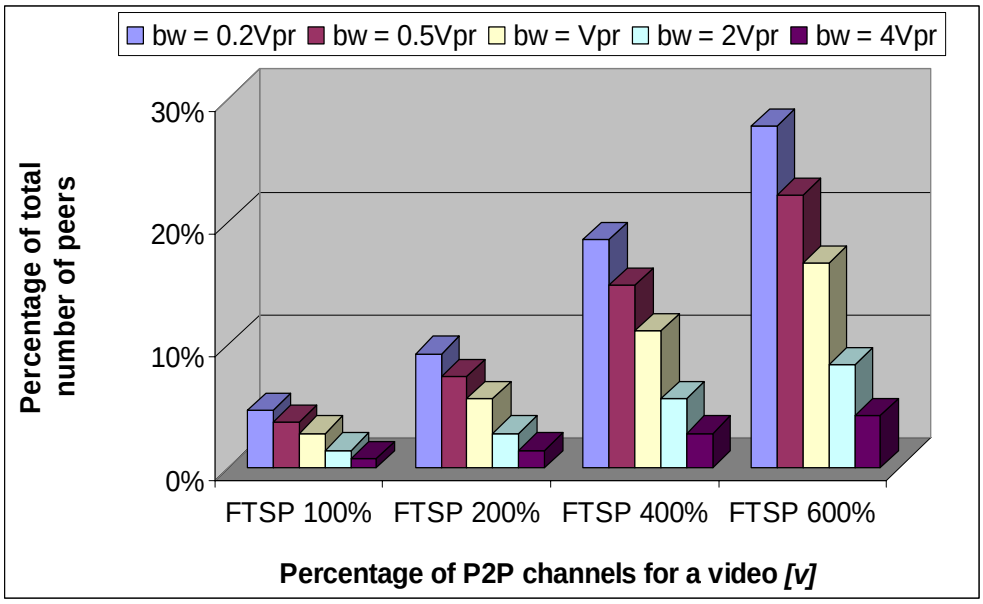


Figure 69. Fault Tolerance Scheme service performance evaluation with 100 video sessions and peers with 338 MB of buffer capacity.

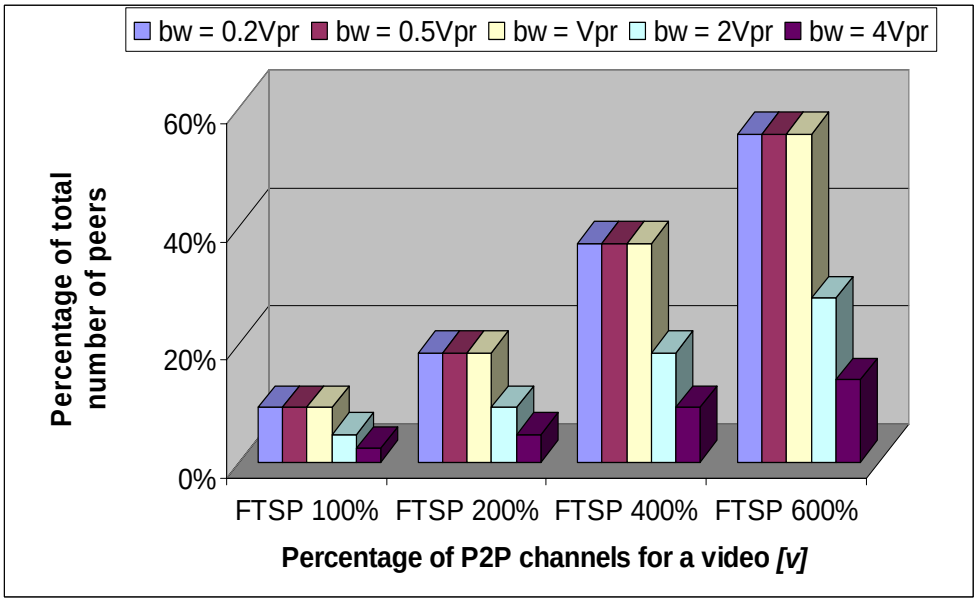


Figure 70. Fault Tolerance Scheme service performance evaluation with 100 video sessions and peers with 102 MB of buffer capacity.

These results show that the *FTSP* tuning is not a trivial task; it depends on the client/group relation, number of multicast groups, number of clients in the system and service scheme. Drastic situations are also very difficult to control; for example, if peers

massively fail at the same time, the service inevitably falls down and the scenario radically changes, including the FTS, so there is no sense in trying to handle an immense volume of simultaneous failures.

We assessed two scenarios: a few groups with many clients and a few clients spread out in many groups. Each case demands a different FTSP level to achieve the same results. Hot spots demand more copies of the same stream because groups are larger; on the other hand, clients distributed throughout many groups need less than one replica for each channel. The service scheme also has influence on the FTSP definition, since it determines the level of P2P collaboration that the system depends on. If IP Multicast is enabled, as in PCM/MCDB, the P2P interaction is not the core of the service. Information is replicated from the source to many destinations at the network level; a few peers are responsible for creating the bypass from the main multicast channel (multicast channel branching). In this case, a lower FTSP could give a satisfactory performance since one source is enabled to feed all the destinations in the same group. On the other hand, P2Cast is practically fully based on P2P communications (ALM distribution tree); thus, service depends much more on the end hosts and a high level of *FTSP* could be crucial to the service.

With regard to figures 67, 68, 69 and 70, one can see that a range from 10% to 20% of the peers is sufficient to handle 200 simultaneous faults in both scenarios, 1000 (FTSP = *20%*) and 100 (FTSP = *200%*) video channels. FTGs can handle failures simultaneously when orphans are visualising different points of the content. Tables 22 and 23 summarise the capacity of the FTS in dealing with simultaneous failures considering clients at distinct playback ranges. For example, groups of ten peers with 750kb/s can serve five faults simultaneously if each orphan is playing the content at a different point (minutes 1-9 or 10-18 or 19-27, and so on); hence the FTS is enabled to handle 1000 simultaneous failures (9% of system clients) demanding the collaboration of 18% of the peers. These results show that larger FTGs demand more peers, but at the same time increase the FTS performance; by distribution of the content throughout many peers reliability is added to the FTS.

