UNIVERSITÉ DE VERSAILLES SAINT-QUENTIN-EN-YVELINES

U.F.R DE SCIENCES

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE VERSAILLES

SAINT-QUENTIN-EN-YVELINES

Discipline **: INFORMATIQUE**

Présentée et soutenue publiquement par

Toufik AHMED

le : 25 Novembre 2003

TITRE

Adaptive Packet Video Streaming Over IP Networks: A Cross Layer Approach

Diffusion Adaptative des Paquets Vidéo sur IP: Une Approche Cognitive

JURY

Samir Thomé	Professeur, ENST Paris	Président
Ahmed Karmouch	Professeur, Université de Ottawa	Rapporteur
Eric Horlait	Professeur, Université de Paris-6	Rapporteur
Jean-Pierre Claudé	Professeur, Université de Versailles	Directeur de thèse
Ahmed Mehaoua	MCF, Université de Versailles	Directeur de thèse
Raouf Boutaba	Professeur, Université de Waterloo	Examinateur
Olimpiu Negru	Dr., Thales Broadcast & Multimedia	Examinateur

Abstract

While there is an increasing demand for streaming video applications on IP networks, various network characteristics make the deployment of these applications more challenging than traditional Internet applications like email and the Web. These applications that transmit audiovisual data over IP must cope with the time varying bandwidth and delay of the network and must be resilient to packet loss and error. This dissertation thesis examines these challenges and presents a cross layer video streaming system design and implementation that ameliorates some of the important issues with packet video streaming over large scale IP networks with statistical quality of service (QoS) guarantee.

Video sequences are typically compressed according to the emerging MPEG-4 multimedia framework to achieve bandwidth efficiency and content-based interactivity. The original characteristic of MPEG-4 is to provide an integrated object-oriented representation and coding of natural and synthetic audio-visual content for manipulating and transporting over a broad range of communication infrastructures.

The originality of this work is to propose a cross-layer approach for resolving some of the critical issues on delivering packet video data over IP networks with satisfactory quality of service. While, current and past works on this topic respect the protocol layer isolation paradigm, the key idea behind our work is to break this limitation and to rather inject content-level semantic and service-level requirements within the proposed IP video transport mechanisms and protocols.

We leverage the characteristics of MPEG-4 and IP differentiated services frameworks, to propose an efficient and adaptive cross layer video delivery system over IP that is composed of (1) a system-level audio-visual object classification model; (2) a robust and adaptive application level framing protocol with fine-grained TCP-friendly rate control and unequal error protection; and (3) a dynamic network-level packet video marking algorithm for Diffserv edge routers.

To permit a wide share of MPEG-4 multimedia content between heterogeneous wired and wireless IP terminals, we have also designed and implemented an Interworking Signaling Gateway that supports both IETF Session Initiation Protocol (SIP) and ISO MPEG-4 Delivery Multimedia Integrated Framework (DMIF) Session Control Signaling Protocol. This multimedia signaling gateway performs various translation functions for transparent establishment and control of multimedia sessions across IP networking environment, including, session protocol conversion, service gateway conversion and address translation.

Keywords: IP, MPEG-4, Adaptive Quality of Service, Video Object Classification, Application Level Framing, TCP-friendly Rate control, Error Resilience, Session control signaling.

Résumé

Nous constatons aujourd'hui une forte demande de services vidéo sur les réseaux IP. Cependant, plusieurs caractéristiques de ces réseaux font que le déploiement à grande échelle de tels services présente un réel challenge par rapport à des applications telles que l'émail et le Web. Ces applications audiovisuelles doivent faire face aux différentes variations de la bande passante et du délai de transfert, tout en restant robuste aux pertes de paquets et aux erreurs de transmission. Cette thèse examine donc ces différents problèmes et présente un système de diffusion vidéo adaptatif et intégré (« cross-layer »), qui résout certains problèmes induits par le transport de la vidéo en mode paquet sur les réseaux IP avec garantie statistique de la qualité de service (i.e. IP Diffserv).

Les flux vidéo sont généralement compressés selon la norme MPEG-4 qui permet d'assurer une utilisation optimale de la bande passante ainsi qu'un meilleur degré d'interactivité basé sur la description structurelle et sémantique des flux. L'originalité de MPEG-4 est de fournir une représentation objet du contenu multimédia qui peut être naturel ou de synthèse afin de le transporter sur une large variété d'infrastructures de communication.

L'originalité de notre contribution est de proposer un système adaptatif de diffusion vidéo respectant l'approche intégrée ou également appelée « cross-layer ». En effet, tandis que la plupart des travaux de recherche dans ce domaine respectent le paradigme d'isolation et d'indépendance des couches protocolaires hérité du modèle de référence ISO, notre approche conceptuelle supprime cette limite en autorisant une meilleure prise en charge de la sémantique et des contraintes applicatives au niveau des couches et des mécanismes dédiés au transport.

Conformément aux modèles architecturaux MPEG-4 et IP Diffserv, nos contributions s'articulent autour (1) d'un modèle de classification automatique d'objets audiovisuels intégré à la couche système MPEG-4; (2) d'un protocole robuste de fragmentation et d'encapsulation de flux vidéo avec contrôle d'erreurs et de débits ; et (3) d'un algorithme de marquage dynamique de paquets vidéo IP pour routeurs d'accès asservi aux variations de bande passante et aux caractéristiques des flux multimédia.

Pour favoriser une plus grande interopérabilité des services multimédia MPEG-4 avec les terminaux IP fixes et mobiles, nous avons également proposé, conçu et implémenté une architecture de passerelle de signalisation supportant les protocoles de contrôle de sessions IETF SIP (Session Initiation Protocol) et ISO MPEG-4 DMIF (Delivery Multimedia Integrated Framework). Cette passerelle multimédia de signalisation permet d'assurer la continuité du service, la résolution d'adresse et la négociation des capacités des terminaux.

Mot clés: IP, MPEG-4, Qualité de Service Adaptative, Classification d'Objets Vidéo, Transport Temps Réel, Contrôle de Flux, Codes Correcteurs d'Erreurs, Signalisation.

Table of Contents

List of Figures

List of Tables

Acknowledgments

The first person I would like to thank and express my gratitude is my direct supervisor, Ahmed Mehaoua, without him this thesis will not be accomplished.

Thank to my thesis director, Jean-Pierre Claudé, who accepts to direct this thesis and being always friendly.

Many thanks to Raouf Boutaba, who advises me over the time and encourages me to give the best of me.

I would like also to thank Vincent Lecuire, form CRAN (Centre de Recherche en Automatique de Nancy). My collaboration with him was very beneficial.

Thanks to my colleagues of the PRiSM that all gave me the feeling of being at home at work. Without forgetting Souheila, Saida, Naçera, Karim, Guillaume, Alain, Isabelle, Octavio, Najim, Jalil, Abdelhamid, Yassine, Daniel, Armando, Adlen, Mathieu, Stéphane, Huaizhong, and many others colleagues. I thank you all for having shared many experiences and being for me as sisters and brothers.

Finally, I would like to thank the jury for their time, assistance and comments.

This chain of my gratitude would be definitely incomplete if I forget my parents without their foresight and help, I would not be where I am today.

Publications Arising From This Thesis

Contributions to Chapter's Books:

[1] *Toufik Ahmed*, Ahmed Mehaoua, Raouf Boutaba, Youssef Iraqi**, "IP Video Streaming With Fine-Grained TCP-Friendly Rate Adaptation",** in Lecture Notes in Computer Science, "Management of Multimedia Networks and Services", Springer-Verlag Editor, vol. 2839, pp. 18-31, 2003.

[2] *Toufik Ahmed,* Ahmed Mehaoua and Raouf Boutaba **"Dynamic QoS Adaptation using COPS and Network Monitoring Feedback"** in Lecture Notes in Computer Science, "Management of Multimedia on the Internet", Springer-Verlag Editor, vol. 2496, pp.250-262, 2002.

[3] *Toufik Ahmed*, Guillaume Buridant and Ahmed Mehaoua **"Delivering of MPEG-4 Multimedia Content Over Next Generation Internet"** in Lecture Notes in Computer Science, "Management of Multimedia on the Internet", Springer-Verlag Editor, vol. 2216, pp.110-127, 2001.

Articles in Refereed Journals:

[4] *Toufik Ahmed*, Raouf Boutaba and Ahmed Mehaoua**, "A measurement-based Approach for Dynamic QoS Adaptation in Diffserv Networks",** accepted for publication in Computer Communications Journal, Special issue on End-to-End Quality of Service Differentiation, Elsevier Editor, winter 2003.

[5] Vincent Lecuire*, Toufik Ahmed, et* Ahmed Mehaoua, **"Protection des Flux Audiovisuels MPEG-4 Transportés sur Internet",** à paraître dans la revue Technique et Science Informatique: Réseaux et Protocoles, Mai 2004.

Articles in Refereed Conference Proceedings:

[6] *Toufik Ahmed,* Ahmed Mehaoua and Vincent Lecuire **"Streaming MPEG-4 Audiovisual Objects Using TCP-Friendly Rate Control And Unequal Error Protection",** IEEE International Conference On Multimedia & Exo, ICME'03, Volume 2, pp. 317-320, Baltimore, July 2003.

[7] *Toufik Ahmed,* Abdelhamid Nafaa and Ahmed Mehaoua **"An Object-Based MPEG-4 Multimedia Content Classification Model for IP QoS Differentiation"** in proceedings of the Eighth IEEE Symposium on Computers and Communications ISCC'2003,Volume 2, pp. 1091-1096, Kemer - Antalya, Turkey, June 2003.

[8] Abdelhamid Nafaa, *Toufik Ahmed,* Yassine Hadjadj Aoul and Ahmed Mehaoua **"RTP4Mux: A Novel MPEG-4 RTP Payload for Multicast Video Communications Over Wireless IP"** in Packet Video PV'2003, April 28-29, Nantes, France, April 2003.

[9] *Toufik Ahmed*, Ahmed Mehaoua, Raouf Boutaba, **"Interworking Between SIP and MPEG-4 DMIF for Heterogeneous IP Videoconferencing"**, in IEEE International Conference on Communications (ICC'02), Vol. 4, pp. 2469-2473, April 2002, NewYork

[10] *Toufik Ahmed*, Ahmed Mehaoua and Guillaume Buridant **"Implementing MPEG-4 Video On Demand Over IP Differentiated Services"** in Globecom'2001, Volume 4, pp. 2489-2493, San Antonio, Texas, USA, November 2001.

[11] *Toufik Ahmed*, Guillaume Buridant and Ahmed Mehaoua **"Encapsulation and Marking of MPEG-4 Video over IP Differentiated Services"** in proceedings of the Sixth IEEE Symposium on Computers and Communications ISCC'2001, pp. 346-352. Hammamet, Tunisia. June 2001.

Articles in Refereed Conference Proceedings (French):

[12] *Toufik Ahmed*, Ahmed Mehaoua, Abdelhamid Nafaa and Raouf Boutaba **"Gestion Dynamique de la Qualité de Service de Flux Vidéo sur l'Architecture IP Diffserv"** in 5ème Colloque Francophone GRES, Février 2003.

[13] Abdelhamid Nafaa, *Toufik Ahmed*, Yassine Hadjadj aoul, Ahmed Mehaoua **"RTP4mux: Un Protocole de Transport et d'Agrégation de Flux Vidéo MPEG-4 pour Réseaux IP Sans Fils"** in 5ème Colloque Francophone GRES, Février 2003.

[14] *Toufik Ahmed,* Ahmed Mehaoua and Raouf Boutaba **"Un modèle de contrôle de service de vidéoconférence IP impliquant des terminaux multimédia SIP et MPEG-4"** in CFIP, Montréal Canada, Mai 2002.

[15] *Toufik Ahmed*, Guillaume Buridant, Ahmed Mehaoua et Jean-Pierre Claude **"DVMA: Un Algorithme de Marquage des Flux MPEG-4 sur IP Diffserv"** in 4ème Colloque Francophone GRES, Marrakech, Maroc, décembre 2001.

Posters in Refereed Conference Proceedings:

[16] *Toufik Ahmed,* Raouf Boutaba and Ahmed Mehaoua **"On Enhancing Diffserv Architecture by Dynamic Policy Provisioning Using Network Feedback"** in IFIP/IEEE Conference on Network Control and Engineering (Net-Con'02), Paris, France, pp. 273-278, October 2002.

Others Contributions:

[17] *Toufik Ahmed*, Ahmed Mehaoua, Raouf Boutaba " Interworking Between SIP and MPEG-4 DMIF" **Internet Draft** draft-ahmed-DMIF-SIP-00.txt work in progress, September 2001.

Chapter 1

1 Résumé de Thèse

Le transport de contenus multimédia codés au standard MPEG-4 sur Internet est soumis à des contraintes de débit, de délai et de fiabilité. La charge et la nature du trafic sur le réseau ont des effets sur les pertes et les délais d'acheminement des paquets qui doivent être contrôlés. Dans cette thèse, nous proposons un système adaptatif de diffusion vidéo respectant l'approche intégrée ou également appelée « *cross-layer* ».

Conformément aux modèles architecturaux MPEG-4 et IP Diffserv, nos contributions s'articulent autour de trois axes de recherches:

Premièrement, nous proposons une couche de transport adaptative des flux MPEG-4. Cette couche implémente les fonctionnalités suivantes :

- un modèle de classification automatique d'objets audiovisuels intégré à la couche système MPEG-4 : ce système permet d'assurer la mappage de la qualité de service du niveau applicatif à celle du niveau réseau. L'algorithme est basé sur l'utilisation des descripteurs d'objets audio vidéo MPEG-4, et leur classification de manière automatiques. Cela permet d'affecter une priorité relative à tous les objets présents dans la scène MPEG-4.

- un protocole robuste de fragmentation, d'encapsulation et de transport de flux vidéo avec contrôle d'erreurs et de débits. Ce protocole réalise la synchronisation, la fragmentation, l'encapsulation et le multiplexage optimal des flux élémentaires MPEG-4 sur les réseaux IP, réduisant fortement la surcharge de contrôle associée à la couche transport temps réel (RTP), tout en maintenant une synchronisation inter-flux forte et une robustesse face aux erreurs et pertes de données. Ainsi, les objets MPEG-4 exigeant les mêmes critères de QoS, seront regroupés ensemble, encapsuler sur RTP et protéger contre les erreurs en fonction de leur indice de priorité.

- Un algorithme de contrôle de débit compatible TCP (*TCP-Friendly*). Cet algorithme permet d'ajuster finement le débit du serveur vidéo en fonction du débit autorisé pour sa connexion. L'ajustement est contrôlé en ajoutant / supprimant des objets MPEG-4 en fonction de leurs priorités initialement calculées par le modèle de classification.

Deuxièmement, nous proposons un algorithme de marquage de flux vidéo appelé DVMA (*Diffserv Video Marking Algorithm*), qui tient compte de la priorité relative de chaque objet MPEG-4. Cet algorithme de marquage peut être implémenté sur le serveur vidéo ou déployé sur les routeurs

d'accès. Les interactions de QoS entre le serveur vidéo et le réseau de transport se traduisent par l'association d'un flux élémentaire MPEG-4 (vidéo, audio, descripteurs d'objets, descripteur de scène) et d'un PHB (*Per Hop Behavior*) de DiffServ tel que AF/EF. Nous montrons aussi par une plateforme, comment configurer, administrer et déployer les politiques de marquage des flux vidéo. L'algorithme proposé peut être aussi asservi aux variations de la bande passante et aux caractéristiques des flux multimédia. Ce qui permet d'assurer un marquage dynamique en fonction de l'état du réseau.

Finalement, pour favoriser une plus grande interopérabilité des services multimédia MPEG-4 avec les terminaux IP fixes et mobiles, nous avons également proposé, conçu et implémenté une architecture de passerelle de signalisation supportant les protocoles de contrôle de sessions IETF SIP (*Session Initiation Protocol*) et ISO MPEG-4 DMIF (*Delivery Multimedia Integrated Framework*). Cette passerelle multimédia de signalisation permet d'assurer la continuité du service, la résolution d'adresse et la négociation des capacités des terminaux.

1.1 Transport Adaptatif Des Flux Vidéo

1.1.1 Système de Classification des Flux MPEG-4

MPEG-4 est un standard ISO/IEC qui fournit une spécification des outils d'encodage naturels et synthétiques des flux audio et vidéo. MPEG-4 cible un large rayon d'applications multimédia telles que la vidéoconférence classique et la télédiffusion. L'intérêt principal de ce standard est de fournir un langage commun de description pour l'encodage 2D et 3D des objets audiovisuels (ISO/IEC 14496-1/14496-2/14496-3, 1998). Cette section présente les caractéristiques des flux MPEG-4 et définit un système de classification qui sera utilisé dans notre proposition.

1.1.1.1 Notion d'Objets Audiovisuels

Une approche orientée objet est adoptée dans MPEG-4 en vue de représenter des scènes complexes. Ce modèle respecte le système audiovisuel humain, qui perçoit un ensemble d'objets plutôt qu'une matrice de pixels. Une scène est décrite comme une composition d'objets audiovisuels élémentaires, les AVO (*Audio-Visual Objects*), qui sont combinés selon une structure particulière. La Figure 1-1 montre un exemple d'une scène MPEG-4 composée de plusieurs AVO.

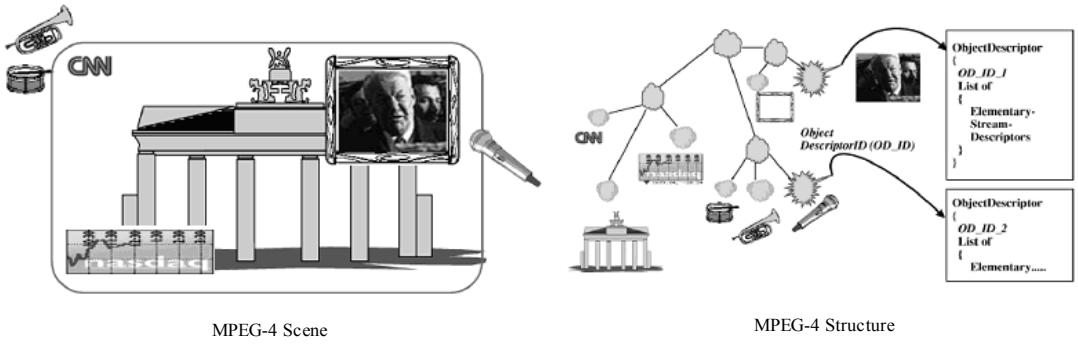


Figure 1-1: Structure d'une scène MPEG-4

1.1.1.2 Architecture MPEG-4

Un terminal MPEG-4 est composé de trois couches, comme cela est présenté dans *l'appendix A* : Compression, Synchronisation et Livraison (ISO/IEC 14496-1, 1998). La couche Compression génère la représentation des données audiovisuelles dans des flux élémentaires, les ES (*Elementary Streams*). Les relations hiérarchiques, l'emplacement et les propriétés des ES sont décrits par un ensemble de descripteurs d'objets, les OD (*Object Descriptors*). Les OD sont eux-mêmes transportés dans un ou plusieurs flux élémentaires. Une scène MPEG-4 est décrite par un descripteur de scène, le SD (*Scene Description*), qui contient les informations sur l'adresse et l'organisation des objets audiovisuels de la scène, en terme d'emplacement spatial et temporel. Le descripteur de scène utilise un langage basé sur VRML (*Virtual Reality Modeling Language*) appelé BIFS (*Binary Format for Scene*). Les ES sont fractionnés en une ou plusieurs unités d'accès, les AU (*Access Units*). Une AU est la plus petite entité de données à laquelle on peut assigner une information de temps. Un flux élémentaire MPEG-4 peut être une vidéo MPEG-1 par exemple, et dans ce cas chaque AU va correspondre à une image compressée de type I, B ou P.

Sur la couche Synchronisation, les AU sont transportées en une séquence de paquets appelés SL-PDU. Un SL-PDU est constitué d'un entête de paquet SL et de données dans le paquet SL. L'entête contient des informations telles qu'un numéro de séquence et une estampille temporelle pour les besoins liés au contrôle des pertes de paquets, ainsi qu'à l'ordre et à l'échéance de leur livraison.

Enfin, l'interface entre la couche Synchronisation et la couche Livraison offre des procédures indépendantes qui permettent d'établir des sessions MPEG-4 et d'accéder à des canaux de transport comme RTP/UDP/IP.

1.1.1.3 Classification des AVOs

La classification des AVOs est assurée par un modèle à deux niveaux. Le premier niveau correspond à une classification des AVOs selon leur importance dans la scène, et le deuxième correspond à une classification inter-AVO, c'est-à-dire entre les unités qui composent le flux d'un même AVO.

Dans l'architecture MPEG-4, un AVO est décrit par un descripteur d'objet (OD) qui spécifie la configuration du flux de données. Dans un OD, plusieurs descripteurs coexistent : on y trouve par exemple *DecoderConfigDescriptor* qui décrit le type du décodeur utilisé ainsi que les ressources nécessaires pour bien interpréter le flux de données du coté du client. Le descripteur qui nous intéresse pour les besoins de la classification des flux est celui qui décrit les paramètres de QoS demandés par chaque objet MPEG-4. Ce descripteur est appelé *QoSDescriptor* (voir annexe). *QoSDescriptor* encapsule des paramètres de qualité de service sur le canal de transport demandé par un objet. La Table 1-1 illustre la structure du descripteur de QoS.

Métrique	Description
MAX_DELAY	Délai maximum accepté de bout en bout
PREF_MAX_DELAY	Délai maximum préférable de bout en bout
LOSS_PROB	Probabilité de perte des AU acceptée
MAX_GAP_LOSS	Nombre de perte d'AU consécutives accepté
MAX_AU_SIZE	Taille maximale d'une AU
AVG_AU_SIZE	Taille moyenne d'une AU
MAX_AU_RATE	Débit maximal des AU
PRIORITY	Priorité du flux

Table 1-1: Les paramètres de qualité de service associés aux AVOs

Nous exploitons les informations contenues dans ce descripteur pour classer les différents objets existants dans une scène MPEG-4. Nous utilisons pour cela un algorithme de classification à base de réseaux de neurones.

La technique de codage utilisée dans MPEG-4 permet le codage de chaque AVO par un codeur adapté pour obtenir le meilleur ratio de compression. Dans une scène MPEG-4, on peut donc trouver un objet vidéo codé en H.263 et un autre en MPEG-1 par exemple.

Le codage MPEG génère trois types d'images codées, fournissant des niveaux de scalabilité temporelle. Ces trois types d'images sont appelées : I-VOP (*Intra coded Video Object Plane*), P-VOP (*temporally Predicted Video Object Plane*), et B-VOP (*Birectionally predicted Video Object Plane*).

Le codeur MPEG autorise la configuration de la fréquence et de la position des images clés. Cette configuration détermine comment les images sont rassemblées en groupes appelés GOP (*Group Of Pictures*), une séquence vidéo étant considérée comme une suite de GOP. Un GOP est constitué de N images. Il commence par une image de type I suivie par des séquences constituées par un nombre (M-1) d'images B et d'une image P. Par exemple, une configuration fixée à M=3 et N=12 donne des groupes d'images formés sur la série IBBPBBPBBPBB. Il est clair que l'importance des AU appartenant aux différents types d'images n'est pas la même. L'image de type I-VOP est la plus importante puisque sans elle, c'est tout le GOP (et même un peu plus, N+M−1 images précisément) qui ne pourra pas être reconstitué. Les images de type P-VOP ont une importance qui dépend de leur position dans le GOP.

D'une manière générale, les sources vidéo MPEG-4 sont codées selon un modèle de représentation multi-niveau. La source vidéo est codée en plusieurs flux compressés correspondant chacun à un incrément de débit et de qualité en termes de corrélation temporelle, de résolution spatiale, ou de rapport signal sur bruit. On distingue un flux de base (BL ou *Base Layer*), qui est le plus important, et un ou plusieurs flux d'améliorations (EL ou *Enhancement Layer*)

Nous avons utilisé un algorithme de classification basé sur la technique de classification RBF (*Radial Basis Function*). RBF permet de trier un ensemble d'objets en utilisant une fonction basée sur un vecteur de distance. Nous exploitons la caractéristique du réseau de transport sous-jacent (Diffserv dans notre cas) pour marquer chaque objet avec une étiquette – le label – qui définit la classe de service de l'objet. Les objets sont classés un par un et indépendamment les uns des autres.

Considérons un objet MPEG-4 à classifier, caractérisé par un vecteur $\mathrm{X} \in \Re^n$. Ce vecteur X est défini par $(x_1, x_2, \ldots x_n)^T$, où chaque composante x_i représente une caractéristique de qualité de service associée à l'objet, parmi celles données en Table 1-1 Section 1.1.1.3. Soit T, l'ensemble des étiquettes qu'un objet peut recevoir. Notre système de classification correspond à une fonction $C : \Re^n \to T$ qui assigne une étiquette (ou une priorité) $c \in T$ à chaque vecteur $X \in \Re^n$ en se basant sur un ensemble de vecteurs caractéristiques des modèles $\{U_1, U_2, \ldots U_m\}$.

La Figure 1-2 illustre le principe de fonctionnement du classificateur RBF. La méthode utilise un ensemble de blocs reliés entre eux pour former un réseau. Le chemin parcouru de l'entrée à la sortie du réseau est appelé neurone. Un bloc correspond à un traitement particulier identifié par une fonction de transfert qui doit prendre en compte les caractéristiques de l'objet MPEG-4 en entrée.

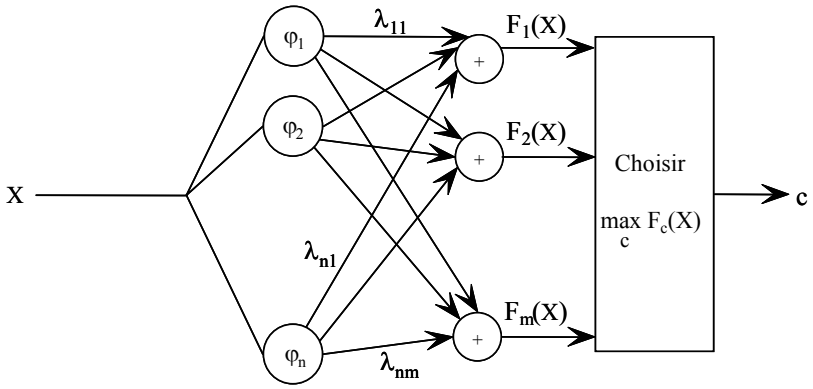


Figure 1-2: Architecture du classificateur RBF

La fonction de transfert $\varphi_{j=1,2,\ldots n}$ est une fonction gaussienne centrée sur un point de l'espace d'entrée. Pour une entrée donnée, chaque sortie du bloc φ_j est calculée selon la formule :

$$\varphi_j(x) = e^{-\frac{D^2}{2\sigma_j^2}}$$

où $D = \|X - U_k\|^2$ est la distance entre le vecteur X et un vecteur modèle U_k pour la caractéristique de rang j et σ_j est la déviation standard permettant de régler la sensibilité du résultat aux données d'entrée.

La fonction gaussienne permet aux neurones de ne répondre qu'à une petite région de l'espace d'entrée, région sur laquelle la gaussienne est centrée. La sortie du réseau est simplement une combinaison linéaire des sorties des neurones RBF multipliés par le poids λ_{jk} de leur connexion respective. Le poids permet à l'utilisateur de donner plus d'importance à certains paramètres de qualité de service qu'à d'autres.

Avec notre classificateur RBF, l'équation générale du résultat d'un neurone est donnée par la relation: $F_k(X) = \sum_{j=1}^{n} \lambda_{jk} \varphi_j(X)$. Le classificateur choisit l'étiquette correspondant au résultat du neurone le plus grand qui correspond à la classe de service choisie par le réseau RBF.

1.1.2 Proposition dun Protocole d'Encapsulation d'Objet MPEG-4 Sur RTP

1.1.2.1 Format de Payload RTP pour les Flux Audio/Vidéo MPEG-4

Cette approche reste le seul standard, proposé par l'IETF dans le RFC 3016 [156], pour le transport des flux audiovisuels MPEG-4. Le format du paquet spécifie un mécanisme de fragmentation des flux audio/vidéo MPEG-4 pour pouvoir ensuite l'encapsuler dans un flux RTP, sans utiliser le système MPEG-4 (*Sync Layer*) pour la synchronisation et la gestion des flux élémentaires MPEG-4. Cette spécification fournit des règles pour l'utilisation des champs de l'entête RTP et la fragmentation du flux vidéo en entités indépendamment décodables. Elle donne enfin des indications pour l'utilisation du MIME (*Multipurpose Internet Mail Extension*) et du SDP (*Session Description Protocol*) [141] pour la signalisation et la configuration, au niveau du récepteur, des types et des paramètres des médias utilisés dans la session MPEG-4.

Les flux MPEG-4 (audio et vidéo seulement) sont directement encapsulés dans des paquets RTP, après fragmentation préalable. Chaque flux élémentaire est acheminé dans une session RTP individuelle, identifiée par une adresse réseau et un numéro de port. Le RFC 3016 ne spécifie pas l'utilisation des flux élémentaires du système MPEG-4 (OD, BIFS, etc.), mais ceci peut être réalisé par les applications à travers, par exemple, le protocole de signalisation H.245. Ce qui représente un atout pour le déploiement d'un streaming MPEG-4 par rapport à d'autre média non MPEG-4. Le transport des flux audio passe par un formatage LATM (*Low-overhead Audio Transport Multiplexing*), ce qui optimise l'utilisation du payload RTP à travers le multiplexage de plusieurs flux audio.

1.1.2.2 Format de Payload RTP avec une Entête SL Réduite

Cette approche, proposée par l'IETF dans [157] et [159], préconise l'utilisation de la couche SL, puis l'encapsulation du flux de paquets SL dans le payload de RTP avec un éventuel mapping, de quelques champs redondants, de l'entête SL (voir Figure 1-3) vers l'entête RTP. Ce mode d'encapsulation fait intervenir un mapping (transfert) de certains champs pertinents de l'entête SL vers l'entête RTP.

Parmi les apports de cette approche, nous citons :

- Un payload générique et qui ne dépend pas de la nature du flux transporté (audio, vidéo, OD, SD,....etc.)
- Un payload hautement configurable (i.e. la structure du payload RTP dépend des paramètres MIME) et ajustable pour convenir à chaque flux élémentaire individuel.
- Deux style d'encapsulation (singulière et multiple). Une encapsulation d'une AU, ou un fragment d'AU, permet la compatibilité avec le RFC 3016, tandis que l'encapsulation (concaténation) de plusieurs AUs dans le payload RTP exploite mieux la bande passante en diminuant l'overhead.
- Possibilité de transporter plusieurs AUs (ou fragment d'AU) entrelacées dans le payload du paquet RTP, ce qui permet une plus grande tolérance aux pertes de paquets RTP.

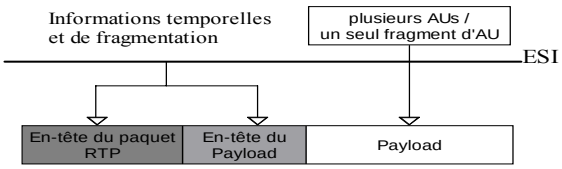


Figure 1-3: Transport des flux MPEG-4 avec une entête SL réduite

1.1.2.3 Format de Payload RTP pour les *FlexMux* MPEG-4

Cette approche, proposée par l'IETF dans [158], repose sur l'encapsulation d'un nombre entier de paquets FlexMux complets dans la charge utile de RTP. Contrairement aux autres approches, proposées par l'IETF dans, [156], [157] et [159], qui s'appuient sur l'utilisation d'autant de sessions RTP que de flux élémentaires dans la session MPEG-4, cette approche permet le multiplexage de plusieurs ESs sur une même session RTP, ce qui entraîne un gain en performance au niveau du serveur et du player. Nous distinguons aussi les fonctionnalités suivantes:

- Utilisation d'une session commune pour des flux qui sont étroitement synchronisés à travers la table *codetable* du *FlexMux* (table qui doit être signalée au player au préalable).
- L'utilisation de la technologie *FlexMux* permet une interconnexion entre le réseau Internet et le réseau de la télévision numérique (avec l'utilisation de la syntaxe *FlexMux* de MPEG-4 pour transporter du flux MPEG-4 sur les canaux de transport MPEG-2).
- Possibilité de regrouper des ES, de même exigences en QoS, dans un même flux *FlexMux*, puis session RTP, pour réaliser un mapping efficace de la QoS sur le réseau de transport sous-jacent (comme par exemple l'agrégation des flux DiffServ sur des réseaux IP).

1.1.2.4 Discussion et Analyse des Approches Existantes

L'approche [156] propose un mécanisme de fragmentation de la vidéo MPEG-4, issue de la couche de compression MPEG-4, qui prend en compte la hiérarchie du codage MPEG-4 et permet par conséquent une plus grande tolérance aux pertes de paquets. De plus, cette spécification propose un formatage du flux audio de façon à permettre le multiplexage de celui-ci et offre la possibilité d'exploiter tout le payload (dans la limite fixée par le path-MTU). L'absence du système MPEG-4 (OD, BIFS, etc.), pour la gestion et le synchronisation des flux élémentaires, entraîne une faible interaction avec la séquence MPEG-4. De plus, du fait que chaque flux élémentaire doit être acheminé, du serveur vers le player, dans une session RTP distincte, cette approche peut ne pas être optimale, en particulier pour une séquence MPEG-4 contenant plusieurs dizaines d'objets audiovisuels.

Dans l'approche [157] et [159], une encapsulation du flux de paquets SL est proposée avec un mapping, ou suppression, de quelques champs redondants ou inutiles de l'entête SL. D'où, l'utilisation optionnelle de quelques champs pertinents de l'entête SL (par exemple: les champs CTS « *Composition Time Stamp* » et DTS « *Decoding Time Stamp* » qui sont particulièrement utiles pour un codage prédictif bidirectionnel plus efficace). Cette approche permet aussi une concaténation de plusieurs AUs avec un éventuel entrelacement optionnel pour une meilleure tolérance aux pertes

des paquets RTP. Reste toujours le problème de gestion de plusieurs dizaines de sessions RTP, au niveau du serveur et du player, pour une session MPEG-4 et l'absence d'un mécanisme de multiplexage au niveau de RTP. Mais aussi, le problème de la faible exploitation de la bande passante (charge utile de RTP) pour les flux à bas débit, problème qui est aussi retrouvé dans [156].

Le gain apporté par le multiplexage réalisé par l'approche [158], avec une gestion plus simplifiée des sessions RTP au niveau du serveur et du player, permet aussi un mapping efficace de la QoS sur le réseau de transport sous-jacent. Ceci dit le principal inconvénient reste l'excès d'overhead provoqué par la redondance de certains champs existants dans l'entête RTP et l'entête SL, avec l'impossibilité de mapper ces champs redondants, de l'entête SL vers l'entête RTP, vu l'intercalation de l'entête FlexMux.

La synthèse de ces approches est représentée dans la Figure 1-4 ci-dessous.

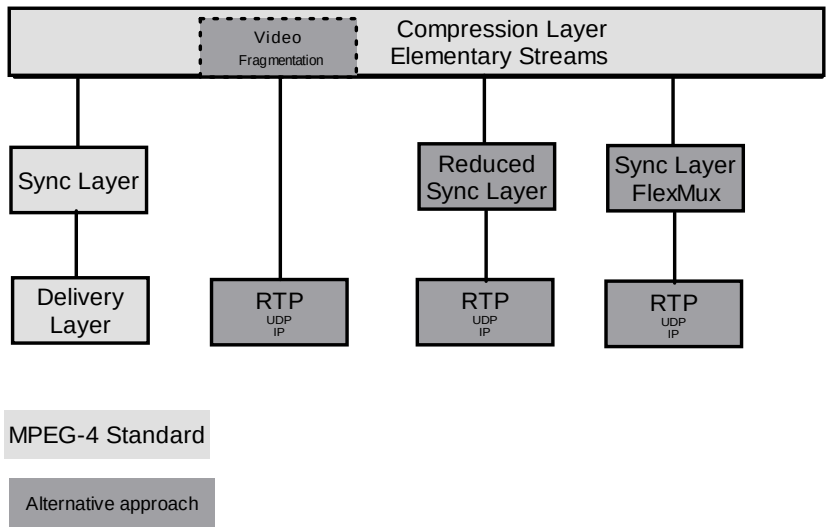


Figure 1-4: Synthèse des approches du transport de la vidéo MPEG-4 sur les réseaux IP

1.1.2.5 Proposition Du Protocole RTP4mux

Un protocole d'encapsulation efficace réduisant l'overhead (surcharge d'entête) est essentiel pour le transport optimal des flux MPEG-4 sur les réseaux IP. Ce protocole doit minimiser le problème de surcharge (voir Table 1-2) tout en permettant une gestion et une synchronisation des flux élémentaires. Ce protocole spécifie un format de charge utile RTP générique, tout en intégrant des fonctionnalités de gestion temporelle existantes dans la couche MPEG-4 Sync Layer (Temps de composition »CTS » et le temps de décodage « DTS »). Le format de charge utile doit, également, permettre le multiplexage de plusieurs flux élémentaires dans une même session RTP pour faciliter la gestion des flux élémentaires, exploiter convenablement la charge utile de RTP et favoriser la scalabilité du serveur.

Audio Object	Audio Payload Length	Overhead (RTP/UDP/IP)	Overhead Rate
AAC (64 kbit/s, 24 kHz)	342 bytes (average)	12+8+20 bytes	10,9%
CELP (6 kbit/s, 8 kHz)	15 bytes (fixed)	12+8+20 bytes	72,7%

Table 1-2: Exemple d'overhead introduit par le transport des flux MPEG-4 à bas débit

Quand un codage AAC (Adaptive Audio Coding) est utilisé, pour le codage d'un signal sonore stéréo à 64 Kbps, la couche de compression génère des trames AAC avec une taille moyenne de 200 octets. Pour optimiser l'utilisation de la bande passante (diminuer l'overhead), nous devons effectuer une concaténation de plusieurs AUs, en particulier pour les flux audio ayant des AUs de petite taille (AAC, CELP, etc.), qui pour un WLAN avec 1500 octets de MTU permettra de concaténer 7 trames AAC et avoir un gain conséquent en bande passante.

Dans les flux à bas débit (AAC, FA « Facial Animation », CELP « Coding Expedited Linear Predective audio », flux TTSI «Text-to-Speech Interface», etc.) l'intervalle temporel entre la génération de 2 AUs successives peut s'avérer très important. La concaténation de plusieurs AUs de ce flux, pour diminuer l'overhead, impliquera des délais de packetization très importants et intolérables pour des communications à temps réel. De plus, ces flux MPEG-4 à bas débit sont destinés à être très largement déployés dans les réseaux WLAN, d'où la nécessité d'optimiser la bande passante à travers une bonne exploitation du payload RTP.

A partir de ce constat, nous proposons l'encapsulation, dans le payload RTP, de plusieurs AUs issues de différents flux élémentaires – nous allons voir par la suite que nous regroupons les flux ayant les même exigence de QoS. Ce qui réduira les délais au niveau de l'encapsulation RTP tout en bénéficiant des avantages apportés par le multiplexage des ESs. Nous pouvons, ainsi, palier le problème de la faible exploitation de la charge utile de RTP et la sur utilisation de la bande passante entraînée.

La motivation de concaténer plusieurs AUs dans le payload RTP est la réduction de l'overhead, d'où une meilleure exploitation de la bande passante. Tandis que la motivation d'encapsuler des AUs, issues de flux élémentaires (ESs) différents, est la réduction des délais de *packetization*. De plus, ce double multiplexage permet une meilleure robustesse aux pertes en réduisant les dépendances entre les paquets RTP adjacents.

Dans le protocole RTP4mux, le multiplexage (regroupement) d'un certain ensemble de flux élémentaires dans un même payload RTP sera fait :

- Sur la base de la nature des flux élémentaires (flux étroitement synchronisés ou ayants un même besoin en terme de champs d'entête du payload RTP). Par exemple: flux d'animation d'une bouche et le flux CELP de parole associée, etc.
- Ou encore, sur la base des exigences en QoS des flux, c'est-à-dire réaliser une sorte d'agrégation des ESs dans un même paquet RTP (ESs ayants les mêmes besoins en QoS) pour faciliter le mapping de la QoS sur le réseau de transport sous-jacent (particulièrement intéressant pour un réseau DiffServ).

Pour remédier au surplus d'overhead, dû à la redondance ou l'inutilité de certains champs de l'entête SL, nous proposons d'adopter, avec RTP4mux, une encapsulation des flux SL avec un mappage des champs redondants, mais surtout avec la possibilité de concaténer plusieurs AUs dans le payload (dans la limite fixée par le path-MTU). Ainsi, nos pallierons le problème d'overhead, dû à des AUs de petite taille. De plus l'entête SL peut être configuré selon le besoin et la nature du flux transporté (définir la présence et taille en bits des champs de l'entête du payload).

Nous pouvons distinguer trois possibilités pour le multiplexage des ESs dans les flux RTP:

- ***Une session RTP pour chaque ES:*** cette possibilité présente l'avantage de pouvoir bénéficier de la possibilité de différencier les flux élémentaires selon leurs besoins en QoS. De plus, les ES_IDs sont signalés à l'établissement de la session MPEG-4 et ne nécessitent pas une autre signalisation, ou une maintenance d'état, au cours de la session. Cependant, vu que les ESs sont dynamiques dans une session MPEG-4, l'apparition d'un nouveau flux élémentaire dans la session MPEG-4, introduit des délais d'établissement de la session RTP associée à l'ES (comme c'est déjà le cas pour l'approche adoptée par le RFC 3016 [156] pour le multiplexage des flux élémentaires MPEG-4).

- ***Une session RTP pour tous les ESs:*** cette possibilité offre une gestion simplifiée des ESs mais ne permet pas la différenciation entre les flux élémentaires selon leurs besoins en QoS.

- ***Une session RTP pour quelques ESs:*** cette possibilité permet, en plus de la gestion simplifiée des ESs, le regroupement des ESs fait sur la base d'une classification préalable et sur des considérations de QoS, en permettant une différenciation des différents regroupements et l'exploitation du codage hiérarchique. Ce mécanisme sera retenu pour la suite de la définition de notre format de payload RTP.

Avant de détailler l'implémentation des différents modules du protocole RTP4mux, nous exposons, ci-dessous, brièvement les principales fonctionnalités de protocole RTP4mux:

- ***Fragmentation et encapsulation des flux élémentaires:*** cette fonction prend en considération la nature du flux à transporter, en fragmentant ce dernier en entités indépendamment décodables. De plus, la partie encapsulation améliore la tolérance aux pertes au niveau application à travers le déploiement d'un mécanisme d'entrelacement des flux élémentaires.

- ***Multiplexage et démultiplexage des flux élémentaires :*** cette fonction de multiplexage, qui est réalisée à deux niveaux, est destinée aux réseaux IP à bas débit (WLAN). D'une part, nous multiplexons plusieurs flux élémentaires sur une même session RTP. D'autre part, nous multiplexons des AUs, appartenant à des ESs différents, dans un même payload RTP.

- ***Synchronisation des flux élémentaires:*** cette fonction, également à la charge de RTP4mux, permet la synchronisation inter et intra flux élémentaires agrégés. Cette synchronisation repose sur le système MPEG-4 et évite, par conséquent, une synchronisation au niveau transport, beaucoup plus complexe et contraignante.

- ***Protection inégale aux erreurs:*** La protection inégale aux erreurs s'applique bien sur les sources vidéo codées selon un modèle objet. Le taux de protection assigné à un objet dépend de son niveau relatif de priorité. Le rapport entre la bande passante utilisée et la distorsion induite par les pertes est le critère à optimiser.

Dans le reste de cette section nous exposons plus en détails, l'implémentation des fonctionnalités du protocole RTP4mux.

1.1.2.5.1 Fragmentation et Encapsulation des Flux Elémentaires

Puisque le payload RTP sera constitué d'une concaténation d'AUs appartenant à différents flux élémentaires (encapsulation de plusieurs flux SL), nous n'aurons plus besoin de déployer un mécanisme d'entrelacement des AUs, dans la mesure où la perte d'un paquet RTP ne provoquera pas la perte d'AUs appartenant qu'à un seul ES. Ceci rend l'utilisation du champ Index (resp. Index-Delta), de l'entête de chaque AU, optionnelle et permettra de réduire l'overhead.

L'entrelacement des ESs, utilisé avec RTP4mux, s'avère plus efficace que le mécanisme d'entrelacement d'AUs proposé dans [159]. En fait, l'encapsulation adoptée dans le protocole RTP4mux répartit les conséquences de perte d'un paquet RTP sur tout les flux multiplexés.

Nous proposons d'encapsuler, dans le payload RTP, plusieurs paquets SL réduits appartenants à différents flux élémentaires (voir Figure 1-5). Contrairement au format adopté par RFC 3016, qui utilise deux configurations distinctes du payload RTP pour les flux audio et les flux vidéo avec un acheminement en hors bande des flux système (à travers H.245, .., etc.), notre format de payload RTP sera unique et conviendra à tous les flux élémentaires MPEG-4 grâce, notamment, à l'entête SL qui peut être configuré selon la nature du flux.

Parmi les regroupements efficaces des paquets SL réduits dans un paquet RTP, nous citons celui qui est fait sur la base de flux élémentaires étroitement synchronisés. Qui, en cas de pertes de paquet RTP, préserve la synchronisation entre les flux SL multiplexés dans le même payload RTP.

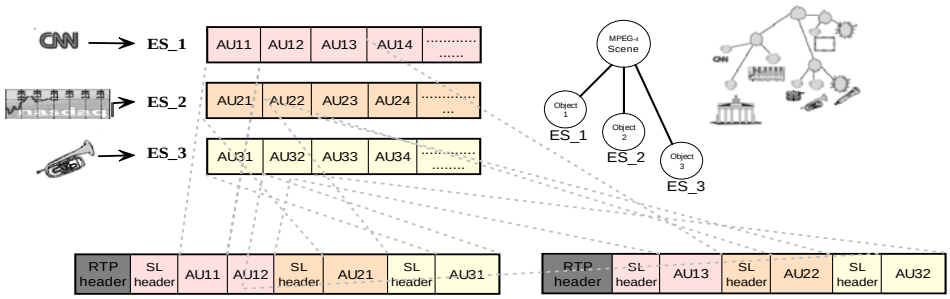


Figure 1-5: Processus d'Encapsulation des Flux Elémentaires (ES) MPEG-4

Il est clair que notre format de payload RTP exploite efficacement la bande passante. Le gain en overhead est réalisé à travers :

- Une concaténation de plusieurs AUs dans un paquet SL (avec le partage de quelques champs du SL Header par les AUs du même paquet SL réduit et par conséquent une réduction de l'overhead).
- Une concaténation de plusieurs paquets SL, flux élémentaire, dans un même paquet RTP (avec le partage du RTP Header et par conséquent une réduction de l'overhead). Contrairement à plusieurs approches où chaque ES est encapsulé dans une session RTP à part même si le payload RTP n'est pas complètement utilisé.

1.1.2.5.2 Multiplexage des Flux Elémentaires MPEG-4

Contrairement aux approches existantes [156], [157] et [159] qui transportent un ES par session RTP et Un AU par paquet RTP. Le protocole RTP4mux, à travers sa fonction de multiplexage à 2 niveaux, exploite mieux la bande passante (i.e. réduit l'overhead de transport). De plus, notre multiplexage réduit considérablement les délais de transmission de bout en bout mesurés au niveau application.

Le multiplexage des différents ESs dans un paquet RTP passe par une signalisation et une identification des différentes AUs contenues dans le payload RTP (à travers leurs ES_IDs respectifs).

1.1.2.5.3 Démultiplexage des Flux Elémentaires MPEG-4

Du fait que chaque flux élémentaire est acheminé dans une session RTP à part. Dans ce cas, le player connaît à priori la configuration des champs de l'entête du payload RTP et l'identifiant du flux transporté dans la session RTP en question (puisque la configuration des champs de l'entête du payload est faite à l'établissement de la session MPEG-4 par l'intermédiaire du message SDP). Dans notre approche, à la réception d'un paquet RTP, le player n'a pas connaissance de l'organisation du payload ni des flux élémentaires encapsulés dans le paquet RTP.

A partir de ce constat, et pour permettre un démultiplexage sans ambiguïté des ESs, deux approches sont envisageables (i.e. approches de signalisation des ES_IDs pour identifier les paquets SL contenus dans le payload de RTP). Une signalisation mappée sur un champs de l'entête RTP, impliquant une signalisation hors bande et signalisation insérée dans l'entête du paquet SL réduit, impliquant une signalisation dans la bande.

1.1.2.5.3.1 Signalisation Des Identifiants De Flux (ES_Ids) Dans l'Entête RTP

D'une manière similaire au mode *codeMux* de l'outil *FlexMux* (*codetable*), cette signalisation indique, pour un codeMux donné, l'organisation du payload (séquence de paquets SL « réduits » identifiée par leurs ES_IDs). Ce code pourra être mappé dans l'entête RTP (le champ SSRC, voir Figure 1-6).

Ceci dit, cette signalisation reste complexe et lourde dans l'établissement de la session, dans la mesure où il faudra signaler, au player, tout les regroupements possibles des paquets SL, dans le paquet RTP, ainsi que le code associé à chaque regroupement donné. Cette signalisation, en hors bande, des tables de correspondance entre codeMux et ES_IDs associés, sera sujette au risque de perte d'où un blocage dans le démultiplexage puis dans le décodage.