N_{CFTG} \ bw (kb/s)	sf
5 \ 300	200
4 \ 750	400
3 \ 1500	600
3 \ 3000	1200
3 \ 6000	2400

Table 22. Simultaneous failures with peers' buffer of 30 minutes.

N_{CFTG} \ bw (kb/s)	sf
10 \ 300	400
10 \ 750	1000
10 \ 1500	2000
10 \ 3000	4000
10 \ 6000	8000

Table 23. Simultaneous failures with peers' buffer of 9 minutes.

Finally, peers' buffer capacity or processing power is not the main restrictions on today's Internet VoD service. Bandwidth and playback rate (video quality) are the most important players for designing the system. Moreover, clients' behaviour strongly affects the service performance of the FTS and the VoD system as a whole, principally when P2P and multicast are used as the core strategies to increase system performance and scalability.

Chapter 6

Conclusions

6.1 Conclusions and Main Contributions

The research on Video on Demand (VoD) systems is of keen importance and still presents many challenges and open issues; the great goal in designing such systems must be achieving a feasible large-scale and high-quality service with lower costs and fewer deployment constraints.

A large-scale Video on Demand service (LVoD) makes reference to systems which are able to support a great number of concurrent requests generated by geographically distributed clients; such clients can be spread out widely in areas such as metropolitan regions, countries or even a continent. They are also enabled to choose a multimedia content from a diversified catalogue.

The Internet is the most popular environment of connected users; it is a publicly accessible environment of interconnected computer networks accessible worldwide. The Internet is a 'network of networks' made up of domestic, academic, business and government entities. Hence, the Internet has become the most important environment to deploy large-scale video service. VoD represents an attractive commercial service to be offered in a public and global scale environment like the Internet.

In the last few years, multimedia streaming technology has shown incredible growth on the Internet. The applications are various, such as news, education, training, entertainment or advertising. Currently, video streaming on the Internet basically relies on real-time transmission of live events (e.g. Justin.tv) and streaming of on-demand contents (e.g. YouTube). Nevertheless, multimedia services on the Internet still present strong restrictions of quality and availability; a user commonly deals with long start-up delays, glitches, frozen-frames or even the removal of some content for reason of copyright infringement.

The Internet is a non-dedicated environment with a high degree of dynamicity and heterogeneity. Meanwhile, the VoD service presents hard constraints such as storage and bandwidth requirements as well as time restrictions in order to guarantee Quality of Service

(QoS) to the clients. Therefore, owing to such features and constraints, combining a VoD service with the Internet environment is not a trivial task.

As VoD service over the Internet has become more popular, the researches have intensified. One of the main targets of a LVoD system deployed over the Internet is to provide multimedia content with QoS; owing, however, to system dimension many issues play important roles in the design, such as the heterogeneity and scalability aspects.

The Peer-to-Peer (P2P) and multicast paradigms appeared in the multimedia research environment as promising solutions to improve system scalability and performance; both approaches achieve their aims by making use of shared resources. P2P is based on the free cooperation of equals in view of the performance of a common task; it takes advantage of available resources at the end host side (storage, content, bandwidth, power processing, etc.). Multicast is a communication strategy where a sender has the capability to transmit information that can be received concurrently by a group of interested destinations. The mechanism is based on delivering the information over each link of a topology only once, creating copies when the links to the multiple destinations split.

P2P and multicast paradigms, however, place new demands on the design of Internet VoD services. Peers are free and thus can arrive or depart from the system at any time; besides, end hosts present different availability of resources. Meanwhile, multicast needs topology reconfiguration when a peer departs (owing to change of source (IP Multicast) or peer replacement on the distribution structure (ALM)). Such characteristics demand a strict control mechanism on the system to guarantee the designed service performance. In this work we focused on peer failure, since these faults can be one of the most recurrent causes of errors on Internet VoD systems based on P2P and multicast approaches.

Fault Tolerance mechanisms to maintain the service running even when peers depart still shows poor of investigation; efforts must be orientated to guarantee good user experience during the entire session (i.e. avoiding errors). Current VoD researches intend to improve system performance and have the focus more orientated to the application level logic for data transmission. The control complexity added to the system by P2P and multicast strategies is strongly neglected in VoD designs. The soft real-time nature of the service imposes an important constraint on Fault Tolerance: the failure treatment has a deadline to guarantee error absence and consequently maintain the QoS. The control sub-system must be in charge of managing peer failures through processes of detection and recovery. These processes depend on communication messages that implement the control logic; the extra load created by the control messages (overhead) can affect the whole system performance. A good

control mechanism is very important because it provides coordination and synchronisation among system elements and guarantees the reliability and accuracy of the service. Nevertheless, the control scheme demands a careful design owing to the soft real-time restriction and the overhead imposed on the system.

In this work we systematise peers' Failure Management Process (FMP). The FMP represents the dynamic of the control mechanism when dealing with peer failures and therefore we defined three components which describe the FMP: detection, recovery and maintenance. Using these phases as background, we establish Load and Time costs as metrics to evaluate the performance of the FMP. The Load cost represents the control overhead imposed on the system in managing failures (volume of control messages) while the Time cost metric is meant to express the time consumption of an FMP.

Using these tools (FMP, Load and Time costs) we assessed PCM/MCDB and P2Cast VoD service schemes. These service mechanisms were selected because they represent different aspects of VoD systems design. The PCM/MCDB is the delivery mechanism of the Pn2Pn VoD architecture. This architecture is designed for large-scale VoD service and considers distributed servers, P2P collaborations and multicast communications. The service scheme implements collaboration groups of peers organised in a mesh-based topology; also, it assumes local networks enabled with IP Multicast capability. On the other hand, the P2Cast is an architecture proposed to provide scalability on VoD service through Application Layer Multicast strategy; the service policy uses the P2P approach to stream multimedia data cooperatively while only relying on unicast connections among peers. In P2Cast each client acts as a data source while it receives the video. This strategy creates an ALM tree-based architecture, which can be applied at the Internet AS level. Moreover, both service schemes adopt the patching multicast strategy to group clients and serve petitions. PCM/MCDB and P2Cast are also enabled to use heartbeats and income-stream monitoring as detection techniques; each of them applies distinct recovery and maintenance strategies.

Results showed that the control overhead during a video session can reach considerable values. Large-scale systems present control overheads that can consume the same amount of resources as an important parcel of the server data traffic (10% to 25%). The ranges of resource consumption make clear the importance of the control scheme in P2P-VoD service on the Internet.

In a general way, it was possible to observe that the Load cost grows when the system parameters rise. The PCM/MCDB service scheme presents a lower cost compared with P2Cast. This behaviour occurs because of the different P2P and multicast implementations

adopted by each one. PCM/MCDB takes advantage of the multicast capability at the network layer (IP Multicast) and depends on P2P collaborations less than P2Cast: in this scheme the packet diffusion from a source to many recipients is the responsibility of the routers; the P2P collaboration groups are made up by a few peers in order to provide the bypass on the multicast channels. The service mechanism used by P2Cast is based on an ALM distribution tree and totally relies on P2P connections; hence, multiple unicast paths are used at the network layer in order to build the overlay topology. In addition, P2Cast does not have a maintenance phase to update the status of the online peers; the recovery is based on subsequent queries for seeking a substitute collaborator; meanwhile, in PCM/MCDB the server periodically receives status messages from the peers; thus, in the recovery phase the server is able to inform directly about the replacement peer. These differences between the service schemes are responsible for the distinctive Load cost behaviours. Results also indicate that the Time cost behaviour is strongly affected by data exchange during the FMP, i.e. approaches which need to query information at real time in the process of solving a failure demand more time to finish the task. The heartbeat frequency also has great impact on the Time cost; high monitoring frequencies diminish the total time consumed in handling peer failures.

These results make clear the trade-off between Load and Time costs. A high detection rate provides lower response times in failure processes but increments the Load cost. In the same vein, the maintenance stage inserts a considerable amount of messages at the system, but guarantees few messages at the recovery phase, improving the response time. The distribution of the service using P2P and multicast techniques improves system performance and scalability; nevertheless, at the same time, it demands a well-designed control mechanism to achieve the planned service with the desired level of QoS.

Impelled by these observations, we propose a mechanism to provide fault tolerance to peer disconnections in a robust and reliable way; the proposed scheme tries to guarantee the QoS with low Load and Time costs to the system. Our proposal is referred to as the *Fault Tolerance Scheme (FTS)* and is based on efficient handling of the asynchronous redundancy present in the system. We use the time redundancy to create data replication; the replication is applied in order to improve reliability and accessibility of the fault tolerance service. A fault tolerance service can be achieved by aggregating a dedicated infrastructure of backup elements; such elements would be able to provide hardware and information redundancy. This solution founders, however, on the service deployment and maintenance extra costs.

The aim of our Fault Tolerance Scheme is to build a backup system in a distributed way based on peers' capabilities. The FTS is designed to organise a small set of peers to store statically portions of the multimedia files; it assumes that clients have a portion of buffer reserved to collaborate on fault tolerance services and this buffer portion is called the 'altruist' buffer. Peers are designated to cache a well-defined video window of a requested content while they are online.

The selected set of clients that form the distributed backup is based on the intention of peers to reserve buffer space and upload bandwidth to fault tolerance matters; peers with more buffer and bandwidth availability have priority to integrate the fault tolerance scheme. Hence, the selected peers compose a Fault Tolerance Group (FTG); the FTG maintains a full copy of a specific content and is able to deal with simultaneous peer failures. The distributed backup system takes advantage of peers' resources to provide reliability to the VoD service; also, the strategy guarantees content legality issues, since servers controlled by the service provider host the video catalogue and peers hold information only on their volatile memory (when the end host turns off, the data vanish).

We applied the FTS to the PCM/MCDB and P2Cast service schemes in order to assess the Load and Time costs of the proposed mechanism. The application of the Fault Tolerance Scheme can reduce PCM/MCDB Load cost by 60%, on average. The elimination of the status messages by available peers is the strongest reason for the reduction. In P2Cast, the usage of the FTS generally represents a low increment on the Load cost, which is 4% on average. In addition, the detection strategy of clients monitoring the input-stream quality can decrease the Load cost by 86% for PCM/MCDB and P2Cast, on average. Nevertheless, this detection scheme is more prone to trigger recovery process erroneously (clients with poor resources can request recovery even when the source and network are running well) and is more expensive at the client side.

As regards the time aspect, the benefit provided by the usage of the proposed Fault Tolerance Scheme relies on the fact that it limits the communication during the search process for a new peer. The location of the information is already known, an FTG member; thus, subsequent queries are not necessary. We also observed that the cushion buffer may not be sufficient to avoid errors because, in some cases, the detection and recovery process can consume more time than the cushion buffer can guarantee. If the FTS is enabled to limit the Time cost, the start-up delay on user visualisation can be reduced and the buffer usage can be improved. The proposed Fault Tolerance Scheme reduces the number of messages and time response by organising and managing distributed back-up copies of the contents.

Finally, we evaluated the performance of the FTS. The construction of the FTGs is related to the time interval between member's arrivals, the order of arrival, the buffer capacity and the available bandwidth. The time consumed in the construction of the FTGs represents a transient phase; once the transient interval is overcome the group reaches a steady state where it is enabled to act on failures at any playback point. We consider as mandatory the resource aggregation to the VoD service over the Internet, since generally peers have low upload bandwidth availability; therefore, the input rate will be always equal to or greater than the playback rate. The transient interval then will be equal to or lower than the video exhibition length, always the set of peers necessary to build a FTG is available at the beginning of the process and the buffer capacity does not represent a hard constraint.