De plus, le caractère changeant de la séquence vidéo MPEG-4, avec entre autre la disparition et l'intervention de plusieurs objets média (et les ESs qui leur sont associés), au cours de la session MPEG-4, implique une signalisation continue des tables de correspondance entre CodeMux et ES_IDs qui leurs sont associées.

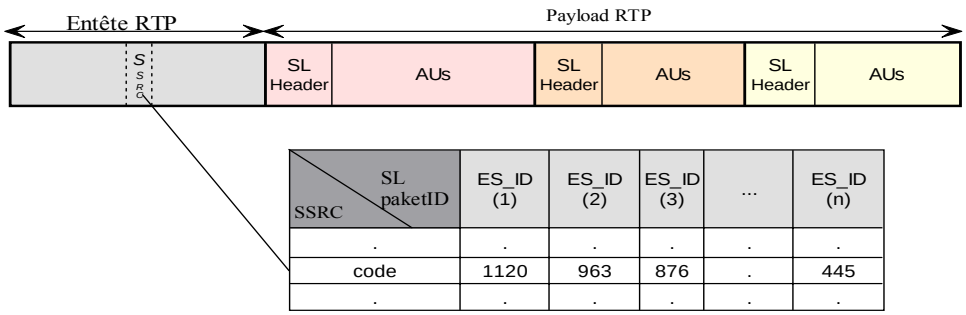


Figure 1-6: Signalisation des ES_ID dans l'Entête RTP.

1.1.2.5.3.2 Signalisation Des Identifiants De Flux (ES_Ids) Dans Le Payload RTP

Cette signalisation permet la concaténation des AUs avec la possibilité d'avoir plusieurs occurrences d'AUs d'un même flux élémentaire dans le même paquet RTP (voir Figure 1-7). Cette signalisation permet: D'une part, de réaliser un démultiplexage, sans ambiguïté, basé uniquement sur les ES_IDs qui sont acheminés dans l'entête de chaque paquet SL (i.e. signalisation dans la bande). D'autre part, d'éviter une signalisation continue des tables de correspondance (en plus de la signalisation faite pour les ODs et BIFS) qui peuvent être sujettes aux pertes et provoquer par conséquent un blocage dans le démultiplexage. De plus, le champ SSRC sera ainsi laissé pour une utilisation conforme aux recommandations du RFC 3550 [110]. Ce champ pourra être exploité dans des sessions multicast de communication bidirectionnelle (pour identifier l'originaire du flux et les participants actifs).

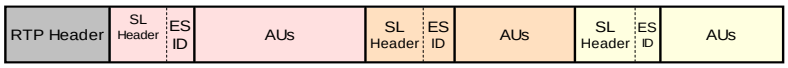


Figure 1-7: Signalisation des ES_IDs dans le Payload RTP.

Cette signalisation sera retenue pour la suite de la définition de notre format de payload RTP.

1.1.2.5.4 Format du paquet RTP

Dans ce qui suit, nous décrivons tout les champs qui interviennent dans la synchronisation (désencapsulation sans ambiguïté) des flux agrégés dans un flux RTP4mux (voir Figure 1-8).

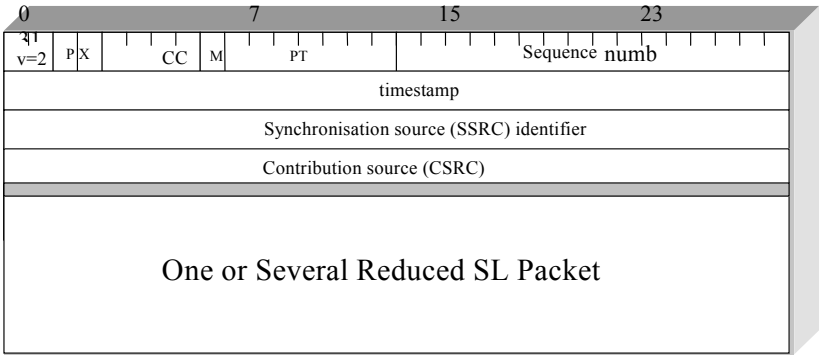


Figure 1-8: Structure d'un paquet RTP.

Timestamp : est dérivé de l'horloge de référence de l'émetteur. Si ce champ n'est pas connu, la valeur du « *Timestamp* » sera égale à la valeur de l'horloge locale du système. La résolution de

l'horloge est par défaut égale à 90 Khz, sinon elle doit être spécifiée durant la phase d'initialisation de la session (exemple par SDP).

SSRC : ce champ est utilisé comme défini dans le protocole RTP RFC 3550 [110], il sera en particulier utile, afin d'identifier les participants actifs dans la session, pour des sessions de communication multicast et bidirectionnelle.

M : c'est un bit qui est tout le temps mis à 1 sauf quand la charge utile contient un ou plusieurs fragments d'AU. Dans le cas où c'est le dernier fragment d'une AU, ce bit est également mis à 1.

Le reste des champs, de l'entête RTP, seront utilisés comme spécifié dans le protocole RTP RFC 3550 [110].

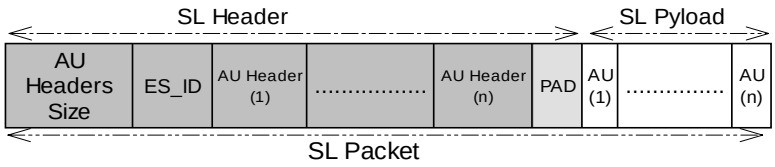


Figure 1-9: Structure de Paquet SL Réduit.

Dans l'entête du paquet SL nous définissons les champs suivants (voir Figure 1-9):

Au Header Size : c'est un champ sur 2 octets utilisé pour indiquer la taille de l'entête SL sans les bits de bourrage. Ceci permet une décapsulation des AUs, du paquet SL, sans ambiguïté.

ES_ID : c'est un champ, qui est également sur 2 octets, qui permet d'indiquer l'identificateur du flux élémentaire auquel les AUs du paquet SL appartiennent. Ce champ est commun à toutes les unités d'accès concaténées dans la charge utile du paquet SL. Ceci permet de réduire l'overhead en évitant le mapping du ES_ID dans chaque AU Header.

PAD : c'est une suite de bits de bourrage (plus de 7 bits mis à zéro) pour aligner l'entête SL sur un nombre entier d'octets. D'où un accès facile aux AUs concaténées.

Il existe autant d'entêtes d'AU concaténés dans l'entête du paquet SL que de AU dans la charge utile du paquet SL. Les AUs sont associées aux entêtes d'AU selon leur ordre d'apparition.

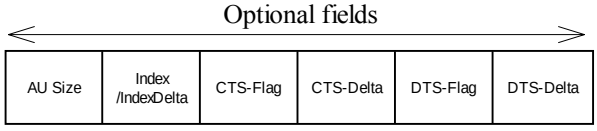


Figure 1-10: Entête de chaque Unité d'Accès (AU).

Nous définissons dans AU Header (voir Figure 1-10) Les champs de l'entête d'AU sont les suivants :

AU Size : utilisé pour indiquer la taille en octets de l'AU associée dans le payload du paquet SL.

Index / IndexDelta : utilisé pour indiquer le numéro de séquence de l'AU associée. Le premier entête d'AU, du paquet SL, contient le champ Index tandis que les entêtes d'AU restants contiennent le champ IndexDelta qui est codé en différentiel par rapport au numéro de série de l'AU précédente (Index ou IndexDelta précédent).

CTS-Flag : utilisé pour signaler la présence ou pas du champ CTS-Delta.

CTS-Delta : codé en différentiel par rapport au champ Timestamp de l'entête RTP et indique l'instant de composition de l'AU associée.

DTS-Flag : utilisé pour signaler la présence ou pas du champ DTS-Delta.

DTS-Delta : codé en différentiel par rapport au champ CTS-Delta du même entête d'AU et indique l'instant pour le décodage de l'AU associée.

Tout les champs de l'entête de l'AU sont configurables, et peuvent exister ou pas, en prenant des tailles variables selon la nature et les besoins du flux élémentaire à acheminer. Cette configuration est faite, à l'initiation de la session MPEG-4, grâce au message SDP à travers lequel les informations, sur la taille et présence des champs du payload RTP, sont véhiculées.

1.1.3 Contrôle de Débit Vidéo

La majorité des applications multimédia utilise la pile protocolaire RTP au dessus de UDP/IP. Cependant, le protocole UDP n'offre aucun mécanisme de contrôle de congestion et est inéquitable envers d'autres trafics. Le trafic Internet est dominé par des flux TCP, ce dernier utilise plusieurs mécanismes de contrôle de congestion tel que : *AIMD* (Additive Increase and Multiplicative Descrease), s*low start, congestion avoidance,* etc. De ces faits, il est nécessaire que UDP réalise aussi un contrôle de congestion [68] ou tout simplement être compatible avec TCP (TCP-Friendly)

1.1.3.1 Algorithme de Contrôle de Débit

L'idée d'un transport TCP-Friendly est d'émuler le fonctionnement de TCP sans toute fois le dupliquer. Plusieurs algorithmes TCP-Friendly ont été développés dans le cadre du transport de la vidéo sur IP. L'IETF a récemment proposé l'algorithme TFRC [84] qui semble être le meilleur protocole de réalisation d'équité de trafic entre plusieurs flux. Notre système de streaming vidéo utilise le module TFRC. Il opère comme suit :

- Le client mesure les pertes rencontrées et retourne ces informations au serveur. Cela est possible grâce au rapport de contrôle RTCP. Puisque chaque paquet RTP contient dans son entête des informations telle que le *timestamp* et le numéro de séquence, il est donc possible de calculer les valeurs du taux de pertes et du RTT (*Round-Trip-Time*)
- Les valeurs du taux de perte et du RTT sont utilisées par le module TFRC pour calculer le débit autorisé.
- Le serveur ajuste son débit en fonction du débit calculé en ajoutant/supprimant des objets dans la scène en fonction de leur importance.

Le débit calculé est obtenu par la formule suivante :

$$R_{TCP} \cong \frac{s}{RTT\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{3bp}{8}})p(1+32p^2)}$$

La valeur R_{TCP} représente le débit autorisé pour la transmission, s est la taille d'un paquet, RTT est la valeur du temps aller-retour, p le taux de perte, t_{RTO} est le temps de retransmission et b est le nombre de paquets acquitter par un seul paquet d'acquittement.

Soit E, un ensemble d'objets MPEG-4 triés par rapport à leur importance dans la scène par exemple en utilisant le modèle de classification présenté dans la section 1.1.1.

$E = \{L_{1,1}, L_{1,2}\ldots L_{1,m1}, L_{2,1}, L_{2,2}\ldots L_{2,m2}, \ldots, L_{n,1}, L_{n,2}\ldots L_{n,mn}\}$ qui est noté

$E = \{e_1, e_2, \ldots, e_W\}$ avec $w = |E| = \sum_{j=1}^{n} m_j$

Le serveur vidéo ajoute un objet audiovisuel lorsque le débit estimé R_{TCP} excède le débit actuel du serveur. Supposons que le serveur est entrain d'envoyer k entités (objet ou couche audiovisuel) à l'instant t_i, nous nous intéressons a ce qui se passera à l'instant t_{i+1}. Le serveur doit obéir à la règle suivante :

$$\sum_{j=1}^{k+1} R_{i+1}(e_j) \leq R_{TCP}$$

Au niveau du client, les nouveaux objets reçus seront bufférisés et synchronisés pour être joués.

De la même manière, si le débit estimé R_{TCP} indique que le serveur consomme plus qu'il en faut, alors il doit réduire son débit en arrêtant le streaming des objets les moins importants, et ceci tant que la règle suivante est vrai :

$$\sum_{j=1}^{k} R_{i+1}(e_j) > R_{TCP} \qquad \text{(Eq. 11)}$$

1.1.3.2 Gestion de la Stabilité Vidéo

Le module TFRC calcule le débit autorisé du serveur pour chaque RTT. De ce fait, ajouter et supprimer des objets audiovisuels à chaque RTT peut produire des oscillations indésirables et une qualité faible au niveau du player vidéo. Pour prévenir ce comportement, plusieurs mécanismes ont été pris en considération.

Premièrement, une implémentation optimale de la moyenne exponentielle EWMA (Exponentially Weighted Moving Average) permet de détecter les situations hors contrôle de manière rapide. En effet, EWMA est utilisé pour répondre dynamiquement au changement de valeur du RTT et des taux de perte et régularise ces valeurs pour traduire la réalité. Cela permet de régulariser le débit afin qu'il ne change pas de manière agressive.

Le second mécanisme opère en collaboration avec le mécanisme précédent. Il permet d'assurer la synchronisation des images vidéo envoyées par un lissage au niveau du **GOV** (Group of video object plane). Le serveur vidéo est autorisé à changer son débit seulement au début de chaque GOV. La Figure 1-11 illustre le principe de lissage et cd transmission de la vidéo pour gérer la stabilité.

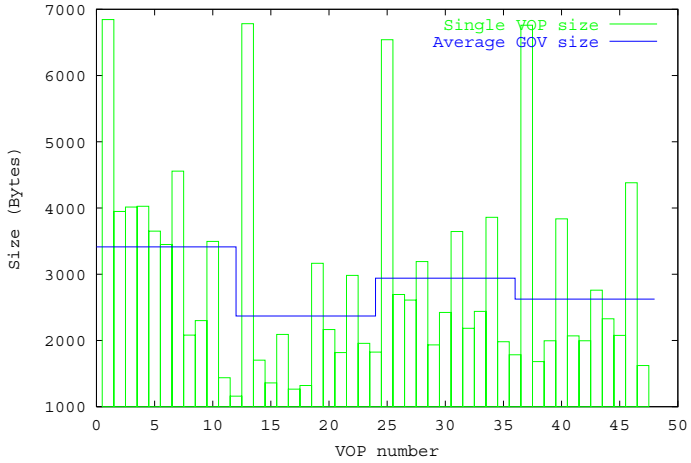


Figure 1-11: Gestion de la stabilité de la video par GOP

1.2 Marquage Dynamique des Flux Vidéo

Les récentes recherches sur le support de la qualité de service sur Internet a conduit à deux approches distinctes : l'architecture Intserv accompagnée de son protocole de signalisation RSVP (*Resource Reservation Protocol*) [136] et l'architecture Diffserv [126]. Contrairement à la gestion de flux par RSVP, les réseaux Diffserv séparent le trafic en plusieurs classes de paquets. La classe à laquelle un paquet est agrégé est marquée dans un champ de l'entête IP, le DSCP (*DiffServ Code Point*). Quand un paquet arrive sur un routeur Diffserv, celui-ci effectue le relayage du paquet en adoptant un comportement qui est fixé par la classe du paquet, appelé son PHB (*Per-Hop Behavior*).

Deux schémas de PHB ont été proposés pour fournir des traitements différenciés sur les paquets :

Le PHB EF (*Expedited Forwarding*) [128]: ce type de comportement est destiné aux applications qui ont besoin de garanties sur la bande passante, avec un délai d'acheminement des paquets réduit, peu d'effets de gigue et peu de pertes.

Le PHB AF (*Assured Forwarding*) [128]: ce type de comportement est destiné aux applications qui ont besoin d'un service meilleur que le *best-effort*. Il distingue trois niveaux de priorité de perte et quatre niveaux de priorité temporelle à travers les classes de services AF1, AF2, AF3 et AF4. Les paquets de plus faible priorité sont écartés en premier en cas de congestion du réseau.

Diffserv ne fournit pas de garantie absolue sur la qualité du service rendu mais son mécanisme résiste au facteur d'échelle. C'est essentiel dans l'Internet. Avec la gamme des services proposés, Diffserv permet de prendre en compte les besoins de qualité de service des applications audiovisuelles. Reste à définir objectivement comment le trafic va être classifié car cela conditionnera en grande partie les performances. Les fonctionnalités nécessaires aux extrémités d'un domaine Diffserv pour classifier le trafic et affecter les ressources réseau sont définies dans une recommandation de l'IETF. Plusieurs algorithmes de marquage au niveau des routeurs de bordure sont proposés, parmi lesquels **TSWTCM** (*Time Sliding Window Three Color Marker*) [130] et **TRTCM** (*Two Rate Three Color Marker*) [131]. Mais ce marquage des paquets est effectué de manière systématique, sans aucune connaissance du type de flux, et en conséquence peut se relever inadapté.

Pour ces raisons, nous nous intéressons aux interactions de QoS entre les applications vidéo MPEG-4 et les réseaux IP DiffServ. Les paquets vidéo sont marqués en fonction de leur importance. Ainsi, un paquet issu d'un objet MPEG-4 important sera marqué avec une priorité de perte faible et un paquet jugé non important sera marqué avec une priorité de perte importante. La Figure 1-12 présente un exemple simple de marquage de paquet vidéo pour un objet utilisant le codage en couche. Sachant que la couche de base est la couche la plus importante, elle sera marquée avec une priorité de perte la plus faible possible. Les couches d'améliorations seront marquées avec les priorité de perte moyenne et importante

```
if stream is "audio stream" then (application of property 2)
      if  coder rate is "low rate"
            then DSCP=AF Low Drop Prec
            //example AF11
      if coder rate is "medium rate"
            then DSCP=AF Medium Drop Prec
            //example AF12
      if coder rate is "high rate"
            then DSCP=AF  High Drop Prec
            //example AF13

if stream is "video stream" (application of property 3)
      if  "base layer video stream"              (level 1 = mimimum QoS)
            then DSCP = AF low Drop Prec
            //example AF21
      if  " enhanced layer video stream 1"       (level 2 = medium QoS)
            then DSCP = AF Medium Drop Prec
            //example AF22
      if  "enhanced layer video stream 2"        (level 3 =  maximum  QoS)
            then DSCP = AF Medium Drop Prec
            //example AF23

if stream is "objects descriptor, scene descriptor" (application of property 4 )
            then DSCP = EF
            //descriptions streams  are significant, no loss
            //it's necessary that these streams will be available
            //as soon as possible in MPEG-4 player to interpret
            //correctly the scene
```

Figure 1-12: Un exemple simple d'algorithme de marquage IP Diffserv

Notre contribution est ciblée sur deux points : d'abord, un protocole d'encapsulation pour les paquets MPEG-4 à travers RTP/IP est mis en place. Et ensuite, un mécanisme de marquage pour les routeurs Diffserv est proposé. L'intégration de ces deux composants est essentielle pour fournir une QoS de bout en bout. Les performances sont évaluées à l'aide d'un réseau de routeurs DiffServ. Ce réseau est constitué de PC fonctionnant sous Linux avec le noyau 2.4. Ce dernier permet d'implémenter DiffServ de manière logicielle.

1.3 Passerelle de Signalisation SIP/MPEG-4

Internet est confronté au développement rapide d'applications et de services multimédia tels que la vidéoconférence et la téléphonie. Cet essor s'explique par l'utilisation de standards et de normes bien unifiés. Actuellement, plusieurs normes dominent le marché de la vidéoconférence et de la voix sur IP. L'IUT-T a proposé le standard H.323 [142] [143], permettant de mettre en place tous les mécanismes nécessaires à l'établissement d'une session de vidéoconférence. L'IETF a par ailleurs proposé le protocole SIP (Session Initiation Protocol) [140] avec SDP (Session Description Protocol) [141]) comme un moyen simple et efficace d'initier et de contrôler des sessions multimédia entre terminaux. Finalement, l'ISO avec son standard MPEG-4 DMIF (Delivery Multimedia Integration Framework) [144] offre un moyen transparent aux applications pour

initialiser toute forme de communications multimédia sur des réseaux IP. Dans un contexte de terminaux et de serveurs multimédia implémentant ces différentes normes, un problème majeur d'interaction et de contrôle de services multimédia apparaît. L'objectif de ce travail est de proposer une solution pour permettre l'interopérabilité des systèmes et une continuité de service de vidéoconférence entre clients et serveurs MPEG-4 et SIP. Cette solution passe par la définition et l'implémentation d'une passerelle de signalisation MPEG4 DMIF et SIP.

1.3.1 Architecture de la passerelle de signalisation

Dans ce qui suit, nous proposons une architecture fonctionnelle d'une passerelle permettant d'assurer l'interconnexion entre les standards SIP et MPEG-4 DMIF. Cette entité est appelée **DMIF-SIP IWF** (DMIF-SIP Interworking Function). La Figure 1-13 montre un domaine utilisant la signalisation SIP interconnecté à un domaine utilisant la signalisation MPEG-4 DMIF.

La passerelle d'interconnexion est composée de deux entités : une entité logique pour traduire les messages de signalisation du domaine SIP au domaine DMIF, qui est appelé entité **SIP2DMIF IWF** et une partie pour traduire les messages du domaine DMIF au domaine SIP, qui est appelé entité **DMIF2SIP IWF**.

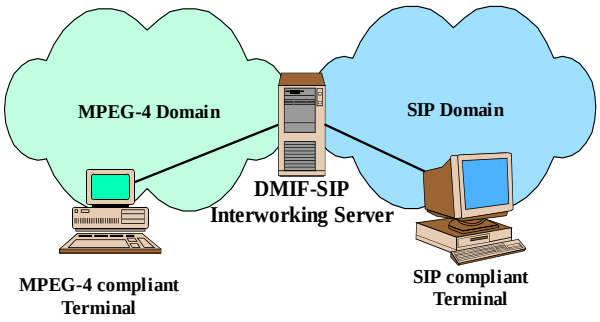


Figure 1-13: Interconnexion entre MPEG-4 DMIF et SIP

1.3.1.1 L'entité SIP2DMIF

L'entité SIP2DMIF IWF est une partie du serveur d'interconnexion DMIF-SIP IWF. L'entité permet de commencer ou de terminer un signal SIP/DMIF depuis et vers un terminal SIP/DMIF. L'entité SIP2DMIF permet à un terminal SIP (SIP User Agent Client « UAC ») d'appeler un terminal MPEG-4 DMIF. Le terminal SIP dialogue avec la passerelle DMIF-SIP IWF en utilisant la spécification SIP.

Quand l'entité SIP2DMIF IWF reçoit un message « INVITE » de la part de terminal SIP, elle envoie alors un message de signalisation DMIF (DS_Message) à l'interface DNI (DMIF Network Interface) du terminal MPEG-4. Une fois que la session et le service sont établie, entité SIP2DMIF envoie un message d'acquittement de type « 200 OK message » au terminal SIP. Le signale d'acquittement confirme au terminal MPEG-4 que la connexion est bien établie. La Figure 1-14 illustre les messages échangés entre le terminal SIP, DMIF-SIP IWF et l'interface DNI du terminal MPEG-4.

Telles que illustrés par la Figure 1-14, les différentes étapes nécessaires à l'établissement d'une session entre un terminal SIP (client A) et un terminal DMIF (client B) sont:

Etape 1 : L'utilisateur SIP envoie une requête d'invitation au client B (message INVITE). Ce message invite le client B à participer à une vidéoconférence ou un appel téléphonique, le message « INVITE » contient les paramètres d'appel suivants :

1. Un identificateur (adresse mail, numéro de téléphone) du client B. Ces informations sont insérées dans le champ « Request-URI » sous forme d'un URL SIP.
2. Un identifiant unique est assigné à cet appel est sera inséré dans le champ « Call-ID ».
3. Un numéro de transaction sera inséré dans le champ « CSeq ».
4. Les capacités de gestion de media du terminal SIP sont spécifiées via le protocole SDP.

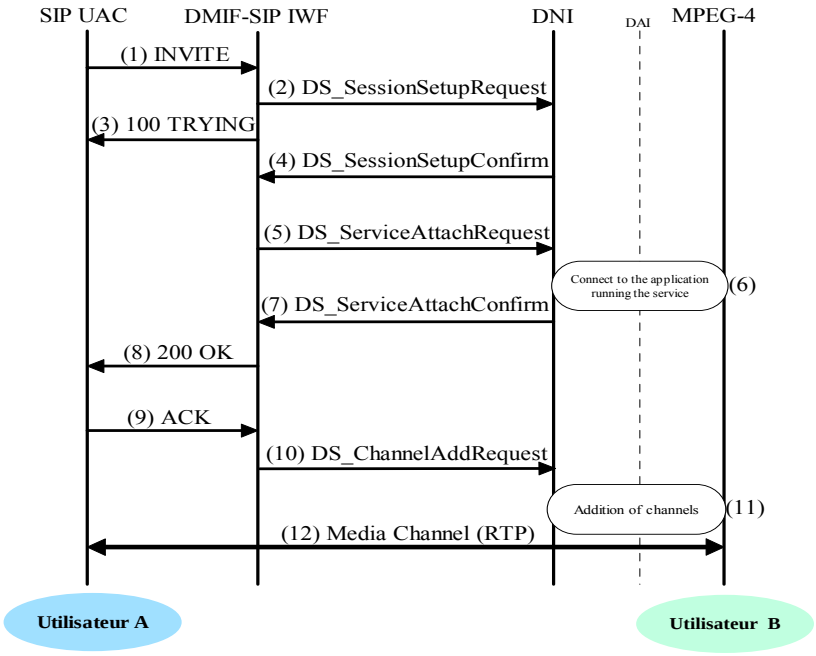


Figure 1-14 : L'entité SIP2DMIF

Etape 2 : SIP2DMIF IWF reçoit le message « INVITE » du client A, il transforme l'identifiant du client B sous forme d'un URL DMIF, celui-ci est obtenu lorsque le terminal MPEG-4 DMIF sera allumé. Nous supposons que l'adresse du client B est bien traduite dans un URL DMIF, cette traduction est présentée dans la prochaine sous-section. L'unité d'interconnexion SIP2DMIF passe alors le message « DS_SessionSetupRequest » au terminal DMIF dans le but d'activer une session réseau. Ce message contient la description du contenu multimédia que le client A est capable de gérer. La description du contenu multimédia d'un terminal MPEG-4, quand à elle, est définie dans [175].

Etape 3 : SIP2DMIF IWF envoie un message « 100 TRYING » au client A.

Etape 4 : Quand l'unité d'interconnexion reçoit le message « DS_Session-SetupConfirm », les deux terminaux (SIP UAC et MPEG-4 DMIF) ont connaissance l'un de l'autre. Le message reçu

contient aussi la description du contenu multimédia commun entre les deux terminaux dans l'ordre de préférence.

Etape 5: SIP2DMIF IWF envoie le message « DS_ServiceAttachRequest » contenant l'URL DMIF.

Etape 6: L'interface DNI informe le terminal MPEG-4 que le client A veut établir une session de vidéoconférence ou de téléphonie avec le client B.

Etape 7 : SIP2DMIF IWF reçoit le message « DS_ServiceAttachConfirm » indiquant que le client B est capable de rejoindre la vidéoconférence.

Etape 8 : SIP2DMIF IWF envoie le message « 200 OK » en retour au client A. Ce message informe l'utilisateur A que la connexion est établie, le message contient aussi l'intersection des descripteurs du contenu multimédia des deux terminaux. S'il n'y a aucun descripteur en commun le message « 400 Bas Request » avec une réponse « 304 warning » dans l'entête est transmis.

Etape 9 : Le client A envoie un message « ACK » à l'entité SIP2DMIF.

Etape 10 : En recevant l'acquittement, les canaux de média doivent être ouverts. Pour cela, le message « DS_ChannelAddRequest » est envoyé par DNI du terminal MPEG-4.

Etape 11 : Le terminal MPEG-4 confirme la création de ces canaux.

Etape 12 : Quand les canaux de transport RTP sont ouverts entre les clients A et B, les flux de données multimédia peuvent maintenant être véhiculé et le service de vidéoconférence débute.

1.3.1.2 L'entité DMIF2SIP

La seconde entité logique DMIF2SIP est une composante du serveur d'interconnexion DMIF-SIP IWF. L'entité permet d'initier, contrôler et clore une session de vidéoconférence hybride DMIF/SIP depuis et vers un terminal DMIF/SIP. L'entité DMIF2SIP permet à un terminal MPEG-4 DMIF d'appeler un terminal SIP. Les étapes nécessaires pour établir une connexion multimédia entre les deux terminaux sont illustrées dans la Figure 1-15 et sont expliquées ci-dessous :

Etape 1 : Le terminal MPEG-4 invoque la primitive DA_ServiceAttach() afin d'établir une session en indiquant des informations concernant l'autre partie (client B) comme son adresse mail ou son numéro de téléphone.

Etape 2 : L'interface DNI envoie le message « DS_SessionSetupRequest » à DMIF2SIP pour demander l'ouverture de la session avec le terminal SIP.

Etape 3 : Une fois cette demande d'ouverture de session reçue, DMIF2SIP envoie un message « INVITE » au terminal SIP pour l'inviter à participer à la vidéoconférence. Le message « INVITE » contient essentiellement :

1. L'adresse de l'utilisateur B qui sera insérée dans le champ « Request-URI » sous forme d'un URL SIP
2. L'adresse de l'utilisateur A est insérée dans le champ « From »

3. DMIF2SIP vérifie si son propre adresse est contenue dans le champ « VIA » (pour prévenir les boucles), autrement, il insère sont adresse dans ce champ.
4. DMIF2SIP crée un identifiant qui sera assigné à l'appel, celui-ci sera inséré dans le champ « Call-ID »
5. Le numéro de transaction pour chaque appel est identifié par le champ « Cseq »
6. Les capacités de gestion du media du terminal DMIF seront transformées sous forme de message SDP.

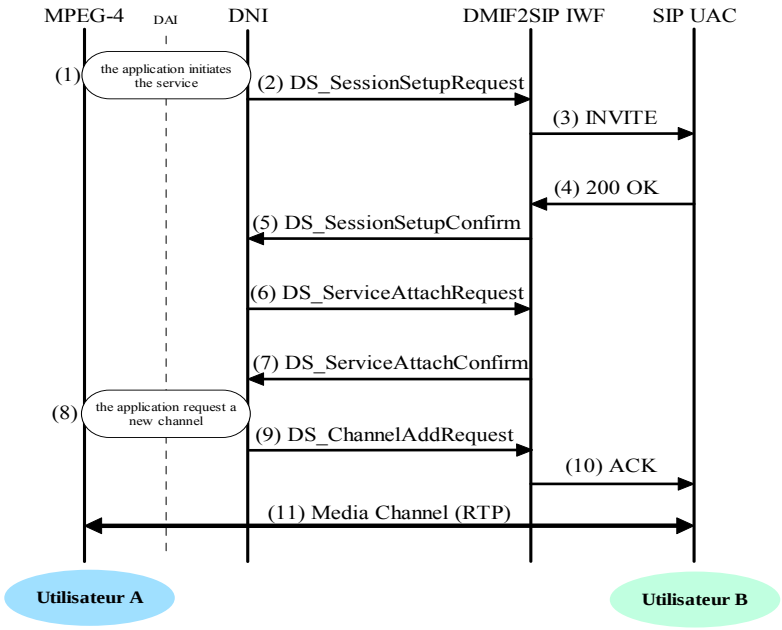


Figure 1-15 : L'entité DMIF2SIP

Etape 4 : Apres avoir envoyé les messages TRYING et RINGING, le client B (Terminal SIP) envoie le message « 200 OK » qui informe la passerelle que la connexion est établie. Si le terminal SIP supporte les médias véhiculés dans le message « INVITE », il dresse l'intersection de ses propres capacités avec les capacités du client A dans le message de réponse « 200 OK ». Si le terminal SIP ne supporte aucun média, il répond avec un message de réponse « 400 Bad Request » avec un champ « 304 Warning » dans son entête.

Etape 5 : Quand le message « 200 OK » est reçu, DMIF2SIP envoie le message « DS_SessionSetupConfirm » en retour au terminal MPEG-4 DMIF et spécifie l'intersection des capacités de gestion de media que le terminal SIP a engendré.

Etape 6 : Le terminal MPEG-4 DMIF est capable maintenant d'assigner une signification locale à la session du réseau et il doit ouvrir un service réseau à l'intérieur de la session. Cette tâche est réalisée par le message « DS_ServiceAttachRequest »

Etape 7 : DMIF2SIP reçoit alors le message « DS_ServiceAttachRequest » qui identifiera le service du coté de l'unité DMIF2SIP.

Etape 8 : Le terminal MPEG-4 DMIF est prêt à demander l'ouverture de nouveaux canaux de media.

Etape 9 : Le message « DS_ChannelAddRequest » envoyé pas le terminal MPEG-4 DMIF demande l'ouverture d'un canal de transport multimédia.

Etape 10 : DMIF2SIP envoie un message d'acquittement « ACK » au terminal SIP (client B) confirmant que les deux parties sont capables d'échanger (envoyer et recevoir) les médias négociés.

Etape 11 : Les canaux de media sont ouverts, le dialogue entre le terminal MPEG-4 DMIF et le terminal SIP commence.

1.3.1.3 Gestion des Capacités des Terminaux

La capacité d'un terminal à gérer les différents types de media est associée aux algorithmes de traitement audio, vidéo et données que le terminal supporte. Cela peut être influencé par des contraintes extérieures. Par exemple, du à la limitation de la bande passante, un terminal peut indiquer qu'il est disposé à échanger uniquement de l'audio G.711, soit de l'audio G.723.1 associé à de la vidéo au format H.261.

L'intersection des capacités de gestion de media entre deux terminaux doit être non vide pour permettre l'établissement d'une connexion multimédia entre eux, et éventuellement sélectionner la pile protocolaire à utiliser. Dans le cas de SIP et DMIF, cette étape est réalisée au moment de l'établissement de la session. SIP définit SDP [141] comme un protocole de description de capacité de gestion de media, quant à MPEG-4, cette fonction est décrite dans la recommandation DSM-CC ISO/IEC 138148-6 [175].

Quand le terminal SIP initialise la session avec le terminal DMIF, il envoie le message « INVITE ». La description de capacité est véhiculée dans ce message d'invitation à travers des messages SDP. Tandis que lorsque le terminal DMIF ouvre la session, il envoie le message « *DS_SessionSetupRequest »*, dans ce cas la description de capacité est véhiculée dans le champ « *comptabilityDescriptor »* appartenant au message précédent.

L'algorithme employé pour déterminer le dénominateur commun entre les capacités des deux terminaux est décrit dans [177] et est présenté ici à titre indicatif.

1. Mettre l'ensemble du résultat C à vide (C contient à la fin de l'algorithme le maximum d'intersection des capacités des deux terminaux).
2. Pour chaque paire de description de capacité (d1, d2) où d1 tiré de l'ensemble C1 et d2 tiré de l'ensemble C2 (avec C1 : la description de capacité du terminal 1 et C2 la description de capacité du terminal 2) faire les permutations des ensembles alternatifs s1 et s2. Pour chaque permutation, où s1 pris de d1 et s2 pris de d2, faire l'intersection de s1 et s2 (noté : s=s1 ^ s2) et ajouter « s » à C.
3. Supprimer les entrées dupliquées de C.

1.3.1.4 Conversion entre L'adresse SIP et L'URL DMIF

1.3.1.4.1 Définition de l'URL MPEG-4 DMIF

L'URL DMIF permet à la couche DMIF d'établir une session avec le terminal DMIF distant afin de localiser un service (service de VoD, service de téléphonie, etc.). L'URL DMIF peut être

utilisée optionnellement pour localiser des sources de données (généralement des flux élémentaires MPEG-4). L'URL DMIF suit la syntaxe générique des URL Internet qui est définie dans le RFC1738 [182].

La syntaxe d'un URL DMIF pour un réseaux IP est :

xdmif://<user>:<password>@<target dmif>:<dmif port>/<service-entity-path> or <stream-source-path>

1.3.1.4.2 Format d'adresse de l'IETF SIP

Pour SIP, une adresse peut être soit un URL ou un URI. La grammaire BNF (Backus-Naur Form) suivante donne la définition des adresses SIP.

SIP-Address = (name-addr | addr-spec)

name-addr = [display-name] "<" addr-spec ">"

addr-spec = SIP-URL

SIP-URL = "sip:" [userinfo "@"] hostport url parameters [headers]

userinfo = user [":" password]

hostport = host [":" port]

host = hostname | IPv4address

url-parameters = *(";" url-parameter)

url-parameter = user-param | ...

1.3.1.4.3 Conversion entre adresses SIP et URL DMIF

Une adresse SIP peut être convertie en URL DMIF facilement. L'URL SIP est copié mot à mot en URL DMIF sans aucune information supplémentaire.

1.4 Conclusion

Dans cette thèse, nous avons exploré un transport adaptatif des paquets vidéo sur les réseaux IP. Nous avons proposé un système intégré qui est constitué de plusieurs blocs fonctionnels tel que indiqué dans la Figure 1-16.

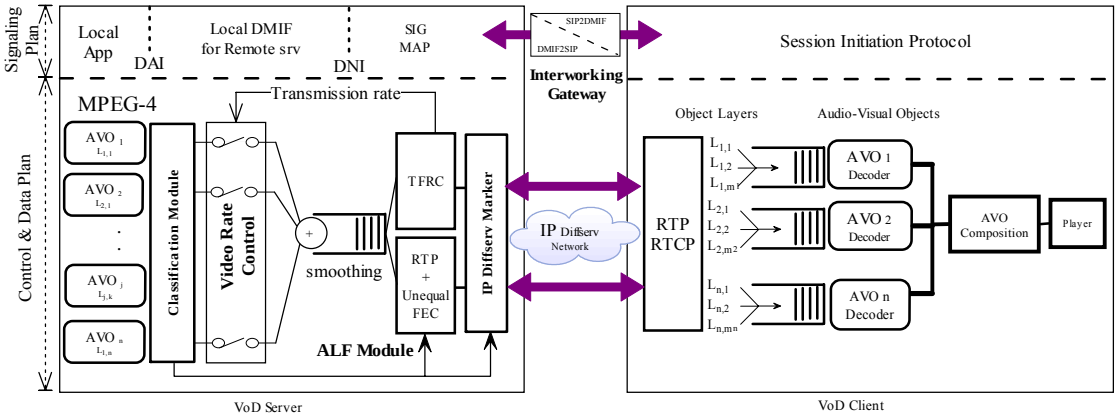


Figure 1-16: Diagramme général des différents blocs de notre système de diffusion vidéo

Ce système est composé d'un serveur et d'un client vidéo. Le serveur transmis les paquets vidéo par le biais d'un réseau IP diffserv. Le module de classification des objets vidéo permet de distinguer les objets importants et les moins importants dans la scène. En fonction de la priorité relative de l'objet, le module de contrôle de débit régularise son débit en arrêtant la transmission des objets les moins important en cas de détection de congestion. Le module de contrôle d'erreur utilise les codes correcteurs dynamique FEC (Forwarding Error Correction). Les objets vidéo les plus important seront mieux protégés contre les erreurs par rapport aux objets les moins importants. Au niveau du player vidéo, les objets reçus seront bufférisés et synchronisés avec la scène et joués au bon moment.

Chapter 2

2 Introduction

2.1 Motivation

The rollout of Internet opens up a new frontier for the digital video broadcasting industry. Two worlds that were barely connected are on the verge of being merged: namely, real-time video and Internet. Thanks to the transition of the video medium from analog to digital, and the rise of powerful video compression techniques, such as H.264 and MPEG-4, it is now possible to combine video, audio, and data within the same signal and transmit it over an multi-service packet switching network infrastructure, such as the Internet.

These advances will lead to the emergence of new powerful multimedia applications, with limitless possibilities of great commercial potential. For example, computers can be turned into traditional TV receivers and the digital set-top boxes can host applications such as interactive TV, e-commerce, and customized programming.

Using the Internetworking Protocol (IP) as a medium for the transmission of real-time multimedia applications is a challenging problem. These applications require real-time performance guarantee in term of bounded transfer delay and jitter, low packet loss, and bandwidth guarantee. IP is a best effort packet switching technology. It does not guarantee itself any Quality of Service (QoS) such as transfer delay, jitter, loss, and bandwidth. These performance metrics are affected by many parameters, some of them are related to the end systems (e.g. video server load) and others are related to the network (e.g. link capacity, router buffer, etc.). The provision of quality of service for IP has been the subject of significant research and development efforts recently. Two approaches have drained the attention of the IETF: namely the Integrated Service (Intserv) and the Differentiated Service (Diffserv) architectures. The key difference between Intserv and Diffserv is that while Intserv provides end-to-end deterministic QoS service on a per-flow basis, Diffserv is intended to provide better scalability through flow aggregation, class differentiation and statistical QoS guarantee over a long timescale.

In this thesis, we study the problems involved in the transport of real-time video streams over IP Differentiated Service networks. We fellow a cross-layer approach for resolving some of the critical issues on delivering packet video data over IP networks with satisfactory quality of service. While, current and past works on this topic respect the protocol layer isolation paradigm, the key idea behind our work is to break this limitation and to rather inject content-level semantic and service-level requirements within the proposed IP video transport mechanisms and protocols.

2.2 wxContribution

Video sequences are typically compressed according to the emerging MPEG-4 multimedia framework to achieve bandwidth efficiency and content-based interactivity. The original characteristic of MPEG-4 is to provide an integrated object-oriented representation and coding of natural and synthetic audio-visual content for manipulating and transporting over a broad range of communication infrastructures.

We leverage the characteristics of MPEG-4 and IP differentiated service frameworks, to propose an efficient and adaptive cross layer video delivery system over IP Diffserv that is composed of:

- **A system-level audio-visual object classification model**: That implements an efficient transmission of object-based MPEG-4 video over IP networks with QoS management capabilities, the MPEG-4 Audio Visual Objects (AVOs) are classified based on application-level QoS criteria and AVOs semantic descriptors according to objects descriptors and MPEG-7 meta-data. This lead to a relative priority score (RPS) for each video packet. The MPEG-4 AVOs (e.g. video elementary streams, video layers, video plans, audio, …) requiring the same QoS performance (same RPS) from the network are automatically classified and multiplexed within one of the available IP Diffserv Per Hop Behaviors (PHB). Consequently, we propose to extend the MPEG-4 System architecture with a new "Media QoS Classification Layer". This layer provides automatic and accurate mapping between MPEG-4 application-level QoS metrics, data relevancy and underlying QoS-capable network services. This "Media QoS Classification Layer" makes use of a neural network classification model that is configurable to match various object-encoded multimedia applications and network services.

- **A robust and adaptive application level framing protocol with data multiplexing and unequal error protection**: The Application Level Framing (ALF) prepares the video stream to be transmitted over the selected transport layer. In case of IP network, RTP protocol is the most suitable protocol for ALF. The Audio/Video Transport Working Group (AVT) was formed to specify a protocol for unicast and multicast real-time transmission over IP. RTP has to be customized to the transported media. Therefore, we propose a new RTP payload for MPEG-4 media that provides intelligent data fragmentation for error resilience, efficient stream multiplexing for bandwidth saving and unequal error protection for error-prone environment (i.e. wireless IP).

- **A fine grained TCP-Friendly Video Rate Adaptation Algorithm**: video servers perform video source rate adaptation to tackle with varying network resource

conditions. Based on end-to-end feedback measurements conveyed by Real Time Transport Control Protocol (RTCP) reports, video source can estimate the allowed transmission rate and conform to it. The originality of this contribution is to finely adjust the source video rate according to the relative priority score of the AVOs calculated by the Classification Model.

- **A dynamic network-level packet video marking algorithm for IP Differentiated Service**: We demonstrate that the interaction and cooperation between application-level (i.e. data classification model) and network-level (i.e. packet marking) mechanisms can improve the efficiency of the overall video delivery system. In our proposal, the interaction is performed through a dynamic mapping between the application-level RPS (relative priority score) of each video packet – calculated form the classification model - and the network-level differentiated forwarding mechanism. This Diffserv Video Packet Matching/Marking Algorithm (DVMA) can be located at the video server side or simply deported to the access edge router. The marking algorithm can exploit the flexibility of multimedia traffic and process network level parameters to adapt the traffic according to the current state of the network. The latter is determined based on reports sent by bandwidth monitors installed on each node of a Diffserv domain. The bandwidth monitors interact with a policy server which, depending on the network state, decides the policy(ies) that should be enforced by the Diffserv network. The implementation of the selected policies typically leads to accepting, remarking, or dropping the MPEG-4 video traffic entering the network.

- **A SIP/DMIF Multimedia Interworking Signaling Gateway:** To permit a wide share of MPEG-4 multimedia content between heterogeneous wired and wireless IP terminals, we have also designed and implemented an Interworking Signaling Gateway that supports both IETF Session Initiation Protocol (SIP) and ISO MPEG-4 Delivery Multimedia Integrated Framework (DMIF) Session Control Signaling Protocol. This multimedia signaling gateway performs various translation functions for transparent establishment and control of multimedia sessions across IP networking environment, including, session protocol conversion, service gateway conversion and address translation.

2.3 Dissertation Overview

This dissertation is organized as follows:

Chapter 3 presents a state of the art on issues and solutions for supporting packet video applications over IP. This chapter starts with a categorization of current packet video applications. Then, we present the QoS requirements and challenges for such demanding applications.

Chapter 4 describes the components of our adaptive cross-layer packet video transport system over IP Diffserv. At the system-level, we present the classification model that affects a relative priority score to different media type based on AVO classification and prioritization. At the transport-level, we define a robust application-level framing protocol with video stream multiplexing, unequal error protection and TCP-Friendly video rate control features.

Chapter 5 presents our dynamic packet video marking algorithm for IP networks with differentiated services. The proposed marking algorithm implements a flow matching between the relative priority score of the video packets and the targeted preconfigured Diffserv Code Points. The marking can be applied dynamically and according to available bandwidth variations and data importance.

Chapter 6 is devoted to the SIP/MPEG-4 multimedia interworking signaling gateway. We address the problems of session protocol conversion, service gateway conversion and address translation.

Chapter 7 concludes this dissertation and addresses some of our future work.

Appendix A presents an overview of the MPEG-4 Standard, in particular the MPEG-4 system architecture and the Audio Visual Object Descriptor Framework.

Chapter 3

3 Related Work

Nowadays, multimedia is becoming indispensable feature on networking environments. Audio and video content become more and more popular on the Internet. Many systems are being designed to carry this media such as video conferencing, video on demand, IP Telephony, Internet TV, etc.

Multimedia networking is defined by a set of hardware and software infrastructure that support multimedia stream on network so that users can communicate in multimedia. Using IP based network as a communication infrastructure is not a trivial task. Many problems rise for delivering multimedia data over IP.

First, multimedia applications require a higher bandwidth compared to a traditional textual applications. A typical uncompressed video movie of 1 minute length, with a temporal and special resolution of 25 images per second, and 320 x 240 pixels respectively requires 330 Mega Bytes and about 46 Mbps(Mega bit per second) bandwidth for real-time transmission. As a consequence, audio-visual compression is vital for transporting such media on networking environments. This compression process leads to new problems that we expose later.

Second, most multimedia applications require a real-time traffic. Audio and video data must be played-back continuously at the rate they are sampled. If the data does not arrive at the expected time the play-back process will stop. Buffering some parts of this data on the receiver can reduce this problem but the latency remains a challenge. For example, with IP telephony, one can tolerate an end-to-end transmission delay of about 250 millisecond. If the latency exceeds the limit, the voice will sound with a poor quality and will encounters echo. In addition to high bandwidth and low latency, network congestion has more serious effects on real-time traffic. If the network is congested, the transfer takes longer to complete and real-time data becomes obsolete and unused if it doesn't arrive at the expected time.

Third, the compression process will transform the traffic characteristic of audio video data stream from CBR (constant Bit Rate) to VBR (Variable Bit Rate). For most multimedia applications, the receiver has a limited buffer. Smoothing the data can reduce the burstiness problem of multimedia traffic. Sending bursty traffic may overflow or underflow the applications buffer. When data arrives too fast, the buffer will overflow and some data will be lost, resulting in poor quality. When data arrives too slowly, the buffer will underflow and the application will starve.

Fourth, using a large scale networking environment such as the Internet with millions of users using the network, a fairness problem in bandwidth share arises.

The possibility of answering this challenge comes from the existing network software and hardware architecture. Developing a robust and reliable multimedia transport layer will fact problems of transmitting multimedia data over the Internet.

3.1 Taxonomy of Packet Video Applications

In this section we present a taxonomy of packet video applications. Different granularity levels are considered to address this classification. There exist very diverse ranges of video communication. Video communication applications may be unicast, multicast, broadcast or anycast. The video may be pre-encoded (stored) or real-time encoded (e.g. videoconferencing applications). The communication channel may be static or dynamic, reserved or not, packet switched or circuit-switched, may support some quality of service or may only provide best effort service.

3.1.1 Unicast, Multicast, Broadcast, and Anycast Video Communications

There are different service models used for sending data from a sender to one or multiple receiver(s). Four main possibilities are discussed here (see Figure 3-1): Unicasting, Broadcasting and Multicasting, Anycasting.