We also observed that a range from 10% to 20% of the peers connected to the system is sufficient to handle 200 simultaneous faults at the same visualisation range (hardest restriction). The FTGs can handle failures simultaneously, however, when orphans are visualising different points of the multimedia file; in such a situation the service performance of the FTS is improved. Results show that larger FTGs demand more peers, but at the same time increase the FTS performance by being able to handle more simultaneous faults; the content split on various portions and distributed throughout many peers improves the reliability of the FTS. Tuning the FTS service performance is not a trivial task; it depends on the client/group relation, number of multicast groups, number of clients in the system and service scheme. Moreover, drastic situations are very difficult to control; if peers massively fail the service inevitably falls down and the scenario radically changes, including the FTS.

In conclusion, peers' buffer capacity and processing power are not the main restrictions for Internet VoD service deployment nowadays. Bandwidth and playback rate (video quality) are the most important players in the system design. Moreover, clients' behaviour has a crucial role in the service; it strongly affects the performance of the FTS and the VoD system as a whole, principally when P2P and multicast are used as the core strategies to increase system performance and scalability.

The present work resulted in a series of technical publications where the developed research could be assessed by the scientific community.

The published articles **I**, **II**, **III** and **IV** treat the following subjects: the Failure Management Process; the analytical models for Load cost applied in a P2P mesh-based topology and IP Multicast environment (i.e. PCM/MCDB service scheme); experimental results using networks arranged on tree topology and transit-stub model (GT-ITM) and the verification of the control relevance on an FMP.

I. Rodrigo Godoi, Xiaoyuan Y. Xu and Porfidio Hernández. "*Fault-Tolerance Evaluation in a PCM-MCDB Peer-to-Peer Multicast Scheme*". II Spanish Congress of Informatics. Zaragoza, Spain, September 2007. Vol. I, pp. 335-342, ISBN: 978-84-9732-593-6.

II. Rodrigo Godoi, Xiaoyuan Y. Xu, Porfidio Hernández and Emilio Luque. "*Control Evaluation in a LVoD System Based on a Peer-to-Peer Multicast Scheme*". XIII Argentinean Congress of Computer Science, CACIC. Corrientes y Resistencia, Argentina, October 2007. pp. 1192-1203, ISBN 978-950-656-109-3.

III. Rodrigo Godoi, Xiaoyuan Y. Xu and Porfidio Hernández. "*Analytical Evaluation of Clients' Failures in a LVoD Architecture Based on P2P and Multicast Paradigms*". 14th International European Conference on Parallel and Distributed Computing, EuroPar. Las Palmas de Gran Canaria, Spain, August 2008. Vol. 5168, pp. 856-865, ISBN: 978-3-540-85450-0.

IV. Rodrigo Godoi, Xiaoyuan Y. Xu, Porfidio Hernández and Emilio Luque. "*Control Evaluation in a LVoD System Based on a Peer-to-Peer Multicast Scheme*". Journal of Computer Science & Technology. Vol. 8, No. 2, July 2008 - ISSN 1666-6038.

The articles **V**, **VI** make reference to: the analytical models for Load cost applied in a P2P tree-based topology with data distribution through an ALM mechanism and unicast channels (i.e., P2Cast service scheme); the analytical model for the Time cost of an FMP; experimental results using a network arranged on the transit-stub model (GT-ITM); the validation of the Load and Time costs as well as the FMP through simulation applying the

VoDSim tool; the verification of the control importance in P2P-VoD services and the verification of the relation between the time consumed in handling failures and the QoS.

V. Rodrigo Godoi, Alvaro Chalar and Porfidio Hernández. "*Assessing Control Impact Over a P2P-VoD System*". International Conference on Parallel and Distributed Processing Techniques and Applications, PDPTA. Las Vegas, USA, July 2009. (**accepted as regular research paper**).

VI. Rodrigo Godoi, Alvaro Chalar and Porfidio Hernández. "*Cost Models for Failure Management on a Peer to Peer VoD System*". IEEE International Conference on Image Processing, ICIP. Cairo, Egypt, November 2009. (**under evaluation by the conference committee).**

Lastly, the article **VII** includes the following themes: the analytical models for Load cost applied in PCM/MCDB and P2Cast service schemes; the analytical models for the Time cost; the proposed Fault Tolerance Scheme; experimental results using a network arranged on the transit-stub model (GT-ITM); the verification of the control importance in P2P-VoD services; the verification of the relation between the time consumed in handling failures and the QoS as well as the verification of the benefits provided by the application of the proposed FTS.

VII. Rodrigo Godoi, Alvaro Chalar and Porfidio Hernández. "*A Robust and Reliable Fault Tolerance Scheme for P2P-VoD Systems*". 17th Annual Meeting of the IEEE/ACM International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems, MASCOTS. London, UK, September 2009. (**under evaluation by the conference committee**).

6.2 Future Work

In the development of this work we have indentified several topics that can be better exploited in order to expand knowledge of designing VoD systems and improve presented solutions.

Expanded evaluations: The proposed Fault Tolerance Scheme was applied in two VoD service mechanisms, the PCM/MCDB and P2Cast. These service policies were selected owing to their characteristics in the Multicast implementation (i.e. IP Multicast, ALM, patching), P2P approach (mesh and tree topologies) and fault tolerance strategy (detection, recovery and maintenance approaches); PCM/MCDB and P2Cast comprise many crucial features necessary for the system evaluation. Nevertheless, an interesting work would be the expansion of the analysed scenario, i.e. the application and assessment of the FTS in a wide range of VoD architectures and service policies; results considering, for example, a scenario of multiple servers acting cooperatively, inter-AS communications or resource aggregation service policies for peers with poor bandwidth, can be very useful to the deployment of P2P-VoD services over the Internet.

Implementation of the FTS: The Failure Management Process of peers was evaluated according to metrics of Load and Time costs. The analytical models developed to describe such costs were validated through simulated results provided by the VoDSim tool; the verification corroborates the usage of Load and Time expressions to analyse the FMP of the PCM/MCDB and P2Cast service policies. The proposed FTS is applied in both service schemes and results are assessed by mean of the analytical model. However, the behaviour of the FTS was not evaluated in the simulation environment; the aggregation of the FTS in the VoDSim can provide significative results for tuning the performance of the scheme and to check its behaviour in a more dynamic scenario.

FTS approach: The FTS assumes peers reserving some upload bandwidth and buffer capacity to collaborate in fault tolerance; such peers cache portions of visualized contents to form a distributed back-up copy. Few modifications in the approach may generate performance improvement; interesting changes to test can be peers storing parts of non-visualized contents or even using non-volatile storage devices like SSDs (Solid State Disk drives) [116].

Network protocols accuracy: In this work we assume two different communication strategies between source and destine: Unicast and IP Multicast. We have analysed some protocols that implement such mechanisms (OSPF, PIM-SM, IGMP etc.) and identified the messages used by Unicast and IP Multicast protocols in the FMP. However, analytical and simulation models abstract the protocols implementation. A fine measurement of the control load generated by IP Multicast and ALM (Unicast) implementations in P2P-VoD service can provide more accurate results and help to improve the models.

VCR and DVD-like operations: The interactivity of the user with the content through Video Cassette Recorder (VCR) commands (ability to Pause, Play, Fast-Forward and Rewind) is a recurrent issue in video systems. Such concept tries to emulate the user perception when handling the content stored in a magnetic tape, i.e. the image is continuously displayed during the execution of any VCR operation. An approach commonly used to provide interactivity in the digital environment is to adopt reference points to provide Fast-Forward and Rewind commands; the visualisation jumps to a reference point when these commands are executed (this strategy is the same adopted in DVD systems and is referred as DVD-like commands). A DVD-like operation can be faced as a failure by the VoD system since the source may not be enabled to provide the content in a large visualisation window. More research is necessary to tune of the FTS for dealing with DVD-like operations and observe if the FTS scheme can help in handling such commands.

Users' behaviour: The P2P-VoD approach makes use of clients resource to improve system performance. The system shifts service responsibility to the client side, therefore clients play an important role in the service performance. One of the aims of this work is to treat some issues created by the usage of the P2P paradigm; however, more research in clients' behaviour is indispensable. Incentive policies for collaboration, for example, can increment resource availability; the records of the lifetime of a peer can help to designate the most suitable end host to act as Manager Node in the FTS. By knowing the patterns of connections, interests or availability, current protocols can be improved and new ones can be designed on a more efficient manner.

Bibliography

[1] Amazon VoD. Available at: http://www.amazon.com/gp/video/ontv/start. Accessed on May 2009.

[2] Telefonica Imagenio. Manual de usuario, versión 2.3, Noviembre 2008. Available for download at: http://www.telefonica.es. Accessed on May 2009.

[3] ONO Video Club. Available at: http://www.ono.es/television/tv-videoclub-ojo.aspx. Accessed on May 2009.

[4] Broad & TV. Available at: http://www.broadntv.com/. Accessed on May 2009.

[5] R. Bleidt, G. Bulycz, F. Aghdasi and M. Bourges-Sevenier. "*Understanding MPEG-4: Technologies, Advantages and Markets*". MPEG Industry Forum White Paper, 2005.

[6] T. Wiegand, G. J. Sullivan, G. Bjøntegaard and A. Luthra. "*Overview of the H.264/AVC Video Coding Standard*". IEEE Transactions on Circuits and Systems for Video Technology. Vol. 13, nº 7, pp. 560-576, 2003.

[7] YouTube. Available at: http://www.youtube.com/. Accessed on May 2009.

[8] Apple iTunes. Available at: http://www.apple.com/itunes/. Accessed on May 2009.

[9] F. Cores, A. Ripoll, X.Y. Yang, B. Qazzaz, R. Suppi, P. Hernández, and E. Luque. "*Exploiting Traffic Balancing and Multicast Efficiency in Distributed Video on Demand Architectures*". International European Conference on Parallel and Distributed Computing, EuroPar. LNCS 2790, pp. 859-869, 2003.

[10] Jalote P., *Fault tolerance in Distributed Systems*. Prentice Hall, New Jersey, 1998.

[11] A. Dan, D. Sitaram and P. Shahabuddin. "*Scheduling Policies for an On-Demand Video Server with Batching*". Proc. of 2nd ACM Int. Multimedia Conference. San Francisco, CA, pp. 15-23, 1994.

[12] K. A. Hua, Y. Cai and S. Sheu. "*Patching: A multicast technique for true video-on-demand services*". Proceedings of ACM Multimedia, pp. 191-200, 1998.

[13] J. Lu. "*An Architecture for Delivering Broadband Video over the Internet*". Proceedings of the International Conference on Information Technology: Coding and Computing. Washington, USA, p. 542, 2002.

[14] A. Biliris, C. Cranor, F. Douglis, M. Rabinovich, S. Sibal, O. Spastcheck and W. Sturm. "*CDN brokering*". In Proceedings of 6th International Workshop on Web Caching and Content Distribution. pp.393-402, 2001.

[15] F. Cores. "*Arquitecturas Distribuidas para Sistemas de Video-bajo-Demanda a gran escala*". PhD thesis, Universitat Autònoma de Barcelona, 2003.

[16] Y. Zhao and C.-C.J. Kuo. "*Scheduling design for distributed video-on-demand servers"*. IEEE International Symposium on Circuits and Systems. Vol. 2, pp.1545-1548, May 2005.

[17] Y. Zhao and C.-C.J. Kuo. "*Video server scheduling using random early request migration*". Multimedia Systems. Vol. 10, pp. 302-316, 2005.

[18] S-H. G. Chan and F. Tobagi. "*Distributed Servers Architecture for Networked Video Services*". IEEE/ACM Transactions on Networking. Vol. 9, n. 2, pp. 125-136, April 2001.

[19] A.Vlavianos, M. Iliofotou and M. Faloutsos. "*BiToS: Enhancing BitTorrent for Supporting Streaming Applications*". Proceedings of 25th IEEE International Conference on Computer Communications. Barcelona, pp. 1-6, April 2006.