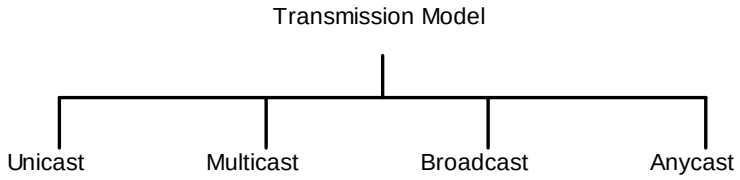


Figure 3-1: Taxonomy of video transmission approaches

- unicasting: called also point-to-point communication, in which data is only sent to one recipient at the same time with several receivers. Unicast wastes bandwidth by sending multiple copies of the data. In unicast, a back channel for the receiver can be used to provide feedback to the sender. In this case, the sender has more knowledge about the receiver and the channel. The sender can use this feedback to adapt dynamically the data being sent to the receiver, and accommodate capacity variations.
- Broadcasting: the data is sent to every terminal in the link even if they are not all concerned by this data. Furthermore the service scope is limited to resources on the Local Area Network. Broadcast is widely used for digital television and radio (e.g. Satellite Broadcast). Broadcast wastes bandwidth by sending the data to the whole network. It can also needlessly slow the performance of client machines because each client must process the broadcasted data whether or not the service is of interest. The main challenge for broadcasting is the scalability problem. Receivers may experience different channel characteristics, and the sender must cope with all the receivers. In this case, feedbacks from the receivers are unmanageable for the sender.

- multicasting: it takes the strengths of both of these previous approaches and try to avoid their weaknesses. This technique is between unicast and broadcast communication. It is similar to broadcasting but the data is sent only to limited identified receivers. There are mainly two different approaches for achieving multicast. Application such as IP video streaming requires sending the same data to multiple receivers. It can be achieved by using multipoint communication in the transport layer. This technique is called simulcast. In this context, sending the same data, using separate point-to-point unicast connections, result in wasted communication resources as shown in Figure 3-2 (a). IP multicast is an example of scalable multicast providing data delivery for communication group as shown in Figure 3-2 (b).
- Anycasting: the anycasting communication paradigm is designed to support server replications to easily select and communicate with the best server, according to some performance or policy criteria, in a group of content-equivalent servers.

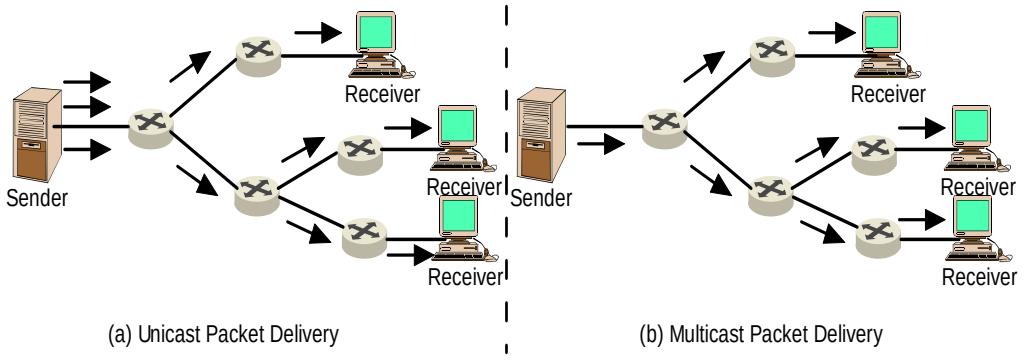


Figure 3-2: Application layer multicast vs. IP multicast

3.1.2 Live vs. Pre-Encoded Video Applications

We can find video in two forms. Video captured by camera devices and encoded in real-time for live communication, and pre-encoded and stored video used for later viewing. There are many applications using real-time video encoding such as videoconferencing, videophone, interactive games, and live broadcasting. However, many applications use pre-encoded and stored video content. The content can be accessed locally or from remote system. A common example of local storage is a DVD-Rom. Video streaming system is also an example of remote access.

The disadvantage of real-time encoding is the time constraint. Encoding video is a time consuming. This can leads to choose fast encoding methods while achieving low compression efficiency. The advantage of pre-encoded video is the ability to encode the video with an efficient codec such as multi-pass codec. However, the main disadvantage of the pre-encoded video is the lack of flexibility. For example, it is difficult for pre-encoded video to adapt to channel condition and clients heterogeneity since it requires real-time transcoding.

3.1.3 Video Downloading vs. Video Streaming

There have been significant research efforts in delivering multimedia content over IP networks. Ideally, multimedia content such as video and audio are streamed from the server to the client in response to a client request. The client plays the incoming multimedia stream in real time as the data is received. Below, we have reviewed the most important technique for delivering multimedia content to users.

Video downloading technique allows delivering video content through a download process. We call this technique video downloading to discern that the downloaded file is a video and not a generic file. The download is achieved using common protocols like FTP and HTTP. Actually many peer-to-peer systems exist to share video between communities, such as Kazaa [1], gnutella [2] , iMesh [3], and eDonkey [4]. The video file is very large and needs to be downloaded entirely before viewing the video. This technique requires patience from the viewer and also reduces flexibility since neither the quality nor the content can be checked before the end of the download. Other constraints are related to the size of the video file. Since a video is very large file, the download could take a long time and a large storage spaces. These are the main constraints of the video download approach.

Video streaming technique enables simultaneous delivering and playback of the video. This technique overcomes the problems of video delivering via file download. It provides considerable additional features. The principle is as follows. The video is divided into units. Each unit is transmitted over the network. The receiver collects the units, reconstructs the video and begins the decoding. The playback begins without having to wait for the entire video to be delivered. In general, there are a short delay (5 – 15 seconds) between the start of the delivery and the beginning of the playback at the client. This delay, namely pre-roll delay, determines also the buffer size for storing the received units.

The advantages of video streaming are considerable. First, it is not necessary to wait for a long delay before viewing the video. A low storage space is required to store the portions of video to view. These two benefits are determined by the duration of the pre-roll delay.

Many applications are based on video streaming techniques such as Video on Demand (VoD), video conferencing, peer-to-peer video streaming, etc.

3.1.3.1.1 Video on Demand Streaming

Video on Demand (VoD) streaming enables the distribution of live and recorded audio and video streams across LANs, intranets and the Internet after a client request. The client contacts the video server and requests the appropriate stream. Many VoD applications are available such as Microsoft Netshow, RealVideo of Real Networks, StreamWorks of Xing Technology, VDOLive of VDONet, Vosaic, Cisco Precept and PictureTel of Starlight Networks.

3.1.3.1.2 Video Conferencing

Videoconferencing is an application that uses video to join people in some live interaction. Videoconferencing makes possible interactive meetings with individuals who may be in other country. The video is delivered to all the participants in real-time. Video conferencing requires

expensive, fixed delivery and reception installations and high transmission costs in term of bandwidth consumption. It allows full two-way audio and video communication between several users.

3.1.3.1.3 Peer-to-peer Video Streaming

The main concept of peer-to-peer computing is that each peer is client and server at the same time. In this context, the multimedia content playing by the user is shared among peers. Peer-to-peer sharing uses the 'open-after-downloading' mode, while peer-to-peer video streaming uses the 'play-while-downloading' mode. One of the advantages of peer-to-peer video streaming is that peers have direct connection to other peers avoiding communication via mediating servers.

Significant research effort has addressed the problem of efficiently streaming multimedia, both live and on demand using peer-to-peer video streaming. We can found systems like *SpreadIt* [5] for streaming live media and *CoopNet* [6], [7] for both live and on-demand streaming. Both systems build distribution trees using application-layer multicast while relying on cooperating peers. Video delivering can be in a multicast [8], [9] model or on demand as presented in [10] and [11]. Peer to peer layered video is also experienced in the work presented in [12].

3.1.4 Interactive vs. Non Interactive Video Applications

Interactive applications have real-time data delivery constraints. The data sent has a time-bounded usefulness, after this time the received data is useless. In Figure 3-3 we give a brief classification of packet video applications based on delay and loss requirements over packet switching network according to the ITU-T recommendation G.1010 [13]. This presents the classification of performance requirements. In this Figure, various applications can be mapped onto axes of packet loss and one-way delay. The size and shape of the boxes provide a general indication of the limit of delay and information loss tolerable for each application class. We can found these classes of applications.

- Interactive video applications. They need a few milliseconds of transfer delay such as conversational voice and video, interactive games, etc.
- Responsive video applications. Typically, these applications response in few second, so that human does not wait for a long time, such as voice and video messaging, transactions, Web, etc.
- Timely video application. The transfer delay can be about some second, such as streaming audio and video.
- Non-critical video application. The transfer delay is not a critical for those applications, such as audio and video download service.

From loss point of view, we can find two types of applications:

- Error sensitive video applications such as highly compressed video.
- Error insensitive video applications such as non-compressed video.

The loss has a direct impact on the quality of the information finally presented to the user, whether it is voice, image, video or data. In this context, loss is not limited to the effects of bit errors or packet loss during transmission, but also includes the effects of any degradation introduced by media coding.

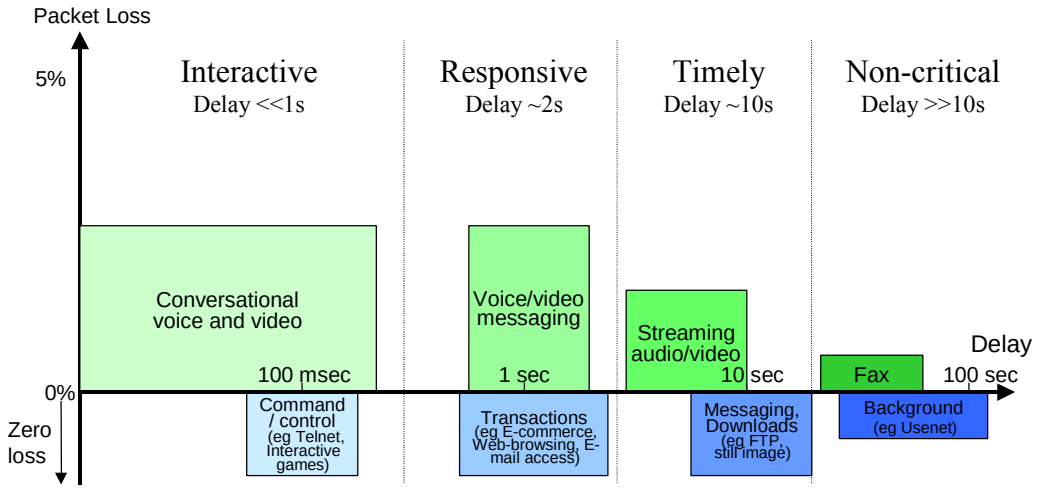


Figure 3-3: Classification of user application according to loss and delay requirements

3.1.5 Constant Bit Rate vs. Variable Bit Rate Video Applications

Some packet video applications generate Constant Bit Rate (CBR) traffic and others generate Variable Bit Rate (VBR) traffic. Theses applications typically have time-varying complexity which cannot be generally achieved by a CBR coding. Therefore, coding a video to perform a constant visual quality requires VBR coding while CBR coding would produce a time-varying quality. In variable network channel condition, it is important to match the video bit rate to what the network support. Many combinations can be used that are presented in Section 3.2.1.

Figure 3-4 illustrates a video encoder generates a VBR video stream which can be transformed into a CBR stream using an encoding buffer, after which theses streams can be transmitted over CBR or VBR channel.

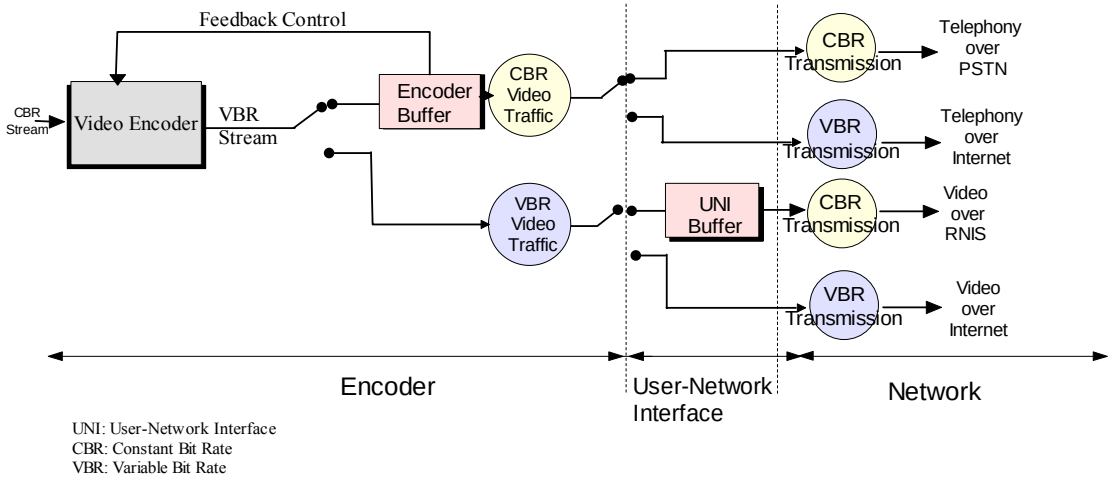


Figure 3-4: CBR vs. VBR coding and CBR vs. VBR transmission

3.1.6 Static vs. Dynamic Channels

The quality of the delivered video over networking environment is affected by the nature and the characteristic of the communication channel such as bandwidth, delay, and loss. These parameters may be static or dynamic. In the static channel, the bandwidth, delay, and loss are known in advance and bounded, whereas in the dynamic channel, it is difficult to determine they upper bound.

It is clear that delivering video over static communication environment is much easier. Moreover, many of researches ongoing on video streaming are related to capacity varying communication environment such as IP.

Another attribute that affect the design of multimedia streaming system is whether the network is packet-switched or circuit-switched. Packets switched networks, such as Ethernet and Internet are by nature shared networks in which packets may encounter losses, may arrive out of order, and exhibit variable delay. On the other hand, circuit-switched networks such as public switched telephone network (PSTN) or ISDN has a fixed delay and guaranteed fixed bandwidth allocated. However, the errors still a challenge since as corrupted bit-errors or burst errors may affect data integrity.

3.2 Requirements and Challenges for Packet Video Applications over IP

There are a number of problems that affect packet video streaming. In this section, we discuss the problems that influence video streaming quality over IP packet switching networks. We present their solution found in the literature.

Video streaming over IP is a challenging task since IP is a shared environment providing best effort service. It offers no quality of service and no guarantee of resources in term of (1) bandwidth, (2) packet losses, (3) delay variation (jitter), and (4) transfer delay. Moreover, these parameters are unpredictable and unbounded. Therefore, a key goal of video streaming is to design a system that efficiently handles the video over IP and deals with the above problems. Table 3-1 summarizes issues with video delivering and their solutions.

Video Transmission Issues	Solutions	Section
Bandwidth Management	Codec choice Adaptive Video Techniques	3.2.1
Packet Losses	Loss and Error Management	3.2.2
Delay Variation (Jitter)	Playout Buffer	3.2.3
Transfer Delay	Network QoS	3.2.4

Table 3-1: Video delivering problems and solution

The bandwidth is the most important characteristic that affects directly the video quality of service. The bandwidth between the receiver and the sender is generally known and it is time-

varying. If the sender transmits more than the available bandwidth, video packets may be lost in the channel, especially in the bottleneck links. If the sender transmits lower than the available bandwidth than the receiver playback a poor video quality. One of techniques to deal with bandwidth variation is to use adaptive video streaming. The server estimates the available bandwidth and then adapts it sending rate to much the available bandwidth. Specially, to deal with the bandwidth it is necessary to choose the best codec that can produce video at certain rates and when the channel conditions change it applies an adaptive behavior. Some of the video codec designed for video streaming are presented in Section 3.2.1.1. On other hand, the techniques for adaptive video streaming are presented in Section 3.2.1.

Packet looses is affected by the traffic pattern and its volume in the network. Packet loss occurs in general in network routers. Before the packet will be routed, it must be buffered. If the router buffers are full, all incoming packets will be automatically dropped because of the drop tail policy used by best effort routers. Many techniques can be used to make TCP source reducing it sending rate such as implementing RED queue management. This allows avoiding network congestion and preventing the application entering congestion collapse in which the network link is heavily used and little useful work is being done. Another type of packet loss is considered when there is a bit-error in the received packet. Bit error is generally afflicted in wireless environment. To combat the effect of losses, the video streaming system must be designed with error control in mind. There are many techniques that can be used to deal with this problem and are presented in Section 3.2.2.

The transfer delay is the time that the packet experiences from the time when leaving the server until its reception by the client. This delay varies from one packet to another. It is affected by the pattern and volume of traffic entering the network. The variation of this delay is called the jitter. Jitter is a problem, because when a video packet arrives at the client too late it becomes unused. On other hand, when a packet arrives too fast, it must be buffered and then could produce buffer overflow. To compensate delay variation, the client includes a customized smoothing buffer also called playout buffer. Section 3.2.3 addresses this problem.

3.2.1 Bandwidth Management

As seen before, transmitting packet video stream over the IP encounters three major problems: bandwidth fluctuation, delay variation, and packet losses. It is desired to have techniques to adapt coding video stream to channel condition, namely adaptive video streaming. Adaptive video streaming aims to adapt the sending rate to the available data-rate in the network. It addresses also the problem of heterogeneous receivers. Many techniques exist to implements adaptive video streaming. In this subsection, we focus on existing techniques for adaptive video streaming.

3.2.1.1 Video Compression Standards

Audio video coding algorithms are greatly correlated to the growth of computation speed of systems. The most important compression standard for streaming video are proposed by ITU (International Telecommunication Union) and ISO (International Organization for Standardization) standardization bodies such as H.261, H.263, MPEG-1, MPEG-2, MPEG-4 and recently H.264. The majority of available video codec does not survive to streaming in IP

environment because they require a great scalability, lower computational complexity, great resiliency to network losses and lower encoding / decoding latency. Building a video codec that response to all these requirements is not sufficient. The codec must be aware of network condition in order to achieve the highest possible user perceived quality. Current research is investigating a new scalable and flexible coding algorithm, and ways for adapting the existing codecs to heterogeneous environment such as Internet. We present in this subsection some of video codec that are widely used in packet video applications.

3.2.1.1.1 Overview of Video Compression

Since Newton's time, it has been known that a wide spectrum of colors can be generated from a set of three primaries colors. Television displays generate colors by mixing lights of the additive primaries: red, green, and blue (RGB). The classic color chart used in early television specifications was established in 1931 by the Commission Internationale de L'Eclairage (CIE). It defines a special concept of isolating the luminance, or brightness, from the chrominance, or hue. From this chart, two systems were defined, namely, NTSC YIQ and PAL/SECAM YUV. YIQ color space represents the luminance, in-phase chrominance, and quadrature chrominance coordinates respectively. The only change between the PAL/SECAM YUV color space and the NTSC YIQ color space is a 33 degree rotation in UV plan.

The digital equivalent of YUV is YCbCr, where the Cr chrominance component corresponds to the analog V component, and the Cb chrominance component corresponds to the analog U component, though with different scaling factors. The codec will use the terms Y, U, and V for Y, Cb, and Cr for a shorthand description of the digital YCbCr color space. The result is that the YCbCr elements are less correlated compared to RGB format and, therefore, can be coded separately without much loss in quality.

The gain of data volume by converting from RGB to YCbCr is 2 to 1 (denoted 2:1). For example, if the RGB format is specified by eight bits for each color, then each RGB picture element is described by 24 bits; and after conversion and decimation, each YCbCr pixel is described by an average of 12 bits: the luminance at eight bits, and both the chrominances for every other pixel (horizontally and vertically) at eight bits. This technique was the first step toward video compression.

Furthermore, video compression is achieved by exploiting similarities or redundancies that exists in typical video signal. A video sequence consists of a sequence of video frames or images. Each frame may be coded as a separate image. A frame to be coded is divided into a set of blocks, which can be compressed using two mains techniques. Block Discrete Cosing Transform (DCT) and Discrete Wavelet Transform (DWT). Coding each frame separately is not efficient since neighboring video frames are typically very similar. Much higher compression can be achieved by exploiting similarity between frames. A given frame can be predicted from a previous frame. The difference between the two frames is considered rather than the complete frame. Consecutive video frames typically contain the same imagery with possible different special locations because of motion. To improve the prediction, it is important to estimate the motion between the frames and to form an appropriate prediction that compensates this motion. This process is known as Motion Estimation. The process of forming a prediction in motion is called Motion Compensation. It

represents the difference or error between the best-match frame block and the current to-be-coded frame block.

In term of temporal location in video frames, three types of coded frames are generally generated by the codec. (1) Intra-coded frame (I-Frame) where the frame is coded independently of all the frames. (2) Predicatively coded frame or (P-Frame) where the frame is coded based on a previously coded frame. (3) Bi-directionally predicted frame (B-Frame) where the frame is coded using both previous and future coded frames. Figure 3-5 illustrates an MPEG video Group of Picture (GOP) coded using predictions dependencies between frames.

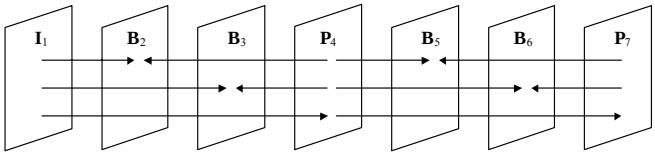


Figure 3-5: Example of the prediction dependencies between frames

Current video codecs achieve video compression by applying combination of the previously described techniques and which are: color space conversion, motion compensation, motion estimation, and signal transformation.

Currently there are three families of video compression standards, performed under the auspices of ITU-T (International Telecommunication Union-Telecommunication), ISO (International Organization of Standardization) and IEC (International Electro-Technical Commission) as explained in Figure 3-6. A brief description of these standards is given in the next subsections.

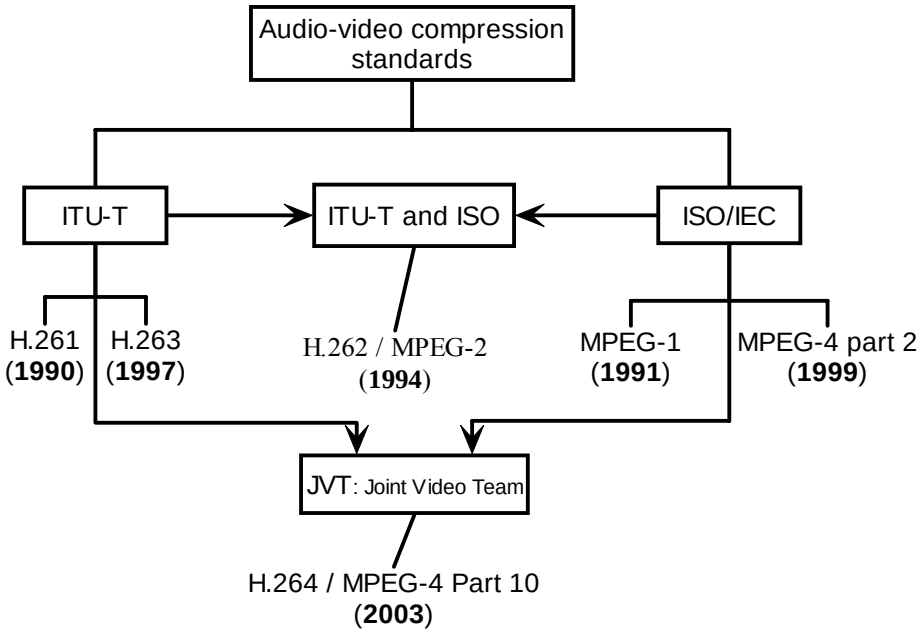


Figure 3-6: Current and emerging video compression standards

3.2.1.1.2 H.261

H.261 is called video codec for audiovisual services at p * 64 kbit/s (know also as p64 algorithm) [14]. This ITU-T recommendation was developed to target a constant bit rate videophone, videoconference and other audiovisual services over ISDN. The H.261 model consists of five stages: a motion compensation stage, a transformation stage, a lossy quantization stage, and two lossless coding stages.

The coding algorithm is a hybrid of inter-picture prediction, transform coding, and motion compensation. The H.261 specification is already implemented in several telecommunications devices and is integrated onto custom chips by several manufacturers. The difference between H.261 and MPEG-1 algorithm is in the bit-rate. MPEG-1 was not designed for packet video streaming. It has a bit rates–0.9 Mbps to 1.5 Mbps and consequently achieves higher quality than H.261.

3.2.1.1.3 H.263 and H.263+

H.263 is a recommendation that specifies a coded representation that can be used for compressing moving picture at low bit rates [16]. The basic configuration of the video source coding algorithm is based on ITU-T recommendation H.261. H.263 differs from H.261 recommendation in the flowing. Half pixel precision is used for the motion compensation compared to full pixel precision. In addition to the core H.263 coding algorithm, four negotiable coding options are included to improve performance. All these options can be used together or separately. These options are: Unrestricted Motion Vector mode, Syntax-based Arithmetic Coding mode, and Advanced Prediction mode and PB-frames mode. H.263 supports five resolutions. In addition to CIF (352 x 288) and QCIF (176 x 144) that were supported by H.261, there is sub-QCIF (128 x 96), 4CIF (704 x 576), and 16CIF (1408 x 1152). H.263+ was formally known as H.263v2 [17]. This codec enhances H.263 codec by adding new types of options which are scalability pictures, improved PB frames, custom source formats, and nine new coding modes.

3.2.1.1.4 MPEG-1, MPEG-2(H.262) and MPEG-4

Moving Picture Experts Group (MPEG) was created to develop video packing standards. At the moment there are three MPEG-standards available and two on the process: MPEG-1 (ISO-11172) [18], MPEG-2 (ISO-13818) [15] and MPEG-4 (ISO-14496) [19]. The ISO/ IEC IEC13818 namely MPEG-2 is known also as ITU-T recommendation H.262, This recommendation is designed for high definition video quality [15]. Also multimedia document description interface MPEG-7 (ISO-15938) [20] and frame of reference standard MPEG-21 (ISO-18034) [21] is under development. MPEG-1, 2 and 4 are currently the wide well known codec in the word. Thanks to MPEG standards that Digital Television is now possible.

The first standard developed by MPEG committee is MPEG-1 codec. This codec targets a bit storage rate of 0.9 - 1.5 Mbps offering VHS quality at CIF resolution and 30 frames per second. MPEG-1 is not adapted for transmission over loosely packet switching environment such as IP, due to the dependencies present in the P (Predicted) and B (bi-directional predicted) frames. For this reason, MPEG-1 does not offer resolution scalability and the video quality is highly vulnerable to packet losses.

MPEG-2 extends MPEG-1 by including support for higher resolution video and many others capabilities. It is designed for Digital Video Broadcasting (DVB) and high quality digital video storage (DVD). The target bit rate for MPEG-2 is 4 -5 Mbps. Capabilities introduced in MPEG-2 concern essentially the scalability. Three types of scalability are proposed: Spatial scalability allows the decoder to treat a subset of streams produced by the coder to rebuild and display video with a reduced space resolution. Temporal scalability allows the decoder to treat a subset of streams produced by the coder to rebuild and display a video with reduced temporal resolution. With SNR Scalability (Signal to Noise Ratio), the coder transmits the difference between the original image and the preceding. This allows improving subjective quality of the final image to get maximum quality. However, MPEG-2 was not designed for transmission over IP due to bandwidth consumption. Figure 3-7 presents the hierarchical organization of an MPEG-2 video sequence.

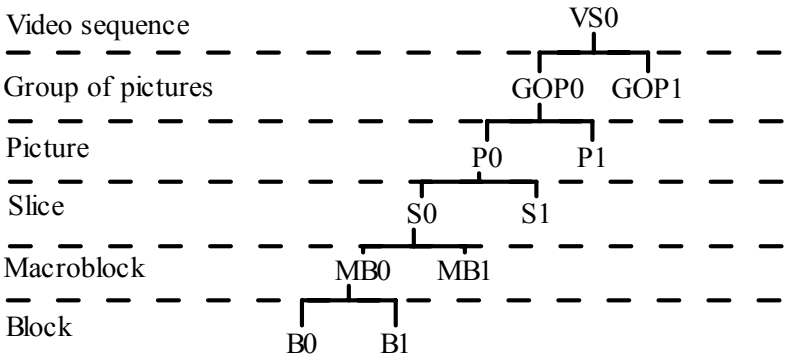


Figure 3-7: Hierarchical organization of MPEG-2 data stream

MPEG-4 standard [22][23][24] is an emerging digital multimedia standard with associated protocols for representing, manipulating and transporting natural and synthetic multimedia content (i.e. audio, video and data) over a broad range of communication infrastructures including IP and ATM .

The original characteristic of MPEG-4 is to provide an integrated object-oriented representation of multimedia content for the support of new ways of communication, access, and interaction with digital audiovisual data, and offering a common technical solution to various telecommunications, broadcast, and interactive services. MPEG-4 addressed a broad range of existing and emerging multimedia applications such as multimedia broadcasting, content-based audiovisual database access, games, audiovisual home editing, advanced audiovisual communications and video over mobile networks.

The MPEG-4 standard introduces a new technique of coding multimedia scenes called "object-based video coding". This technique allows the encoding of different audio-visual objects in the scene independently. An MPEG-4 scene consists of one or more Audio Visual Object (AVO), each of them is characterized by temporal and spatial information. The hierarchical composition of an MPEG-4 scene is depicted in Figure 3-8. Each audio visual object may be encoded in a scalable (multi-layer) or non scalable (single layer) form. A layer is composed of a sequence of a Group of Video-Object-Plane (GOV). A Video Object Plane (VOP) is similar to the MPEG-2 frame. VOP supports intra coded (I-VOP) temporally predicted (P-VOP) and bi directionally predicted (B-VOP).

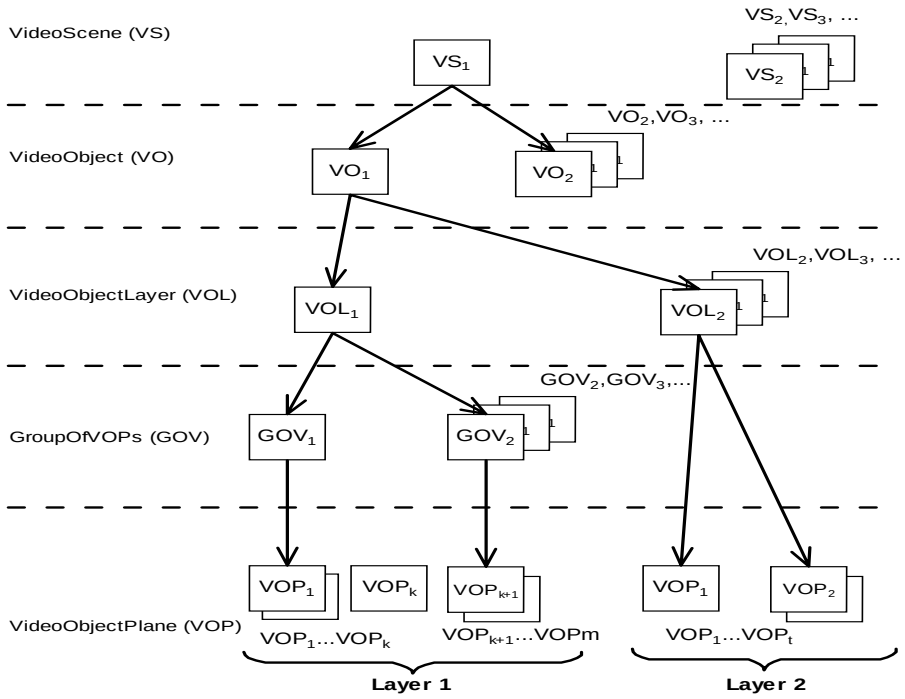


Figure 3-8: Hierarchical organization of MPEG-4 data stream

3.2.1.1.5 H.264

Since 1997, the IUT-T video experts Group (VCEG) has been working on new coding standard, namely H.26L. In late 2001, MPEG video group and VCEG decided to work together as Joint Video Team (JVT) to create a single technical design for forthcoming ITU-T Recommendation and for new part of ISO/IEC MPEG-4 standard. The final working version carries the denomination H.264 and ISO/IEC 14496-10 (MPEG-4 part 10) it has been adopted in July 2003 as a standard in Berlin Meeting.

The codec specification [25] itself distinguishes conceptually between a video coding layer (VCL), and a network abstraction layer (NAL). The VCL contains the signal processing functionality of the codec, things such as transform, quantization, motion search/compensation, and the loop filter. It follows the general concept of most of today's video codecs, a macroblock-based coder that utilizes inter picture prediction with motion compensation, and transform coding of the residual signal.

The basic configuration of the H.264 codec is similar to H.263 and MPEG-4, Part 2. The image width and height of the source data are restricted to be multiples of 16. Pictures are divided into macroblocks of 16x16 pixels. A number of consecutive macroblocks in coding order can be organized in slices. Slices represent independent coding units in a way that they can be decoded without referencing other slices of the same frame. The outputs of the VCL are slices. The NAL encapsulates the slices into Network Abstraction Layer Units (NALUs) which are suitable for the transmission over packet networks or the use in packet oriented multiplex environments [26].

Earlier test demonstrates that the future H.264 standard can achieve 50% coding gain over MPEG-2, 47% coding gain over H.263 baseline, and 24% coding gain over H.263 high profile

encoders within the motion compensation loop in order to reduce visual artifacts and improve prediction [27].

One of the main properties of the H.264 codec is the complete decoupling of the transmission time, the decoding time, and the sampling or presentation time of slices and pictures. The codec itself is unaware of time, and does not carry information such as the number of skipped frames (as common in the form of the Temporal Reference in earlier video compression standards). Also, there are NAL units that are affecting many pictures and are, hence, inherently time-less. IETF is working now on defining RTP payload format for H.264 codec, which defines essentially timing information of NAL units [28].

3.2.1.1.6 Proprietary Format

There exists a number a proprietary solution for video compression such as Windows Media Format by Microsoft [29] and Real Video by Real Network [30].

Windows Media 9 Series product was developed and published in November 2002. It includes many technologies and tools for media creation, delivery, security and playback. However Windows Media is proprietary, OS-based system and details information of this format is not available for public domain. A little work considers streaming using Windows Media Format.

The last release of Real Network has developed recently Real Video 9 and published in July 2002. A compression format designed for broadband and high definition television. It promises a good quality with a reasonable bandwidth. Furthermore, real video is a multiplatform solution. Recently there were many efforts among the Real Network to move to an open source solution. Helix project was the result of these efforts. Helix initiative consists of three major elements: (1) Helix Platform, (2) Helix Community, and (3) Helix Universal Server. Helix platform is an open source platform for media production and applications for any format and operating system.

3.2.1.1.7 Other Related Standards: MPEG-7 and MPEG-21

The Moving Expert Group (MPEG) has not stop after getting the MPEG-4 ready for use. Currently it is defining standards related to multimedia such MPEG-7 [20], and MPEG-21 [21].

(a) MPEG-7

Before one can use any information (audio, video, data, etc), it have to be identified and located first. More and more audio-visual information is available in digital form, in various content delivery systems and networks. At the same time, the increasing availability of potentially interesting material makes this search harder. The question of finding content is not restricted to database retrieval applications; also in other areas similar questions exist. For instance, there is an increasing amount of (digital) broadcast channels available, and this makes it harder to select the broadcast channel (radio or TV) that is potentially interesting.

In October 1996, MPEG started a new work item to provide a solution for the urging problem of generally recognized descriptions for audio-visual content, which extend the limited capabilities of proprietary solutions in identifying content that exist today. The new member of the MPEG family is called "Multimedia Content Description Interface", or in short MPEG-7 [20].

The emphasis of MPEG-7 will be the provision of novel solutions for audio-visual content description. It will standardize:

- A set of description schemes and descriptors,
- A language to specify description schemes, i.e. a Description Definition Language (DDL),
- A scheme for coding the description.

(b) MPEG-21

The general goal of MPEG-21 [21] activities is to describe an open framework which allows the integration of all components of a delivery chain necessary to generate, use, manipulate, manage, and deliver multimedia content across a wide range of networks and devices.

The MPEG-21 multimedia framework will identify and define the key elements needed to support the multimedia delivery chain, the relationships between and the operations supported by them. Within the parts of MPEG-21, MPEG will elaborate the elements by defining the syntax and semantics of their characteristics, such as interfaces to the elements. MPEG-21 will also address the necessary framework functionality, such as the protocols associated with the interfaces, and mechanisms to provide a repository, composition, conformance, etc. The seven key elements defined in MPEG-21 are:

- Digital Item Declaration: a uniform and flexible abstraction and interoperable schema for declaring Digital Items,
- Digital Item Identification and Description: a framework for identification and description of any entity regardless of its nature, type or granularity,
- Content Handling and Usage: provide interfaces and protocols that enable creation, manipulation, search, access, storage, delivery, and (re)use of content across the content distribution and consumption value chain,
- Intellectual Property Management and Protection: the means to enable content to be persistently and reliably managed and protected across a wide range of networks and devices,
- Terminals and Networks: the ability to provide interoperable and transparent access to content across networks and terminals,
- Content Representation: how the media resources are represented,
- Event Reporting: the metrics and interfaces that enable Users to understand precisely the performance of all reportable events within the framework.

3.2.1.2 Video Scalability Coding Model

As seen before, the second item for managing bandwidth for packet video application is to use the video scalability coding model. There are two main techniques used for video coding. Block Discrete Cosing Transform (DCT) and Discrete Wavelet Transform (DWT). DWT has received

much attention for its inherent multi-resolution and progressive characteristics. Many works [31][32][33][34] have explored this techniques for adaptive video coding. DCT-based coding also proposes video scalability techniques in many codec such as scalability provided in MPEG-2 and H.263+. In video scalability, the video is coded in multiple representations. The major applications of scalability include internet video, wireless video, multi-quality video services, video database browsing etc.

Although a simple solution to scalable video is the simulcast technique that is based on transmission /storage of multiple independently coded reproductions of video. A more efficient alternative is scalable video coding, in which the bandwidth allocated to a given reproduction of video can be partially reutilized in coding of the next reproduction of video. Figure 3-9 depicts the different kinds of video scalability.

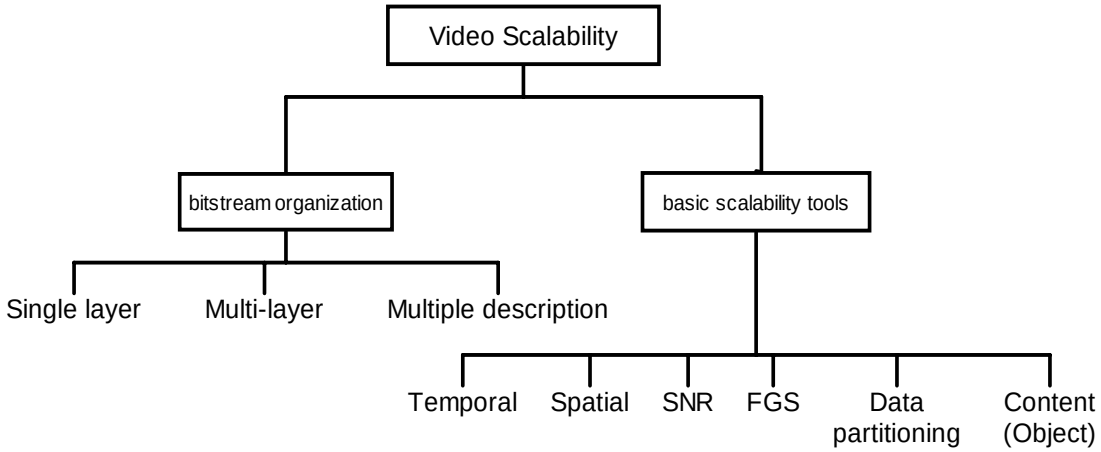


Figure 3-9: Video scalability coding modes

Grouping scalability according to the organization of the bitstream leads to single layer scalability, multi-layer scalability, and multiple description scalability. In the single layer scalability, the data is coded in a single segmented stream. In this case, a simple truncation can be performed to get lower quality layers. This leads to a fine granularity. In multi-layer scalability the data is coded in a base layer (BL) stream and one or more enhancement layers (EL) streams. The base layer is a separately decodable bitstream. The enhancement layers can be decoded in conjunction with the base layer to increase perceived quality by increasing the picture rate, increasing the picture quality, or increasing the picture size. A third type of scalability is multiple description scalability, in which the data is coded in several (at least two) different streams. Multiple description coding provides two important properties (1) each description can be independently decoded to give a usable reproduction of the original signal, and (2) the multiple descriptions contain complementary information so that the quality of the decoded signal improves with the number of descriptions that are correctly received. Unfortunately this scalability type is in principle less efficient, because dependencies between the streams cannot be exploited. A number of video coding algorithms have recently been proposed and they offered much functionality among which error resilient [35], [36], [37], [38], and [39].

Grouping scalability for video coding according to basic scalability tools lead to temporal scalability, spatial scalability, SNR scalability, fine grain scalability and object scalability.

Temporal scalability involves partitioning of the video into layers, where the lower layer is coded by itself to provide the basic temporal rate and the enhancement layer is coded with temporal prediction with respect to the lower layer. These layers, when decoded and temporally multiplexed, yield full temporal resolution. There is also support for temporal scalability by the use of B pictures. B pictures allow enhancement layer information to be used to increase perceived quality by increasing the picture rate of the displayed enhanced video sequence. This mode can be useful for heterogeneous networks with varying bandwidth capacity and also in conjunction with error correction schemes.

Spatial scalability refers to enhancement information to increase the picture quality by increasing picture resolution either horizontally, vertically, or both. Spatial scalability involves generating two spatial resolutions video layers from a single video source such that the lower layer is coded by itself to provide the basic spatial resolution and the enhancement layer employs the spatially interpolated lower layer and carries the full spatial resolution of the input video source.

SNR scalability refers to enhancement information to increase the picture quality without increasing picture resolution. SNR scalability and spatial scalability are equivalent except for the use of interpolation. Because compression introduces artifacts and distortions, the difference between a reconstructed picture and its original in the encoder is (nearly always) a nonzero-valued picture, containing what can be called the coding error. Normally, this coding error is lost at the encoder and never recovered. With SNR scalability, these coding error pictures can also be encoded and sent to the decoder, producing an enhancement to the decoded picture. The extra data serves to increase the signal-to-noise ratio of the video picture, and hence the term SNR scalability.

The Fine Grain Scalability (FGS), often known as the Streaming Profile is intended to support applications and environment where the bandwidth and/or computational power cannot be predicted and may vary dynamically. It was developed specially in MPEG-4 in response to the growing need on a video coding standard for streaming video over the Internet [40]. Three proposals was submitted to MPEG-4 for achieving FGS, namely, bit-plan coding of the predicted DCT residue [41][42], wavelet coding of image residue [43][44][45], and matching-pursuit coding of the predicted DCT residue [46][47]. The bit-plan coding of the predicted DCT residue was finally accepted after several experiment as a standard for FGS [48].

FGS and its combination with temporal scalability address a variety of challenging problems in delivering video over the Internet. FGS provides a mechanism that permits a single encoding process, producing a bitstream that can be modified subsequently in two different ways. Prior to transmission, the bitstream may be processed to scale it down to known bandwidth limitations. This can be performed dynamically, for example in response to the requirements of a statistical multiplexing system. Downstream distribution points may reduce the bit rate if necessary. After the transmission, the receiver terminal can adapt the received stream to its capabilities.

In [49] [50] [51], another type of FSG was introduced called Progressive Fine Granularity Scalable coding (PFGS). It improvement over FGS by introducing two prediction loops with different quality references.

An additional advantage of video scalability is its ability to provide resilience to transmission errors as the more important data of the lower layer can be sent over a channel with better error

performance, whereas the less critical enhancement layer data can be sent over a channel with poor error performance. A lot of scalable schemes were presented as presented in [52], [53], [54], [55], [56], [57], and [58].

3.2.1.3 Simultaneous Store and Stream

"Simultaneous Store and Stream" also know as Simulstore [59] is the technique used to store at the server different streams with different spatial resolutions, temporal resolutions and SNR levels. The client connects to the server and selects the appropriate stream from a multitude of stored streams. The quality of the video is not degraded, because each stream is codec optimally. Furthermore, this technique has an easy selection of the appropriate stream at server side and low complexity at server and client.

Simulstore can easily be combined with end-to-end retransmission but it has a major disadvantage that streams cannot cope with degradations in network conditions, hence, cannot be adapted to network conditions.

3.2.1.4 Stream Switching

Streams switching overcomes the disadvantage of Simultaneous Store and Stream by adapting the video transmission rate to network conditions. This technique is developed specially for controlling Simulcast streaming. It works as follows. If the transmission condition change (less or more network capacity), another stream is selected. This can lead to a synchronization problem. To overcome this problem, streams are synchronized according to intra-coded frames (I-Frame) or special switching frame (SP-Frame). Signaling in this case is crucial for a stream switching.

In [60] and [61], authors propose a seamless switching scheme for scalable video bitstreams. In this proposal, small bandwidth fluctuations are accommodated by the scalability of the bitstreams, while large bandwidth fluctuations are tolerated by switching between scalable bitstreams.

In [62] stream switching is also investigated in case of unicast and multicast video streaming. The proposed system can provide network adaptation with receiver-driven congestion control by estimating the network conditions. Clients are able to adjust their subscription levels by joining or leaving some multicast groups according to the estimated bandwidth. This allows users to switch up and down over different-rate bitstreams with minimized drift errors. Each group delivers a particular quality of the video stream.

RealVideo products developed by real network employs in some kind the method of stream switching called SureStream [63]. It consists of multiple streams encoding at different rate and stored in a single file. It provides a mechanism for servers and clients to detect changes in bandwidth and choose an appropriate stream. As a user is watching a video, the delivery bit rate can up shift or downshift, adjusting to changing network bandwidth conditions, to ensure the user a smooth playback experience. SureStream works for both live and on-demand content.

3.2.1.5 Simulcast

Simulcast means simultaneous transmission of different streams with different spatial resolutions, temporal resolutions and SNR level. The client selects the appropriate stream out of

multitude of streams. It can switch from one stream to another very easy. Here, the client is a simple single layer decoder.

3.2.1.6 Real-Time Single Stage Encoding

Real-time coding means to encode the video stream right before the transmission process. The client connects to the servers and specifies the encoding parameters. This result in a high complexity at the server and rate adaptation in the network is not possible.

3.2.1.7 Transcoding

One of the methods to modify the media bit stream is the transcoding. The transcoding aims to transform of one media format to another media format. For example transcoding from MPEG-2 format to H.264 format. This allows bit rate reduction, frame rate reduction, and temporal and special downsampling [64]. The transcoding is applied by decoding the video to raw format (uncompressed video) and then re-encode the video to the target format. There are mainly two drawbacks when using this approach. First, the quality of the result format is generally lower than the original format. Second, media transcoding generally requires extensive computation and large storage spaces, which makes this approach very expensive. The advantage in transcoding is that it allows media adaptation both at the server and in the network. However, it can be efficient using a dedicated media gateway [65] [66].

3.2.1.8 Streaming over TCP

The Transmission Control Protocol (TCP) is the most widely used protocol in the current IP networks. It can be used for packet video streaming. However, it has been noted that TCP is not adapted for real time traffic for theses reasons. First, TCP guarantees delivery via retransmissions, but because of the retransmissions its delay is unbounded and not suitable for real-time traffic. Second, TCP provides flow control and congestion control that make packet video streaming unpractical. Finally, TCP requires a back channel for packets acknowledgements. All theses reasons, led us to choose UDP for media streaming applications

3.2.1.9 Streaming over UDP

The User Datagram Protocol (UDP) is simply a user interface to IP. It is therefore unreliable and connectionless. UDP does not guarantee delivery of packet video applications, but for those delivered packets their delay is more predictable. UDP does not provide any flow control or congestion control. If the video application needs theses techniques, it should implement its own one like TCP-Friendly congestion control.

3.2.1.10 Streaming over Rate Controlled UDP: TCP-Friendly

In order to avoid network congestion, in which the transmitted traffic exceeds the designed limit, it is necessary to implement functions that perform congestion control such as rate control. The congestion is a phenomenon that has symptoms on packet losses, transmission delay, and delay jitter. As we have discussed in Section 3.1.6, such symptoms represent significant challenges for media streaming systems. The congestion control can be performed at many levels, by the network (e.g. edge router policy, source quench, DEC-bit, RED) or by the end-hosts. In

environment such as the Internet, which involves multiple networking technologies, the congestion control is commonly performed by the end-hosts (e.g. video server).

The majority of multimedia applications perform over an RTP stack that is implemented on top of UDP/IP. However, UDP offers no congestion control mechanism and therefore is unaware of network condition and unfair towards other competing traffic. Today's Internet traffic is dominated by TCP. TCP uses several mechanisms to handle network congestion such as: AIMD (Additive Increase and Multiplicative Decrease), slow start, congestion avoidance, fast retransmit and fast recovery [67]. Thus, it is crucial that UDP traffic be TCP-compatible (TCP-friendly). So UDP must perform TCP-Friendly congestion control [68].

The idea of TCP-friendly helps to prevent the application entering congestion collapse in which the network link is heavily used and little useful work is being done. So to prevent such situation, all applications must perform TCP-like congestion control mechanisms. Traffic that does not perform in TCP-friendly manner is dropped by the router [69].

Recently, it has been observed that the average throughput of TCP can be inferred from end-to-end performance of measured quantities such as round-trip-time and packet losses [70][71]. This observation has led to the definition of TCP-Friendly Rate Control that can be performed by the end-hosts to maintain them rate to much a rate of en equivalent TCP session having the same conditions.

Video scalability is the key approach for achieving rate adaptation. Many works exploit the video scalability to implement the video rate control via TCP-Friendly congestion management. McCanne et al. [72], proposes a method that enables different receivers to adapt to bandwidth fluctuations by adjusting the number of layers to which multicast users subscribe. The video stream is divided into a number of multi-resolution layers. Each layer is transmitted to a separate multicast group.