[20] W. T. Leung and J. Y. B. Lee. "*A Server-less Architecture for Building Scalable, Reliable, and Cost-Effective Video-on-demand Systems*". Internet2 Workshop on Collaborative Computing in Higher Education: Peer-to-Peer and Beyond. USA, pp. 30-31, January 2002.

[21] P. Rodriguez, S. Tan and C. Gkantsidis. "*On the feasibility of commercial, legal P2P content distribution*". ACM SIGCOMM Computer Communication. Review. Vol. 36, n. 1, pp.75-78, January 2006.

[22] Peer-to-Peer article on Wikipedia. Available at: http://en.wikipedia.org/wiki/Peer-to-peer. Accessed on May 2009.

[23] GNU General Public License. Available at: http://www.gnu.org/copyleft/gpl.html. Accessed on May 2009.

[24] Creative Commons. Available at: http://creativecommons.org/. Accessed on May 2009.

[25] C. Shirky. "What Is P2P…And What Isn't?". Article available at: http://openp2p.com/. Accessed on May 2009.

[26] Napster. Available at: http://free.napster.com/. Accessed on May 2009.

[27] J. Li. "*On peer-to-peer (P2P) content delivery*". Peer-to-Peer Network Applications. No.1, pp. 45-63, 2008.

[28] B. Cohen. "*Incentives Build Robustness in BitTorrent*". 2003. Available for download at: http://www.bittorrent.org/.

[29] FastTrack. Available at: http://gift-fasttrack.berlios.de/. Accessed on May 2009.

[30] S. Saroiu, K. Gummadi and S. D. Gribble. "*A Measurement Study of Peer-to-Peer File Sharing Systems*". Multimedia Computing and Networking, 2002.

[31] A. Dan and D. Sitaram. "*Buffer Management Policy for an On-Demand Video Server*". RC 19347, IBM Research Division, 1993.

[32] E. Rescorla. "*Introduction to Distributed Hash Tables*". In Proceedings of the 65th Internet Engineering task force. Dallas, TX, USA, March 2006.

[33] B. Cheng, X. Liu, Z. Zhang, H. Jin, L. Stein and X. Liao. "*Evaluation and optimization of a peer-to-peer video-on-demand system*". Journal of Systems Architecture. Vol. 54, n. 7, pp. 651-663, 2008.

[34] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma and S. Lim. "*A Survey and Comparison of Peer-to-Peer Overlay Network Schemes*". IEEE Communications Surveys & Tutorials. Vol. 7, n. 2, pp. 72-93, second quarter 2005.

[35] J. Peltotalo, J. Harju, A. Jantunen, M. Saukko, L. Vaatamoinen, I. Curcio, I. Bouazizi and M. Hannuksela. "*Peer-to-Peer Streaming Technology Survey*". Seventh International Conference on Networking Cancun. pp. 342-350, 2008.

[36] L. Yong, Y. Guo and C. Liang, "*A survey on peer-to-peer video streaming systems*". Peer-to-Peer Networking and Applications. Vol. 1, pp. 18-28, 2008.

[37] LimeWire. Available at: http://beta.limewire.com/. Accessed on May 2009.

[38] Bearshare. Available at: http://www.bearshare.com/. Accessed on May 2009.

[39] KaZaA. Available at: http://www.kazaa.com/. Accessed on May 2009.

[40] I. Stoica, R. Morris, D. Kerger, M. F. Kaashoek and H. Balakrishnan. "*Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications*". In proceedings of SIGCOMM'01. San Diego, California, USA, 2001.

[41] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Schenker. "*A Scalable Content Addressable Network*". In proceedings of ACM SIGCOMM. pp. 161–72, 2001.

[42] P. Maymounkov and D. Mazieres. "*Kademlia: A Peer-to-Peer Information System Based on the XOR Metric*". In proceedings of IPTPS. Cambridge, MA, USA, pp. 53–65, 2002.

[43] M. Hosseini, D. T. Ahmed, S. Shirmohammadi and N. D. Georganas. "*A Survey of Application-Layer Multicast Protocols*". IEEE Communications Surveys & Tutorials. Vol. 9, n. 3, pp. 58-74, 2007.

[44] S.-W. Tan, G. Waters and J. Crawford. "*A performance comparison of self-organizing application layer multicast overlay construction techniques*". In Computer Communications. Vol. 29, n. 12, pp. 2332-2347, 2006.

[45] S. Sheu, K. A. Hua, and W. Tavanapong. "*Chaining: A Generalized Batching Technique for Video-On-Demand Systems*". In proceedings of IEEE Int. Conference On Multimedia Computing and Systems. Ottawa, Ontario, Canada, pp. 110-117, 1997.

[46] M. G. W. Jones, S. A. Sorensen and S. Wilbur. "*Protocol Design for Large Group Multicasting: The Message Distribution Protocol*". Computer Communications. pp. 287-297, 1991.

[47] A. Paul, K. K. Sabnani, J. C. H. Lin and S. Bhattacharyya. "*Reliable Multicast Transport Protocol (RMTP)*". In IEEE Journal on selected areas in communications, special issue for multipoint communications. pp. 407-421, April 1997.

[48] J. Liebeherr and B. S. Sethi. "*A Scalable Control Topology for Multicast Communications*". In Proceedings of IEEE INFOCOMM '98. San Francisco, CA, pp. 1197-204, March 1998

[49] K. L. Cheng, K. W. Cheuk and S. H. Gary Chan. "*Implementation and Performance Measurement of an Island Multicast Protocol*". In proceedings of IEEE ICC'05. Vol. 2, pp. 1299-1303, 2005.

[50] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee and S. Khuller. "*Construction of an efficient overlay multicast infrastructure for real-time applications*". In Proceedings of INFOCOM, San Francisco, CA, April 2003.

[51] S. Banerjee, B. Bhattacharjee and C. Kommareddy. "*Scalable application layer multicast*". In proceedings of ACM SIGCOMM. Pittsburgh, PA, pp. 205-217, 2002.

[52] D. Pendarakis, S. Shi, D. Verma and M. Waldvogel. "*ALMI: An Application Level Multicast Infrastructure*". In Proceedings of the 3rd USNIX Symposium on Internet Technologies and Systems. San Francisco, CA, USA, March 2001.