Another type of server rate adaptation using the video scalability is to adjust the codec quantization parameters. In [73] and [74] the video server continually negotiates the available bandwidth and modifies the codec quantization values accordingly. We recall that adapting codec quantization values is a CPU-intensive task which can affects the performance of the video server. The idea of quality adaptation was also used in the context of MPEG-4 in works presented in [75] and [76]. These later employ Fine Granular Scalability which uses a layered coding algorithm. A TCP-friendly rate control algorithm adjusts the rate of each video layer by regulating the level of quantization.

In order to apply a video rate control, it is necessary to have a scheme that returns the transmission rate. The Loss-Delay Based Adaptation Algorithm (LDP) [77] is an end-to-end rate adaptation scheme for adjusting the transmission rate of multimedia applications to the congestion level of the network. Another class of congestion control schemes applies the additive increase, multiplicative decrease (AIMD) algorithm in some form [78], [79]. The Rate Adaptation Protocol (RAP) proposed by Rejaie *et al.* The work presented in [78] is an example of an AIMD-based scheme, where the source performs rate adaptation based on acknowledgments sent by the receivers (feedback information). The acknowledgments are used to detect losses and estimate round trip time (RTT). Rate adaptations are performed once per RTT, with transmission rate being

increased linearly in the absence of loss, and transmission rate being decreased multiplicatively when congestion is detected. RAP uses the ratio of short-term to long-term averages of RTT to fine tune the sending rate. The RAP protocol was applied in the context of unicast video delivery [80] [81]. The video is a layered constant-bit rate. All the layers have the same throughput. The rate control algorithm used by the server adapts the video quality to network state by adding and dropping layers to efficiently use the available bandwidth. The algorithm takes into consideration the status of the receiver buffer, making sure that base layer packets are always available for playback. In [82], the authors propose a spectrum of window-based congestion controls schemes which perform TCP-Friendly compatibility under RED control. These window-based schemes use history information to improve traffic fairness. The proposed schemes are fundamentally different from memoryless schemes such as AIMD and can maintain TCP-compatibility or fairness across connections using history information for different protocols.

In [83] Zhang et al. present an end-to-end transport architecture for multimedia streaming over the Internet. They propose a new multimedia streaming TCP-friendly protocol (MSTFP) that combines forward estimation of network conditions with information feedback control to optimally track the network conditions. This scheme improves end-to-end QoS by allocating resources according to network status and media characteristics.

The TCP-friendly congestion control mechanism that was developed recently is TCP-friendly Rate Control Protocol (TFRC) [84]. It seems to be more robust protocol and is recently accepted as an RFC by the IETF. TFRC provides sufficient responsiveness by taking into consideration all the parameters that affect the TCP rate such as loss, Round-Trip Time (RTT) and retransmission timeout value. The key advantage of TFRC is that it has a more stable rate during the session lifetime. The calculated rate is obtained by using the TFRC is [84] (see Eq. 1):

$$R_{TCP} \cong \frac{s}{RTT\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{3bp}{8}})p(1+32p^2)} \qquad \text{(Eq. 1)}$$

Where RTCP is the target transmission rate or the allowed transmission rate, s is the packet size, RTT is the round trip time, p is the loss rate, t_{RTO} is the TCP retransmission timeout value and b is the number of packets acknowledged by a single TCP acknowledgement.

3.2.1.11 Media Caching

Multimedia Caching is an important technique for enhancing the performance of streaming system. Proxy caching has been the key factor in the scalability of the Web, because it reduces the network load and access latency. Video caching aims to store some part of the video near the clients. It allows improving the quality of the delivered stream when there is a presence of bottleneck links between the server and the client. Many works has experienced the media proxy caching [85], [86], [87], [88], [89], and [90].

3.2.1.12 Video Smoothing

The rate of the encoded video stream is in general a variable bit rate having a bursty behavior. The bursty nature of this traffic causes in the majority of case network congestion. In order to

reduce the variability of the traffic and bandwidth requirement of the video stream, smoothing technique is necessary. Smoothing makes use of a client side buffer to receive data in advance of playback [91], [92], [93], [94], and [95].

3.2.2 Loss and Error Management

Packet loss is a problem that affects considerably the quality of the received video at the client. It can have a very destructive effect of the reconstructed video because of frame dependency. Packet loss may arrive at different level and under different consideration. In wired packet networks, the congestion is the first circumstance on packet loss. Entire packet can be discarded by routers. Whereas in wireless networks, the transmission channel may cause bit error. Thus, the packet could be rejected by the application layer.

Before presenting the approaches for reliable video streaming, it is necessary to understand the effect of packet loss on compressed video stream. In general, there are two basic problems induced by video packet losses. First, the loss of bitstream synchronization or when important synchronization marker is lost. In this case, the decoder may loss track of what bits correspond to what parameters. Second, since the video has a particular structure composed of frames mutually dependent, the loss of one frame may make all depending frame unusable.

To deal with the effect of losses, it is necessary to have functions to muffle the loss. There are two types of approaches for reliable video transmission. The first approach is a sender-based approach, in which the sender can be active or passive to recover from losses. The second approach is a receiver-based approach in which the client conceals the loss by some techniques. Figure 3-10 illustrates the different approaches. These approaches are discussed in subsequent subsections.

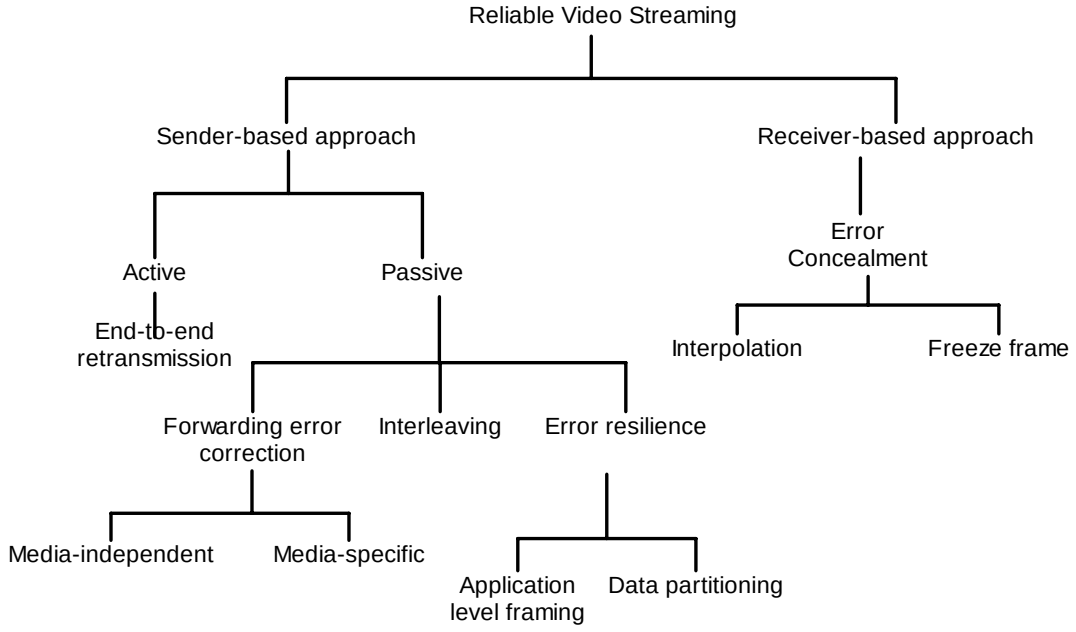


Figure 3-10: Taxonomy for reliable video transmission

3.2.2.1 End-to-End Retransmission

Retransmission is an Automatic Repeat ReQuest (ARQ) scheme for reliable communication. In this approach, lost packets are requested by the receiver from the sender. In this case, it is necessary to have a back channel to notify the sender which packets to send. The retransmission technique has the advantage to uses efficiently the available bandwidth. However, it has the major disadvantage that led it impracticable. Indeed, retransmission leads to additional delay which corresponds to the RTT (round trip time) experienced between the sender and the receiver. The retransmission approach is commonly considered as not useful in case of real-time traffic, particularly when the network RTT is large. Buffering at the receiver can be used to reduce delay constraint but it introduces high memory requirement. There exist a new approaches based on retransmission. For example, lost packets are retransmitted only if they can be playback by the client. This means that a particular packet must arrive before the deadline time. Many works are based on this idea [96], [97], [98], [99], and [100]. However, [101] argues that error recovery using retransmission is not feasible in case of multimedia traffic due to the imposed delay.

3.2.2.2 Forwarding Error Correction

Forwarding Error Correction (FEC) provides a mechanism to rebuilt lost packets by adding redundant information to the original data stream. The redundant data is derived from the original data using techniques from coding theory [102]. Many schemes can be applied to implement the FEC. Parity-based scheme using XOR and Reed Solomon Code are the most important schemes and are proposed as an RTP payload [103]. These techniques are called media-independent because the operation of the FEC does not depend on the contents of the packets and the repair is an exact replacement for a lost packet. In contrary to retransmission approach, FEC-based approach does not need a back channel. However, sending redundant data consumes additional bandwidth and can aggrieve the packet losses. Furthermore, it imposes an additional delay and makes the decoder more difficult for implementation.

FEC can be applied efficiently when knowing the characteristics of the data transmitted. In case of video, it can be applied unequally, so that important video part are more protected than less important part, as shown in [104], [105], [106], [107], [108], and [109].

3.2.2.3 Data Interleaving

Data Interleaving is a useful technique for reducing the effects of loss. The sender re-sequences the packets before sending them to the receiver, so that originally adjacent packets are separated by a guaranteed distance. The received packets are returned to their original order at the receiver. The interleaving disperses the effect of packet losses. This allows avoiding the effect of burst losses. Thus, error concealment techniques at the receiver will work better on interleaved packets. The disadvantage of the interleaving is that it increases the latency. This limits the use of this technique for interactive applications. The key advantage of interleaved is that it does not increase the bandwidth requirement of a stream.

3.2.2.4 Error Resilience: Application Level Framing and Data Partitioning

We expose, in this approach, two techniques for error resilience (1) Application Level Framing (ALF) and (2) Data partitioning.

In a packet video application, the packet is either received correctly or lost entirely. This means that the boundaries for lost information are exactly determined by the packet boundaries. To combat the effect of packet loss, it is necessary to design a specific packet payload format for each video standard. The video server knows best, it can apply a good packetization. For example, if the network path-MTU is known, the video encoder can design video packets having the MTU size and independently decodable. Many codec such as H.263+, MPEG-4, and H.264 support the creation of different form of independently decodable video packets.

Since the reliable transmission mechanism offered by TCP incurs considerable overhead by retransmission, RTP [110] does not rely on TCP. Instead, applications are put on top of UDP. How to cope with the lost packets is up to the applications. Following the application-level framing principle, RTP functionality is usually integrated into the application rather than being implemented as a separate protocol entity. RTP provides basic packet format definitions to support real-time communication but does not define control mechanisms or algorithms. The packet formats provide information required for audio and video data transfer, such as the incoming video data packet sequence. Continuous media such as non-compressed PCM audio can be synchronized by the use of sequence numbers. Non-continuous data such as MPEG can be resynchronized by using the time stamp fields. Many RTP payload format was proposed for many video codec such as for H.261 [111], for MPEG1/MPEG2 Video [112], for H.263 [113], for MPEG-4 Audio/Visual Streams [114], and in progress RTP payload for JVT/H.264 video [28].

In MPEG-2, at the start of a new slice, information called a slice header is placed within the bitstream. The slice header provides information which allows the decoding process to be restarted when errors have occurred. In MPEG-4 a periodic synchronization marker is inserted to allow the decoder to resynchronize after synchronization loss.

In the data partitioning technique the video bitstream is organized and prioritized. Encoded video data is organized into high priority data such as headers, motion vectors, low frequency DCT coefficients, and addresses of blocks, and low priority data such as higher frequency DCT. Such organization follows from this idea that bits which closely follow a synchronization marker are more likely to be accurately decoded that those further away. This approach is referred to data partitioning in MPEG-4. Thus, if there are two available channels, higher data priority is transmitted in channel with better error performance and less critical data is transmitted in the channel with poor error performance. The degradation to channel errors is minimized since the critical parts of a bitstream are better protected. Data from neither channel may be decoded on a decoder that is not intended for decoding data partitioned bitstreams.

3.2.2.5 Error Concealment

In order to conceal the fact that an error has occurred on decodable video, it is necessary that the receiver apply error concealment techniques which can estimate the lost information or missing pixels. Since the video compression removes redundant data, estimating lost information is not an easy task. A trivial technique to conceal packet lost is to replace the corresponding block by a green or black square. However, this approach can be highly disturbing for human eyes. Other techniques more efficient are considered such as (1) interpolation (2) freeze frame, and (3) motion estimation compensation. The interpolation is used by smoothly extrapolate the missing block from the

correctly received block. The missing block can be regenerated from adjacent block. This technique is refereed as spatial interpolation. In temporal interpolation, the missing block is replaced by the equivalent from previous correctly received frame (freeze frame). This approach is very useful when there is little motion. The combination of these methods (interpolation and freeze frame) can be effective when applying a motion estimation and compensation. The problem of this approach is how to accurately estimate the motion for the missing block. Possible approach use motion estimation vector of neighboring motion vector. This approach is applied efficiently for one missing block, but the problem could be more difficult when more than one block is lost. For example, a packet loss may lead to the loss of entire frame. In this case, it is not possible to apply interpolation or motion estimation. Generally, the lost frame is replaced by the last correctly received frame (freeze frame).

3.2.3 Delay Management

Delay is an important performance metric of packet video applications. The delay concerns the end-to-end transfer delay end delay variation.

3.2.3.1 End-to-End Delay Management

The end-to-end delay is viewed as the time when the packet leaves the media server until the time it is consumed by the client. It is highly correlated to the time induced by the network element including queuing delay and link-level retransmission. The management of the end-to-end delay is assured by network QoS solutions which are presented in Section 3.2.4

3.2.3.2 Jitter Management

The jitter is the variation of the end-to-end delay experienced between the server and the client. The effect of the jitter can be very disaster on the quality of the received video. It can lead to interruption in continuous playback at the receiver. Jitter destroys the temporal relationships between periodically transmitted media unit that constitute a real-time media stream [115]. To overcome the effect of the jitter it is necessary to have a buffer to absorb the variation of the delay. Buffering provides a number of advantages such as jitter reduction, error recovery through retransmission, error resilience through interleaving and smoothing capabilities. However, the playout buffer introduces undesired additional delay before the playback can begin. Some works have experienced an analytic model to bound the jitter and design a model for adaptive playout buffer such as works presented in [116], [117], [118], and [119]

3.2.4 IP QoS Network Management

Multimedia applications require stringent delay and loss requirement and a large bandwidth. The current widely supported best-effort service model does not provide any guarantees of bandwidth, loss and delay. Currently, there is a considerable effort by various organizations such as IEEE, IETF, ISO, and ITU-T to design and implement a quality of service framework for various networking technologies especially for IP. In the following subsection we give an overview of the methods developed by the IETF, because it provides the more promising QoS framework approaches for the Internet and we present related work that provide QoS-support for packet

video applications. There are many IETF working group that progress architecture and concept for QoS framework for IP as. They are summarized in what follows:

- Intserv: Integrated service working group. Define a set of service models to support quality of service requirements of various applications.
- Diffserv: Differentiated service working group. Differentiated traffic based on application type and provides appropriate service.
- QoSr: QoS-based Routing working group. Provides a framework for QoS-based routing protocols. QoS Routing allows the network to determine a path that supports the QoS needs of one or more flows in the network [120].
- RSVP: Resource reservation setup protocol working group. Design a signaling mechanism to reserve resources along the connection to support specially integrated service.
- MPLS: Multi-protocol label switching. Provides standards for label switching which can be used for building fast routers/switches.

3.2.4.1 Best Effort Services

The best effort service model is the basic service provided by IP. All users' packets, entering the network, compute equally for network resources. In best effort service model, the network would make its best attempt to deliver packets to their destination with no guarantees and no special resources allocated for any of the packets. Network elements use in general FIFO discipline as a simple queuing discipline.

3.2.4.2 Integrated Services

Integrated Service (IntServ) defines an architecture that is able to reserve network resources for individual flows in order to provide QoS guarantees to end users. The integrated service working group has specified *Control Load Service* (CLS) and *Guaranteed Service* (GS) in addition to the best effort service. RFC 2216 [121] gives a framework which specifies the functionalities of network components which can support multiple, dynamic quality of service. The framework defines a set of functionalities including the setup protocol needed to define the service and the end-to-end behavior.

The control load service (CLS) [122] aims to provide a QoS to applications that closely approximating the QoS a flow would receive from an unloaded network. CLS uses capacity (admission) control to assure that the service is received even when the network element is overloaded. This assumption reposes on defining traffic specification (TSpec) [123] for each flow traversing the network. TSpec consists of a token bucket plus a peak rate, minimum policed unit, and a maximum packet size. Policy is applied for a particular traffic using a token bucket filter. The RSVP protocol is used to reserve the appropriate resource along the packet flow.

The guaranteed Service (GS) [124] aims to deliver a guaranteed on delay and bandwidth of a particular flows in IP. It provides a mathematically provable approach for bounding the end-to-end delay. Thus, it provides a service that guarantees both delay and bandwidth. GS uses also TSpec to

represent user traffic. The delay can be bounded using an envelope curve calculated for each network element. Thus, the transfer delay of a particular packet is computed from the time it enters the network until it leaves a particular network element [125]. Furthermore, policy is done at the edge router of the network to check weather a particular flow is conforming to TSpec or not. When non-conforming packet arrives, they receive best-effort service. Appropriate scheduling mechanisms, such as WFQ (Weighted Fair Queuing), are used each network element.

Intuitively, the most straightforward method for assuring service guarantees to packet video applications is to reserve resources according to an application's request. However, reservation based schemes involve admission control algorithms, reservation protocols such as RSVP, monitoring protocols to track the change in network conditions, and signaling mechanisms. These kinds of schemes are generally complex and require extensive network support. This can lead to network scalability and robustness problems.

3.2.4.3 Differentiated Services

The Differentiated Service Architecture (Diffserv) [126] has been motivated by the market need for immediate deployment of a QoS solution for IP networks. In contrast to the per-flow orientation of RSVP, Diffserv networks classifies packets into one of a small number of aggregated classes, based on the DSCP (Diffserv Code Point) in the packet's IP header [127]. At each Diffserv router, packets are subjected to a PHB (Per-Hop Behavior), which is invoked by the DSCP. A PHB is defined to provide a specific forwarding behavior at each router within the Diffserv domain.

Two PHB schemas known as EF (Expedited Forwarding) [128] and AF (Assured Forwarding) [129] has been proposed to provide differential treatment of traffic. The EF PHB can be used to build a low loss, low delay, low jitter, assured bandwidth, and end-to-end service through Diffserv domains, while AF PHB group provides different levels of forwarding assurance for IP packets by discarding more low priority packets during times of congestion than high priority packets. AF PHB provides four IP packet forwarding classes. Within each AF class, an IP packet is assigned one of the three different levels of drop precedence (i.e. a total of twelve code points).

Diffserv architecture defines, at a lower level, four types of data-path elements: traffic classifiers, actions elements, meters and queuing elements as shown in Figure 3-11.

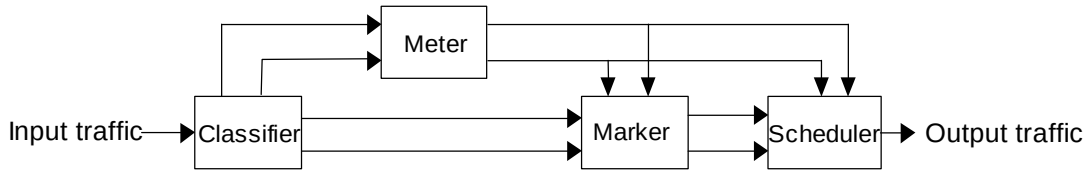


Figure 3-11: IP differentiated service components

- The Classifier performs the classification of traffic based on multiples fields in the IP packet such as source / destination address, source / destination port, Type of Service (TOS) field, etc.
- The Meter measures the amount of traffic received of a particular flows. It checks weather the traffic is conforming to the traffic specification or not. It also apply some policy on confirming and non-conforming traffic.

- Action Elements, such as the Marker. The Marker allows marking the IP packet with a particular DSCP. Diffserv working group has discussed functionality required at Diffserv domain edges to select (classifier) and condition (e.g., policy and shaper) traffic according to traffic management policies. Several marking algorithms were proposed, among which, TSWTCM (Time Sliding Window Three Colour Marker) [130] and TRTCM (A Two Rate Three Color Marker) [131].
- The scheduler implements a particular queuing discipline to serve the packet.

There are some related work that takes benefit from the Diffserv architecture to assure network QoS to multimedia applications. In [132], the authors present a content-based packet video forwarding mechanism where the QoS interaction between video applications and Diffserv network is taken into consideration. The interaction is performed through a dynamic mapping between the priority of each video packet and the differentiated forwarding mechanism.

In [133], the authors introduce a QoS management system over Diffserv network for multimedia servers that benefits from the scaling properties of layered media streams. The presented system enable to map application QoS demands to available streams according to inter-stream QoS dependencies.

Authors in [134] present a bit-stream classification, prioritization, and transmission scheme designed for MPEG-4 video. Different type of data are re-assembled and assigned to different priority using IP Diffserv model.

3.2.4.4 Multiprotocol Label Switching

Multiprotocol Label Switching (MPLS) provide a label switching functionalities at the network layer. In connectionless network, as a packet protocol travels from one router to the next, each router makes an independent forwarding decision for that packet based on information contained in the packet header [135]. In this case, each router analyzes the packet's header, and each router runs a network layer routing algorithm. Each router independently chooses a next hop for the packet, based on its analysis of the packet's header and the results of running the routing algorithm. In MPLS, the assignment of a particular packet to a particular FEC is done just once, as the packet enters the network. The packet is assigned an encoded short fixed length value known as a "Label". The MPLS-enabled router, called Label Switching Router (LSR). When a packet is forwarded to its next hop, the label is sent along with it; that is, the packets are labeled before they are forwarded. The label which is put on a particular packet represents the Forwarding Equivalence Class (FEC) to which that packet is assigned. A label is similar to the VPI /VCI (Virtual Path Identifier / Virtual Circuit Identifier) in ATM. The distribution and management of labels among MPLS routers is done using the Label Distribution Protocol (LDP), which ensures these labels have a uniform meaning. MPLS reduce considerably the cost of packet classification that must be done for each packet. It provides a QoS support using a fast and consistent forwarding. Te major advantage of MPLS is that provides a traffic engineering support for services provides. As MPLS is a protocol-independent mechanism resident in network level switches with no application control, higher layer QoS protocols such as Differentiated Services can leverage the management support provided by MPLS.

3.2.5 IP Signaling Protocols for Packet Video Applications

Signaling protocols over IP networks such as the Internet offer intelligent network services to the applications in particular packet video applications. Two core signaling protocols are discussed in these subsections. The first category is related to QoS signaling protocol such as RSVP, RTP/RTCP, and COPS. The second category concerns session signaling protocol such as H.323, SIP, and MPEG-4 DMIF.

3.2.5.1 QoS Control Signaling Protocols

QoS Control signaling protocols are defined in order to provide Quality of Service support between network nodes and the applications. The signaling protocol defines a set of algorithms, messages, and parameters used by the network to assure Quality of Service to user traffic. In particular, QoS signaling helps to provide differentiated delivery services for individual flows or aggregates, network provisioning, admission control, service guarantee, control load service, etc. Dynamic QoS management could be provided using QoS signaling protocols.

We present in the following subsections two QoS signaling mechanisms which were developed to implement Integrated Services Framework and Differentiated Services Framework. These protocols are RSVP (Resource Reservation Protocol) and COPS (Common Open Policy Service).

3.2.5.1.1 RSVP

RSVP (Resource Reservation Protocol) [136] is a signaling protocol for resource reservation in IP networks. Communally it is used for Intserv architecture (controlled load and guarantee service). RSVP is an object-based message protocol which provides mechanisms to the application to request a specific QoS from the network. RSVP carries the request through the network, visiting each network element and attempts to make a resource reservation for the stream. To make a resource reservation, the RSVP communicates with two local decision modules, *admission control* and *policy control*. Admission control determines whether the network element has sufficient available resources to supply the requested QoS. Policy control determines whether the user has administrative permission to make the reservation. RSVP supports both multicast and unicast traffic, for IPV4 and IPV6 clients.

3.2.5.1.2 COPS Protocol

The COPS (Common Open Policy Service) [137] Protocol is a QoS signaling protocols supporting policy control over a simple client/server model. The protocol describes a simple query and response protocol that can be used to exchange policy information between a policy server (Policy Decision Point or PDP) and its clients (Policy Enforcement Points or PEPs). One example of a policy client is an RSVP router that must exercise policy-based admission control over RSVP usage (COPS-RSVP) [138] or a Diffserv client that receives configuration information about Diffserv elements like meter, classifier, scheduler, shaper, etc. (COPS-PR) [139].

3.2.5.2 Session Control Signaling Protocol

Packet video application has grown rapidly in the last few years. This rapid expansion and potential is essentially due to standardization bodies that enable protocols and services. Actually, it appears likely that both IETF SIP (Session Initiation Protocol) [140] with SDP (Session

Description Protocol) [141] and the ITU-T recommendation H.323 [142][143] will be used for setting up packet video applications such and multimedia conferences and telephone calls. While these two protocols are comparable in their features, SIP provides a higher flexibility to add new features; a relatively easier implementation; and a better integration with other IP related protocols. In the other hand, the recent ISO/IEC MPEG-4 standards target a broad range of low-bit rates multimedia applications: from classical streaming video and TV broadcasting to highly interactive applications with dynamic audio-visual scene customization (e.g. e-learning, videoconferencing). This is achieved by the MPEG-4 control plan, namely DMIF (Delivery Multimedia Integration Framework) [144]. Figure 3-12 illustrates the different body that originates signaling protocols. The rest of this section discusses some features about these protocols.

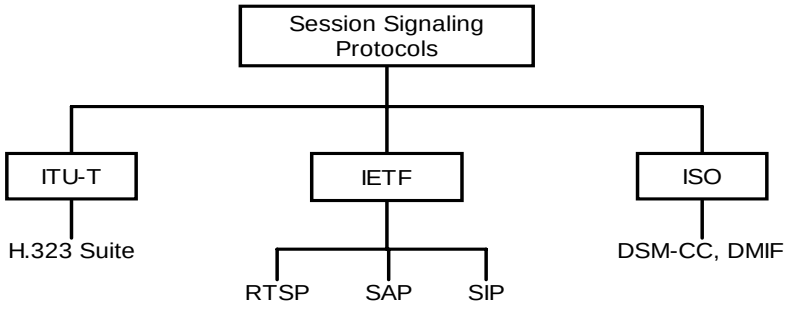


Figure 3-12: Taxonomy of multimedia session control signaling protocols

3.2.5.2.1 ITU-T H.323 Suite

The documents covering the H.323 protocol suite are created mainly by the International Multimedia Teleconferencing Consortium (IMTC) and are distributed by the International Telecommunications Union (ITU). The IUT first published the H.323 suite of protocols, including the ISDN-derived signaling mechanism in 1996 [142], and has updated the specification since then [143]. H.323 has proven to be a strong candidate for large-scale service providers and for enterprise telephony, video, and data conferencing applications. This standard is based on the RTP and RTCP, with additional protocols for call signaling, and data and audiovisual communications. H.323 defines how audio and video information is formatted and packaged for transmission over the network.

The H.323 protocol architecture is defined by a set of specific functions for framing and call control, audio and video codecs and data communication.

- Registration, admission, and status (RAS): RAS is the protocol between endpoints (terminals and gateways) and Gatekeepers (GKs). The RAS is used to perform registration, admission control, bandwidth changes, status, and disengage procedures between endpoints and GKs. An RAS channel is used to exchange RAS messages. This signaling channel is opened between an endpoint and a GK prior to the establishment of any other channels.
- H.225 Call Signaling and Q.931: H.225 call signaling is used to establish a connection between two H.323 endpoints. This is achieved by exchanging H.225 protocol messages on the call-signaling channel. The call-signaling channel is opened between

two H.323 endpoints or between an endpoint and the GK. H.225 specifies the mandatory Q.931 messages that are used for call signaling. Q.931 is actually an ISDN-related protocol used to set up and clear calls. A Q.931 packet may contain a number of parameters known as information elements. For example, a Q.931 packet may contain a user information element. H.323 specifies that the user information element must contain an H.225 message. All the H.323 parameters are coded in user information element of Q.931 messages

- H.245 Control Signaling: it is used to exchange end-to-end control messages governing the operation of the H.323 endpoint. These control messages carry information related to the following: capabilities exchange, opening and closing of logical channels used to carry media streams, flow-control messages and general commands and indications.
- T.120 Data interoperability: T.120 is a series of multimedia communications protocols and is the default basis of data interoperability between an H.323 terminal and other terminals.

3.2.5.2.2 IETF Protocols Suite

The IETF has defined a set of protocol suitable for packet video application. The latest one is SIP (Session Initiation Protocol) that regroup the majority of others protocols. SIP is an application-layer control and signaling protocol for creating, modifying and terminating sessions with one or more participants. These sessions include IP multimedia conferences, IP telephony calls and multimedia distribution [140]. SIP has been approved in early 1999 as an official standard by the IETF for signaling communications services on the Internet. SIP can be used to initiate sessions as well as invite members to sessions. The SIP architecture includes the following protocols:

- RTP (Real-Time Transport Protocol) for transporting real time audio, video and data [110], and RTCP (Real-Time Transport Control Protocol) is the control protocol for RTP, and provides mechanisms for data distribution monitoring, cross media synchronization, and sender identification. The sender transmits a multimedia data stream by using RTP packets. The receiver periodically sends RTCP packets that contain information about the received RTP packets. The information includes feedback and statistics such as the highest sequence number received, inter-arrival jitter, or packet loss rate.
- RTSP (Real-Time Streaming Protocol) for setting up and controlling on-demand media streams [145]. RTSP is used for initiating and controlling delivery of stored and live multimedia content to both unicast and multicast destinations. RTSP borrows time concept from MPEG-2 DSM-CC, but unlike DSM-CC, it does not depend on an entire set of supporting protocols. RTSP is transport-independent, and can use either

TCP or UDP as the transport protocol. The state machine makes RTSP suitable for remote digital editing and retrieval. RTSP is also text-based protocol, and therefore easy to parse. RTSP reuses the HTTP concept, but unlike HTTP, RTSP is a state full protocol.

- MGCP (Media Gateway Control Protocol) and Megaco for controlling media gateways [146].
- SDP (Session Description Protocol) for describing multimedia sessions [141]. SDP is intended for describing multimedia sessions for the purposes of session announcement, session invitation, and other forms of multimedia session initiation. The purpose of SDP is to convey information about media streams in multimedia sessions to allow the recipients of a session description to participate in the session. Therefore terminal capability is described by SDP.
- SAP (Session Announcement Protocol) for announcing multicast session [147]. SAP is used to assist the advertisement of multicast multimedia conferences and other multicast sessions, and to communicate the relevant session setup information to prospective participants. SAP announces a multicast session by multicasting packets periodically a well-known multicast group. The packets contain a description of the session using SDP.
- TRIP (Telephony Routing over IP) for locating the best gateway between the Internet and the PSTN (Public Switched Telephone Network) [148].
- Suite of resources management and multicast address allocation protocols.

3.2.5.2.3 ISO/IEC 14496-6 (MPEG-4 DMIF)

MPEG-4 DMIF [144] is the control plane of MPEG-4 Delivery layer that allows applications to transparently access and view multimedia streams whether the source of the stream is located on an interactive remote end-system, the stream is available on broadcast media or is located on stored media.

DMIF framework covers three major technologies: (1) interactive network technology, (2) broadcast technology and (3) the disk technology. An application accesses data through the DAI irrespectively whether such data comes from a broadcast source, from local storage or from remote server. In all scenarios the Local Application only interacts through DAI primitives.

Chapter 4

4 Adaptive Packet Video Transport over IP Networks

Rapid Advance in digital video coding and networking technologies is leading to the development of a wide range of new generation audiovisual services and technologies to support them. Such applications include wired / wireless videoconferencing, interactive digital TV, remote diagnosis / surgery, distance / remote sensing, process monitoring and tele-education. Many of these applications involve the use of enabling media coding and content analysis techniques for media compression, indexing, searching and retrieval.

One of the major requirement for the implementation of such applications is the efficient transmission of multimedia content (audio, video, text, image) over a broad rang of telecommunications infrastructure, in particular IP networks such as the Internet.

Multimedia applications have a strict bandwidth, delay, jitter, and loss requirements, which the current IP networks does not guarantee. Today, IP Quality of Service (QoS) mechanisms and architecture (Diffserv, Intserv) are expected to address these issues and enable a wide spread of use of real time IP services. Unfortunately, these QoS control models are not sufficient since they perform on per-IP domain, and not on end-to-end basis. Promising Service Level Agreement (SLA) should address this service provisioning, but in the context of mobile IP multimedia service, the SLA is hard to cope by since there may not be enough resources available in some part of the network as the terminal is moving into.

Therefore, it is important that the multimedia applications be adaptive to system and network resource constraints while insuring that end-user requirements are met.

The key contribution of this chapter is lying in the combination of media content analysis techniques and network control mechanisms for adaptive video streaming transport over IP networks. In order to meet this goal, we have identified the following sub-goals that are addressed and evaluated.

- We design a content-based video classification model for translation from video application level QoS (e.g. MPEG-4 Object Descriptor and / or MPEG-7 Framework) to network system level QoS (e.g. IP Diffserv) (see Section 4.1).
- We design a robust and adaptive application level framing protocol with video stream multiplexing and unequal error protection (see Section 4.2).

- We design an algorithm for fine grained TCP-Friendly video rate adaptation (see Section 4.3).

Thus, this chapter presents our design philosophy in the transport of MPEG-4 video stream over IP network. We enhance the transport layer by building several mechanisms and algorithms that work together in order to achieve a seamless quality of service build around a cross-layer video streaming system. Each mechanism is presented and evaluated both using network simulation and / or experimental network. We finish this chapter by a conclusion which is presented in Section 4.4.

In order to understand our models and mechanisms, readers are invited to examine the *Appendix A* which gives a consistent overview of the MPEG-4 standard.

4.1 A Content-Based Video Classification Model

To implements an efficient transmission of object-based MPEG-4 video over IP networks with QoS management capabilities, the MPEG-4 Audio Visual Objects (AVOs) are classified based on application-level QoS criteria and AVOs semantic descriptors according to AVOs descriptors and MPEG-7 framework [152]. Thus, the classification model lead to a relative priority score (RPS) for each AVO. The MPEG-4 AVOs requiring the same QoS performance (with the same RPS) from the network are automatically classified and multiplexed within one of the IP Diffserv Per Hop Behaviors (PHB). Object data-packets within the same class are then transmitted over the selected transport layer with the corresponding bearer capability and priority level. Thus, we propose to extend the MPEG-4 system architecture with a new "Media QoS Classification Layer". This layer implements the content-based video classification model. In our implementation, the "Media QoS Classification Layer" makes use of a neural network classification model that is transparent to the video application and the network layers.

Classification has been the subject of frequent and profound investigation. It has proved a useful tool in real world applications. In networking environment, packet classification can be used in network elements such as switch and router for packet switching, forwarding and filtering. The proposal [149] presents an architecture that meets this goal. In [150], the authors present some algorithms that can be used for packet classification and can be categorized as basic search algorithms, geometric algorithms, heuristic algorithms, or hardware-specific search algorithms. These algorithms are used in IP services such as firewalls and quality of service.

In the machine-learning mechanisms, classification methods are incredibly used. One well-established approach is Bayesian classification, a technique that has become increasingly popular in the recent years in part due to recent developments in learning with Bayesian belief networks [151]. Another classification method based on similarity such as K-NN (K-Nearest Neighbor), Naïve-Bayes and hierarchical clustering are very used in machine learning.

In classification, neural networks solve many problems that conventional methods cannot, or at least not within acceptable cost or performance criteria. In our design, we have used neural network algorithms for automatic AVO classification.

4.1.1 Video Classification Model Properties

To take benefits from the object-based compression, we propose to classify the MPEG-4 Audio Video Objects (AVOs) at the video server from most important AVO to least important AVO. We deal with the Audio Visual Object as an independent calculation primitive. Several methods can be used for AVOs classification. During scene creation, one can affect the adequate priorities to each object in the scene. For scenes with no assigned object priorities, MPEG-4 objects descriptors and / or MPEG-7 [152] can provide the relevant information needed to compute the relative priority score for each objects.

The MPEG-7 standard describes a generic Description Schemes (DSs) for image, video, multimedia, home media, and archive content. MPEG-7 aims to create a multimedia content description standard in order to facilitate various multimedia searching and filtering applications. We can use this content description to do an intelligent AVOs classification. The main components of the image, video, and multimedia DSs are objects, feature classification, object hierarchy, entity-relation graph, code downloading, multi-abstraction levels, and modality transcoding. Each AVO may have one or more associated features, which are grouped in the following categories: media features, visual features, temporal features, and semantic features. Each feature is described by a set of descriptor.

The user interacts with the MPEG-4 server and can decide at any time to choose some AVOs among several available in the scene. This is the basic kind of classification. The automatic classification is done in the server by the prioritization mechanism which affects a Relative Priority Score (RPS) to each AVO. High RPS value (high priority) are affected to the important AVOs in the scene (e.g. Base layer stream in hierarchical coding) and low RPS value are affected to the less important AVO (e.g. Enhancement Layer stream).

The rest of this section gives the properties of the classification model for flexible, extensible, scalable, and efficient MPEG-4 AVO classification and prioritization. This architecture is very adapted to deal with network Quality of Service and user terminal capabilities. Figure 4-1 shows the new MPEG-4 architecture layers. In this architecture, classification layer is developed between Sync layer and Delivery layer. Classification layer must be aware of the transported media and the adjacent layers. It is a media aware, delivery aware layer. This task is performed by two interfaces.

The interface between Sync Layer and Classification Layer is called "MPEG-4 AVO Classification Interface" it performs a logical MPEG-4 Object identification and retrieval.

The interface between Classification Layer and Delivery Layer is called "MPEG-4 AVO Mapping Interface" it is a logical interface at which the classified MPEG-4 AVO are mapped into various QoS transport mechanism such as IP Diffserv, Intserv, MPLS, etc.

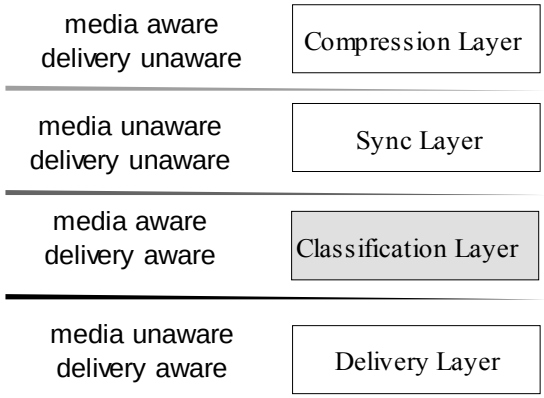


Figure 4-1: Classification layer in the MPEG-4 architecture

The proposed MPEG-4 classification layer satisfies the following features:

- ***Object-Based Abstraction:*** Classification layer deals with MPEG-4 AVOs as a fundamental calculation entity. This allows a flexible, extensible, scalable and simple manipulation of the MPEG-4 scene.
- ***Flexibility:*** The flexibility of the proposed Classification layer is achieved by (1) allowing different classification algorithms to be use (not only the one proposed in this section) (2) enabling or disabling on the fly the classification process.
- ***Extensibility:*** The classification layer can be used to derive new elements for different domains. As example, we mention the video surveillance system, which simply use a classification to detect and track objects within the scene.
- ***Scalability:*** Classifying object into classes according to some criteria increases the scalability of the system. The criteria can be specified in terms of visual features (e.g., size and color), semantic relevance (e.g., relevance to user interest profile, background objects, foreground objects), service quality (e.g., media features, bit rate, ratio loss), and/or temporal features. For example, terminal with less capacity can choose to view only video sequence without logo information, background or others banners.
- ***Simplicity:*** we specify an AVO classification based on media QoS. Additional objects and features can be easily added in a modular and flexible way.
- ***Application Domain:*** The proposed classification layer is generic and supports a very broad range of applications and media transport mechanism.

4.1.2 Audio Visual Object Classification Model

We suppose that we have a collection of MPEG-4 AVOs, which must be classified into classes. The numbers of classes are well defined and each class has a well-known characteristics. A class is viewed as a class of service of a network layer. It can be an IP Diffserv class (Best Effort, EF or AFx class), an ATM class of service (ABR, VBR, UBR, etc.), or any others mechanisms that deliver network QoS. Figure 4-2 shows a collection of MPEG-4 objects described by a collection of object descriptors. These AVOs can be classified according to some attributes (features). The

attributes can refer to the QoS parameters (Bandwidth, loss, and jitter) or any others information founded in their object descriptors or provided by the MPEG-7 tools.

In MPEG-4, the description is done at two levels: **Structure Level** and on **Semantic level**. Structure Level indicates how the scene is composed and how the AVO are arranged in the scene in term of both special and temporal location. The semantic level interests on how the various streams are configured and information regarding the manner that must be delivered to the user. In particular, it describes the expected QoS requirement from the network.

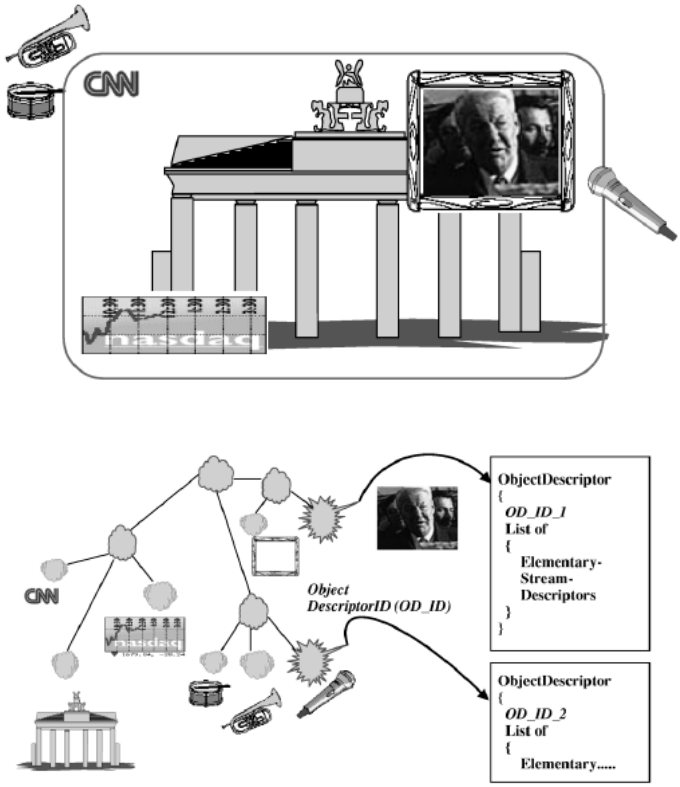


Figure 4-2: MPEG-4 AVO with corresponding objects description

The MPEG 4 video coding standard provides an object-based representation of the video scene by allowing the coding of AVOs separately. Texture and shape coding in MPEG-4 are very similar to the coding of frames in MPEG-2. Temporal instance of a video object is called VOP (Video Object Plane). VOP is divided into macro, luminance, and chrominance blocks. VOP supports intra coded (I-VOP) temporally predicted (P-VOP) and bi-directionally predicted (B-VOP) Frames. The different contents of the video data stream don't have the same importance for the quality of the decoded video. The damages caused by some data loss in a reference picture (I-VOP or P-VOP) will affect subsequent picture(s) due to inter-frame predictions. Subsequently, I-Frame must be protected more than P-Frame and P-Frame more than B-Frame. Let us consider now the example of video object coded with layered wavelet transform techniques. The most important layer contains the low frequency sub-band of the picture, called *Base Layer* (BL). Other layers, which represent a hierarchical level of resolution of the wavelet transform, are less important. These layers are called *Enhancement Layers* (EL).

This is the second step of preparing the MPEG-4 Access Unit (AU) to be transmitted over the network. It operates within a single audio-visual object. The first step is handled by the

classification layer which classifies the object between them. As we said, the result of the classification is a set of AVOs sorted according to their importance in the scene. The classification layer affects a final relative priority score (RPS) to each Access Unit to apply a differentiation mechanism. This priority score reflects both the priority of particular AVO in the scene and the priority of a single frame type (I, P, B or hierarchical stream if any BL or EL). In the rest of this Section, we will interest to the prototype implementation which is based on a neural network algorithm.

Let T = {(x,c)} be a training set of N labeled vectors, where $X \in \Re^n$ is a feature vector $x = (x_1,x_2,\ldots,x_n)^t$ and $c \in T$ is its class label from an index set T (X is an AVO, and T is the set of available classes (e.g. Diffserv PHB), in our context) . The variables x_i are referred to as attributes (e.g. QoS features of each AVO). A class is modeled by one or more prototype, which have $\{U_1,U_2,\ldots U_m\}$ as features.

A classifier is a mapping function called C defined as C: $R^n \rightarrow$ T, which assigns a class label in T to each vector in R^n (a vector is an MPEG-4 AVO features). Typically, the classifier is represented by a set of model parameters $\Lambda = \{\Delta_k\}$. The classifier specifies a partitioning of the feature space into regions $R_j = \{ x \in R^n : C(x) = j\}$ where $\bigcup_j R_j \equiv R^n$ and $\bigcap_j R_j \equiv \Phi$.

It also induces a corresponding partitioning of the training set into subset T.

There are different methods based on this definition, which allow an automatic classification of Vectors. We present in what below the classification method that used Radial Basis Function (RBF) classification.

Figure 4-3 shows the RBF classifier. A vector to be classified is passed to a set of basis functions, each returning one scale value φ_j j=1, 2,…,l. The concrete choice of the basis function is not critical; the common implementations prefer the radial basic function (the Gaussian "bell" functions). Hence the name: RBF network, with :

$$\varphi_j(x) = e^{-\frac{D^2}{2\sigma_j^2}}$$

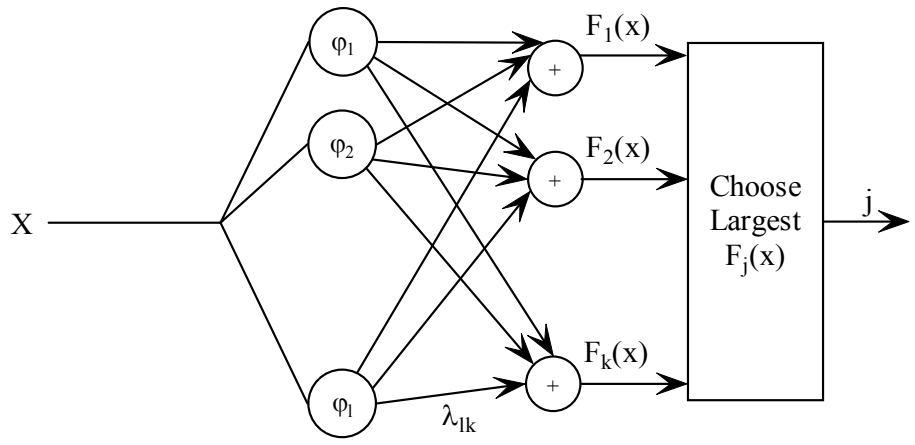


Figure 4-3: RBF architecture for classification

In RBF, $D = \|X - U_k\|^2$ is the distance between X and U_k. φj(x) measures similarity, i.e. the mapping function. λjk in the Figure is a set of scalar weights that connect each of the receptive fields to the class outputs of the network. The general equation of an output of the neuron j is given by: $F_k(X) = \sum_{j=1}^{n} \lambda_{jk} \varphi_j(X)$

The classifier maps the vector x (an Audio Visual Object) to the class with the largest output: $R_j \equiv \{x \in R^n / F_j(x) \geq F_k(x) \forall k\}$

A neural network takes in it input the feature vector x, produces computing class outputs $\{F_j(x)\}$, and then, classification decisions are made based on the largest output. The classification cannot be accomplished without knowing the characteristic of the underlying transport mechanisms.

4.1.3 Performance Evaluation

This section presents our experiment testbed, and some results got by using our MPEG-4 platform. During experiments, two scenarios are investigated, the first on the standard MPEG-4 framework (without any mechanism of QoS) and the second with the new MPEG-4 framework (with Classification Layer and prioritization mechanisms).

We loaded the network by background traffic to generate congestion. Measurements are done to evaluate packet losses and end-to-end one-way delay from the server to the client.

4.1.3.1 System and Network Models

In our experiments, we have used an MPEG-4 video composed of three Video Objects. Each object is carried through an elementary stream. These objects are (see Figure 4-4 for more details):

(1) Object 1: The logo at the top left of the MPEG-4 Video,

(2) Object 2: The background of the MPEG-4 Video,

(3) Object 3: The person.

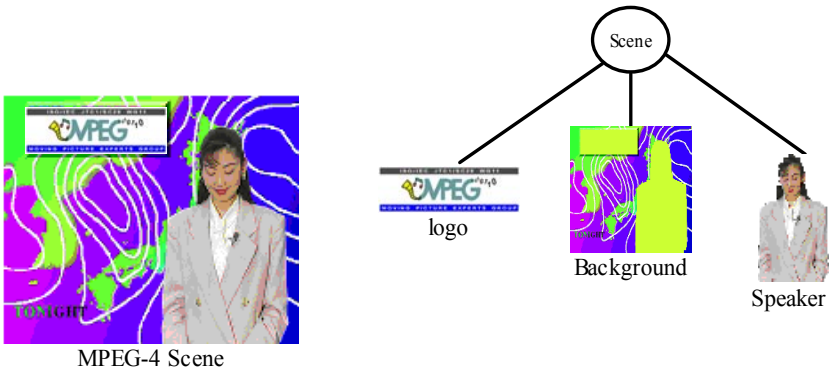


Figure 4-4: MPEG-4 scene used for experiment

An Object Descriptor (OD) describes each of the three video objects. Table 4-1 summarizes QoS value for each object.

In our implementation, each MPEG-4 object is modeled by one parameters vector called $x \in R^3$ containing the flowing structure x= (MAX_DELAY, MAX_DELAY - *PREF_MAX_DELAY*, *LOSS_PROB*) ^{T}. The first element of the vector specify the maximum end-to-end AU transfer delay, the delay variation of the AU is calculated by the formula MAX_DELAY - *PREF_MAX_DELAY*, the third element is the AU loss ratio.

QoS Metric	Object 1 O1	Object 2 O2	Object 3 O3
MAX_DELAY	250ms	200ms	150ms
PREF_MAX_DELAY	200ms	150ms	100ms
LOSS_PROB	$2*10^{-3}$	$3*10^{-3}$	$1*10^{-3}$

Table 4-1: AVO QoS value in QoS_Descriptor

Figure 4-5, shows the throughput of each video object stream used in the experiment. Each stream is read, classed and marked over IP Diffserv network using the set of tools described above. We give more details of the marking in the next Chapter.

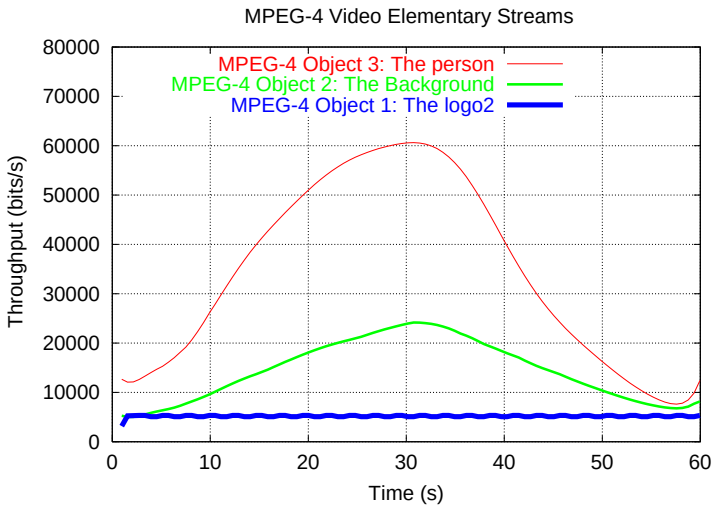


Figure 4-5: Throughput of each MPEG-4 video object

The testbed network is depicted in Figure 4-6. The MPEG-4 server delivers a simulated MPEG-4 AVOs stream on demand to several heterogeneous receivers. The receiver can be simple wired or wireless terminals such as a mobile GSM/GPRS/UMTS capable of rendering MPEG-4 video sequences. The network nodes are IP Diffserv compliant routers. We use Linux-based IP routers with Diffserv implementation as presented in [153], and [154]. The testbed is composed of tow edge routers and a core router running Linux with IP Diffserv components. All Ethernet links are 10 Mb/s.

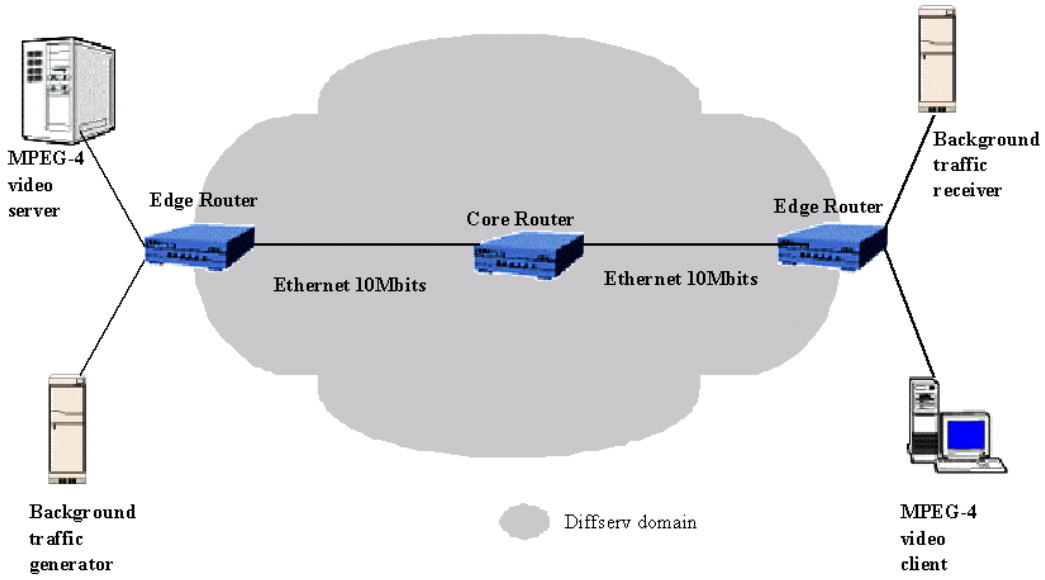


Figure 4-6: Experiment testbed

In the current testbed, we have modeled the Diffserv class by some metrics, which are described in Table 4-2. We have define five different Diffserv QoS classes corresponding to EF (Expedited Forwarding), AF11 (Gold Service), AF12 (Silver Service), AF13 (Bronze Service), and finally BE (Best Effort Service). The transmitted video packets are marked according to two scenarios. In **scenario 1**, the classification layer affects the relative priority score to each packet which is mapped to the Diffserv network. In **scenario 2**, video stream are marked using classical marking algorithm such as Time Sliding Window Three Color Marker (TSWTCM) [130] as explained in details in Chapter 5.

The metrics used to setup the Diffserv domain are taken from [155] and are listed bellow:

(1) **IPTD**: *IP packet transfer delay*,

(2) **IPDV**: *IP packet delay variation*,

(3) **IPLR**: *IP packet loss ratio,*

(4) **IPER**: *IP packet error ratio*,

(5) **BW**: Bandwidth used.

	QoS Classes				
Network Performance Parameter	**EF**	**AF11**	**AF12**	**AF13**	**Best Effort**
IPTD	100ms	150ms	200ms	300ms	Unspecified
IPDV	50ms	50ms	Unspecified	Unspecified	Unspecified
IPLR	$1*10^{-3}$	$1*10^{-3}$	$2*10^{-3}$	$3*10^{-3}$	Unspecified
IPER	$1*10^{-4}$				Unspecified
BW	1,5Mbits	1,5Mbits	1,5Mbits	1,5Mbits	4Mbits

Table 4-2: IP Diffserv QoS class definitions

We run our classification algorithm with the parameters described in the previous section, and we compute the output of each neuron. We choose the largest neuron output which reflects the high similarity. We choose also the value of $\lambda_{1k} = \lambda_{2k} = \lambda_{3k} = 1$ which connect each feature from the vector the network.

The result output of the neuron network is shown is Table 4-3. As said, we choose the largest output from the network. According to these results we can see that O1 is marked with AF11 PHB and that O2, O3 are both marked with AF12 PHB. Regarding the execution time, we have measured the mapping of 100 AVO is about 30 millisecond. The complexity of this algorithm is $O(n^2)$.

	EF	**AF11**	**AF12**	**AF13**	**Best**
O_1	3.96	3.98	4.02	0.70	2.60
O_2	3.90	3.98	4.00	0.69	2.60
O_3	3.81	3.90	3.60	0.62	2.49

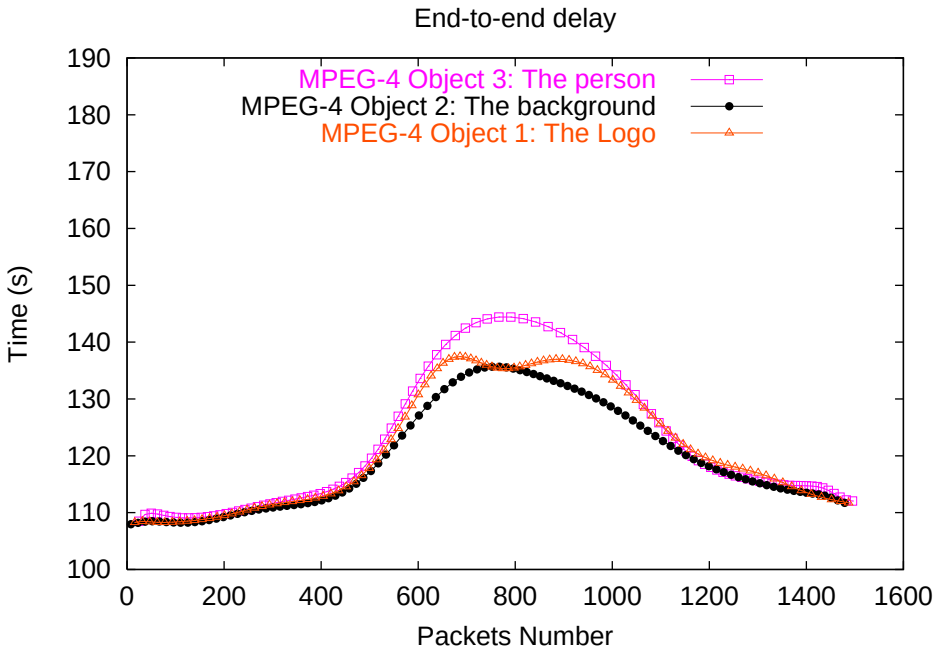
Table 4-3: Results output of the RBF network

4.1.3.2 Experimental Results

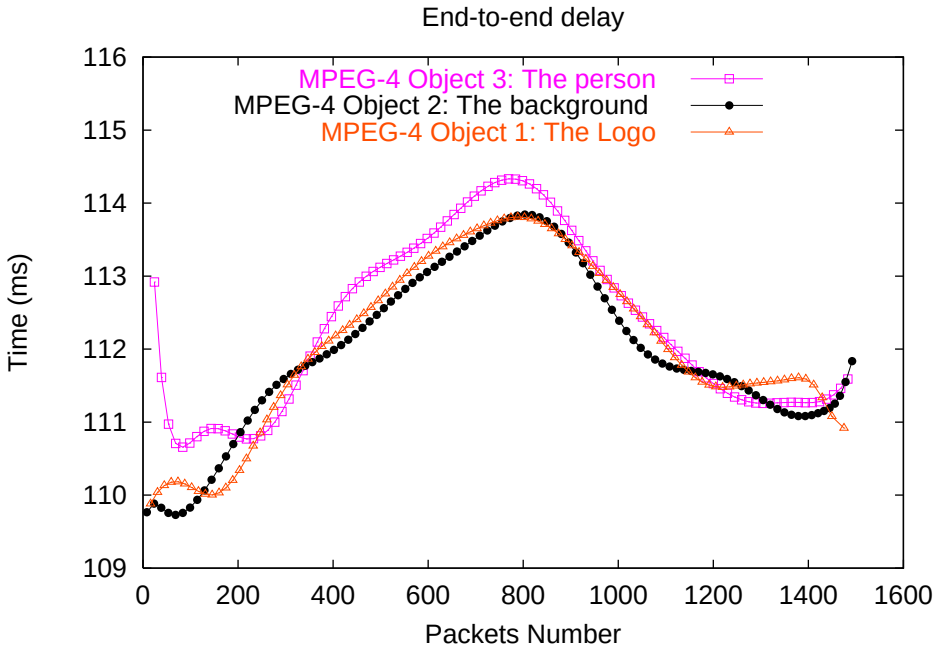
Regarding the performance measurements, we will interest on the network parameters such the end-to-end one-way delay encountered by video packet between the server and the destination, and the packet loss probability for each video object stream. This performance metrics are given only for comparison purpose between the scenario with classification (scenario 1) and without classification (scenario 2) model.

Figure 4-7 shows the end-to-end video packet transfer delay in both scenarios. We remark that the end-to-end transfer delay increases when the amount of video sent by the server increases (time t=30s). The network load is about 85% (8,5 Mbits). When comparing Figure 4-7 (a) and (b), we note that using the classification and prioritization mechanism (scenario), we get the maximum QoS expected. Important streams (AVOs) will be transmitted as soon as possible to the player and with respect to the QoS required by it. The mean transfer delay is about 114 s in scenario 1 and 120s in scenario 2.

Figure 4-8 enforces this measurement by providing loss ratio. We notice that important video objects are protected against loss since they are marker with low drop precedence and this in case of scenario 1. In scenario 2, the loss can affect important object during network congestion (time=30s). This metric, shows a better protection of relevant video objects in the scene during transmission and network congestion.

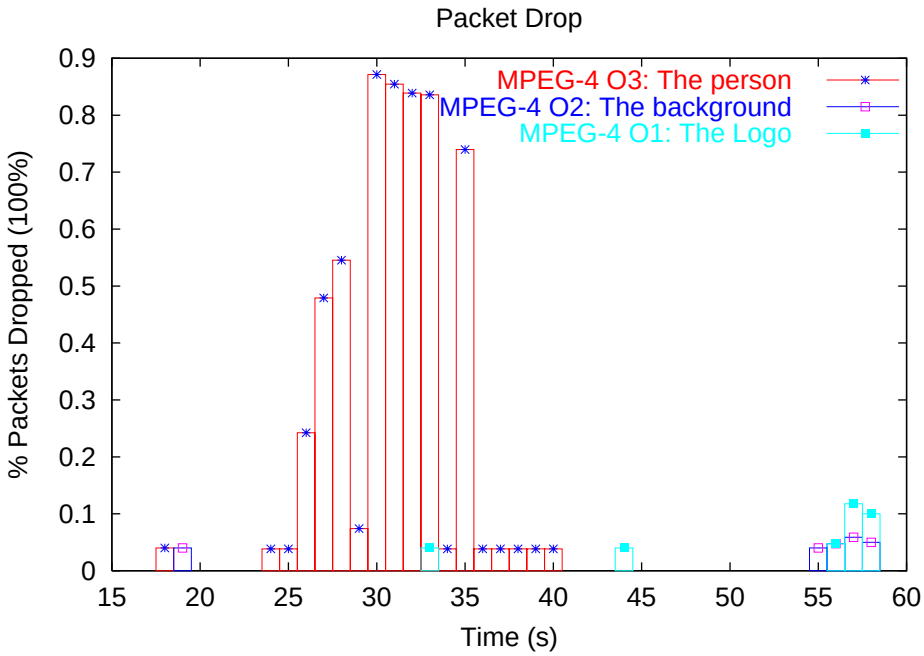


(a) Standard MPEG-4 framework

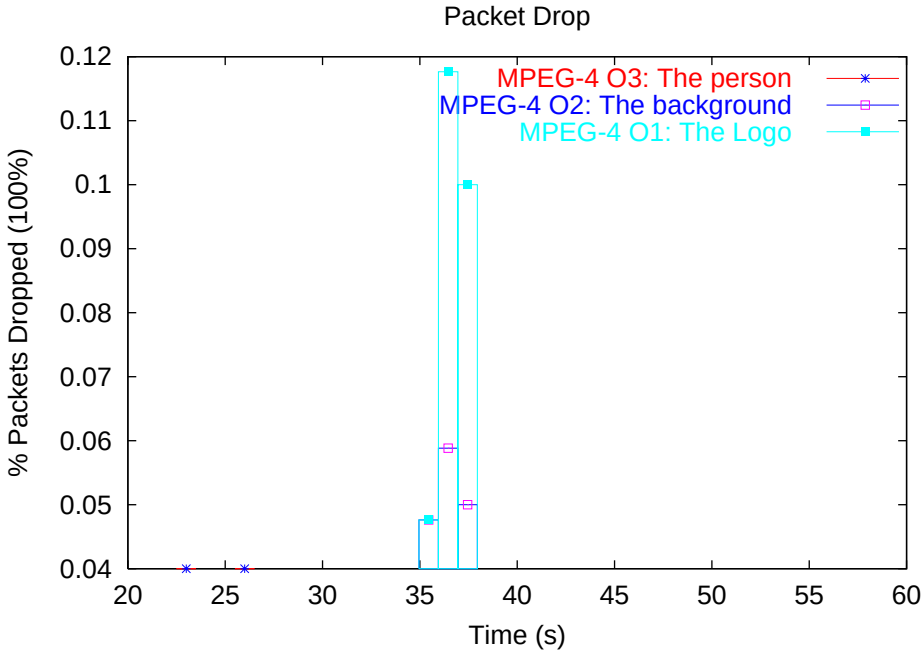


(b) MPEG-4 framework with Classification Layer

Figure 4-7: End-to-end transmission delay



(a) Standard MPEG-4 framework



(b) MPEG-4 framework with Classification Layer

Figure 4-8: MPEG-4 AVO packet loss ratio

4.2 An MPEG-4 Application Level Framing Protocol

As defined in the MPEG-4 Architecture [22],[23],[24], the compression layer compresses the visual information and generates elementary streams (ESs), which contain the coded representation of an AVO. The Elementary Streams are packetized as a sequence of SL-packetized streams at the

Sync Layer. The SL-packetized streams provide timing and synchronization information, as well as fragmentation and random access information. The SL-packetized streams are multiplexed into a FlexMux stream at the Delivery Layer, which is then passed to the transport protocol stacks such as RTP/UDP/IP. The resulting IP packets are then transported over the network.

At the receiver side, the video stream is processed in the reversed manner before its presentation. The received AVOs are decoded and buffered before being composed by the player.

To transmit MPEG-4 stream over the network, it is necessary to adapt this stream to the network. This is performed by the Application Level Framing (ALF) protocols. ALF protocol allows the application to get its own control over mechanisms that traditionally fall within the transport layer e.g., loss detection and recovery. ALF differ from one stream to another. It is specific to a particular stream format and application. A common ALF used for multimedia application is the RTP protocol.

There are a number of RTP packetization schemes for MPEG-4 data. Some works presented in [156], [157], [158], [159] and [160] specify how the MPEG-4 streams should be fragmented and packetized to be conveyed over IP network. We summery these works as follows:

- MPEG-4 System over RTP (Section 4.2.1.1)
- MPEG-4 System over RTP with Error Protection (Section 4.2.1.2)
- RTP Payload Format for MPEG-4 Audio/Visual Streams (Section 4.2.1.3)
- RTP Payload Format with Reduced SL Header (Section 4.2.1.4)
- RTP Payload Format for MPEG-4 FlexMultiplexed Streams(Section 4.2.1.5)

It is clear that many packetization schemes can be implemented together in one terminal. Each packetization scheme is basically adapted to a particular media stream. This technique is called media-aware packetization. For example, a video object plane is fragmented at recoverable sub-frame boundaries to avoid any error propagation.

It is likely that several RTP packetization schemes will be needed to suit the different kinds of media types and encoding. Also, it is clear that the video packetization scheme is not the same as the audio packetization scheme. Thus, we proposed and new RTP payload scheme for MPEG-4 Video transport over IP and for MPEG-4 audio.

4.2.1 Related Work

4.2.1.1 MPEG-4 System over RTP

This is the generic approach for encapsulating audiovisual data over RTP [161]. MPEG-4 SL-PDU packets are directly encapsulated into RTP packet without optimization. This approach is implemented by using the transmission time or the MPEG-4 composition time to initialize the RTP timestamp field. However, due to the duplication of header information, like sequence numbering and time stamps, this approach increases the overhead and it is not considered as efficient.

4.2.1.2 MPEG-4 System over RTP with Error Protection

The design of this payload format has been proposed in [160]. It has been actually inspired by previous proposals for generic payload formats as presented in the previous subsection. Additionally, authors attempts to federate different error control approaches under a single protocol support mechanism. The rationale for this payload format consists in protection against packet loss with a protocol support easily adaptable to varying network conditions, for both "live" and "pre-recorded" visual contents. Despite of offering additional error control features, this approach requires to be aware of the characteristics of the network layer and the path MTU (Maximum Transfer Unit).

4.2.1.3 RTP Payload Format for MPEG-4 Audio/Visual Streams

This approach is defined in [156]. It remains actually the only IETF RFC standard for transporting the MPEG-4 Audio/Visual streams over IP networks. This specification seems to be the most suitable and realist for coordination with the existing Internet QoS. It uses the MPEG-4 Simple profile Level 1 coding with only one video object in the scene (as in MPEG-2). Nevertheless, it does not use MPEG-4 systems at all. The RFC 3016 [156] provides specifications for the use of RTP header fields and fragmentation rules for MPEG-4 audio and video. MIME (Multipurpose Internet Mail Extension) type registration and SDP (Session Description Protocol) are used for the MPEG-4 session establishment. This is useful to configure the decoding parameters associated to a particular ES through the RTP session initialization. In this approach, MPEG-4 audio and video streams are not managed by MPEG-4 systems Object Descriptor but by H.245 session signaling protocol or other out of band means. The BIFS language is also not used to manage the scene that consequently reduces video content manipulation and interaction.

The fragmentation rules described in this document are flexible. It defines the minimum rules for preventing meaningless fragmentation. LATM (Low-overhead MPEG-4 Audio Transport Multiplex) manages the sequences of audio data with relatively small overhead. In audio-only applications, it is desirable for LATM-based MPEG-4 Audio bit streams to be directly mapped onto the RTP packets without using MPEG-4 Systems. This help reducing the overhead caused by the Sync layer.

4.2.1.4 RTP Payload Format with Reduced SL Header

This approach is represented by the MPEG-4 generic payload format described in [157] and its companion simplified draft [159]. Both of them are at status of Internet draft and seem to be an alternative for [156]. The approach presented in [159] implements a subset of the [157] features and addresses the AAC (Adaptive Audio Coding) and CELP (Coding Excited Linear Predictive audio) audio streams encapsulation over RTP.

The approach recommends the use of few significant fields of the SL header with an eventual mapping to RTP header and RTP payload header. The RTP payload header (reduced SL Header) is fully configurable and can fit to different ESs types (video, audio, BIFS and ODs). In addition, this payload format can be configured to be compatible with [156].

With this payload format, only a single MPEG-4 elementary stream can be transported over a single RTP session. Information on the type of MPEG-4 stream carried in RTP payload is

conveyed by MIME format parameters through SDP message or others means. It is possible to carry multiple Access Units originating from the same ES in one RTP packet.

4.2.1.5 RTP Payload Format for MPEG-4 FlexMultiplexed Streams

The approach [158] is a former Internet draft. It recommends encapsulating an integer number of FlexMux packets in the RTP payload. Inversely to other payload formats approaches, this one provides an elementary streams multiplexing over an RTP session and use the complete MPEG-4 terminal specification (i.e. the Sync Layer and FlexMux syntax).

4.2.1.6 Feature Comparison and Opens Issues

MPEG-4 scene may involve a large number of objects, which are encapsulated in several distinct Elementary Streams (ESs). For instance, a basic audio-visual MPEG-4 scene needs at least 5 streams: a BIFS stream for scene description that specifies the number and location of individual media objects present in the scene, two additional streams for describing video and audio objects respectively (Object Descriptor), and finally, two other streams that encapsulate video and audio data. All these elementary streams, originated from the MPEG-4 codec, have to be transmitted to the destination player. Transporting each ES as an individual RTP session may be unpractical or inefficient. Allocating and controlling hundreds of destination addresses for each MPEG-4 session is a complex and demanding task for both source and destination terminals.

The approaches described above are Internet draft and some of them are now obsolete since a number of open issues were identified with these proposals and which are summarized here:

- **Use of MPEG-4 systems vs. elementary streams**. MPEG-4 has a complete system model, but some applications just desire to use the codecs. It is necessary to generate payload formats for both cases. MPEG-4 encompasses codecs which may have an existing RTP payload format, for example H.263 codec. This can lead to stop the use of MPEG-4 specific packetization. In addition, for error resilience, it is desirable to packetize in a media aware manner which does not imply a choice between systems or elementary streams.

- **Multiplexing multiple streams**. The IETF has identified five multiplexing options that can be used for MPEG-4: GeRM (Generic RTP Multiplexing), FlexMux (Flexible Multiplexing), PPP Mux with Compressed RTP, don't multiplex, and don't multiplex, but compress headers.

- **Grouping within a stream**. In order to amortize header overhead and to aid error resilience it is necessary to implement grouping. The open issue relating to grouping is how to group the MPEG-4 Access Unit?

- **Fragmentation**. It is necessary to fragment a codec bit-stream in a media aware manner to achieve error resilience. We believe that the choice of fragmentation is a matter for the encoder, and that MPEG-4 should be able to fragment accordingly.

- **Error protection**: There are two choices to apply an existing error protection mechanisms using packets-based protection such as parity FEC or to apply a specific FEC within the payload.

- **Some MPEG-4 elementary streams must be reliably delivered**. An example of theses streams, we can find a control streams BIFS and OD, or streaming media such as Java class (MPEG-J) files. It is necessary to transport this media in reliable manner. There is no works that define clearly how to transport this kind of media.
- **Addressing the issue of DMIF session management**.
- **Timing model**. MPEG-4 and RTP have different timing models. It is desired to synchronize MPEG-4 data with data using a native RTP packetization we must align the models (capture time vs. composition time).

Figure 4-9 illustrates the approaches of encapsulating MPEG-4 stream over IP compared to the ISO MPEG-4 standard approach. To deal with some of the open issues cited later, we have developed an RTP payload formats for transporting MPEG-4 Audio Visual Object over RTP protocol called RTP4Mux.

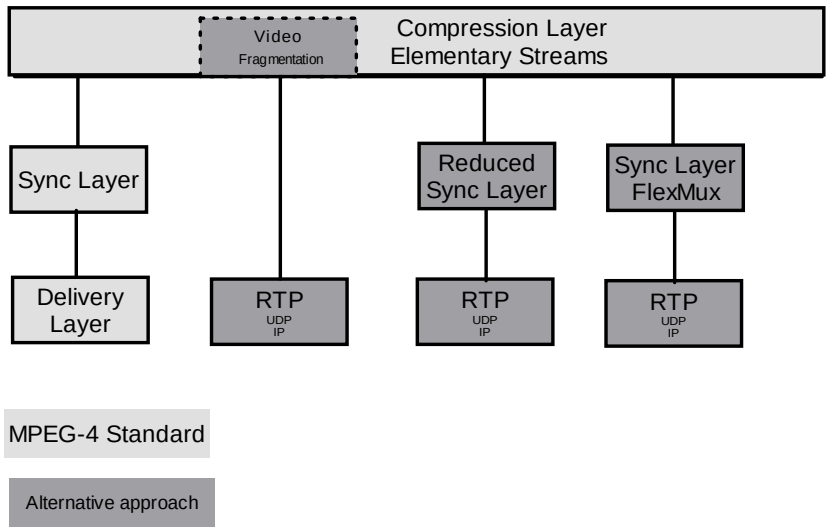


Figure 4-9: Concurrent approaches for encapsulating MPEG-4 stream over the IP networks

4.2.2 An RTP Payload for MPEG-4 Audio / Visual Object

Classical payload format supports a fragmentation mechanism where the full AUs or the partial AUs passed by the compression layer are fragmented at arbitrary boundaries. This may result in fragments that are not independently decodable. This kind of fragmentation may be used in situations when the RTP packets are not allowed to exceed the path-MTU size.

However, this fragmentation is not recommended for error resilience. It is preferable that the compression layer provides partial AUs, in the form of typed segments, of a size small enough so that the resulting RTP packet can fit the MTU size.

Consecutive segments (e.g. video frames) of the same type can be packed consecutively in the same RTP payload. The compression layer should provide partial AUs, of a size small enough so that the resulting RTP packet can fit the MTU size. Note that passing partial AUs of small size will also facilitate congestion and rate control based on the real output buffer management. RTP packets that transport fragments belonging to the same AU will have their RTP timestamp set to

the same value. This timing information is obtained only when AUs are encapsulated in RTP packet. AUs are then transmitted to the RTP layer by the compression layer, with indication of the boundary, the payload type and other control information.

Among the open issues related to MPEG-4 encapsulation and fragmentation, we have identified in the previous section, the grouping and the multiplexing issues. Hence, an appropriate ES packetization and grouping (aggregation) payload is essential for an optimal transport of MPEG-4 streams over the IP networks. The packetization process must address the overhead reduction and provides a generic RTP payload format, which offers some of the Sync Layer features. Furthermore, an efficient ESs multiplexing mechanism is much suitable for transporting multiples MPEG-4 ESs in a single RTP session such as low bit rate audio. This will facilitate the ESs management, optimize the RTP payload use, and favor the MPEG-4 terminal scalability.

In this subsection, we present a new RTP payload format for MPEG-4 audio-visual object, namely, RTP4mux protocol. RTP4mux is based on ESs aggregation and offers both efficiency and robustness for MPEG-4 transport over the low bit rate IP networks such as WLAN (Wireless Local Area Network).

The main problem that an encapsulation scheme must address is the overhead when transporting small AU size. Table 4-4 illustrates the low bit rate MPEG-4 streams transport overhead over RTP. It is clear that the AUs concatenation and grouping reduces the total packetization overhead.

Audio Object	Audio Payload Length	Overhead (RTP/UDP/IP)	Overhead Rate
AAC (64 kbit/s, 24 kHz)	342 bytes (average)	12+8+20 bytes	10,9%
CELP (6 kbit/s, 8 kHz)	15 bytes (fixed)	12+8+20 bytes	72,7%

Table 4-4: Overhead of AAC and CELP audio streams

When AAC is used for encoding of a stereo audio signal at 64 kbit/s, AAC frames contain an average of approximately 342 bytes (with 1024 sample/frame). On a network with 1500 bytes MTU (Maximum Transfer Unit), a four AAC frames can be carried in one IP packet. This allows optimizing network resources and bandwidth usage. However, with the low bit rate MPEG-4 streams (e.g. AAC, CELP, Facial Animation, etc.), the interval between a successive AUs generation may be important. Otherwise, the concatenation of multiple AUs from a single elementary stream will introduce a supplementary delay, which is intolerable in a real-time and interactive multimedia communications. For example, the concatenation of 30 CELP AUs will induce 600 ms packetization delay. In addition, since the low bit rate MPEG-4 streams is expected to be largely deployed on the IP networks, optimizing the bandwidth usage trough a good RTP payload exploitation is indispensable.

Within RTP4mux, we propose to aggregate several MPEG-4 elementary streams on a single RTP session. We achieve a concatenation of several AUs, from several ESs, in the same RTP payload (i.e. It is possible to convey several ESs in the same RTP packet). The motivation for packetizing multiple AUs in the same RTP packet is the overhead reduction. While the motivation for concatenating AUs from different ESs is minimizing the end-to-end ESs transmission delays. Furthermore, this aggregation scheme minimizes the dependency between adjacent RTP packets,

which mitigate the dependency of MPEG-4 decoder on any lost packet. In our case, the regrouping of particular streams in the same RTP payload is made as follows:

Based on taking together elementary streams that are tightly synchronized, such as facial animation stream and its associated CELP speech stream, etc. Or, based on the Quality of Service considerations (i.e. AUs originating from ESs with the same QoS requirements are grouped together in one RTP packet). This facilitates the QoS mapping over the network.

When transporting each ES in a single RTP session, like used in [156] [157], and [159], the multiplexing of the ESs is achieved through signaling mechanism. The ES_ID of each MPEG-4 stream is sent at the RTP session initialization by using the MIME format parameters and SDP messages. Thus, due to the dynamic nature of the MPEG-4 scene with the intervention of a new Media Object (ES) during the MPEG-4 sequence. This signaling mechanism involves additional delays not suitable for real-time communications.

Before giving implementation details of the RTP4mux protocol, we describe briefly the principal RTP4mux features in what follows:

- **Elementary streams fragmentation***:* RTP4mux takes in consideration the nature of the transported streams. In addition, it behaves better against the wireless network losses through an appropriate ESs multiplexing and encapsulation schemes. The fragmentation provides (1) ESs fragmentation into independently decodable entities (2) ESs interleaving over a single RTP payload.
- **Elementary streams multiplexing**: It is a two levels ESs multiplexing scheme, which is based on (1) multiplexing several ESs in a single RTP session and (2) encapsulating different AUs (originating from different ESs) in the same RTP payload. This multiplexing scheme provides better bandwidth exploitation and reduces the end-to-end delays.
- **Elementary streams encapsulation**: it provides a transport mechanism over the RTP/UDP/IP stack. The main purpose of the encapsulation scheme is to allow ESs interleaving which reduces loss impact on the received stream.
- **Elementary streams synchronization**: It is a synchronization based on the MPEG-4 system (i.e. it provides inter and intra ESs synchronization). RTP4mux uses the transport-level synchronization (provided in the RTP header) and the Sync Layer synchronization (provided in the reduced SL header).
- **Elementary stream error protection**: it achieved through an unequal error protection mechanism. We protect each stream according to it priority score. The more the stream is important in the scene, the more the protection is high.
- **QoS provisioning**. In our cross-layer system, the QoS is assured in the Network Using the classification layer, each RTP packet is marked according to it relative priority score.

4.2.2.1 Fragmentation

We propose to carry in the RTP payload several reduced SL packets, which are originated from different ESs (see Figure 4-10).

Our packetization mechanism reduces the RTP packet loss's consequences at the receiver side. It offers losses tolerance without deploying an AU's interleaving mechanism like used in [159]. This is due to the packetization mechanism which carries several AUs originating from different ESs in the same RTP packet. The packetization process must take care of the path MTU constraints. In our case, the loss of a single RTP packet doesn't involve successive losses of AUs from the same elementary stream. Loss of consecutive AUs from the same ES can cause a poor quality at the player.

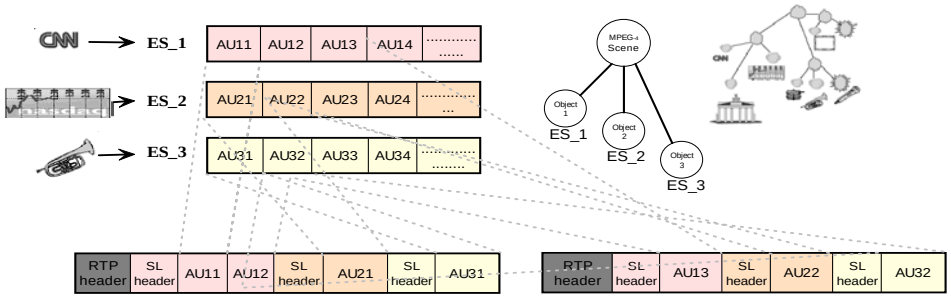


Figure 4-10: Elementary stream encapsulation process

It is clear that our packetization mechanism optimizes the bandwidth usage through the overhead minimization by (1) sharing of a single RTP packet header by a multiple reduced SL packets and (2) sharing of single reduced SL packet header by several AUs.

4.2.2.2 Multiplexing

When deploying the object-based MPEG-4 video (i.e. with several audiovisual ES) over a IP network, multiplexing is unavoidable due to the large number of ESs that may be used in each MPEG-4 session. Our proposed encapsulation process involves an ESs multiplexing design, which is performed at 2 levels. First, we provide an MPEG-4 elementary streams multiplexing over a single RTP session. Second, we transport different elementary streams in the same RTP payload. However, this multiplexing scheme involves the identifying of the different AUs that are carried in the RTP payload through their ES_IDs signaling.

The ES_IDs signaling is achieved at the MPEG-4 session initialization through SDP message. This is possible due to transporting single ES within a single RTP session. In this case, the receiver knows in advance the configuration and the organization of the RTP payload (i.e. presence and size of each RTP payload field). To permit a de-packetization without ambiguity, we have identified two distinct ES_ID signaling approaches. The first approach is ES_ID signaling through the RTP header (in SSRC filed) and the second one is ES-ID signaling through the RTP payload. These two approaches are described in what follows.

4.2.2.2.1 ES_ID Signaling through the RTP Header

In a similar manner with the FlexMux tools (codeMux mode), we transmit in the RTP packet's header a code value that indicates the RTP payload organization. In such a way to make a

correspondence between each ES_ID and its associated reduced SL packet, which is carried in the RTP payload. This code field may be mapped into the *SSRC* RTP header field (see Figure 4-11). Nevertheless, this approach induces additional out of band signaling of the correspondence tables between the codeMux field and the associated ES_IDs. In addition, the dynamic behavior of the MPEG-4 scene (e.g. apparition of a new ESs during the MPEG-4 session) induces a continuous signaling of the correspondence tables, which are exposed to loss. This will result in a multiplexing blocking, then a decoding blocking.

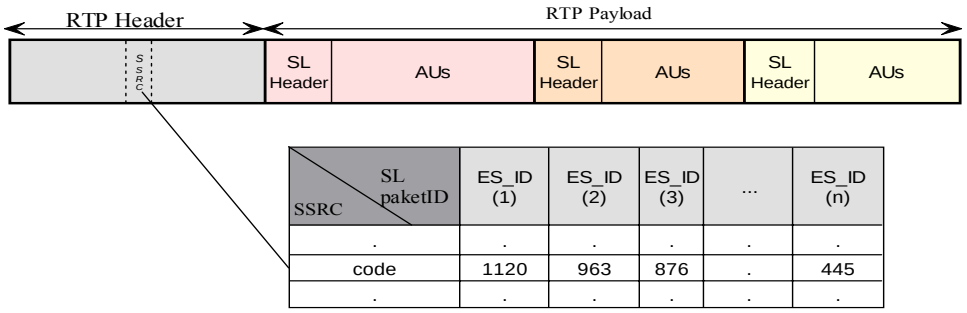


Figure 4-11: ES_IDs signaling though the RTP header.

4.2.2.2.2 ES_ID Signaling through the RTP Payload

This ES_ID signaling approach offers a de-packetization without ambiguity through the transmission of the ES_ID in each reduced SL packet (see Figure 4-12). Otherwise, this mechanism doesn't needs any other out of band signaling stream. This approach will be adopted for the remaining of our RTP payload format definition.

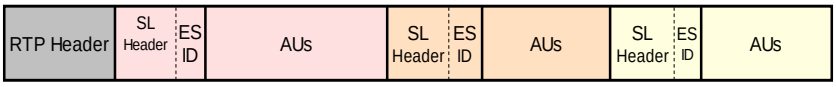


Figure 4-12: ES_ID signaling through the RTP payload.

4.2.2.2.3 MPEG-4 Elementary Streams Synchronization

Inter and intra-ES synchronization is crucial for deploying audiovisual services over IP networks. In corporate communications such as videoconferencing and video surveillance, the audiovisual equipments, such as camera and microphone associated to a talking person, are connected directly to the RTP4mux Server. The RTP4mux server provides interleaving, multiplexing, and synchronization of the different ESs into one MPEG-4 mux stream.

4.2.2.2.4 RTP Packet Format

RTP Packet format is described in Figure 4-13. It is composed of the following fields:

Marker (**M**) bit: The M bit is set to 1 to indicate that the RTP packet payload includes the end of each Access Unit carried in the RTP packet. As the payload either carries one or more complete Access Units or a single fragment of an Access Unit, the M is always set to 1, except when the packet carries one or multiple single fragment of an Access Unit that is not the last one.

Timestamp: Indicates the sampling instance of the first AU contained in the RTP payload. This sampling instance is equivalent to the CTS (Composition Time Stamp) in the MPEG-4 time domain.

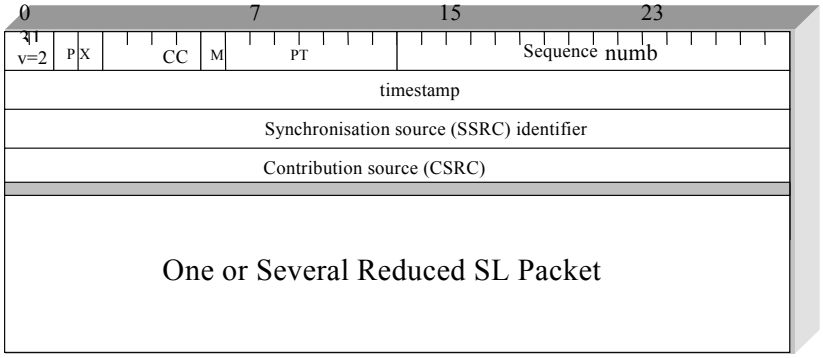
SSRC, CC and **CSRC** fields are used as described in the generic RTP header RFC 1889.

We define in the reduced SL packet the following fields (see Figure 4-13 (b)).

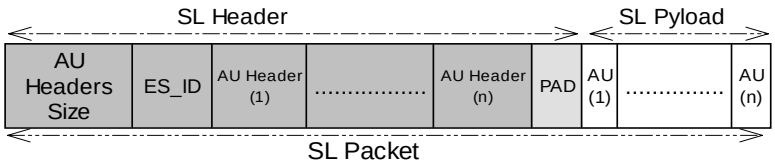
AU Headers Length: is a two bytes field that specifies the length in bits of the concatenated AU-headers. If the concatenated AU-headers consume a non-integer number of bytes, up to 7 zero-padding bits must be inserted at the end (**PAD** field) in order to achieve byte-alignment of the AU Header Section.

ES_ID: is a two bytes field that specifies the ES_ID associated to the AUs carried in the reduced SL packet. This field is common to all the AUs encapsulated into the SL packet. This minimizes the overhead. The ES_ID field is the only one that must be present in the SL packet header.

For each Access Unit in the SL packet, there is exactly one AU-header. Hence, the n^{th} AU-header refers to the n^{th} AU.



(a) Header information



(b) Payload format: reduced SL packet

Figure 4-13: RTP packet format.

We define in the AU Header the following fields (see Figure 4-14):

AU Size: indicates the size in bytes of the associated Access Unit in the reduced SL payload.

Index / IndexDelta: indicates the serial number of the associated Access Unit (fragment). For each (in time) consecutive AU or AU fragment, the serial number is incremented with 1. The AU-Index-delta field is an unsigned integer that specifies the serial number of the associated AU as the difference with respect to the serial number of the previous Access Unit. The Index field appears only on the first AU Header of the reduced SL packet.

CTS-Flag: Indicates whether the CTS-delta field is present. A value of 1 indicates that the field is present, a value of 0 that it is not present.

CTS-Delta: Encodes the CTS by specifying the value of CTS as a 2's complement offset (delta) from the timestamp in the RTP header of this RTP packet. The CTS must use the same clock rate as the time stamp in the RTP header.

DTS-Flag: Indicates whether the DTS-delta field is present. A value of 1 indicates that the field is present, a value of 0 that it is not present.

DTS-Delta: specifies the value of the DTS as a 2's complement offset (delta) from the CTS timestamp. The DTS must use the same clock rate as the time stamp in the RTP header.

We propose to signal each RTP payload configuration through SDP messages at the RTP session initialization.

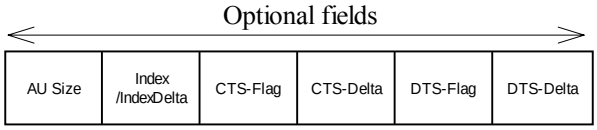


Figure 4-14: AU header's fields.

4.2.2.3 Unequal Error Protection

Error resilience of each *Elementary Stream* associated to one AVO can be enhanced when the sensitive data is protected whereas the less important data is none or less protected, as shown in, [103], [104], [105], [106], [107]. The IETF draft [108] and [109] specify how error protection is unequally applied to different part of the video stream. We extend this idea in case of object based coding (i.e. MPEG-4 AVO). To this effect, the *classification layer* specifies how to assign priority score to each *Access Units* (AU) within an AVO. From such classification, an unequal error protection (UEP) mechanism can be performed through forward error correction.

4.2.2.3.1 Reed-Solomon Codes

The aim of Reed-Solomon (RS) codes is to produce at the sender n blocks of encoded data from k blocks of source data in such a way that any subset of k encoded blocks suffices at the receiver to reconstruct the source data [102]. RS code is called an (n, k) code. RS code (n, k) is defined over the Galois Field $GF(2^q)$ where each block contains q bits. The codeword length n is restricted by $n \leq 2^q - 1$. We choose q to be 8 bits and therefore $n \leq 255$. With this value for q, encoding and decoding are processed easier.

Let $x = x_0 \ldots x_{k-1}$ be the source data, G an $(n \times k)$ generator matrix of the (n, k) RS code, and y the encoded data. Then, y is given by:

$$y = G \cdot x \qquad \text{(Eq.1)}$$

G consists of two parts. The first part is the $(k \times k)$ identity matrix I_k. The second part is an $(h \times h)$ matrix, with $h = n - k$. G is given by Eq.2.

When ***G*** is used as generator matrix, the blocks of encoded data include a verbatim copy of the source. It simplifies the reconstruction of source data when few losses are expected.

$$G = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \ddots & \vdots \\ \vdots & & & \vdots \\ \vdots & \ddots & 1 & 0 \\ 0 & \dots & 0 & 1 \\ (n-1)^1 & (n-2)^1 & \dots & (n-h)^1 \\ (n-1)^2 & (n-2)^2 & \dots & (n-h)^2 \\ \vdots & \dots & \dots & \vdots \\ (n-1)^h & (n-2)^h & \dots & (n-h)^h \end{bmatrix} \quad \text{(Eq.2)}$$

4.2.2.3.2 Unequal Error Protection Using Dynamic RS Code

We propose an Unequal Error Protection (UEP) in order to handle each Access Unit according to its importance. Let us consider U_i, the i^{th} Access Unit in the flow of priority score p. The main block of the proposed UEP is to determine the values n_i and k_i in such a way that the (n_i, k_i) RS code is efficient. The value k_i is defined as the number of packets in which U_i is broken when no error protection is performed. The value n_i depends on the priority score p. It depends also on the length m_i of U_i because the traffic overhead introduced by redundant data does not become excessive.

Once the efficient **(n_i, k_i)** RS code is found, the coding step begins. We also investigate a packetization method known as block of packets which was introduced in [168]. Data of U_i is placed in k_i horizontal packets (S_1, S_2… S_k). Each packet has the same size of t_i bytes. Padding is added to the last packet if m_i is not a multiple of k_i. Then the **(n_i, k_i)** RS code is applied across these packets, vertically. Figure 4-15: illustrates this technique.

We generate $h_i=n_i-k_i$ redundant packets (R_1, R_2…R_{hi}). After appending the RS codes, result packets are transmitted horizontally with a FEC header. Finally, the packet can be transmitted over a transport protocol such as Real-Time Protocol (RTP) [110].

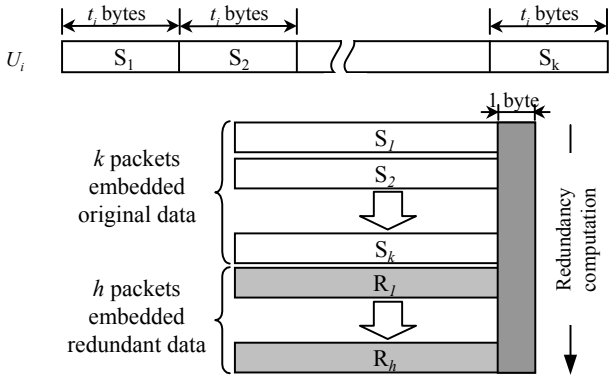


Figure 4-15: Packetization mechanism

FEC header contains both of the Ui sequence number and the values n_i and k_i of the RS code. In case of packet losses, the decoder needs this information to decode correctly the received

packets. If the number of lost packets is not more than h_i, then the decoder will be able to recover Ui. Otherwise, Ui is completely lost. Figure 4-16 shows the format of packets sent on the IP network.

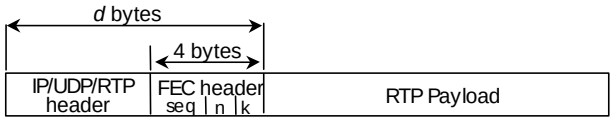


Figure 4-16: Header information and packet format

UEP augments the amount of the traffic sent in the network. To correctly control the volume of transmitted data, we have to provide a certain ratio of traffic overhead called ***r***, for each level of priority score. We assume that moving from one priority score to other increases by a 10 percent ratio. Then, the ratio *r* can be defined by:

$$r = 0.1 \times p \qquad \text{(Eq.3)}$$

Therefore, the traffic overhead is limited to 10 percent (i.e., *r=0.1*) for the data flow of priority score 1, to 20 percent (i.e., *r=0.2*) for the data flow of priority 2, and so on. In order to find the efficient value n_i for the (n_i, k_i) RS code, we proceed as below:

Let $\varsigma^*_{U_i}$ be the reserved byte-budget for error protection. It depends on the number of bytes used to send U_i when no error protection is performed. It is given by:

$$\varsigma^*_{U_i} = r \cdot (k_i \cdot d + m_i) \qquad \text{(Eq.4)}$$

Where *d* is the packet header size (i.e, when RTP/UDP/IP is used with the proposed UEP, *d = (20+8+12+4) = 44* bytes). The relation between the real byte-budget spent on error protection, ς_{U_i}, and the RS code to be used can be stated as follows:

$$\varsigma_{U_i} = (n_i - k_i) \cdot (t_i + d) \qquad \text{(Eq.5)}$$

The error margin between ς_{U_i} and $\varsigma^*_{U_i}$ is $\hat{\varsigma} = \varsigma_{U_i} - \varsigma^*_{U_i}$, that can be positive or negative. It cumulates along the data access unit arrivals. Using the formula (Eq.4) and (Eq.5), the fluctuation of the error margin can be written as:

$$\hat{\varsigma}(n_i) = \begin{cases} 0 \quad , \ i = 1 \\ (r+1) \cdot m_i - n_i \cdot t_i + (r - n_i + k_i) \cdot d + \hat{\varsigma}(n_{i-1}), \ i > 1 \end{cases} \qquad \text{(Eq.6)}$$

To respect the constraint given on traffic overhead, the best value n_i is the one that provides the smallest error margin in formula (Eq.6). Then, n_i is obtained by:

$$\min_{n_i} \left| \hat{\varsigma}(n_i) \right| \quad \textit{such that } n_i \in \aleph, \ n_i \geq k_i, \forall m_i, k_i, t_i \qquad \text{(Eq.7)}$$

With the proposed UEP, the RS code evolves dynamically so the network bandwidth is correctly controlled according to video application requirement.