[53] D.A. Tran, K.A. Hua and T.T. Do. "*ZIGZAG: An efficient peer-to-peer scheme for media streaming*". In proceedings of IEEE INFOCOM. San Francisco, USA, 2003.

[54] Internet Group Management Protocol (IGMP). The Internet Engineering Task Force, RFC 2236. Available at: http://www.ietf.org/rfc.html. Accessed on May 2009.

[55] Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification. The Internet Engineering Task Force, RFC 2362. Available at: http://www.ietf.org/rfc.html. Accessed on May 2009.

[56] Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised). The Internet Engineering Task Force, RFC 3973. Available at: http://www.ietf.org/rfc.html. Accessed on May 2009.

[57] Core Based Trees (CBT version 2) Multicast Routing: Protocol Specification. The Internet Engineering Task Force, RFC 2189. Available at: http://www.ietf.org/rfc.html. Accessed on May 2009.

[58] Distance Vector Multicast Routing Protocol (DVMRP). The Internet Engineering Task Force, RFC 1075. Available at: http://www.ietf.org/rfc.html. Accessed on May 2009.

[59] Internet article from Wikipedia. Available at: http://en.wikipedia.org/wiki/Internet. Accessed on May 2009.

[60] P2P TV article from Wikipedia. Available at: http://en.wikipedia.org/wiki/P2p_tv. Accessed on May 2009.

[61] PPStream. Available at: http://www.ppstream.com/. Accessed on May 2009.

[62] PPLive. Available at: http://www.pplive.com. Accessed on May 2009.

[63] X. Hei, C. Liang, J. Liang, Y. Liu and K. W. Ross, "*Insights into PPLive: A Measurement Study of a Large-Scale P2P IPTV System*". In Proceedings of IPTV Workshop, International World Wide Web Conference. Edinburgh, Scotland, 2006.

[64] Justin.tv. Available at: http://www.justin.tv/. Accessed on May 2009.

[65] B. Cheng, L. Stein, H. Jin and Z. Zheng, "*Towards cinematic internet video-on-demand*". Operating Systems Reviews, EuroSys. New York, USA, vol. 42, n. 4, pp. 109-122, 2008.

[66] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn and S. Moon, "*I tube, you tube, everybody tubes: analyzing the world's largest user generated content video system*". Proceedings of the 7th ACM SIGCOMM conference on Internet measurement. San Diego, USA, pp. 1-14, 2007.

[67] Joost. Available at: http://www.joost.com/. Accessed on May 2009.

[68] MegaVideo. Available at: http://megavideo.com/. Accessed on May 2009.

[69] OVGuide. Available at: http://www.ovguide.com/. Accessed on May 2009.

[70] SurfTheChannel. Available at: http://www.surfthechannel.com/. Accessed on May 2009.

[71] DosPuntoCeroVision. Available at: http://www.dospuntocerovision.com/. Accessed on May 2009.

[72] Y. Guo, K. Suh, J. Kurose and D. Towsley. "*A peer-to-peer on-demand streaming service and its performance evaluation*". In proceedings of International Conference on Multimedia and Expo. 2003.

[73] V. Padmanabhan, H. Wang, P. Chou, and K. Sripanidkulchai. "*Distributing streaming media content using cooperative networking*". In proceedings of NOSSDAV. Miami Beach, USA, 2002.

[74] X. Yang, P. Hernández, F. Cores, A. Ripoll, R. Suppi, E. Luque. "*Dynamic Distributed Collaborative Merging Policy to Optimize the Multicasting Delivery Scheme*". International European Conference on Parallel and Distributed Computing, EuroPar. Vol. 3648, pp. 879-889, 2005.

[75] X. Yang, P. Hernández, L. Souza, A. Ripoll, R. Suppi, E. Luque and F. Cores. "*Multi-Collaboration Domain Multicast P2P Delivery Architecture for VoD System*". IEEE International Conference on Communications, ICC. Vol. 4, pp. 1633 - 1640, 2006.

[76] X. Yang. "*Un Sistema de Vídeo-bajo-Demanda a gran escala basado en la Arquitectura P2P con Comunicaciones por Multidifusión*". PhD thesis, Universitat Autònoma de Barcelona, 2006.

[77] L. Souza, F. Cores, J. Balladini and A. Ripoll. "*Improving VoD P2P delivery efficiency over Internet using idle peers*". In Proceedings of Signal Processing and Multimedia Applications Conference, SIGMAP. Barcelona, Spain, July 2007.

[78] L. Souza, F. Cores, X. Yang and A. Ripoll. "*DynaPeer: A Dynamic Peer-to-Peer based Delivery Scheme for VoD System*". International European Conference on Parallel and Distributed Computing, EuroPar. Vol. 4641, pp. 769-781, 2007.

[79] L. S. C. Souza. "*DynaPeer: A dynamic Peer-to-Peer VoD System over Internet*". PhD thesis, Universitat Autònoma de Barcelona, 2007.

[80] T. T. Do, K. A. Hua and M. A. Tantaoui. "*P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment*". IEEE International Conference on Communications. Vol. 3, 2004.

[81] Y. Guo, K. Suh, J. Kurose, D. Towsley. "*P2Cast: peer-to-peer patching for video on demand service*". Multimedia Tools and Applications. Vol. 33, pp. 109-129, 2007.

[82] M. Hefeeda, A. Habib, B. Botev, D. Xu, and D. B. Bhargava. "*PROMISE: Peer-to-peer media streaming using collectcast*". In proceedings of ACM Multimedia. Berkeley, CA, pp. 45-54, November 2003.

[83] M. Hefeeda, A. Habib, D. Xu, B. Bhargava and B. Botev. "*CollectCast: A peer-to-peer service for media streaming*". Multimedia Systems. Vol. 11, n. 1, pp 68-81. 2005.

[84] M. Rabbat, R. Nowak and M. Coates. "*Multiple Source*, *Multiple Destination Network Tomography*". In proceedings of IEEE INFOCOM, 2004.