4.2.3 Performance Evaluation

Intensive simulations are conducted to evaluate the performance of the proposed RTP payload. We have used the network simulator ns2 in which we have implemented the proposed RTP payload. An MPEG-4 Server (NS2 Agent) and an MPEG-4 client (NS2 Agent) are designed for this purpose. The server reads and sends the different MPEG-4 AVOs found in video trace files to the client though an IP network.

4.2.3.1 System and Network Models

For the simulation, we used the network simulation models depicted in Figure 4-17 for evaluating and comparing our proposal with the classical approach described in IETF draft of the RTP Payload Format for Transport of MPEG-4 Elementary Streams [159]. MPEG-4 audio video streams are transported over an IP network using both approaches. Also, the network topologies and parameters are similar for both approaches. Links characteristics between the different network elements (i.e. the channel bit rate and the transfer delay) are illustrated in the Figure 4-17.

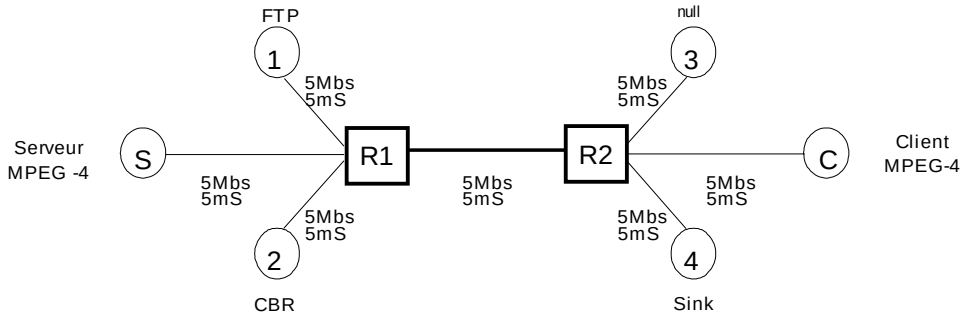


Figure 4-17: IP network model.

The MPEG-4 source terminal is attached to the node "S". It sends a customize MPEG-4 streams to the MPEG-4 destination terminal which is attached to the node "C" according to two scenarios (*scenario 1* and *scenario2*). We include a constant-bit-rate (CBR) traffic over UDP to make the link between the nodes "R1" and "R2" congested A CBR sources allows loading the network differently each time in order to get further information of our packetization scheme.

4.2.3.1.1 Scenario 1

In the scenario 1, we demonstrate the effect of the multiplexing of our RTP payload. We transmit an MPEG-4 scene composed only of CELP stream (6Kbit/s). In order to optimize the bandwidth usage, we maximize the RTP payload exploitation for both approaches. The RTP packet size is fixed to 210 bytes. This permits encapsulating 12 AUs from a CELP stream into one RTP payload. The encapsulation takes care of the RTP payload header fields (i.e. SL packet header fields) using for either [159] or our proposal. In the classical encapsulation algorithm [159], the concatenation of several AUs, from a single CELP stream, into the RTP payload induces the transport of 12 AUs (i.e. 240 ms of voice) per RTP packet. With our encapsulation proposal, we achieve a multiplexing of 3 different CELP streams, which will result with the transport of 4 AUs (i.e. 80 ms of voice) from each multiplexed CELP stream. At the end, we simulate both of the approaches through a network varying conditions.

4.2.3.1.2 Scenario 2

In this scenario, we demonstrate the effect of the unequal error protection (UEP) of our RTP payload. The transmitted MPEG-4 scene is composed of a set of Audio Visual Objects (AVOs). The first object (O1) is a 25-fps video stream composed of 3 hierarchical layers (Base Layer, Enhancement Layer 1 and Enhancement Layer 2). The second object (O2) is also a 25-fps video stream composed of one single layer. Finally, the last object (O3) is an AAC audio stream. The instantaneous throughput of each object during one-minute is described as follows. The full MPEG-4 scene has an average rate of 770 Kbits/s and a peak rate of 1460 Kbits/s. Assume that the audio (O3) has the higher priority score than O2 which have a higher priority score than O1 in the MPEG-4 scene. Diffserv marking uses this information and marks each object stream by a Diffserv code point to reflect a particular Diffserv class of service (high drop, low drop, etc.).

When UEP is performed, it uses the priority value of each A*ccess Unit*. The higher priority score corresponds to higher FEC protection as shown in Table 4-5.

AU types	Object Type	Priority
Base Layer, I-frame	Video	***p**=2*
EL1 Layer, P-frame	Video	***p**=1*
EL2 Layer, B-frame	Video	***p**=0*
CELP Voice Sample	Audio	***p**=0*

Table 4-5: MPEG-4 AVO priority

With respect to this error protection policy, the redundant data generates a global network traffic overhead equal to 7.0 % of the total traffic (i.e., 12.5, 6.0, and 0.0 % for the objects O1, O2, and O3 respectively).

4.2.3.2 Simulation Results

4.2.3.2.1 Result of scenario 1

Figure 4-18 illustrates the end-to-end transmission delay for each AU generated by the CELP stream. This delay was calculated from the AU's coding generation time (CTS) until the AU's decoder reception time. We note that in the Figure 4-18, our proposal provides a better end-to-end delay transmission (means of 0.075s in our proposal compared to 0.175s of the IETF approach). This is especially due to our RTP packetization process (i.e. multiplexing scheme), which takes only 80 ms while in the IETF approach RTP packetization takes 240 ms. Value calculated between the coding generation time and the decoder reception Time.

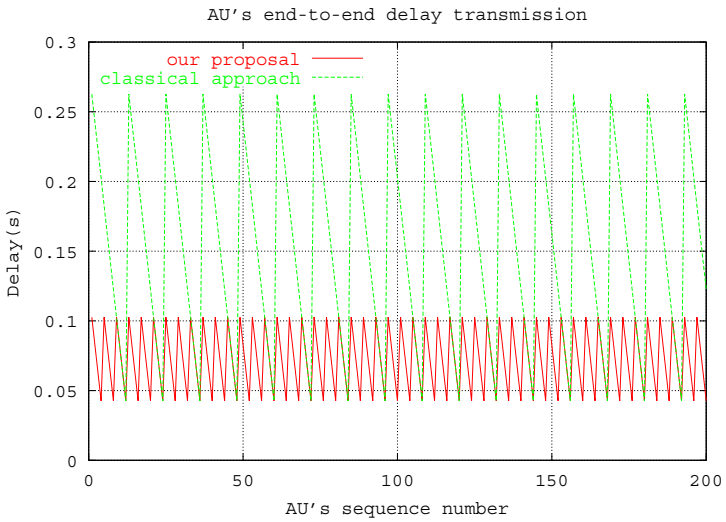


Figure 4-18: End-to-end AU's transmission delays.

In the other hand, the instantaneous losses measurements (see Figure 4-19) reveal a better behavior in our proposal. In Figure 4-20, we experiment a loss of 4 adjacent AUs from a single CELP voice stream per RTP packet lost. While in [159] proposal, the RTP packet loss induces the loss of 12 adjacent AUs from the same CELP voice stream. Thus, the loss of 240 ms of compressed stream will lead to devastating consequences at the decoding side.

Also, when interleaving of AUs is deployed with the [159] approach. Both end-to-end delays and losses tolerance remains an immense inconvenience due to the CELP low bit rate, AU small size, and the number of AUs transported into each RTP packet

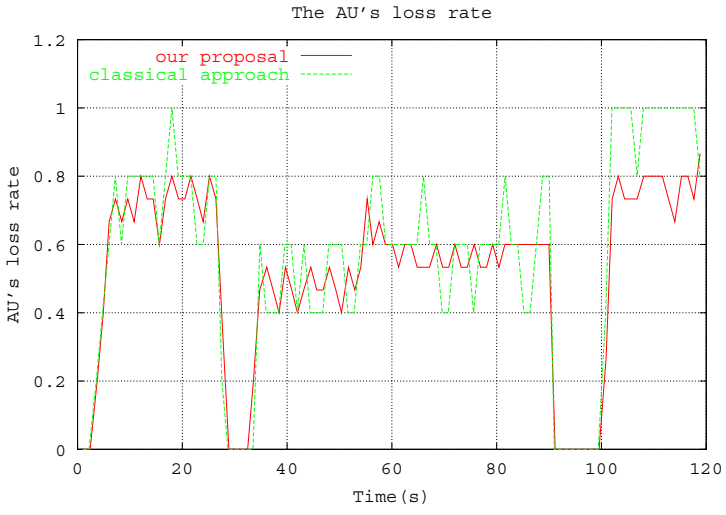


Figure 4-19: Instantaneous AU's loss rate.

Value calculated, every 1,2 seconds, for all the audio conference duration.

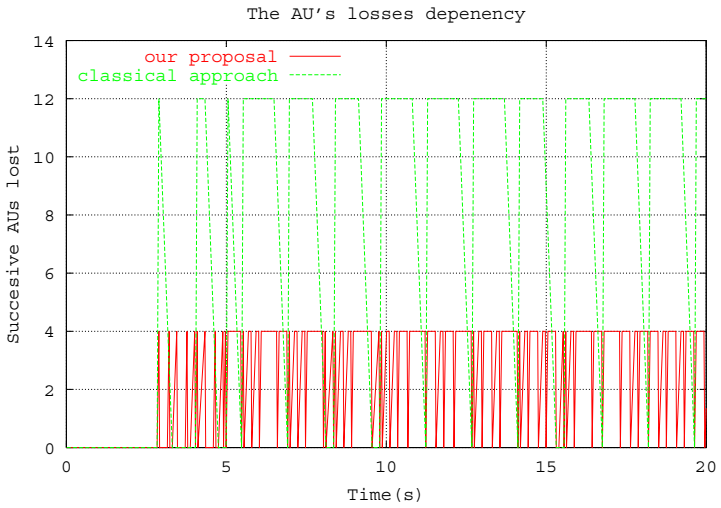


Figure 4-20: Correlated AU's losses.

4.2.3.2.2 Scenarios 2

In this scenario, we have compared between two configurations. The first is RTP payload format with UEP scheme (**Configuration A**), and the second is without UEP scheme (**Configuration B**). For each configuration, we vary gradually the network load to get more information on the behavior of the different mechanisms.

In order to highlight the efficiency of the UEP scheme, we compute the number of AU that can be decoded at the Client. Figure 4-21 shows this result of comparison between the decoded object ratios. The X-axis represents the throughput of the background traffic. As expected, the quantity of the AVOs decoded at the receiver side decreases when the network load increases because it entails more packet losses.

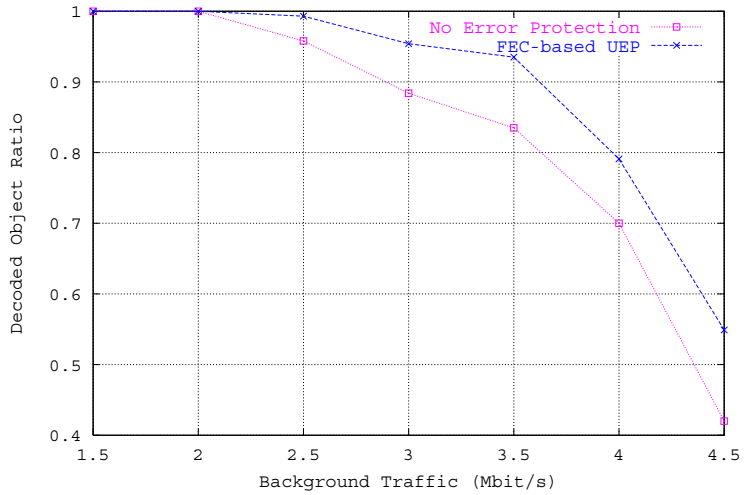
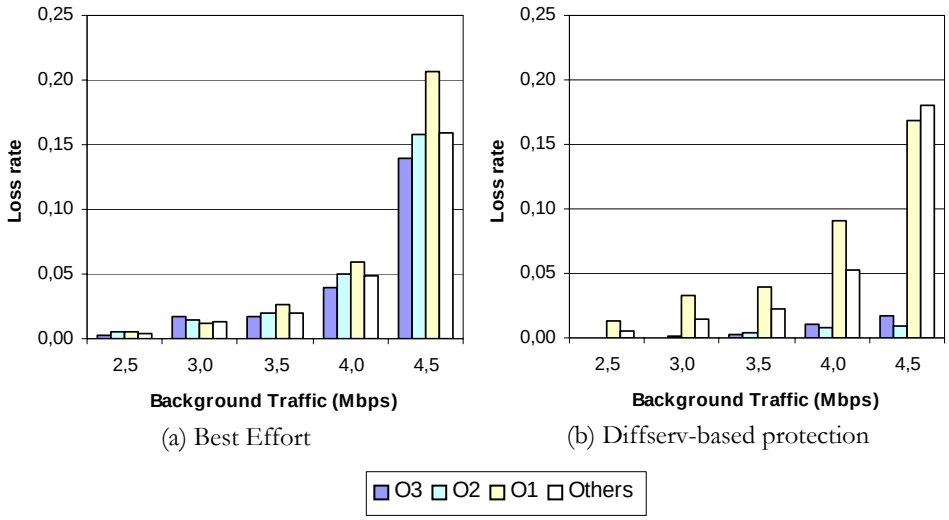


Figure 4-21: Decoded object ratio vs. background traffic throughput

Packet loss is accented by using our FEC-based UEP because UEP increases the MPEG-4 packet-stream throughput by 7 %. For this reason, there is more packet losses with UEP configuration, for a given network load. However, the redundant UEP information better recovers

lost packet at the receiver. Consequently, a particular *Access Unit* can be restored. Failures in the decoding process are rather distributed toward the less important objects, and then UEP reduces the effects of spatial and temporal errors propagation. This observation is shown in Figure 4-21 where the decoded object ratio of configuration A is always better than configuration B.

In this scenario, we setup two networks configurations to demonstrate the effect of the cross-layer system. For this reason, we mark each packet according to its priority score and we study the effect of this marking. We transmit the protected MPEG-4 scene over IP Best Effort network (**Configuration C**) and over IP Diffserv network (**Configuration D**). Figure 4-22 shows results of the comparison between the loss rate for configuration C and D. The X-axis represents the throughput of the background traffic. At each point of load, a set of four bars in Figure 4-22 shows the loss rate for O1, O2, O3, and the background traffic. In configuration D (Figure 4-22 (a)), loss rate approximately follows a uniform distribution. This is due to the IP Best Effort routers which uses a simple drop policy (i.e. Drop Tail). When the queue is full, all the incoming packets are dropped with the same probability. In contrast, Diffserv network provides a differentiated level of QoS for each stream depending to its priority class. It is visible for configuration D(Figure 4-22 (b)), that the losses happen mainly on the lower priority streams (i.e., O1 and some of background traffics which are marked with high drop precedence).



(a) Best Effort (b) Diffserv-based protection

Figure 4-22: Loss rate vs. background traffic load

Figure 4-23 shows in more details the performance of our cross-layer system. The X-axis represents the loss ratio related to the MPEG-4 traffic (i.e., the sum of O1, O2, and O3 traffics).

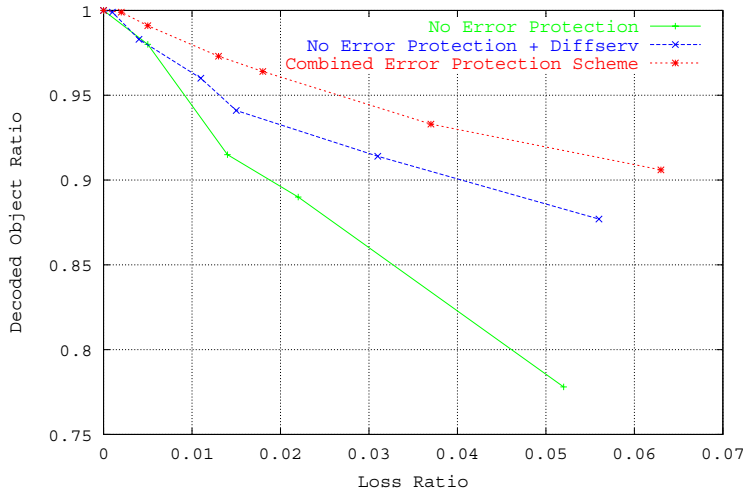


Figure 4-23: Performance evaluation of the different scenarios

Figure 4-23 shows clearly that the proposed RTP payload format performs a better QoS of the received MPEG-4 scene. When the loss ratio increases the rate of the decoded MPEG-4 objects to playback is better in our combined error protection scheme. The combined mechanism provides a better graceful degradation of quality of MPEG-4 scene. Furthermore, the gain increases slightly when the loss ratio increases.

4.3 A Fine Grained TCP-Friendly Video Rate Adaptation Algorithm

Streaming audio and video on the Internet is becoming more popular. This rapid expansion underlies a new challenge for efficient handling of Internet traffic. As seen later, the majority of multimedia applications perform over an RTP stack that is implemented on top of UDP/IP. However, UDP offers no congestion control mechanism and therefore is unaware of network condition and unfair towards other competing traffic. Today's Internet traffic is dominated by TCP. TCP uses several mechanisms to handle network congestion such as: AIMD (Additive Increase and Multiplicative Decrease), slow start, congestion avoidance, fast retransmit and fast recovery. Thus, it is crucial that UDP traffic performs also TCP-friendly congestion control [68].

The idea of congestion control mechanism helps to prevent the application entering congestion collapse in which the network link is heavily used and little useful work is being done. So to prevent such situation, all applications must perform TCP-like congestion control mechanisms. Traffic that does not perform in TCP-friendly behavior can be dropped by the router [69].

In our congestion control, we investigate Quality of Service (QoS) interaction provisioning between an MPEG-4 video application and the IP Diffserv network. To achieve the best possible QoS, all the components involved in the transmission process must collaborate together. In this regards, we propose two mechanisms. The first one is the rate adaptation mechanism. The server performs rate adaptation through the adjustment of the number of streamed object based on network state and relative priority score for each objects. We use a TCP-friendly to adapt the server rate to network condition. The server tries to deliver the maximum number of audio visual objects (AVO) that can fit in the current available bandwidth. The second mechanism is a Diffserv

marking scheme. The server must be aware of each audio-visual object in the scene since it is able to classify these objects in a hierarchical manner, from less important to more important object. This mechanism was presented in Section 4.1 and it allows the server to: (1) deal with network congestion by stopping streaming less important object when congestion is detected and (2) prioritize the transport of important object by an intelligent marking in case of IP Diffserv network. When network congestion occurs less important AVOs will be dropped automatically by network elements. Lost packets notify the server to reduce its transmission rate by stopping streaming less important AVOs.

4.3.1 Related Work

The idea of TCP-friendly transport protocol is to emulate TCP behavior without replicating TCP mechanism. By definition, a flow is said to be TCP-friendly or TCP-compatible, if its arrival rate does not exceed the arrival rate of a conformant TCP implementation in the same circumstances [167]. Many TCP-friendly congestion control mechanisms were developed recently that are either window-based or equation-based, among this mechanism: Rate Adaption Protocol (RAP) [78], Loss-Delay Based Adaptation Algorithm (LDP) [77], a spectrum of TCP-Friendly Algorithm [82] and TCP-friendly Rate Control Protocol (TFRC) [84]. While these protocols and others are comparable in their features in simulating TCP behavior, TFRC seems to be more robust since it was adopted recently by the IETF as a standard. TFRC provides sufficient responsiveness by taking into consideration all the parameters that affect the TCP rate such as loss, Round-Trip Time (RTT) and retransmission timeout value. The key advantage of TFRC is that it has a more stable rate during the session lifetime.

Our video quality adaptation mechanism is based on TFRC. It operates as follows:

- The receiver measures the loss rate and feeds this information back to the sender. This is achieved by a modified version of RTP and RTCP protocols [110]. Each RTP packet has a timestamp and a sequence number that allow the receiver to compute the packet loss and the sender to compute the RTT.
- The loss rate and the RTT are then fed into the TFRC module to get the appropriate transmission rate (cf. Eq.8 below).
- The sender then adds or drops audio-visual objects and the associated layers if any, to adjust its transmission rate to match the target rate (i.e. allowed rate).

The calculated rate is obtained by using the TFRC equation [84]:

$$R_{TCP} \cong \frac{s}{RTT\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{3bp}{8}})p(1+32p^2)} \qquad \text{(Eq. 8)}$$

Where $\boldsymbol{R_{TCP}}$ is the target transmission rate or the allowed transmission rate, $\boldsymbol{s}$ is the packet size, $\boldsymbol{RTT}$ is the round trip time, $\boldsymbol{p}$ is the loss rate, $\boldsymbol{t_{RTO}}$ is the TCP retransmission timeout value and $\boldsymbol{b}$ is the number of packets acknowledged by a single TCP acknowledgement.

4.3.2 Video Object-Based Rate Adaptation Algorithm

Let S be a set of MPEG-4 AVOs containing n AVOs O_j, with $j \in \{1, 2 \ldots n\}$. Without loss of generality, we assume that these objects are sorted in a decreasing order of priority score. Each object O_j may consist of m_j layers ($m_j \geq 1$). Note that lower layers within an object have higher priorities than higher layers.

Let **P** be the function that returns the relative priority score of a particular object or layer. Without loss of generality, we assume that:

$$\forall j, 1 \leq j < n : P(O_{j+1}) \leq P(O_j)$$
$$\forall j, 1 \leq j < n, \forall l, 1 \leq l < m_j : P(L_{j,l+1}) < P(L_{j,l}) \quad \text{(Eq. 9)}$$

$L_{j,l}$ is the Layer number l of the Object O_j

By using Eq. 9 we can construct an Audio-Visual Entity set called E composed of all object layers ordered by their priorities.

$E = \{L_{1,1}, L_{1,2} \ldots L_{1,m1}, L_{2,1}, L_{2,2} \ldots L_{2,m2}, \ldots, L_{n,1}, L_{n,2} \ldots L_{n,mn}\}$. We will note E as follows:

$E = \{e_1, e_2, \ldots, e_w\}$ with $w = |E| = \sum_{j=1}^{n} m_j$

Note that if two objects have the same priority score, then the associated layers of an object have the same priority score as the object (in relation to other objects) with the lower layers having higher priorities than higher layers.

At time t_i, the function R_i gives the instantaneous transmission rate of an audio-visual entity. For example, the audio-visual entity $\mathbf{e_p}$ has an instantaneous transmission rate equal to $R_i(e_p)$, and the object O_j has the instantaneous transmission rate equal to $R_i(O_j)$.

Our object-based quality adaptation mechanism operates as follows: The server evaluates the network state from the information gathered (i.e. RTT and loss rate) at time t_i, then computes the allowed sending rate R_{TCP} using Eq. 1. The server tries to send as much as possible of the audio-visual entities without exceeding R_{TCP} taking into consideration entities priorities. Details of the adding and the dropping process will be presented in section 4.3.2.1 and 4.3.2.2 respectively.

Assume that we have an MPEG-4 scene composed of four audio-visual objects: O_1, O_2, O_3 and O_4. Assume that O_1 is composed of a single layer, and that each of O_2, O_3 and O_4 is composed of three layers (one base layer and two enhancement layers). Also, as presented in Figure 4-24, we assume that the associated priorities are as follows:

- O1 is the most important,
- O2 and O3 have the same priority score,
- O4 is the less important.

Then, $E = \{ L_{1,1}, L_{2,1}, L_{3,1}, L_{2,2}, L_{3,2}, L_{2,3}, L_{3,3}, L_{4,1}, L_{4,2}, L_{4,3}\} = \{e_1, e_2, \ldots, e_{10}\}$. Here w=10.

The video server adds audio-visual entities in the order of their importance (i.e. form left to right in the set E). Entities are dropped in reverse order (i.e. form right to left) until matching the target sending rate.

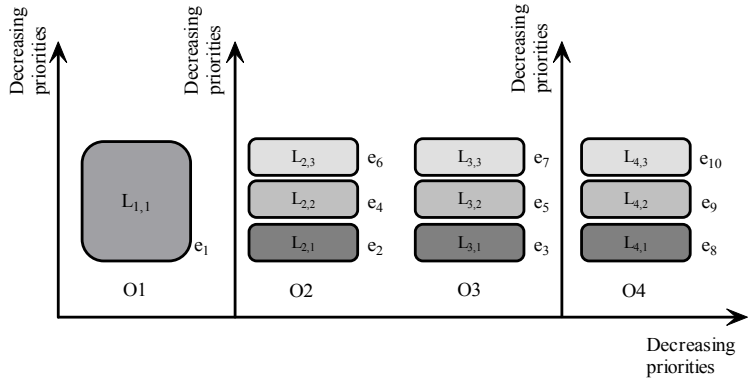


Figure 4-24: Handling priorities between layers and objects

4.3.2.1 Adding Audio-Visual Objects

The server adds a new audio-visual entity as soon as the target rate exceeds the current sending rate of current entities plus the new entity. Assume that the server is streaming ***k*** entities at time t_i. We assume also that the client has sufficient resources to play all the entities being sent by the server. Therefore, at time t_{i+1} the server can add a new entity while the following condition is satisfied:

$$\sum_{j=1}^{k+1} R_{i+1}(e_j) \leq R_{TCP} \qquad \text{(Eq. 10)}$$

At the client side, the new audio-visual entity must be buffered and synchronized to the current playback time.

4.3.2.2 Dropping Audio-Visual Objects

When the estimated throughput of the TCP session indicates that the video server is transmitting more data than it should, then the video server must reduce its sending rate by dropping one or more audio-visual entities. Therefore, the server drops entities while the following condition is satisfied:

$$\sum_{j=1}^{k} R_{i+1}(e_j) > R_{TCP} \qquad \text{(Eq. 11)}$$

4.3.2.3 GOV-Driven Stability

Since the TFRC compute the new target rate each RTT, adding and dropping audio-visual entities can lead to undesired oscillation and poor video quality at the receiver. To prevent from such behavior, several measures are taken into consideration.

First, the TFRC module copes with oscillation behavior by using EWMA (Exponentially Weighted Moving Average) to detect out-of-control situations quickly. EWMA statistics are used to attempt to respond dynamically to the changing value in the measured RTT and loss and attempt to regulate this value to reflect as much as possible the reality. In TFRC, the loss rate is measured in terms of loss interval which represents the number between two consecutive loss events [84]. The mechanism reacts too strongly to single loss events and ensures that allowed sending rate do not change aggressively.

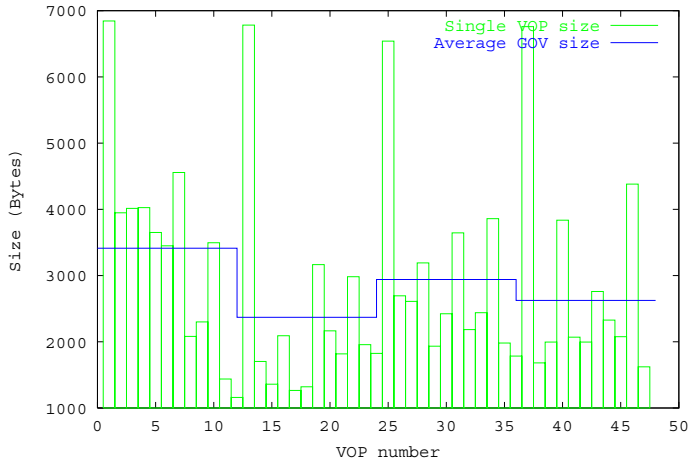


Figure 4-25: Handling stability through video object plan

Second, we propose to adjust the server transmission rate at the beginning of each **GOV** (Group of video object plane). Thus, the new transmission rate obtained from TFRC module is used to adapt video sending rate. Figure 4-25 shows four GOVs (each group has twelve VOPs). The average line in the Figure shows the server transmitting rate at the beginning of each GOV of a current Video Object (VO). If this value does not fit in the current available bandwidth then the server does not stream the object.

4.3.3 Performance Evaluation

4.3.3.1 System and Network Models

Simulations are conduced using the network simulator *ns2*. We used the network architecture shown in Figure 4-26 to simulate a unicast service provided by the MPEG-4 server attached to the node "S". The server sends data to the client attached to the node "C". Our server is an ns2 agent that uses TFRC module to adapt the number of transmitted AVO. The client is also an ns2 agent which extends the capabilities of the RTP sink by reporting statistic information to the server. The network is loaded by *n* FTP streams carried over TCP (*n* ranges from 0 to 8). This allows the link between the routers "R1" and "R2" to be congested differently. FTP sources always have a packet to send and always send a maximal-sized (1000-bytes) packet as soon as the congestion control window allows them to do so. FTP sink immediately sends an ACK packet when it receives a data packet. The queue in the routers has a size of 50 packets. The core IP Diffserv router examines incoming packets and reacts according to the marking, whereas "R1" is an edge router that implements Marking/Classification policy on incoming packets. R1 uses A Two Rate Three Color

Marker (TR3CM) [131] to mark the background. Therefore, background traffic is evenly distributed among the different Diffserv classes. We recall that the video traffic is marked at the MPEG-4 server according to AVOs priorities. The bottleneck link between the core router and R2 has a 5 Mbit/s of bandwidth.

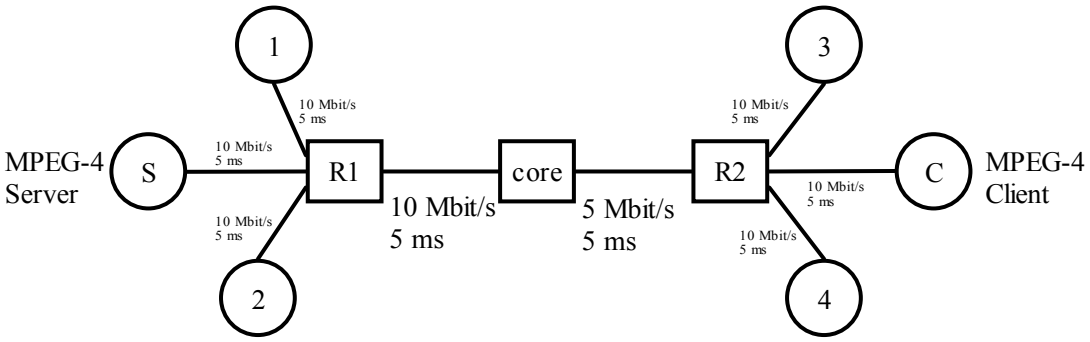


Figure 4-26: Network topology for congestion control

In our simulation, the MPEG-4 presentation was obtained by using a set of AVOs components. We simulate the "Akiyo" video sequence shown in Figure 4-27 by using four multimedia objects: AO (audio speech), VO1 (background), VO1 (speaker) and VO3 (logo). These objects are sorted as follows:

- AO has the priority score of 1, it is the most important object in this scene. It is marked with Diffserv PHB AF11 (low drop precedence).
- VO1 and VO2 have the priority score 2. They are marked with Diffserv PHB AF12 (medium drop precedence). Each Object is composed of 3 layers (one base layer and 2 enhancement layers)
- VO3 has the priority score 3, it is the least important object in this scene. It is marked with Diffserv PHB AF13 (high drop precedence).

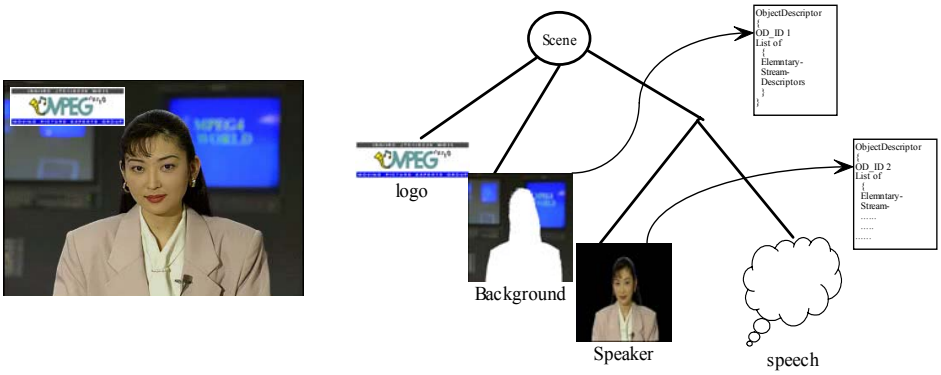


Figure 4-27: Simple composite MPEG-4 scene using "Akiyo" video sequence

Figure 4-28 shows the bit-rate of the MPEG-4 video objects that can be sent from the MPEG-4 server to the client during a period of 120 seconds. The complete scene is shown in Figure 4-28 (a). The Audio Object is a constant bit rate at 64Kbits/s. An Audio packet is sent each 125ms. Video object 1 has an average throughput of 200 Kbit/s and a peak rate of 956 Kbit/s. This object is composed of three Layers: BL (Base Layer), EL1 (Enhancement Layer 1) and EL2 (Enhancement Layer 2). The throughputs of the different layers are shown in Figure 4-28 (b). Video object 2 has an average throughput of 650 Kbit/s and a peak rate of 1722 Kbit/s. This

object is composed of three Layers: BL, EL1 and EL2. The throughputs of the different layers are shown in Figure 4-28 (c). Video object 3 has an average throughput of 124 Kbit/s and a peak rate of 356 Kbit/s. It is composed of one single layer (see Figure 4-28 (c)).

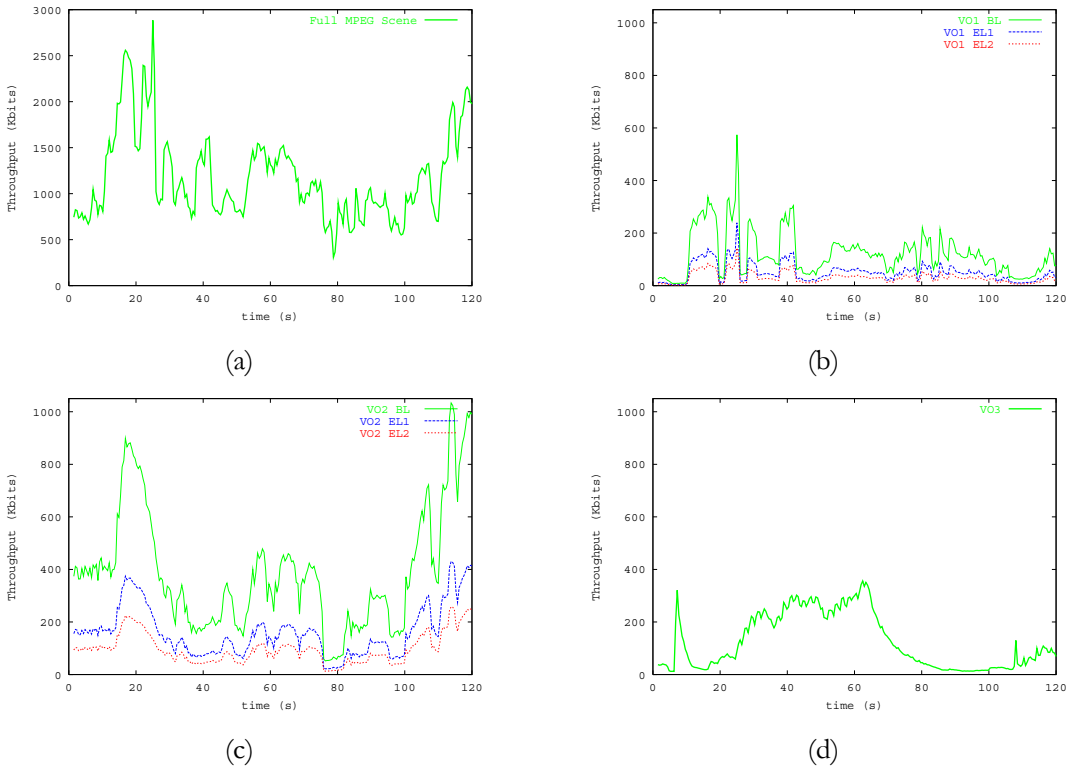


Figure 4-28: Instantaneous throughput of the different MPEG-4 Video Object

4.3.3.2 Simulation Results

We perform an intensive simulation, each time with different parameters to see the behavior of our video on demand system. We vary the number *n* of FTP source according to following scenarios: (1) **Scenario A**: one FTP source; (2) **Scenario B**: two FTP sources; (3) **Scenario C**: four FTP sources; (4) **Scenario D**: eight FTP sources. FTP sources send data from time t=30s until time= 90s.

This section presents some QoS measurement such as, the video server throughput as a function of network state, packet loss and end-to-end packet transmission delay.

4.3.3.2.1 Network Performance

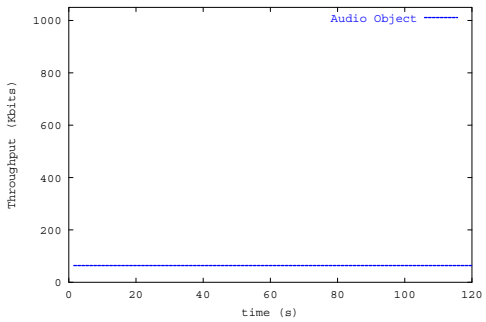
4.3.3.2.1.1 Video Server Throughput

The video server regulates its transmission rate to reflect the allowed rate by adding or dropping audio-visual entities. Results obtained of the different scenarios are shown in Figures below. Also, to simplify the interpretation of the results, Table 4-6 summarizes the transmission ratio per AVO stream observed during the period of the simulations (120s). Note that the FTP sources begin data transmission at time t=30s, and stop at time t=90s. VO3 has the low ratio since it has the lowest priority score is the scene. VO1 and VO2 have the same priority score, so the corresponding layers have more or less the same transmission ratio.

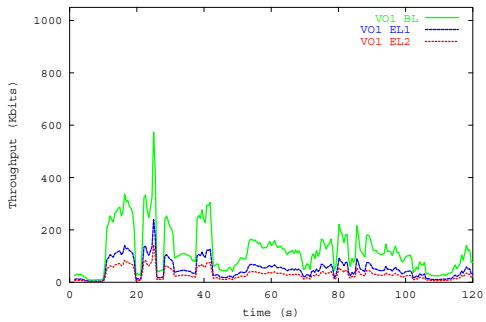
AVO / Scenario	Audio	VO1			VO2			VO3
		BL	EL1	EL2	BL	EL1	EL2	
A	100%	100%	100%	100%	100%	100%	100%	100%
B	100%	100%	100%	100%	100%	100%	100%	100%
C	100%	100%	94%	87%	100%	96%	92%	55%
D	100%	89%	60%	53%	97%	77%	71%	26%

Table 4-6: Transmission ratio per MPEG-4 objects

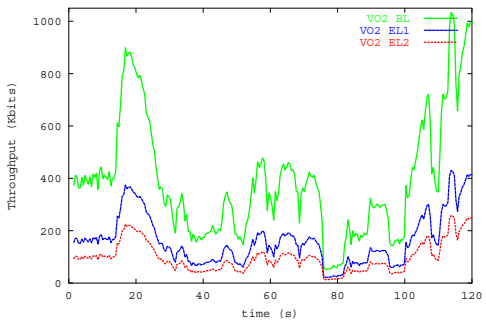
From the result in scenario A (Figure 4-29), we can see that the MPEG-4 video is transmitted entirely. The FTP source adapts to the change caused by the video throughput and tries to consume all the available bandwidth. The bottleneck link is 100% used when the FTP source starts the transmission. In this scenario, there is no loss because we have two streams that fairly share network resources. This gives a portion of 2.5 Mbit/s per stream. Since our MPEG-4 scene consumes less than 2.5 Mbit/s, the rest of the bandwidth is used by the FTP source.



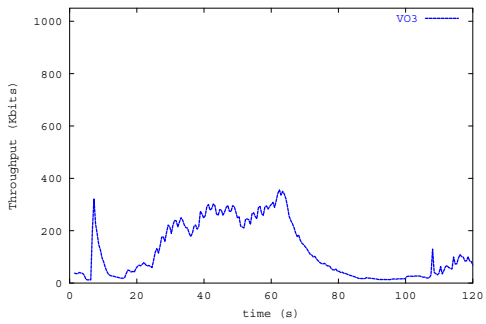
Audio Object



Video Object 1



Video Object 2



Video Object 3

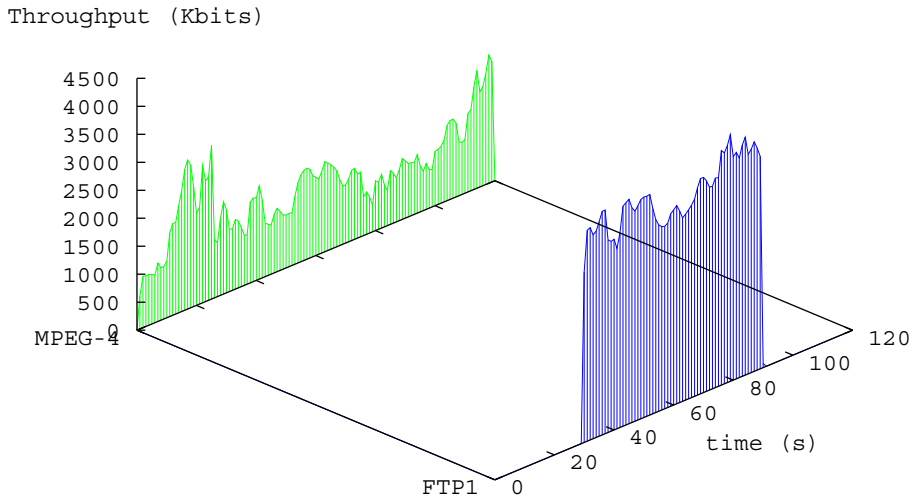
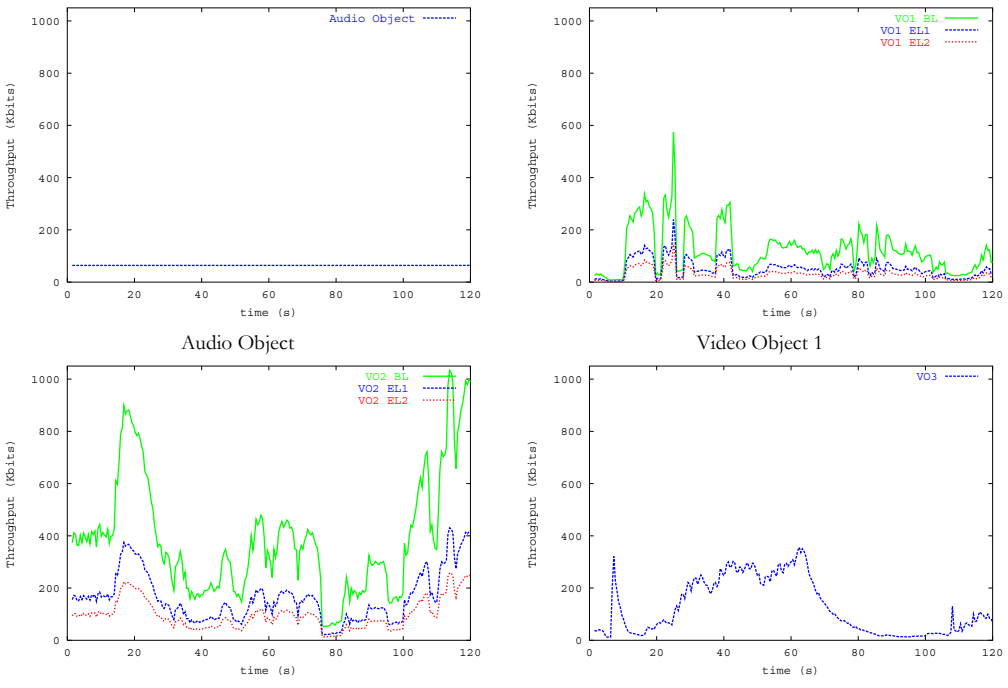


Figure 4-29: Traffic throughput and fairness in scenario A

In scenario B (Figure 4-30), we have two FTP sources. The results show that the video stream is transmitted entirely and that the two FTP sources fairly share the remaining bandwidth. This gives a portion of 1.66 Mbit/s per flow since we have three streams. In this period, when the two FTP streams are active (30s-90s), the video stream consumes less than 1.66 Mbit/s, the remaining bandwidth is fairly shared with the two FTP sources.



Audio Object

Video Object 1

Video Object 2

Video Object 3

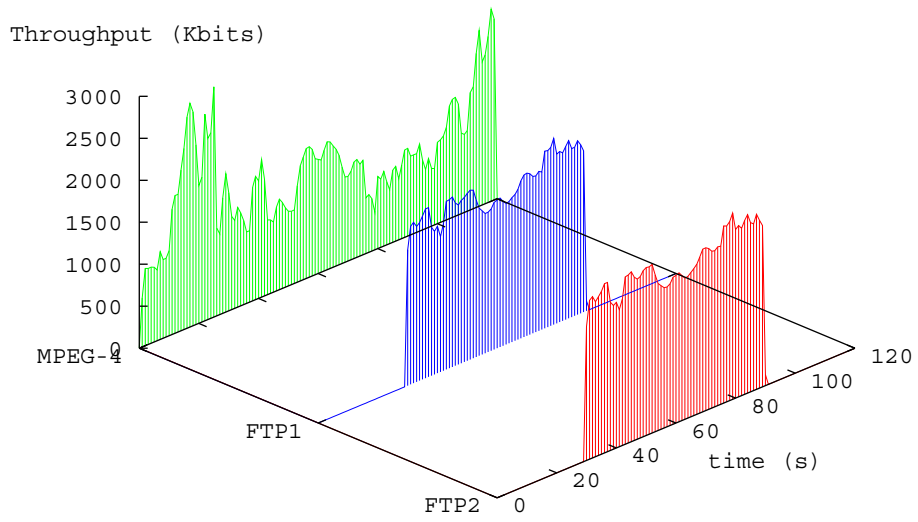
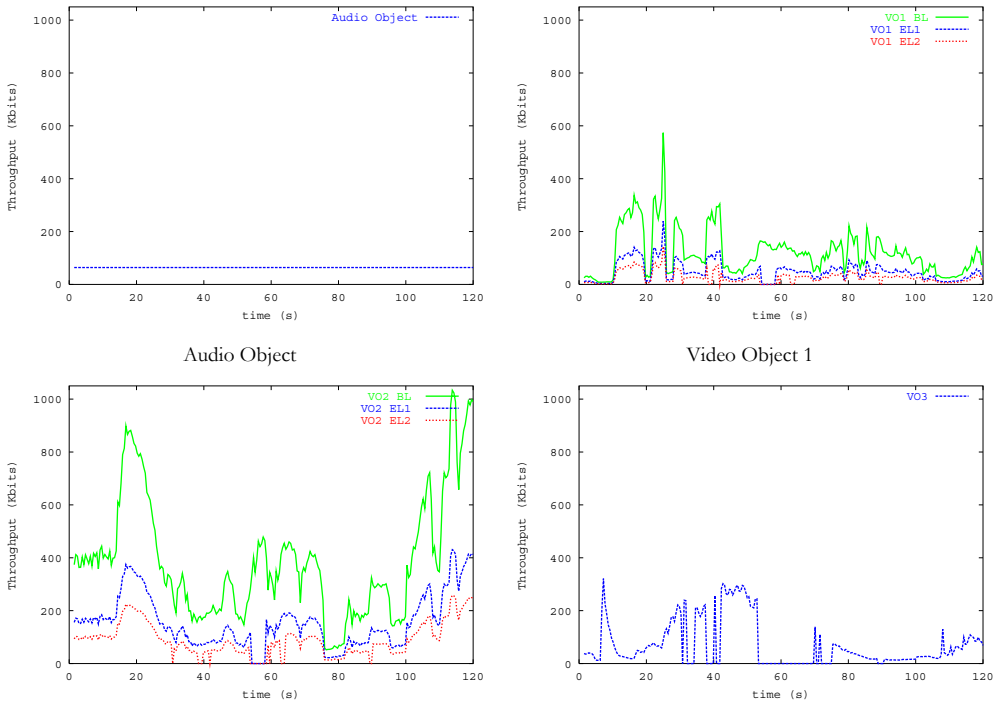


Figure 4-30: Traffic throughput and fairness in scenario B

Scenario D is interesting since we see the effect of our adaptation mechanism. We can see that the audio object is always present and that less important objects (respectively object layers) are not transmitted when the shared bandwidth is not sufficient. Our adaptation mechanism begins transmitting data from important audio-visual entity to less important. We can see that all the streams (FTP and video) fairly share the bandwidth.

Scenario C confirms the previous result and shows the effect of our adaptation mechanism. A minimum of QoS is guaranteed by our adaptation mechanism. The network does not enter in congestion collapse since the video stream is aware of network condition.



Video Object 2 Video Object 3

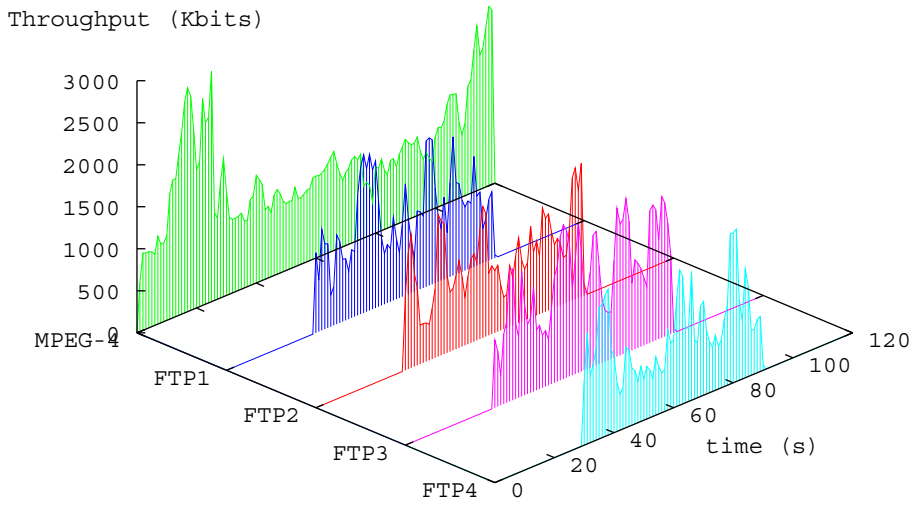
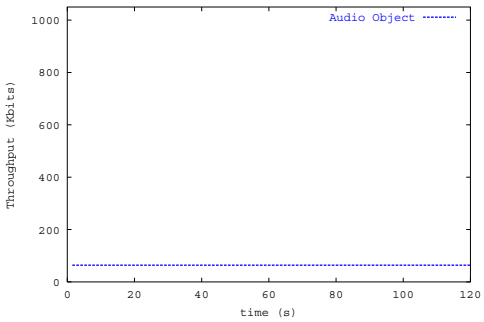
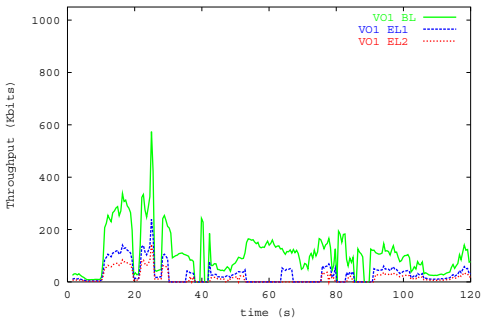


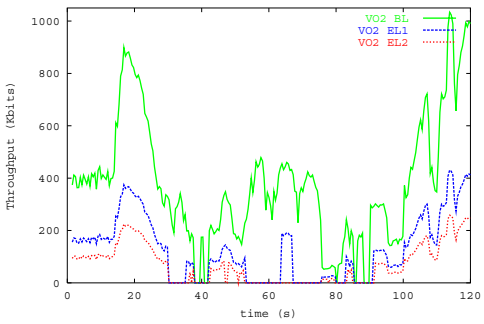
Figure 4-31: Traffic throughput and fairness in scenario C



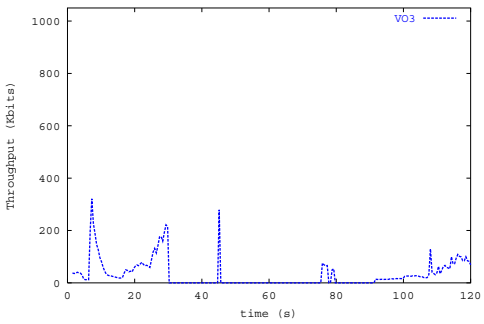
Audio Object



Video Object 1



Video Object 2



Video Object 3

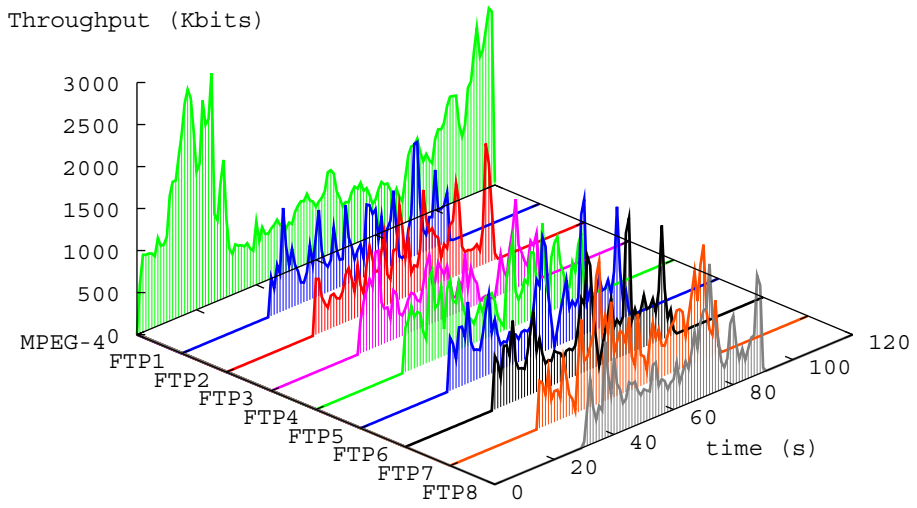


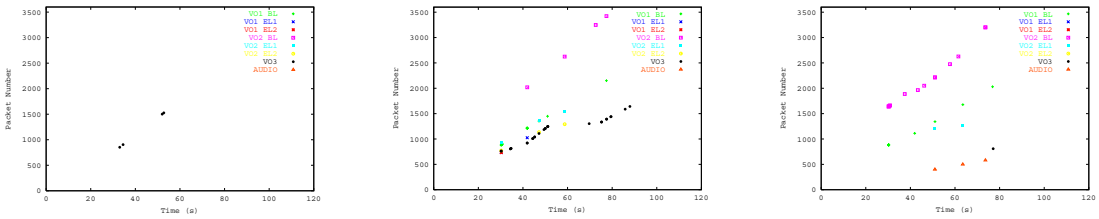
Figure 4-32: Traffic throughput and fairness in scenario D

The Table below shows the server transmission ratio per audio-visual entity observed during the period of the simulations (120s). Note that VO3 has the low ratio since it has the lowest priority score in the scene. VO1 and VO2 have the same priority score, so the corresponding layers have more or less the same transmission ratio.

Without using our congestion control mechanism, the server transmits the audio-visual entities without any regulation as shown in Figure 4-28. The loss may increase and the network may enter in congestion collapse.

4.3.3.2.1.2 Packet Loss

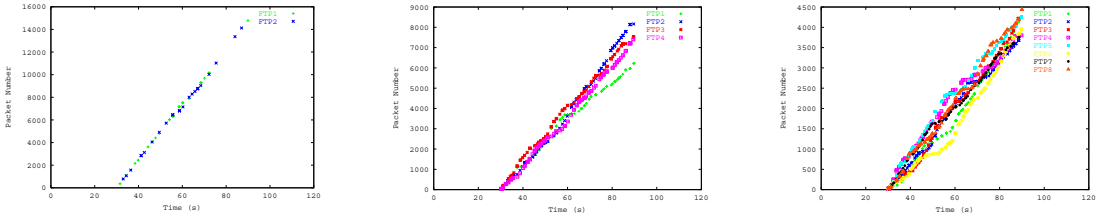
Figure 4-33 shows lost packets for scenarios B, C and D using our congestion control mechanism. Scenario A does not experience any loss. In scenario B, some lost packets are observed on VO3. This is due to the active queue of the Diffserv router which drops lower priority packets when a predefined threshold is reached to prevent congestion. In scenario C, we observe also some loss on lower priority packets but in scenario D high priority packet are also dropped. This is due to: (1) lower priority packets are not transmitted because our congestion control mechanism regulates the server transmission rate by stopping streaming lower priority packets and (2) AVO1 and AVO2 request more bandwidth in our scene and cause some congestion.



Scenario B Scenario C Scenario D

Figure 4-33: Instantaneous MPEG-4 packet loss

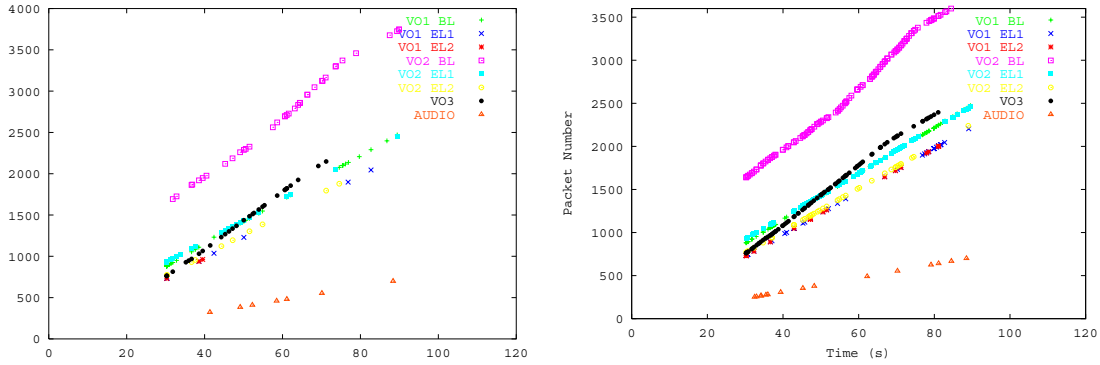
Figure 4-34 shows FTP packet loss observed in the same scenarios. FTP packets encounter more loss than the video packets due to two factors. First factor is that FTP traffic is marked using TR3CM marker which distributed the marked traffic among the different classes of the Diffserv network. We remark that the majority of dropped packets are those marked with high drop precedence. Second factor is that FTP source does not regulate the traffic by computing the allowed transmission rate rather it uses window-based congestion control mechanism.



Scenario B Scenario C Scenario D

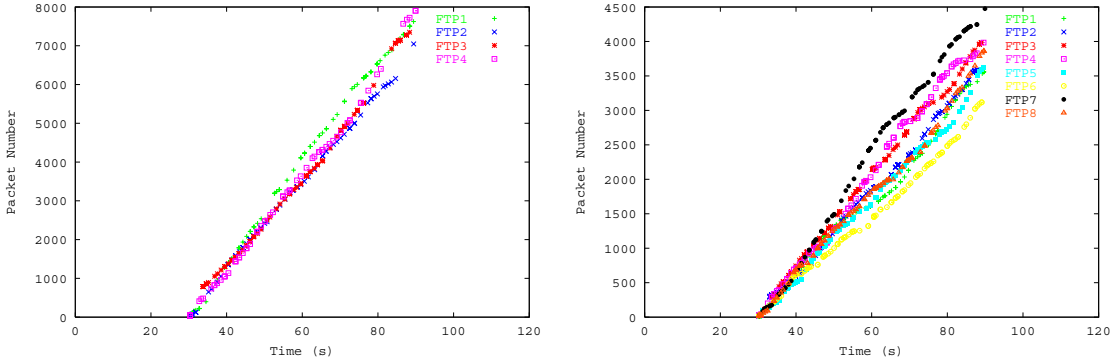
Figure 4-34: Instantaneous FTP packet loss

We have redone the same simulation without using our congestion control mechanism. The results are shown in Figure 4-35 and Figure 4-36. Remark that in scenario B no loss is observed in both of MPEG-4 and FTP streams. In scenario with our congestion control mechanism loss is due to the active queue on the core router which prevents the congestion by an early packet discard mechanism. Without our congestion control mechanism, more packets are dropped because the server does not regulate its transmission rate at the source. Important video packets are also dropped with the same probability due to Drop Tail queue management used by the router. The damage caused by some data loss in some reference picture such as I-VOP or P-VOP will affect subsequent picture(s) due to inter-frame predictions. For example when the I-VOP is lost, the whole dependant P-VOP and B-VOP cannot be decoded. The same conclusion is valid for hierarchical streams. Hence, Enhancement Layer 1 cannot be decoded without the reception of Base Layer, and so on. When using our congestion control mechanism, low important audio-visual entities (those marked with high drop precedence) are not transmitted by the server when the allowed rate decreases. This helps to prevent a future drop by the router. So the regulation is done at the server and demonstrates clearly the advantage of our mechanism.



Scenario C Scenario D

Figure 4-35: MPEG-4 packet loss without our congestion control mechanism

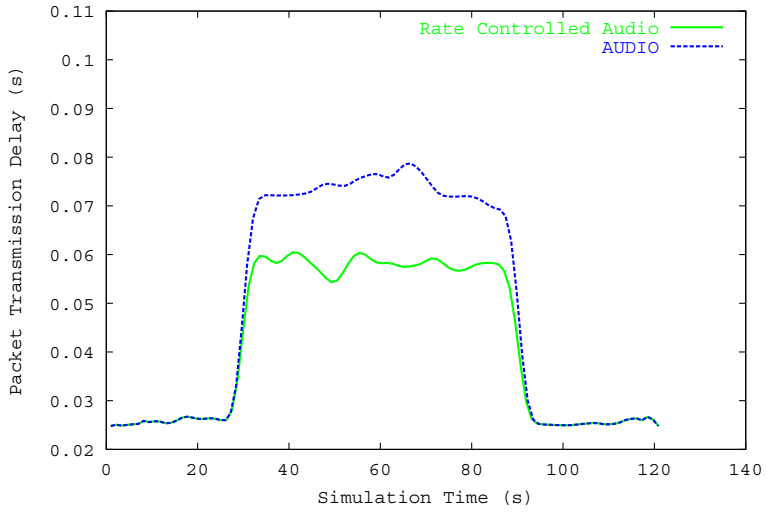


Scenario C Scenario D

Figure 4-36: FTP packet loss using our congestion control mechanism

4.3.3.2.1.3 End-to-end Transmission Delay

Figure 4-37 shows the end-to-end packet transmission delay (PTD) for the audio object in case when using our congestion control mechanism and without it. PTD variations are correlated with the router queue size and the packets lost. The more the queue is on congestion; the more the delay for a packet to reach the destination is increased. Since the queue in the Diffserv is based on active queue management that maintains the queue size as small as possible, then PTD is small in case of our congestion control mechanism. Same results are obtained with the others streams.



Scenario D

Figure 4-37: End-to-end packet transmission delay

4.3.3.2.2 User Perceived Quality

Peak Signal to Noise Ratio (PSNR) is an indicator of picture quality that is derived from the root mean squared error. The PSNR for a degraded N1 x N2 8-bit image *f'* from the original image *f* is computed by the formula in Eq. 12:

$$PSNR = 20\log_{10}\frac{255}{\sqrt{\frac{1}{N_1 N_2}\sum_{x=0}^{N_1-1}\sum_{y=0}^{N_2-1}\left[f(x,y)-f'(x,y)\right]^2}} \qquad \text{(Eq. 12)}$$

In order the compute PSNR value, we should rebuild the received scene. By using ns2 trace file, we create the received MPEG-4 scene. The measured PSNR indicates the difference between the original and the received video sequence. Figure 4-38 shows comparison between the original and the received scene quality for scenarios (A, B, C, and D). As expected, and since there was no loss in scenario A and B, the received quality is the same as the original quality. In Scenario C and D, the degradation of the received quality is due to our adaptation mechanism which only sends pertinent object to the client. This PSNR measurement does not reflect the meaning of what the client received. Because in our case, the logo object is basically not send. This affects the PSNR value but for the end user it does not have any importance (in our assumption). Figure 4-23 presents some snapshot of the received video in scenario D.

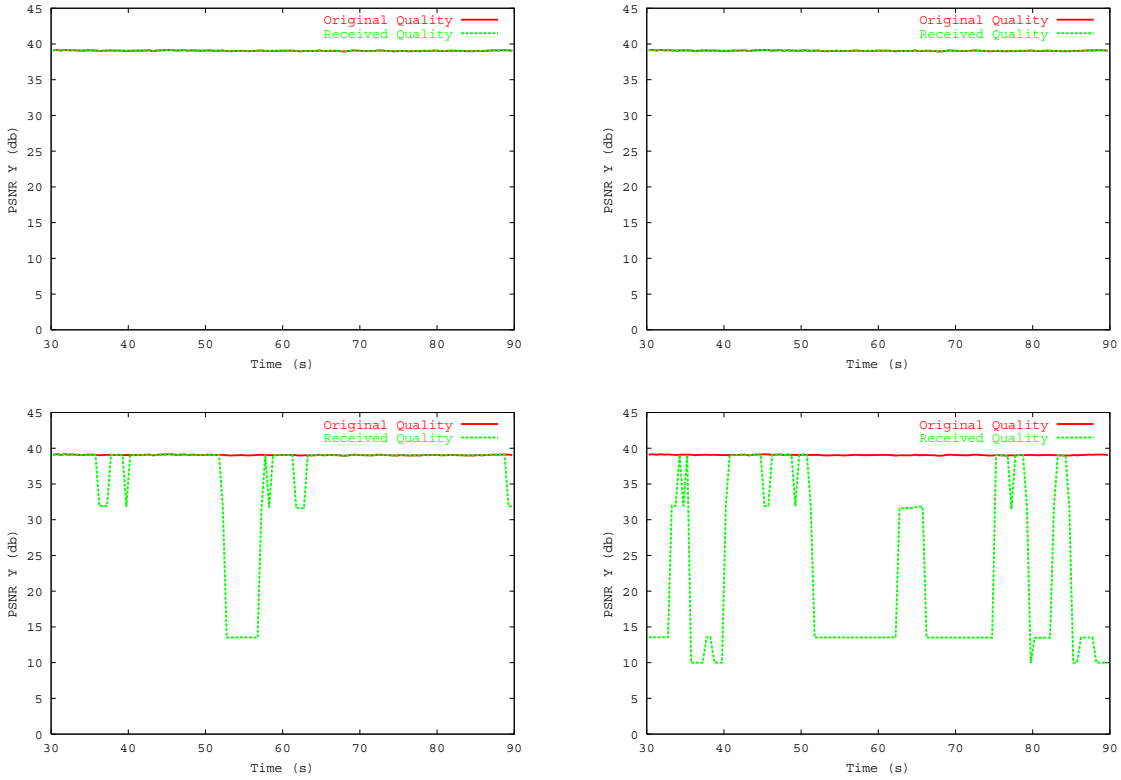


Figure 4-38: Video quality comparison for different scenario

Received video at time t=39.88s
(only the Base Layer of VO2)

received video at time t=42.76s
(All layers of VO1 and VO2)

Received video at time t=60.04 s
(base layer of VO1 and VO2)

Received video at time t=63.88s
(Base layer + Enhancement Layer of VO1 and VO2)

Figure 4-39: Received video presentation

4.4 Conclusion

We proposed in this chapter, an extension to the MPEG-4 System architecture with a new "Media QoS Classification Layer" to provide automatic and accurate mapping between the MPEG-4 Application-Level QoS metrics and the underlying transport and network QoS mechanisms such as IP Diffserv. This "Media QoS Classification Layer" makes use of a neural network classification model to group audiovisual objects of a scene with same QoS requirement to create elementary video streams that are subsequently mapped to IP Diffserv PHB (Per Hop Behaviors). These MPEG-4 Audio Visual Objects (AVOs) are classified based on Application-level QoS criteria and / or AVOs semantic descriptors according to MPEG-7 framework. Thus, MPEG-4 AVOs requiring same QoS from the network are automatically classified and multiplexed within one of the IP Diffserv PHB. Object data-packets within the same class are then transmitted over the selected transport layer with the corresponding bearer capability and priority score. The performance evaluation shows better protection of relevant video objects of a scene during transmission and network congestion.

There are a number of RTP packetization schemes for MPEG-4 data. It is clear that many packetization schemes can be implemented together in one terminal. Each packetization scheme is basically adapted to a particular media stream. Thus, we proposed, a new RTP payload scheme for MPEG-4 video and audio that addresses multiplexing and avoids packet error propagation. The MPEG-4 streams are better recovered against errors when using this payload. The amount of recovered data is related to audio-visual objects priority score in the MPEG-4 scene. The more the object is important, the more the recovered data is valuable and better. Our scheme is beneficial for transmitting MPEG-4 content over Internet.

To enhance the transport mechanism, we have proposed an adaptation mechanism for MPEG-4 video streams that uses a TCP-Friendly Rate Control. Our mechanism adds and drops MPEG-4 Audio-Visual Objects to perform rate adaptation and congestion control. We have evaluated the proposed mechanism through simulations using ns2. The MPEG-4 server implemented in ns2 uses the TFRC module as an equation-based congestion control mechanism. We coupled end-to-end congestion control with a Diffserv network that guarantees objects prioritization within the network. The simulation results show that important multimedia entities are maintained by the router in case of network congestion. Combining these mechanisms into a coherent architecture demonstrates clearly the gains obtained.

Chapter 5

5 A Dynamic Packet Video Marking Algorithm for IP Differentiated Services

The current unprecedented growth of multimedia streams in the Internet creates new challenges and concerns in the research communities. Providing scalable QoS while maintaining fairness and optimal network resource management are key challenges for the future Internet. Successful deployment of networked multimedia systems depends significantly on the ability of the management infrastructure to be scalable, dynamic and adaptive. In the other hand, rich multimedia is increasingly being incorporated into IP applications (like Voice-over-IP, streaming video, videoconferencing, etc.). However, more work is needed to evolve IP and the underlying networking technologies into an infrastructure capable of supporting all-service and all-media traffic. This is why there is a great need to use enabling protocols and services to manage multimedia networks and services. In order to response these questions, we have designed an algorithm that provides automatic and accurate mapping between the MPEG-4 Application-Level QoS metrics and the underlying transport and network QoS mechanisms such as IP Diffserv, this algorithm can be implemented on the media server or deployed / deported on the edge router. This chapter is organized as follows:

- Section 5.2 presents a static Diffserv marking algorithm that provides different level of quality of service to MPEG-4 streams in a Diffserv network.
- Section 5.3 present a mechanism for installing, configuring and managing multimedia stream over a Diffserv domain. The Diffserv domain is governed by a policy decision point which facilitates the management of the domain, and allows a dynamic packet video marking.

5.1 Related Work

Recent works on IP Quality of Service (QoS) Management led to the development and standardization of enhanced protocols and services. The IETF has defined the Policy-based Network Management (PBNM) architecture to configure network services. Currently most efforts are focused on Differentiated Services (Diffserv) in the Internet. The goal of the policy-based network management is to enable network control and management on a high abstraction level by defining configuration rules called policies. Policies specify how a network node must be configured in vendor-independent, interoperable and scalable manner.

Diffserv architecture defines, at a lower level, four types of data-path elements: traffic classifiers, actions elements, meters and queuing elements [168]. Combining these elements into higher-level blocks creates a Traffic Condition Block (TCB), which can be managed by policy-based network management tools. The configuration of Diffserv TCB using PBNM involves the use of administratively prescribed rules that specify actions in response to defined criteria. All the information needed to perform this task such as profiles, user information, network configuration data, and IP infrastructure data such as network addresses and name server information are stored in a policy repository. These configurations do not change frequently because they are not associated with specific application or traffic but with the network management. The more difficult part in the configuration is to have the traffic entering the network appropriately marked (audio, video and other data). Since, the user is signed up for the service, edge devices could be configured to mark user's traffic with the appropriate PHB. With a known IP address and/or IP port number, the administrator can specify a policy that refers to user application IP address and marks traffic coming from that address appropriately.

In the actual configuration, when the user signs up for a particular multimedia service, he/she must specify his/her traffic profile and the action that must be taken when the traffic exceed this predefined profile (out of profile traffic). Generally, the out of profile traffic is dropped or marked as best effort traffic. This model is static and does neither respond to application needs nor favor an optimal utilization of network resources.

It is obvious that the solutions suggested by the IETF IP Diffserv working group makes it possible to privilege some MPEG-4 streams among others by the marking. The marking is performed according to the importance of the Audio Visual Object within the scene. In Chapter 4, Section 4.1, we have presented an algorithm that allows the server to distinguish between most important objects and lower important objects based on application level information found in objects descriptor or provided by the MPEG-7 framework. It is clear that IP Diffserv network will be deployed for next generation Internet since it provides QoS support for end-applications with scalable manner.

Open issues regarding the design of an efficient Diffserv PHB for the transport of packet video applications such as MPEG-4 video applications are summarized in the following: (1) what is the appropriate PHB that guarantee the better QoS for Audio Visual Stream?, (2) What is the IP packet marking policy (marking at the server or marking when entering the network)?, (3) Is it suitable to modify the marking of IP packets with respect to network states? (4) What is the most flexible network architecture to cope with all these requirements?

There are many available algorithms for metering and marking packets at the boundary of the differentiated services (Diffserv) domain. Among these algorithm we find, Time Sliding Window Algorithm (TSW) such as Time Sliding Window Three Color Marker (TSWTCM) [130]and Three Color Marker algorithm (TCM) such as Two Rate Three Color Marker (TRTCM) [131].

The TSW algorithm is based on rate estimation using a window. It provides a smooth rate estimation of the IP traffic throughput. The algorithm can mark the packet as out of profile one the traffic exceeds a certain target rate. It can be used to mark in profile traffic with EF PHB and out of profile traffic with AF PHB.

The TCM algorithm is based on a token bucket model. The token bucket defines three states. Thus, resulting marking can be done using the Olympic service (gold, silver, and bronze). The marking is based on a Committed Information Rate (CIR) and two associated burst sizes, a Committed Burst Size (CBS) and an Excess Burst Size (EBS). A packet is marked as gold if it doesn't exceed the CBS, silver is does exceed the CBS, but not the EBS, and silver otherwise.

It is clear that in case of video traffic, the above marker algorithms do not meet all the requirement of the video stream, since the video can be composed of important data and less important one. The above algorithms mark the video traffic according to the sending rate rather than according to the relevance of the data. For this reason, it is necessary to develop a specific marker for video stream in particular for MPEG-4 AVOs. We suggest that this marker will be implemented in the video server because it is the only element which is aware of the relevance and the importance of the streamed data. The next section presents our design philosophy of the marking algorithm.

5.2 DVMA: An IP Diffserv Packet Video Marking Algorithm

In this section, we propose a novel marking algorithm for IP Diffserv that will manage MPEG-4 video streams (see Figure 5-1). This algorithm can be implemented in the media server as well as in the Diffserv router. We assume that the video packets are encapsulated on RTP/UDP/IP and marked according to the algorithm described in Figure 5-1. Let us note that this algorithm is static and depends largely of our assumptions that are explained in the following (further we describe a generic marking algorithm). We note, that destination terminal (player) is able to specify its QoS requirements since it is aware of its processing, communication and storage capabilities. It is clear that in case of bi-directional real-time video applications such as videoconferencing, audio is more significant than video (property 1). This property led us to distinguish between marking from the audio and the video streams in Diffserv edge nodes.

```
if stream is "audio stream" then (application of property 2)
      if  coder rate is "low rate"
            then DSCP=AF Low Drop Prec
            //example AF11
      if coder rate is "medium rate"
            then DSCP=AF Medium Drop Prec
            //example AF12
      if coder rate is "high rate"
            then DSCP=AF  High Drop Prec
            //example AF13

if stream is "video stream" (application of property 3)
      if  "base layer video stream"              (level 1 = mimimum QoS)
            then DSCP = AF low Drop Prec
            //example AF21
      if  " enhanced layer video stream 1"       (level 2 = medium QoS)
            then DSCP = AF Medium Drop Prec
            //example AF22
      if  "enhanced layer video stream 2"        (level 3 =  maximum  QoS)
            then DSCP = AF Medium Drop Prec
            //example AF23

if stream is "objects descriptor, scene descriptor" (application of property 4 )
            then DSCP = EF
            //descriptions streams  are significant, no loss
            //it's necessary that these streams will be available
            //as soon as possible in MPEG-4 player to interpret
            //correctly the scene
```

Figure 5-1: DVMA- IP Diffserv video marking algorithm

If we look on PHB selection, it is obvious that for loss tolerant multimedia streams, like compressed video or audio they will be transmitted with an AF PHB. Moreover, EF PHB will be more expensive than AF PHB, because it guarantees additional constraints compared to AF PHB group (minimal latency, minimal jitter, and minimal loss). Once this choice is made, it remains us to identify the class of AF drop precedence that we use for IP packet. We recall that AF PHB group defines currently 4 AF classes within each class 3 degrees of drop precedence according to the importance of the packet.

Speech has a greater importance compared to the music or other background sounds in a scene. Thus, we privilege foreground audio speech compared to other audio. Moreover, using high quality voice coding tools, speech bit rates are between 2 and 24 Kbps. Signaling MPEG-4 information bit rates are in the same order of magnitude. Both data are essential for video quality decoding and rendering. Consequently, i.e. low bit rate streams are given higher priority score to raw video data (i.e. property 2).

MPEG-4 uses hierarchical coding for resolving system scalability and heterogeneity issues. Different coding modes exist: "s*patial scalability*" allows the decoder to treat a subset of streams produced by the coder to rebuild and display textures, images and objects video with a reduced spatial resolution. For textures, a maximum of 11 levels of spatial scalability is supported, while for the visual sequence, a maximum of 3 levels is proposed. *Temporal scalability* permits the decoder to treat a subset of streams produced by the coder to rebuild and display a video with reduced temporal resolution. A maximum of 3 levels is supported. With *SNR Scalability* (Signal to Noise Ratio), the coder transmits the difference between the original image and the preceding. This allows improving subjective quality of the final image to get maximum perceived quality (i.e. property 3).

Moreover, MPEG-4 introduces additional data control streams such as: the *scene description* and *media objects description*, all these signaling streams are very sensitive and need to be preserved from errors and loss during transmission (i.e. property 4). If EF PHB is not implemented, scene description stream will be marked as an AF PHB with low drop precedence (i.e. AF31).

We also assume that audio and video are carried with two distinct AF classes. This enables network nodes to better cope with congestion by given some routing privilege to audio compared to video. However, a problem of media synchronization may appear which can be tackled by using RTP Timestamps.

In order to generalize this algorithm we used our classification layer. The classification layer can classify any MPEG-4 AVO in the scene and affect one of the given labels. The label affected can be in our case the Diffserv PHB. Figure 5-2 gives a description of DVMA using the classification layer.

```
BEGIN
Initialize the classification process.
S = {AVO1, AVO2, …, AVOn} // Set of Audio-Visual Object
For each AVOi in S do
  Begin
      Calculate the Label l of the AVOi using the classification / mapping process
      Mark each packet resulting from the stream of the AVOi with the label l
  End
END
```

Figure 5-2: Generalization of DVMA using the classification model

One drawback of DVMA is its static behavior, i.e. the selection of the marking algorithm is known by advance and can not dynamically change according to network state variations. The solution of this problem will be presented later.

Second, the lack of flexibility since the MPEG-4 applications cannot express their needs in terms of particular processing of particular media stream i.e. due to a poor interaction between MPEG-4 applications and network delivery services. This is why we have developed the classification layer that allows building cross-layer QoS architecture by mapping from application level QoS to network QoS.

In order to prove our algorithm we have conducted an intensive experiment using both simulation and real testbed with Diffserv capabilities. We present in next Section 5.5 the results obtained.

5.3 An Adaptive IP QoS Matching Model

5.3.1 Static vs. Dynamic Policy-Driven Decision

In the Diffserv architecture, a particular traffic receives a predefined treatment based on predefined policies. This treatment is interpreted as a particular PHB [128], [129]. This task is done by the TC (Traffic Control) function, which assigns the correct DSCP [127] for the client's traffic according to it SLA (Service Level Agreement). Recall that each client defines it requirements and these are translated into SLAs. The allocation of resources (QoS) still static and can lead to bandwidth wasting and starving clients.

To receive a particular treatment, the user must specify it profile **TSpec** (Traffic Specification). TSpec specifies the temporal properties of a traffic stream selected by a classifier. It provides rules for determining whether a particular packet is in profile or out of profile. The Meter uses a Token Bucket to control user traffic. The following is a non-exhaustive list of potential profile parameters:

- *Peak rate p* in bits per sec (bps)
- *Token bucket rate r* (bps),

- *Bucket depth b* (bytes),

An *Excess Treatment* parameter describes how the service provider will process excess traffic, i.e. out of profile traffic. The process takes place after Traffic Conformance Testing. Excess traffic may be dropped, shaped and/or remarked. Depending on the particular treatment, more parameters may be required, e.g. the DSCP value in case of re-marking or the shapers buffer size for shaping. All these actions are decided once the network element is configured and are not changed over the time. Figure 5-3 gives an example of how out of profile traffic is treated using static configuration. In this Figure, user sends traffic not conforming to his Traffic Specification. Edge router control this traffic by a token bucket. Non-conforming traffic will be dropped always.

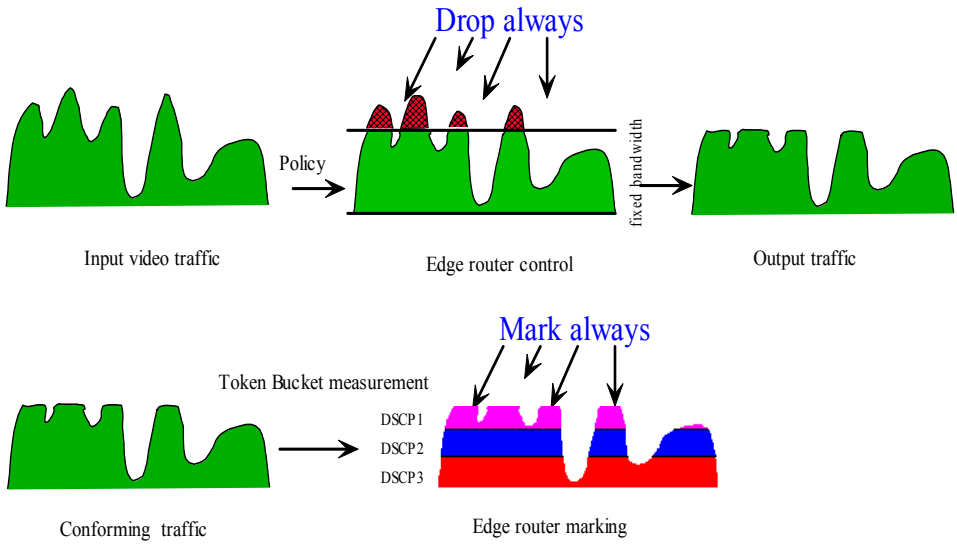


Figure 5-3: Static policy decision

For this reason, there is a great need to control dynamically the action taken by the network element for more flexible resource allocation. For this, different conditioning actions may be performed on the in profile packets and out of profile packets or different accounting actions may be triggered dynamically according to current network state. Clearly, a more flexible resource allocation can be achieved by controlling dynamically network elements behavior.

In the static approach out of profile traffic is simply dropped, remarked or assigned a new profile. This decision is static and is taken once for all, i.e. when the network element is configured.

For example the *Policing Rule* = *drop out-of profile* packets can be applied to all the packets which are out of profile regardless of whether the network is capable or not to transmit this packet.

Figure 5-4 shows where we can dynamically decide what actions must be applied to out of profile packets. In contrast to the static approach, these actions vary according to the network state (network link load, traffic behavior, etc.).

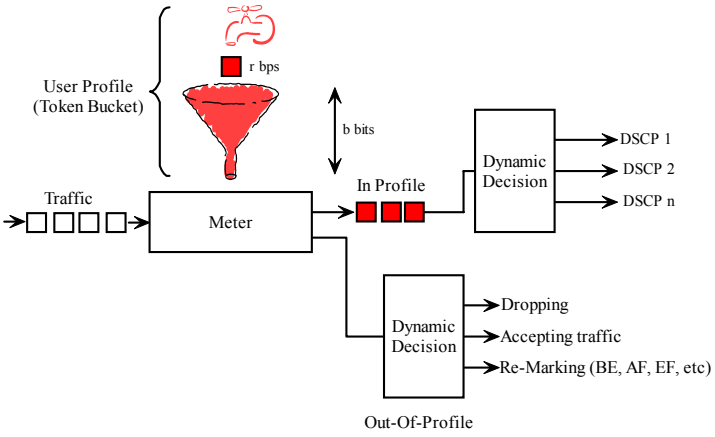


Figure 5-4: dynamic decision

5.3.2 Automatic Diffserv Domain Configuration

When a network element is started, its local PEP requests the PDP for all policies concerning Diffserv traffic marking using COPS (Common Open Policy Service) [137], [138], [139]. The policies sent by the PDP to the PEP, may concern entire router QoS configuration or a portion of it, as an updating of a Diffserv marking filter. The PDP may proactively provision the PEP reacting to external events generated by some monitors such as a bandwidth monitor.

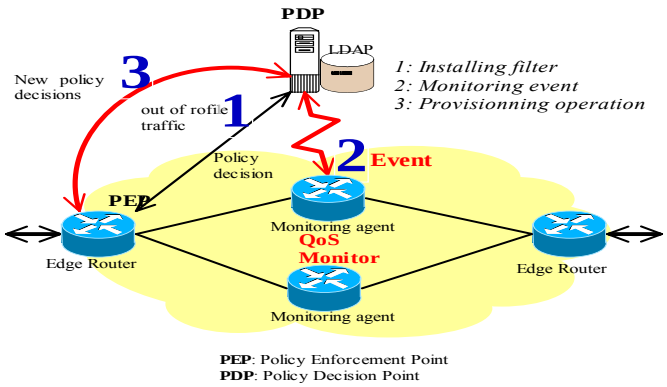


Figure 5-5: COPS-PR with monitoring event

Figure 5-5 shows the steps involved in the configuration of Diffserv domain. These steps are as follow:

- **Step 1**: When the edge router is started, the local PEP requests all policy decisions concerning Diffserv QoS Management (filtering, classes, queuing discipline, and actions for out of profile traffic). All incoming traffics are processed according to the pre-installed rules.
- **Step 2**: When the bandwidth monitor, installed on the core router, detects a significant change in the amount of available bandwidth, it triggers an external event reported to the PDP indicating the current bandwidth availability.

- **Step 3**: The PDP pushes to the edge router (PEP) an update of QoS Management decision.

These steps allow configuring correctly different policies related to the same traffic.

We introduce the following policy rule: let us **On event: If <profile> then <action>.**

A **Profile** is used to determine when a policy rule applies, for instance given pairs of source and destination addresses.

An **Action** is a performed by that the policy enforcement entity to traffic of a given profile. Examples of actions are marking, accepting or rejecting traffic.

Example of policy rules:

- Rule 1: Mark DSCP value EF on all packets with source addresses from 193.51.25.1 to 193.51.25.255 priority 0
- Rule 2: Mark DSCP value AF11 on all packets with destination address 200.200.200.100 priority 1

Assume, an audio application has subscribed to a particular Diffserv class (an Expedited Forwarding Class). Audio traffic is defined by a particular profile. In this example Diffserv class simply mean that the audio stream will be marked with the appropriate DSCP (EF PHB here). The administrator of the Diffserv domain configures the environment to support the Gold, Silver, Bronze and other services. Such configuration can be done through a Bandwidth Broker.

Supporting different classes of service in the core network requires putting in place classifiers, which cause the devices to examine the Diffserv mark on the packet and then treat the traffic accordingly. These configurations do not change frequently because they are not associated with specific application or traffic but with the network management. Since, the application is signed up for the service, edge devices are configured to mark application's traffic with the appropriate PHB. Based on the IP address and/or the port number, the administrator can set a policy that marks traffic coming from that address with EF PHB.

In order for customized traffic going to audio application (e.g. feedback traffic, RTCP, client commands) to receive a Diffserv treatment, policy must be deployed to the opposite edge device of a Diffserv domain.

When the audio application starts sending the data, the edge router must ensure; (1) the data sent by the audio server does not exceed what the application has subscribe-to (SLA) and (2) marking conforming traffic (in profile traffic) with the appropriate PHB (EF PHB in our example). In case of receiving out of profile traffic, the edge router requests a decision from the PDP. Since the PDP knows the current network state - because it receives monitoring information from different monitors installed in the network, it decides a new policy rule, for example dropping, marking or accepting out of profile traffic. This decision varies according to current network state.

5.3.3 QoS Management Algorithm Specification

We have configured 3 rules named Rule1, Rule2 and Rule3 to deal with video traffic. The Policy Server can choose one rule among the several depending on information sent periodically by the monitors. The monitoring information concerns essentially the bandwidth usage of each link in the network. The calculation of shaped value of the bandwidth using *Exponentially Weighted Moving Average* (EWMA) is presented later. Below in Figure 5-6 our algorithm, this uses the predefined policy rules to make a decision depending on bandwidth usage in the network.

```
Initialization:
Start Bandwidth Monitor Mi for each Router i to calculate
the available bandwidth BWi
Lambda ← 0.2      // Fixed value for historical data
X ←50% link capacity    // Initial value of EWMA
Min_th← 40% link capacity
Max_th← 70% link capacity
Loop:
BW ← max(BW1, BW2, …,BWi)  //EWMA available bandwidth X X←
(1-lambda) * BW + lambda * X
if  X < Min_th then
Rule1:
           Accept video traffic (in and out profile)
           Mark video traffic with Gold
else if Min_th<=X<Max_th then
 Rule2:
           Accept video traffic (in and out profile)
           Mark in profile traffic with Gold
           Remark out-of-profile traffic Silver
else
Rule3:
           Accept in profile traffic
           Drop out-of-profile Traffic
           Mark in profile traffic with Bronze
End.
End loop
```

Figure 5-6: Example of a simple algorithm using policies

Our algorithm uses a low-pass filter to calculate the bandwidth usage. Bursty traffic can cause a transient congestion. The bandwidth usage is not affected by this transient congestion since we shape this value. The low-pass filter is an exponential weighted moving average.

The EWMA *(Exponentially Weighted Moving Average)* Chart is used when it is desirable to detect out-of-control situations very quickly. It is an Exponential Smoothing technique that employs one exponential smoothing parameter to give more weight to recent observations and less weight to older observations and vice-versa.

When choosing λ, it is recommended to use small values (such as 0.2) to detect small shifts and larger values (between 0.2 and 0.4) for larger shifts [169].

Policy decision depends on the EWMA statistic calculated by each network monitor and sent to the Policy Decision Point to be aggregated.

Network Monitoring provides a global network status in terms of resource availability and resource consumption, which is required for the management of the available bandwidth on the network. It is an application that tracks on live resources information in the network. A framework

for supporting the traffic engineering of IP-based networks is presented in [170]. Different types monitoring measurements have been identified and are either **passive** or **active**. Passive measurement means that the statistics of the network element are maintained in the form of a Management Information Base (MIB), whereas in active measurement, test packets are injected into the network (like ping test) to gather information. Information collected about these packets are taken as representative of the behavior of the network. Metrics of this data are described in the framework presented in [171]. Different measurement can be made to collect information of the network. Here a list of not-exhaustive measurements:

- Flow-based: gathers information about each flow in the network, but it causes a scalability problem.
- Interface-based, link-based and node-based: collects information of each interface in the network element.
- Node-pair-based: it is used essentially to calculate the performance of edge-to-edge performance of a core network.
- Path-based: it is similar to pair-based measurement and may be used for admission control.

In the rest of this section, we will interest to QoS measurement, essentially the bandwidth usage metrics. This measurement is protocol independent and media independent to ensure both portability and commonality in the measurements and allow a dynamic packet video marking.

Figure 5-7 illustrates how hierarchical measurement estimates the traffic matrix for a large ISP. Each router performs a passive monitoring of incoming traffic (i.e. BW_{ij}: bandwidth usage for router *i* in its interface *j*). Through regular update, each router provides a partial view of the network. MCA (Measurement Collection Agent) aggregate this partial measurement the result is given to the PDP. PDP aggregates all the measurement into a matrix that gives the PDP a global status of the network and can generate feedback from the network.

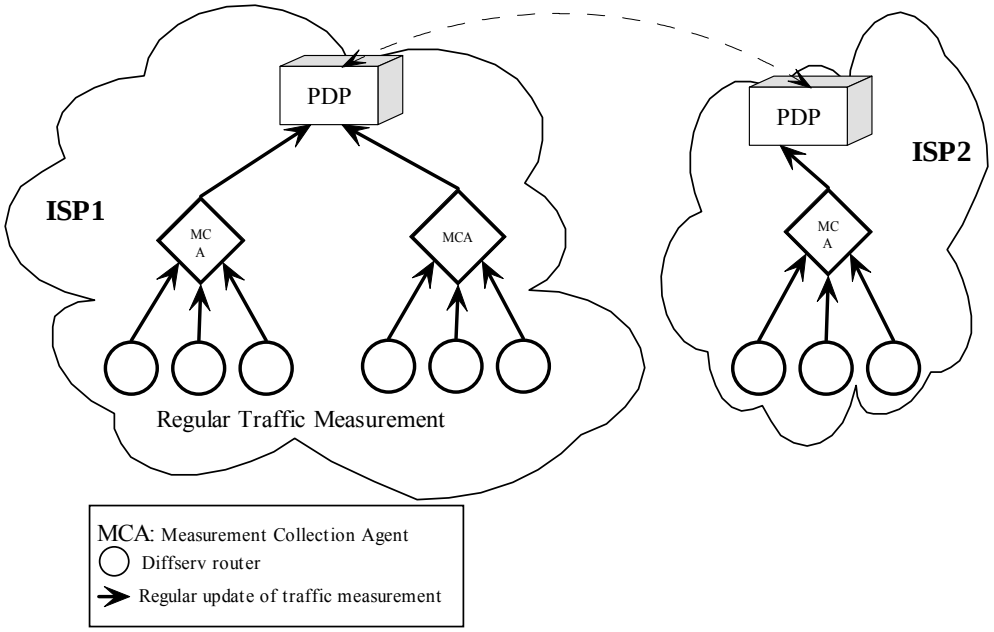


Figure 5-7: Hierarchical aggregation of measurement

5.4 Implementation Issues

Our prototype consists of three modules that perform dynamic packet video marking in an administrative domain; these modules are Policy-based network management tool, Network Monitoring System, and Policy System (Policy Decision Point and Policy Enforcement Point). Figure 5-8 shows the core components of our implementation.

5.4.1.1 Network Monitoring Agent

Our implementation consists of an agent written in Java which collects information on each interface of the router. The collected information consists of a real-time traffic flow measurement in input and output of each interface. This way, the agent augments the functionality of PEP by reporting monitoring information to the PDP in the form of COPS Report State Message. The PDP, when it detects a significant modification in the network state, delivers to the PEP a new policy decision in term of new policy rules. Decision-making is based on the algorithm described in Figure 5-6.

5.4.1.2 Policy Management System

This system is composed of a PDP and a PEP communicating using COPS protocol. All system components are implemented in Java. The COPS-PR implementation is simplified to exchange policy rule between PDP and PEP.

Simplified COPS-PR implementation is used to exchange policy rules between the PDP and the PEP.

The PEP is associated with the interfaces to which the marking must be applied (edge router). It is notified when the policy changes (or is newly) by a COPS provisioning operation. The PEP receives the policy information and transforms it into a form suitable for the device, e.g. using a Linux Diffserv Traffic Control API. After this, all incoming packets to this device will be marked according to the new marking policy.

The PDP is responsible for decision making and uses for that the network monitoring agents. Our implementation is limited to one domain (there is no inter-domain communication).

Our policy tool management is a Policy-based Web Bandwidth Broker. It consists essentially of a web interface installed in web application server. The administrator uses the web interface to configure the Diffserv domain and to enter new policy or to edit an old one. A Java Servlet engine is used to store all the information to a repository. We have used an OpenLDAP [172] server running on Linux. Other functions may be provided, such as validation, verification, conflict detection, etc. which are not yet available in our system.

In the top right of Figure 5-8, a simple web-based interface of the bandwidth broker is shown. It illustrates the edge router configuration, specially the filter configuration and how setting PHB for the traffic entering the network.

The network administrator uses the PBNM tool (BB) to configure the edge and core routers according to a predefined set of policies. Suppose that the administrator's domain can handle EF, AF11 and BE class only. The administrator configures the filters using also PBNM tool. The task of the filter is to mark the traffic entering the network with the appropriate PHB according to user profile. At this step, the administrator chooses how to handle excess of traffic (out of profile traffic) by tuning two control thresholds (Min_th and Max_th).