[85] L. B. Pinho, E. Ishikawa and C. L. Amorim. "*GloVE: A Distributed Environment for Scalable Video-on-Demand Systems*". The International Journal of High Performance Computing Applications. Vol. 17, n. 2, pp. 147-167, 2003.

[86] E. Ishikawa and C. Amorim. "*Cooperative Video Caching for Interactive and Scalable VoD Systems*". In proceedings of the First International Conference on Networking. London, UK, pp. 776-785, 2001.

[87] H. Ericksson. "*MBONE: The Multicast Backbone*". ACM Communications. Vol. 37, n. 8, pp. 54–60, 1994.

[88] K. A. Skevik, V. Goebel and T. Plagemann. "*Evaluation of a comprehensive P2P video-on-demand streaming system*". Computer Networks, vol. 53, n. 4, pp. 434-455, 2009.

[89] H. Weatherspoon and J. D. Kubiatowicz. "*Erasure Coding vs. Replication: A Quantitative Comparison*". International Workshop on Peer-to-Peer Systems. Cambridge, MA, USA, 2002.

[90] G. Carle and E.W. Biersack. "*Survey of error recovery techniques for IP-based audio-visual multicast applications*". In Network IEEE. Sophia Antipolis, France, Vol. 11, n. 6, pp. 24-36, 1997.

[91] OSPF for IPv6. The Internet Engineering Task Force, RFC 2740. Available at: http://www.ietf.org/rfc.html. Accessed on May 2009.

[92] Transmission Control Protocol. The Internet Engineering Task Force, RFC 793. Available at: http://www.ietf.org/rfc.html. Accessed on May 2009.

[93] User Datagram Protocol. The Internet Engineering Task Force, RFC 768. Available at: http://www.ietf.org/rfc.html. Accessed on May 2009.

[94] A. A. Duarte. "*RADIC: A Powerful Fault-Tolerant Architecture*". PhD thesis, Universitat Autònoma de Barcelona, 2008.

[95] V. Pareto, A. S. Schwier and A. N. Page . "*Manual of Political Economy*". A.M. Kelley, 1971.

[96] J. A. Balladini. "*Un Sistema de Vídeo-bajo-Demanda a gran escala Tolerante a Fallos de Rede*". PhD thesis, Universitat Autònoma de Barcelona, 2008.

[97] T. W. J. Ngan, D. S. Wallach and P. Druschel. "*Incentives-Compatible Peer-to-Peer Multicast*". In The Second Workshop on the Economics of Peer-to-Peer Systems, July 2004.

[98] R. Albert and A-L. Barabási. "*Statistical mechanics of complex networks*". Reviews of modern physics. Vol. 74, pp.47-97, 2002.

[99] I. Djonova-Popova. "*Open Shortest Path First – OSPF*". 6th CEENet Workshop. Budapest, 2000.

[100] P. Y. Ho, V. T. Sam and J. Y. B. Lee. "*Providing Probabilistic Performance Guarantees in Multi-Source Video Streaming over the Internet*". IPTV Workshop, International World Wide Web Conference. Edinburgh, Scotland, United Kingdom, May 2006.

[101] P. Eykhoff. "*System Identification: Parameter and State Estimation*". Wiley & Sons, 1974.

[102] T.Sikora. "*MPEG-1 and MPEG-2 Digital Video Coding Standards*". Digital Consumer Electronics Handbook - McGraw-Hill Book Company, 1997.

[103] J. W. Daniel. "*Poisson processes (and mixture distributions)*". Austin Actuarial Seminars, 2008.

[104] G. Zipf. "*Human Behaviour and the Principle of Least Effort*". Addison-Wesley, Cambridge, Massachussets, 1949.

[105] R. Godoi. "*Análisis de la Gestión de Fallos en Sistemas de Video bajo Demanda a gran escala*". Master thesis, Universitat Autònoma de Barcelona, 2007.

[106] L. S. C. Souza. "*Extending VoDSim: A simulation tool for Video-on-Demand Systems*". Master thesis, Universitat Autònoma de Barcelona, 2006.

[107] L. Souza, F. Cores, A. Ripoll, X.Y. Yang and E. Luque. "*On the Relevance of Network Topologies in Distributed Video-on-Demand Servers*". In Proceeding of 14th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, PDP. France, pp. 396-404, 2006.

[108] W. Weibull. "*A Statistical Distribution Function of Wide Applicability*" Journal of Applied Mechanics, ASME. pp. 293-297, September 1951.

[109] T. Silverston and O. Fourmaux. "*Measuring P2P IPTV Systems*". In NOSSDAV '07. Urbana, Illinois, USA, 2007.

[110] E.W. Zegura, K.L. Calvert and S. Bhattacharjee. "*How to Model an Internetwork*". In proceedings of IEEE INFOCOM 96. Vol. 2, pp. 594-602, 1996.

[111] X. Wang, C. Yu, H. Schulzrinne, P. Stirpe and W. Wu. "*IP Multicast Fault Recovery in PIM over OSPF*". In Proceedings of International Conference on Network Protocols, 2000.

[112] D. Li, J. Wu, K. Xu, Y. Cui, Y. Liu and X Zhang. "*Performance Analysis of Multicast Routing Protocol PIM-SM*". IEEE AICT/SAPIR/ELETE'05. pp. 157-162, 2005.

[113] T. Billhartz, J. B. Cain, E. Farrey-Goudreau, D. Fieg and S. Batsell. "*Performance and Resource Cost Comparisons for the CBT and PIM Multicast Routing Protocol*s". In IEEE Journal on Selected Areas in Communications. pp. 304-315,1997.

[114] Tarik C., S. Gjessing and O. Kure. "*Tree Recovery in PIM Sparse Mode*". In Telecommunication Systems. pp. 443–460, 2002.

[115] The International Computer Science Institute Center of Internet Research. Available at: http://www.icir.org/models/linkmodel.html. Accessed on May 2009.

[116] Increase Application Performance with Solid State Disks. White Paper. Texas Memory Systems, February 2008.