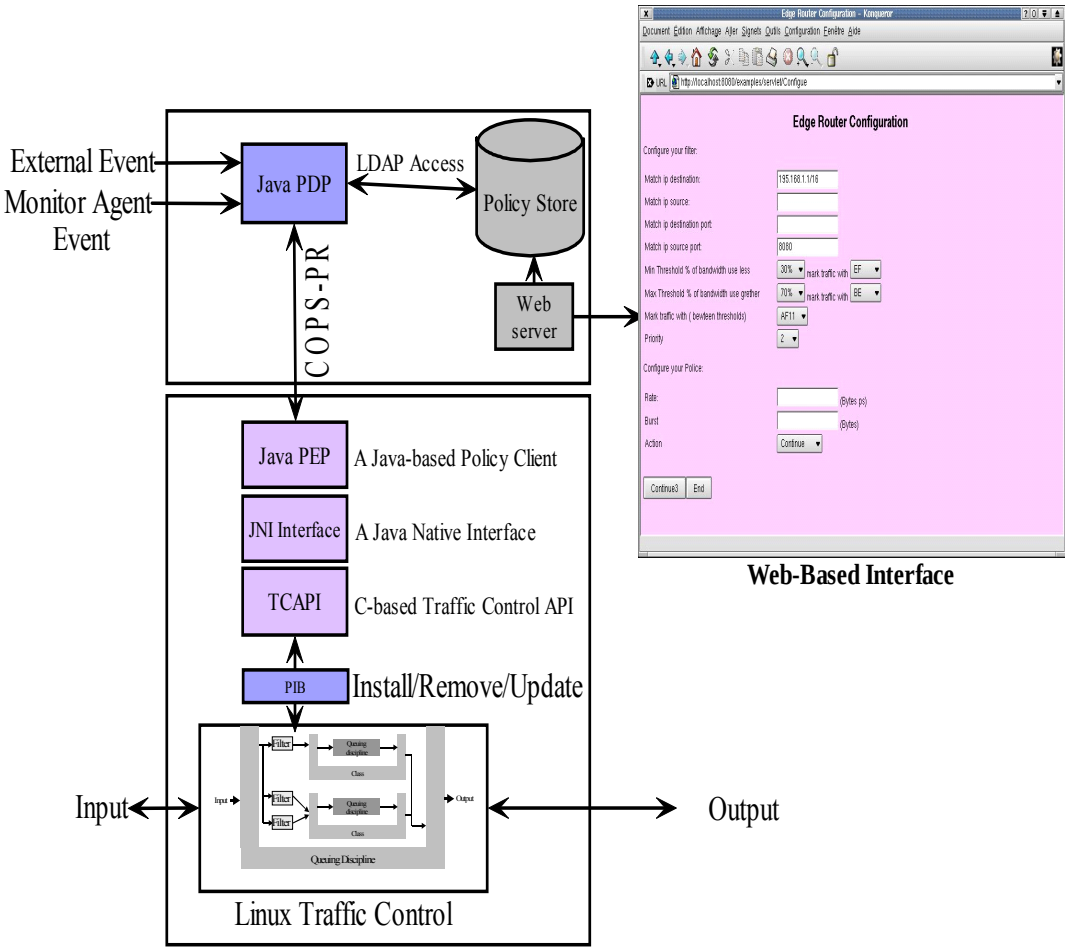


Figure 5-8: Dynamic policy-based management system implementation

The Diffserv router implementation is a based on the Linux traffic control implementation described in [153] and [154]. Each element of the Diffserv router is viewed as a set of components that can be managed via the interfaces specified in [183], [184] and [184]. The following elements are included in our router implementation (see Figure 5-9):

- Queuing disciplines: the queue determines the order in which data is transmitted. Each network interface has a queuing discipline associated with it, which controls how packets are treated.
- Classes: traffic can be divided into classes according to certain rules. Each class maintains a queuing discipline to serve it packets.

- Filters: to put packets into classes we use filters. Filters are used to distinguish among different classes of packets and process each class in a specific way.
- Policing: used to control the amount of traffic from a given class.

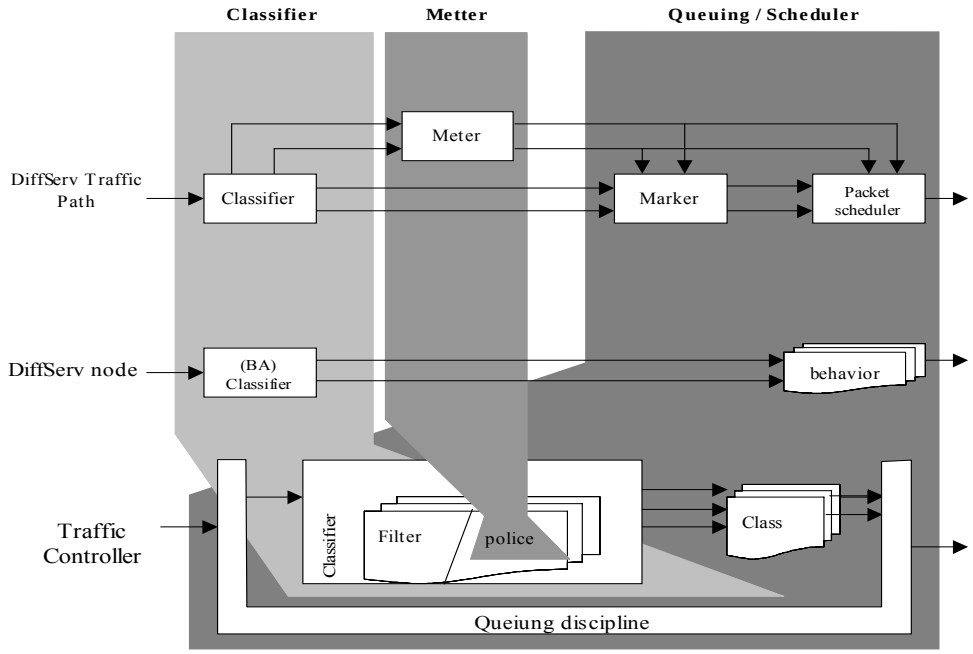


Figure 5-9: Diffserv router implementation

5.5 Performance Evaluation

5.5.1 DVMA Performance Using Network Simulations

In our simulation, the MPEG-4 sequence is obtained by using a set of multimedia components such as audio and video Elementary Streams ESs. Video ES is from the highly bursty Star Wars movie. Voice, Object Description, Scene Description and Data ESs were generated according to an exponential distribution. Three video streams were generated that are associated with:

- The MPEG-4 base layer video stream offers a minimum presentation quality (i.e. **minQoS)**,
- The MPEG-4 enhanced layer 1 video stream improves minQoS to a better quality (i.e. **MedQoS)**.
- The MPEG-4 enhanced layer 2 video stream improves medQoS to the maximum quality (**MaxQoS)**.

5.5.1.1 Network Models

We use the network simulation models depicted in Figure 5-10 and Figure 5-11 for evaluating an MPEG-4 video on demand (VOD) service over classical IP best effort service and IP Diffserv services respectively.

The both network topologies are similar and composed of an MPEG-4 video server "S", a receiver terminal "C" and 2 or 3 routers (R1, R2 and core routers). Background traffic is also transmitted to heavily load the network and charge the bottleneck link between routers "R1" and "R2". This traffic includes FTP data over TCP, and constant-bit-rate (CBR) data over UDP. Steady FTP sources send 1000-bytes length IP packets. A sink node immediately sends an ACK packet when it receives a data packet. CBR sources transmit IP packets of 210 bytes length at 56 kbp/s. Routers queue lengths are 50 packets. Figure 5-11 illustrates the IP Diffserv domain topology. An IP core Diffserv router is inserted between R1 and R2. This core router filters the incoming IP packets and reacts according to the DSCP marking performed by the server "S". The router "R1" implements classification, policing and shaping policies and ensures that incoming traffic obeys to the negotiated traffic.

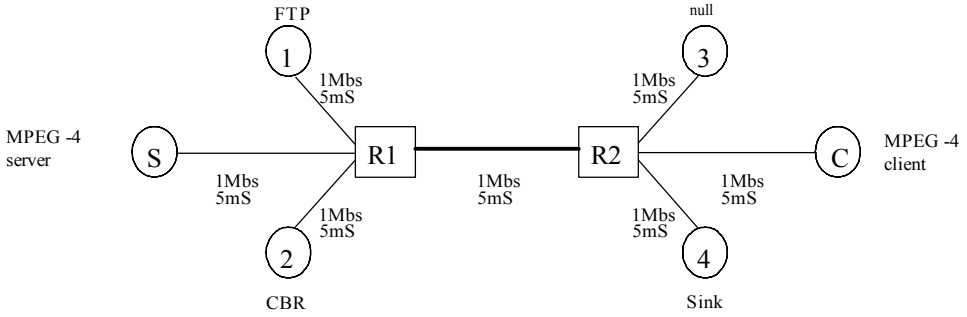


Figure 5-10: Classical IP network model

The peak rate of MPEG-4 stream is about 113 kbps. Two scenarios were experimented, the first one is the best effort model using Drop Tail queue in the router. The second one is Diffserv model in which the video traffic is marked using DVMA. The base layer stream is marked with the low drop precedence, the enhanced layer 1 stream is marked with the medium drop precedence and the enhanced layer 2 stream is marked with the high drop precedence respectively.

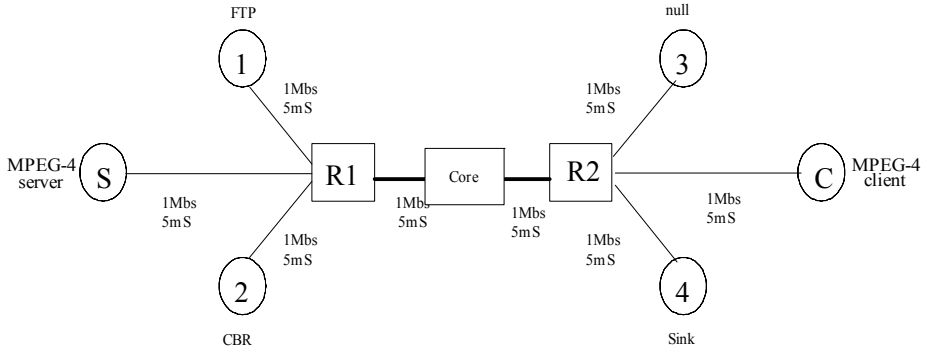


Figure 5-11: Diffserv network model

5.5.1.2 Simulation Results

Figure 5-12 presents the instantaneous throughput of the stream used in the simulation along with the whole MPEG-4 scene. The instantaneous queue size after simulation is presented in Figure 5-13 for the IP best effort and the IP Diffserv models. With the classical IP service, important packet loss appears when load increases as presented in Figure 5-14. There is no distinction between the MPEG-4 sub flows during the dropping process. This is because of the lack of priority

mechanism associated with the Drop Tail Queues. Therefore, all video packets are dropped with an equal probability and regardless to their relevance.

Figure 5-15 shows the end-to-end IP packet transmission delay (IPTD) for the IP best effort. IPTD variations are correlated with the queue size and the packets loss. The more the queue is on congestion; the more the delay to reach the destination is increased for a particular packet.

With IP Diffserv, DVMA is activated. The core router queue size is shown in Figure 5-13, and the video packets loss in Figure 5-16.

FTP and CBR traffics are marked using TSWTCM, whereas, MPEG-4 video is marked according to the proposed DVMA algorithm.

We notice that packets are dropped according to their priority score. Less important packets are dropped first to maintain a low queue size. Consequently, MPEG-4 main video stream is preserved and encounter limited packet loss probability.

Similarly, router queue size is smaller in the IP Diffserv domain than with the IP Best Effort. The early congestion detection algorithm implemented in Diffserv router queue explains this fact. This result to a lower utilization of the queue capacity. The performance can be improved by better tuning the minThreshold and maxThreshold in a situation of multiple shared RED queues.

Finally, Figure 5-17 presents the end-to-end packet transmission delay with IP Diffserv model.

Performance measurements of the Diffserv model are better than with Best Effort. The mean packet delay is about 0,1s with the best effort, and only about 0.05 when using Diffserv AF PHB.

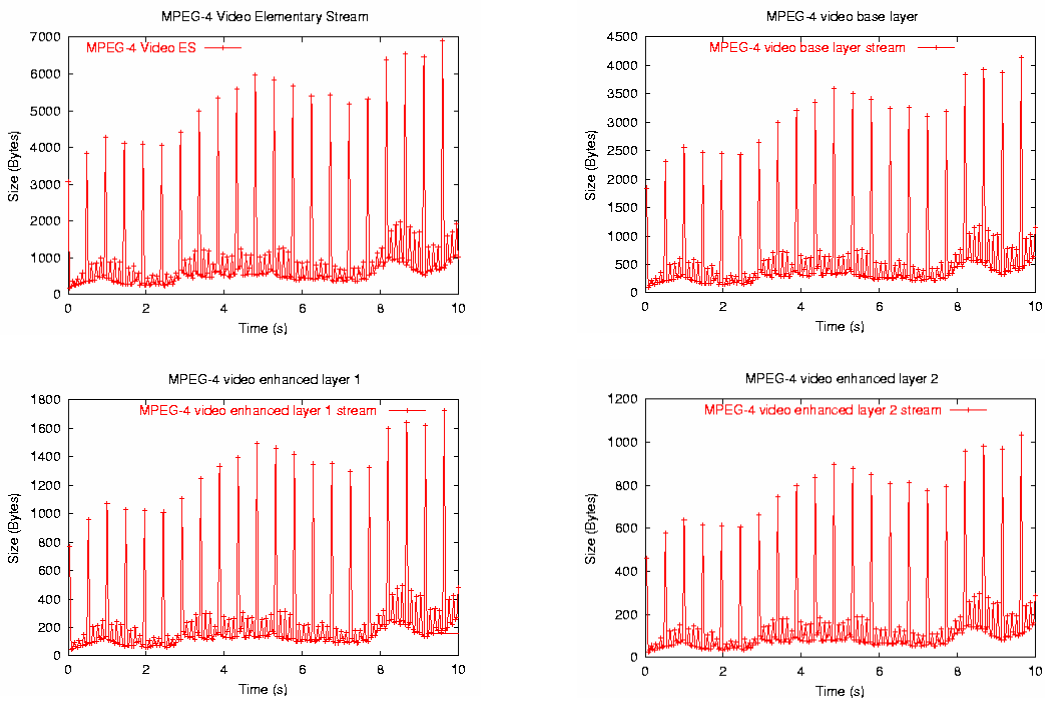


Figure 5-12: The MPEG-4 elementary stream bit rates

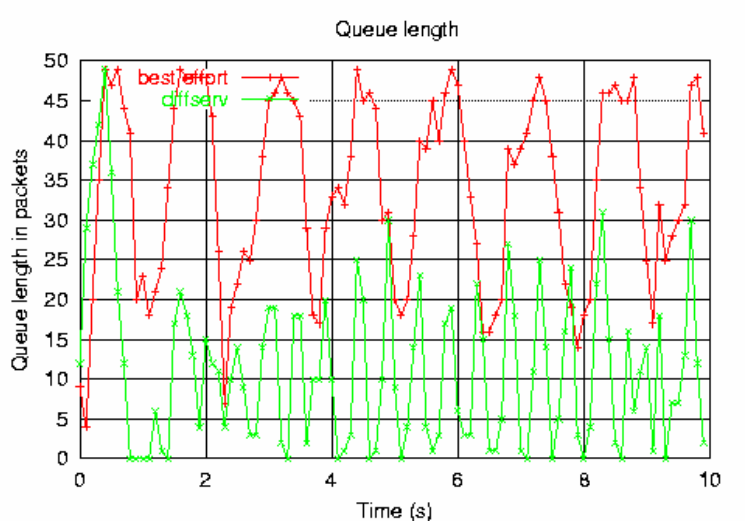


Figure 5-13: Instantaneous queue size with IP Best Effort and Diffserv scenarios.

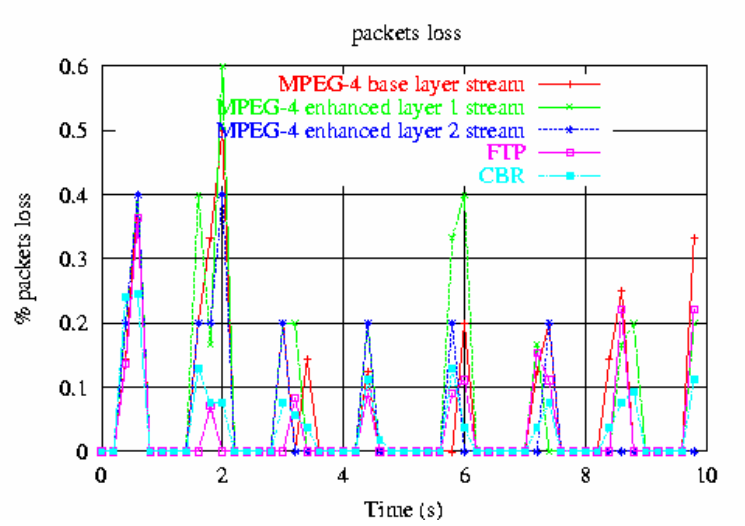


Figure 5-14: Packets loss with IP Best Effort scenario.

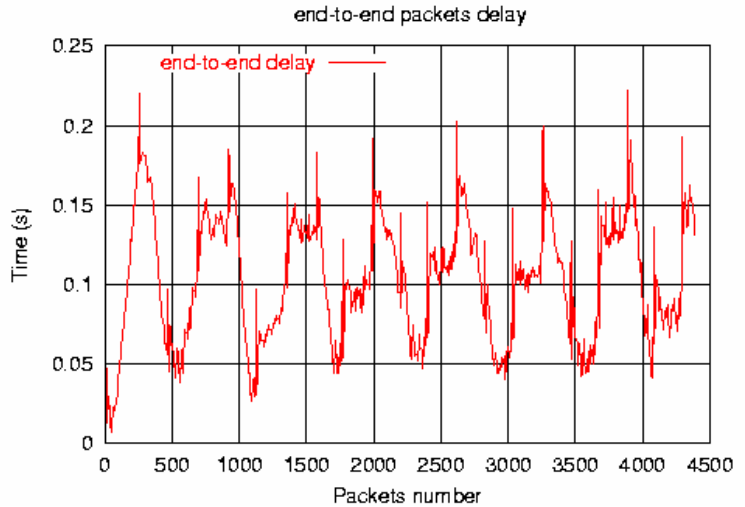


Figure 5-15: End-to-end packet transmission delay with Best Effort scenario

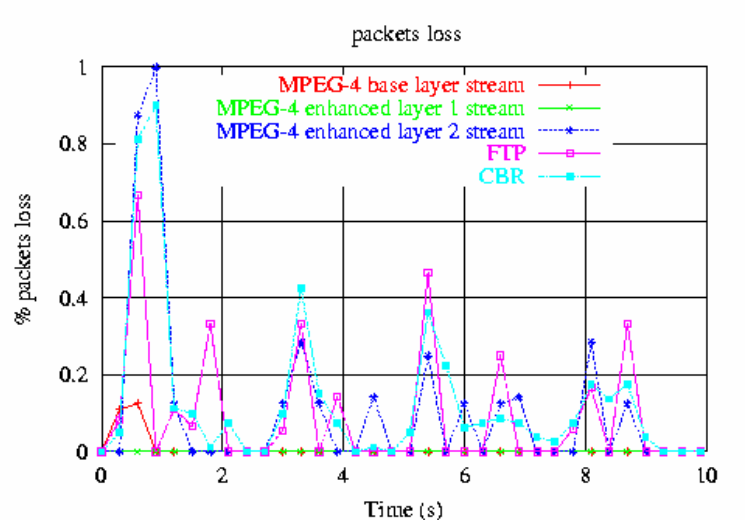


Figure 5-16: Packets loss with IP Diffserv scenario

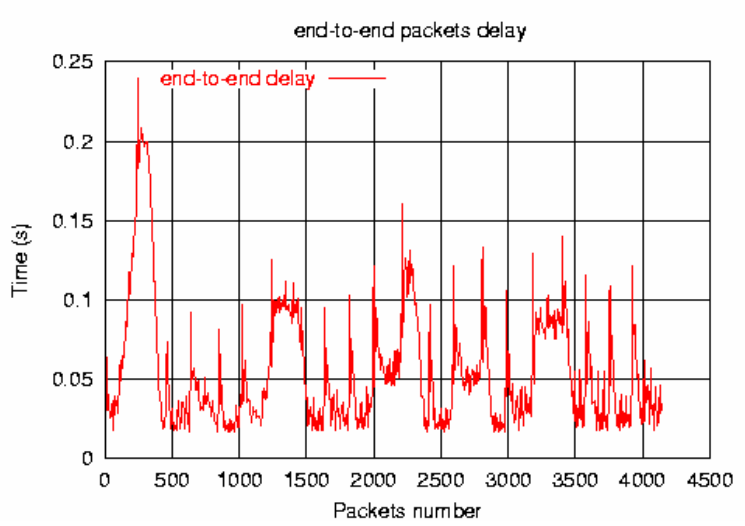


Figure 5-17: End-to-end packet transmission delay with IP Diffserv scenario

5.5.2 DVMA Performance Using Network Experiments

Let is consider the IP network testbed depicted in Figure 4-6., where a server delivers an MPEG-4 video content to several heterogeneous receivers. The receiver can be a simple wired PC or any wireless terminals such as a mobile GSM/GPRS/UMTS phone capable of rendering MPEG-4 video sequences. The network nodes are IP Diffserv compliant routers. We use Linux-based IP routers with Diffserv implementation. The testbed is composed of tow edge routers and a core router running Linux with IP Diffserv components. All Ethernet links are 10 Mb/s.

Our IP Diffserv network testbed is composed of a video server application that plays MPEG-4 video stream for many heterogeneous clients. The server and clients are connected through TCP/IP network. We exploit some test-scenario on the transmission process. We quantify QoS measurement and we compare between transmission process in Best Effort service and in Differentiated Service (with DVMA activated).

5.5.2.1 Edge Router Configuration

Edge routers accept traffic into the network. They can characterize, police, and/or mark customer traffic between other edge or core routers.

Within the Diffserv domain, service association is performed according to the DSCP value in each packet's IP header. Therefore, the video application must mark the packets correctly to obtain a particular level of service within the Diffserv region.

The configuration of the edge router is simple in our testbed. Our edge router limits the amount of EF traffic to 15% of the bandwidth capacity rate i.e. 1.5Mbit. We used a policing mechanism to limit the EF traffic, because EF is more require in term of latency time and losses. Furthermore, the router must make sure that the departure rate configured is greater than the arrival rate and the queuing delay is minimized. This is extensively sufficient since we use EF only for sensitive information such as OD and BIFS signalling, that should be transported to the destination as early as possible, with no loss and with a minimum jitter. The EF flow is bounded and isolated.

For the Edge Router we used a simple *Class-Based Queuing* (CBQ) discipline to classify the incoming traffic.

5.5.2.2 Core Router Configuration

Core routers are configured to perform (1) packet classification based on DSCP, (2) packet scheduling, (3) queue management, (4) policing and (5) packet dropping.

We used CBQ as the packet scheduler, which is a classical assumption as proposed in [164]. For CBQ a single set of mechanisms is proposed to implement link-sharing and real-time services. In our implementation, CBQ is used to classify EF, AF, and BE traffic classes so each connection can get appropriate resources based on packet marking.

Our CBQ mechanisms include:

- a *classifier* to classify arriving packets to the appropriate class. This classification is based on DSCP field in the IP header,
- a *scheduler* to determine the order in which packets from the various classes will be sent. Linux Kernel implements several queuing disciplines (e.g. **RED** "Random Early Detection" or **GRED** "generalized RED"). The GRED queuing discipline is used to support multiple drop priorities as required by AF PHB. One physical GRED queue is composed of multiple **VQ** (Virtual Queue). GRED can operate in ***RIO*** *(*RED with In/Out bit) *mode* [165], with coupled average queue estimates from the virtual queues, or in *standard mode* where each virtual queue has its own independent average queue estimate as required by RED [166]. In our testbed, we used GRED as queuing discipline for the AF classes, since our marker algorithm takes into account these properties to give different level of QoS: *minQoS*, *MedQoS* and *MaxQoS*.

For the AF classes we allocated 1.5Mbit/s for each AF sub-classes namely AF1, AF2, AF3 and AF4, all of which are bounded. For the best effort traffic, we allocated a minimum of 3.5Mbit but this traffic is allowed to borrow any available bandwidth.

5.5.2.3 Experimental Results

Figure 5-12 gives statistical properties of the MPEG-4 video traffic generated by the video server to the client. The network is loaded by TCP and UDP background traffic. In our testbed, the background traffic is marked as best effort traffic.

The first set of performance measurements are on packet loss probability for each video streams, in both IP best effort and Diffserv models. The second set of measures concern the end-to-end one-way delay encountered by video packet between the server and the destination. Two different network loads have been tested, i.e. 80% and 95% of the available bandwidth.

5.5.2.3.1 IP Packet Losses

Figure 5-18 and Figure 5-19 depict the variation of the video packet loss versus network load for IP Best Effort and IP Diffserv respectively. Individual MPEG-4 video layers encounter different packet loss probability. With IP Best Effort, the most essential video layer (i.e. base layer) obtains the highest loss with 60 % for a network load of about 80%. Using IP Diffserv and DVMA, the base layer faces the lowest packet drop probability with a maximum of about 0.2 %.

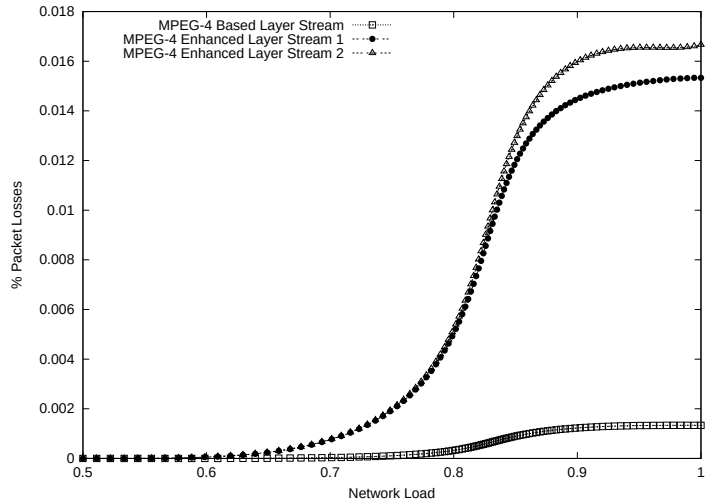


Figure 5-18: MPEG-4 video packet loss ratio vs. network load with IP Diffserv

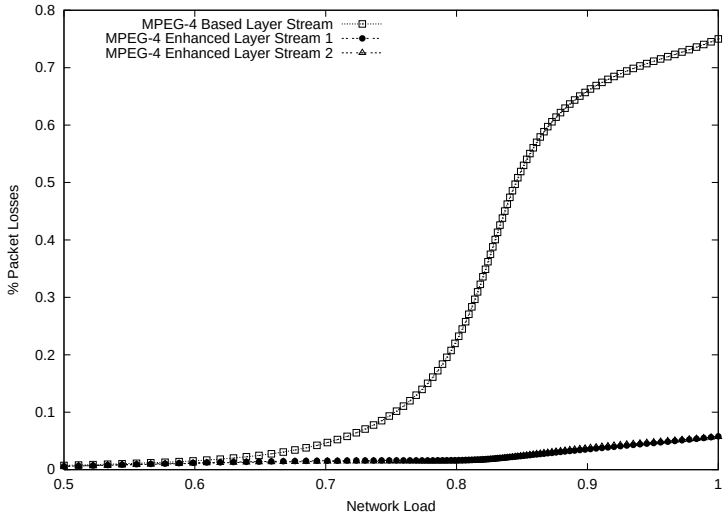


Figure 5-19: MPEG-4 video packet loss ratio vs. network load with IP Best Effort

Figure 5-20 shows the percentage of packets losses when the amount of background traffic is about 80% (6.5 Mbit/s) of the bandwidth. This leads to some losses essentially at time t=30s i.e. when the video server sends at its peak rate. The majority of the losses are concentrated within the base layer stream. This provides a degradation of the video quality at the client. Moreover, the receiver can't decode properly the other elementary video streams without the good reception of the base layer. Loss increases dramatically when the network load increases (Figure 5-21).

The high losses of the base layer are due to its highest requirement of bandwidth. We can compare it with a MPEG-2 video stream where I pictures are bigger than P and B pictures. However, they are much more important. In this case, I packet's losses must be lower than the other packets types losses. When talking about MPEG-4, the base layer stream must have a low packet loss when the enhanced layers streams 1 and 2 must have a respectively increasing drop probability.

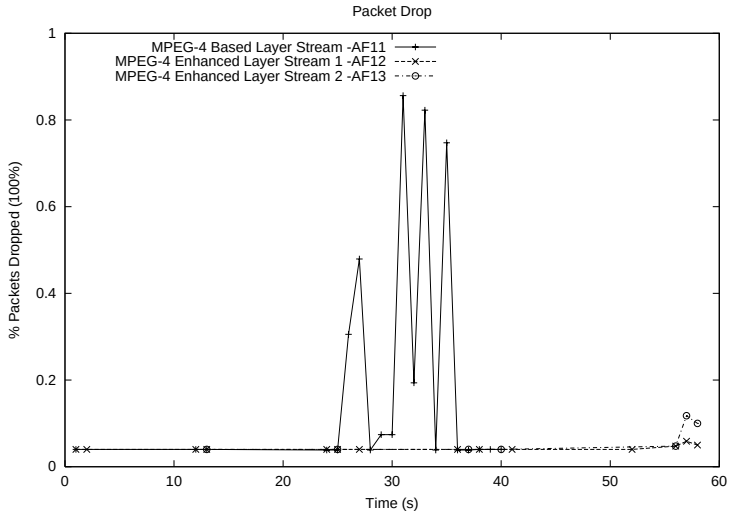


Figure 5-20: Packet drops in Best Effort scenario. - network load 80% -

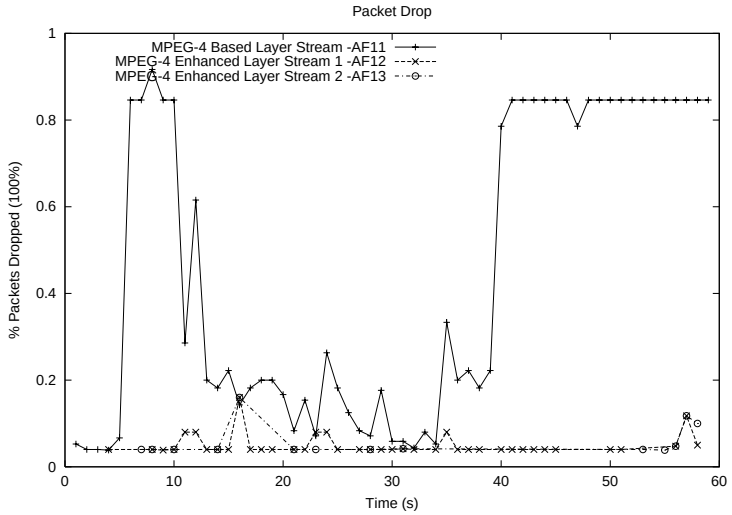


Figure 5-21: Packet drops in Best Effort scenario. - network load > 95%

With IP Diffserv, we can see that the video packet losses are almost equals to 0, and we have ensured that the base layer has no losses. Figure 5-22 and Figure 5-23 illustrate this fact.

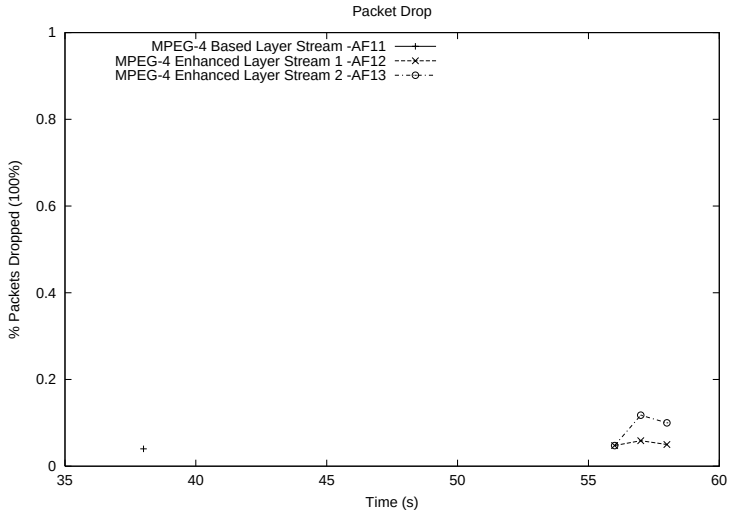


Figure 5-22: Packet drops in IP Diffserv.- network load 80% -

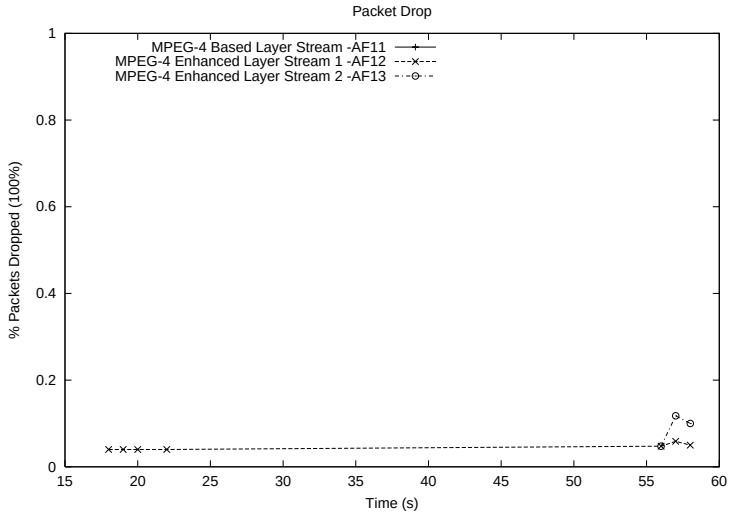


Figure 5-23: Packet drops with IP Diffserv- Network Load >95% -

5.5.2.3.2 End-to-end Transmission Delay

Figure 5-24 and Figure 5-25 illustrate the end-to-end delay, which is an important component of the user-perceived QoS. Since, we are dealing with real-time stream; two much delayed audiovisual IP packets are simply discarded by the destination.

We can notice that the end-to-end transfer delay is almost similar regardless the network's load, i.e. 80% or 95% of the bandwidth. It indicated that traffic load variation have no deep effect upon the video traffic. This is simply due to the static priority-based packet scheduling mechanism performed by the gateways. We also note that best effort traffic class (background traffic) can dynamically use any available bandwidth.

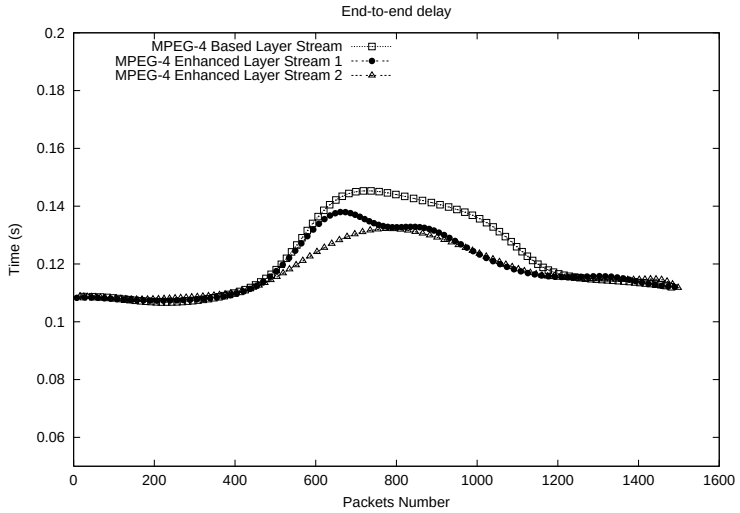


Figure 5-24: End-to-end transfer delay with IP Best Effort- network load of 80% -

When using the Diffserv scenario, the packet delay is decreasing and doesn't really increase when the network load reaches 95%. In both Figures 16 and 17 we can also see that the highest delay is during the peak rate at the middle of the transmission process; i.e. 116 ms.

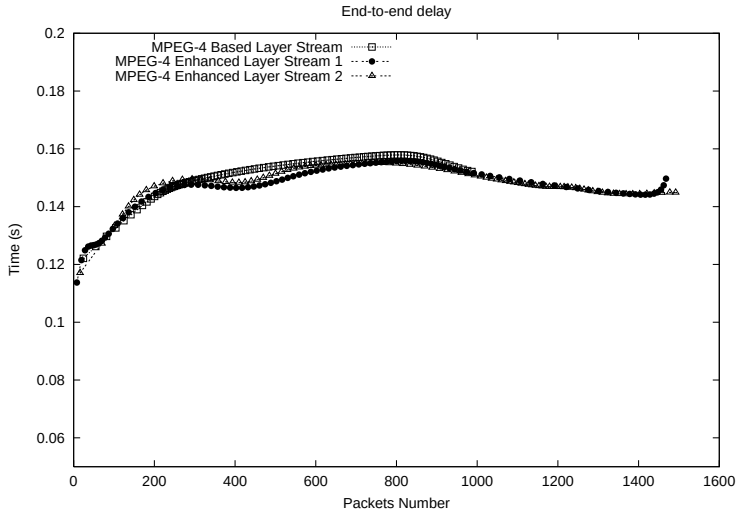


Figure 5-25: End-to-end transfer delay with IP Best Effort - network load > 95% -

Figure 5-26 shows that during the peak rate, the delay dramatically increases. When the network load is about 95%, the one-way delay measurement is the highest, about 150 ms (Figure 5-27).

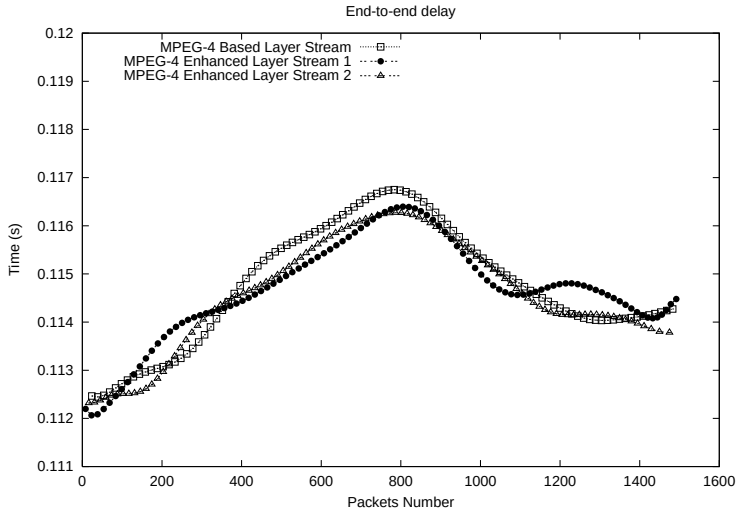


Figure 5-26: End-to-end transfer delay with IP Diffserv - network load of 80% -

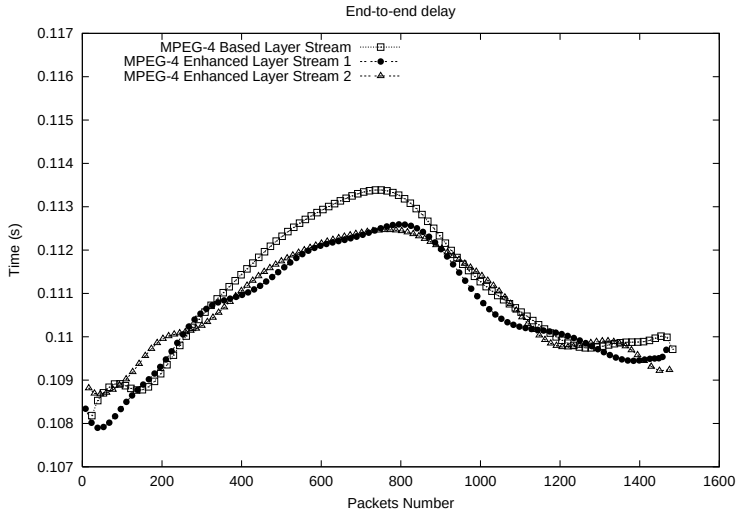


Figure 5-27: End-to-end transfer delay with IP Diffserv - network load > 95% -

5.5.3 Performance Analysis of the Dynamic QoS Adaptation Model

5.5.3.1 System and Network Models

Figure 5-28 depicts our experiments testbed. User transmits a customized traffic (MPEG-4 simple profile traffic) across a Differentiated Services network. The network is composed of Diffserv capable routers. We use Linux-based IP routers with Diffserv implementation [153], [154]. The testbed is composed of two edge routers connecting by 10 Mb/s Ethernet links.

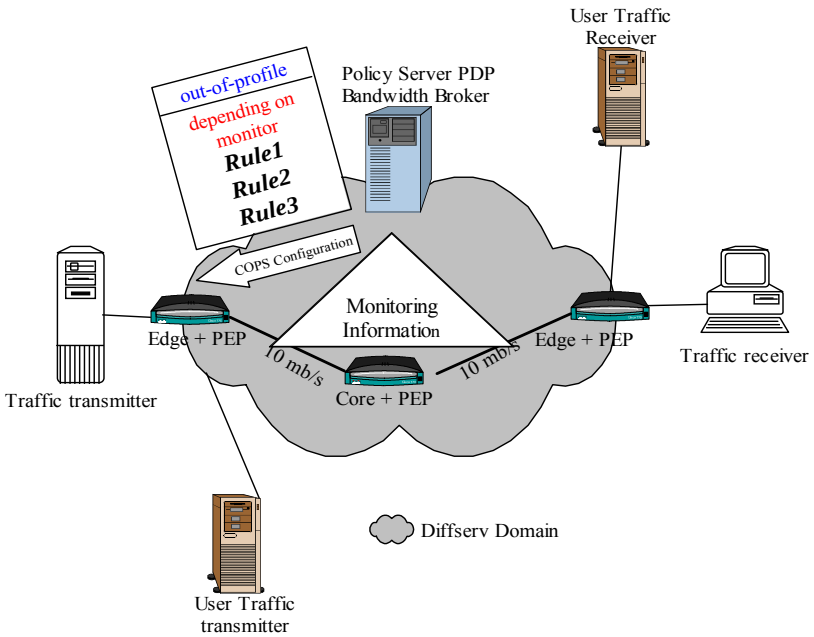


Figure 5-28: Experimental environment

By using our PBNM tool, we allocated 1.5Mbit/s for each AF class (i.e. AF1, 2, 3 and 4), all of which are bounded. We limit the amount of EF traffic to 15% of the bandwidth capacity rate, i.e. 1.5Mbit and we allocated 3.5Mbit for the best effort traffic that are allowed to borrow any available bandwidth. To get distinguish class of service, we used CBQ as our packet scheduler, In our implementation, CBQ is used to classify EF, AF, and BE traffic so that each user can get appropriate resources based on packet marking. The scheduler of the Diffserv core router employs GRED queuing discipline to support multiple drop priorities as required for the AF PHB group.

The network is load using *n* IP traffic generator. One traffic generator is composed of a traffic sender and a traffic receiver. The traffic sender generates a UDP packet of 1024 bytes with IP and UDP headers according to a Poisson distribution with parameter $\lambda = 128$ packet/s that gives 1Mbit/s per traffic generator. In our test, and since our Ethernet links are 10 Mbit/s, we have taken *n=5, n=7 and n=10* in order to load the network differently each time. Each source can be either on or off during exponentially distribution on/off period with an average of $\lambda_{on} = \lambda_{off} = 1s$.

Edge router performs policy for particular video traffic, which is identified by a couple <IP_adr, Port_number>. Policy determines whether the video traffic is in or out-of profile. This task is done by a token bucket (r,b) (**r** is the rate at which the token are placed in the bucket and **b** the size of bucket) optional parameters can be used such as a peak rate (**p**), a minimum policed unit (**m**), and a maximum datagram size (M).

The token bucket and peak rate parameters require that traffic obeys the rule that over all time periods, the amount of data sent cannot exceed M+min[pT, rT+b-M] [173]. M is the maximum datagram size, and T is the length of time period. Datagrams which arrive at an element and cause a violation of the M+min[pT, rT+b-M] bound are considered out of profile (non-conformant) and require a decision from the PDP.

In our experiment, we take these parameters for the token bucket: *r=600Kbit/s* and *b=2K,* to handle user traffic. This means that video traffic must not exceed 600Kbit/s otherwise it will be considered as out-of-profile traffic.

For a testing purpose, we transmit a high quality MPEG-4 simple profile video stream to see the reaction of our system. Figure 5-29 shows the MPEG-4 video stream sent by video application. The average rate of this video is about 800Kbit/s and the peak rate is about 1.4Mbit/s. Video traffic is not conform to the traffic specification, the exceed traffic is considered out-of-profile. Edge router marks the video traffic according to the dynamic policy provisioning.

In our experiment, in profile traffic will be marked with Gold PHB as much as there is no congestion and when congestion occurs it will be marked with Bronze PHB. Out-of-profile traffic will be marked either by Gold or Silver or it can be dropped (according to network status) dynamically.

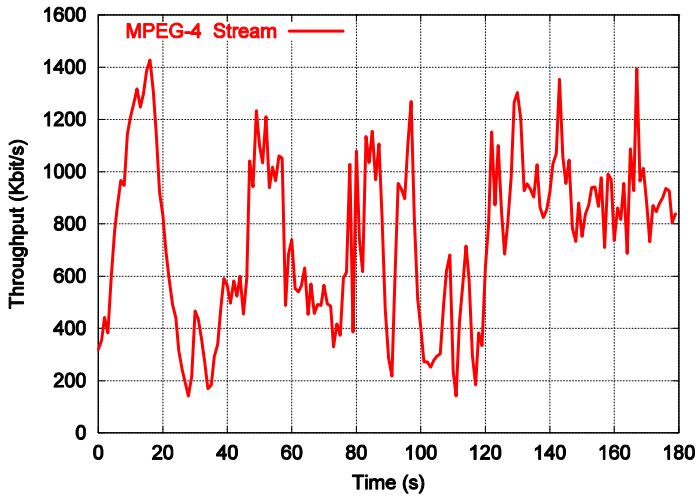


Figure 5-29: Video stream sent by the user

5.5.3.2 Experimental Results

The network is load differently each time. Figure 5-30 shows the network load during the period of the experiment (180s). This load represents the traffic sent from the *n* traffic generators to the receivers. This measure has been taken from the ingress interface of the core router. During to first 60 second there are only *n=5* sources on/off. From 60 to 120 second there are *n=8* sources and in the last 60 second (from 120s to 180s) the number of the source are *n=10*. Each source can be either on or off.

The PDP makes the decision according to the smoothing value of the bandwidth usage (i.e. EWMA). This decision is an identifier of the policy that the PEP must install. The following values Min_th=4Mbit and Max_th=7Mbit determine whether the video path is in congestion or not.

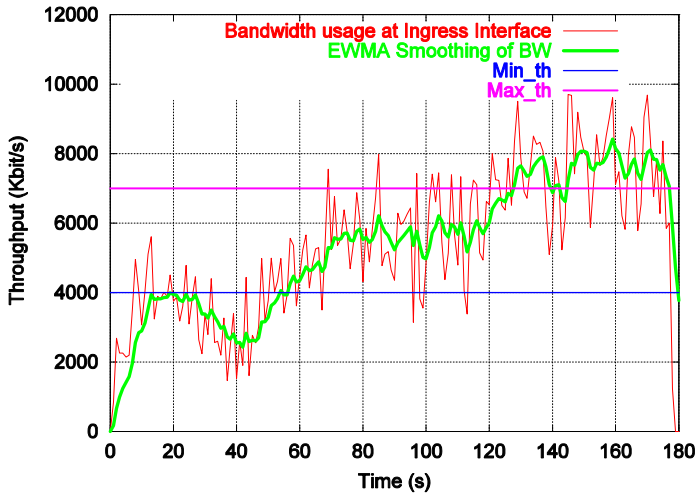


Figure 5-30: Bandwidth usage in core router

Policies used to mark video traffic are listed in Table 5-1.

Events sent from the PDP to the edge router doing the marking are listed in Table 5-2. Each event is time stamped.

Id	Condition Indicator	Action
1	NetworkUnderLoad	• **Accept** video out-of-profile traffic • **Mark** in and out of profile video traffic with **Gold** PHB
2	NormalNetworkLoad	• **Mark** out-of-profile video traffic with **Silver** PHB • **Mark** in-of-profile video traffic with **Gold** PHB
3	NetworkCongestion	• **Drop** out-of-profile Traffic • **Mark** in-of-profile video traffic with **Bronze** PHB.

Table 5-1: Policy for video marking

Time (Second)	Action taken by the edge router (Policy ID)
0	Id=1
53	Id=2
54	Id=1
56	Id=2
127	Id=3
139	Id=2
140	Id=3
142	Id=2
144	Id=3
177	Id=2
179	Id=1

Table 5-2: List of policies event sent by the PDP to the edge router

According to these events, we can say that the video traffic can get each time a dynamic behavior. This-is very interested function, since the Internet Service Provider can make new strategies of traffic engineering. Figure 5-31 shows the video traffic received at the end-user terminal and Figure 5-32 details the different PHB color assigned with video traffic when it enters the network.

In our scenario (see Figure 5-32), there are three significant time intervals, the first one from [0s,56s] where the edge router accepts the video traffic (in profile and out-of-profile) because the network congestion value is set to *NetworkUnderLoad* (id=1). In this case and according to the predefined policies, traffic is marked with Gold PHB. Form [56s,144s], the edge router accept out-of-profile traffic but with a lower priority i.e. marked with Silver and when network congestion occurs it will be first dropped. In profile traffic is marked with Gold. In this time interval the network congestion value indicates *NormalNetworkLoad* (id=2). In last time interval [144s, 180s], the network load augments (*NetworkCongestion* id=3), for that reason, out-of-profile traffic is dropped, and in-of-profile is marked always with Silver PHB.

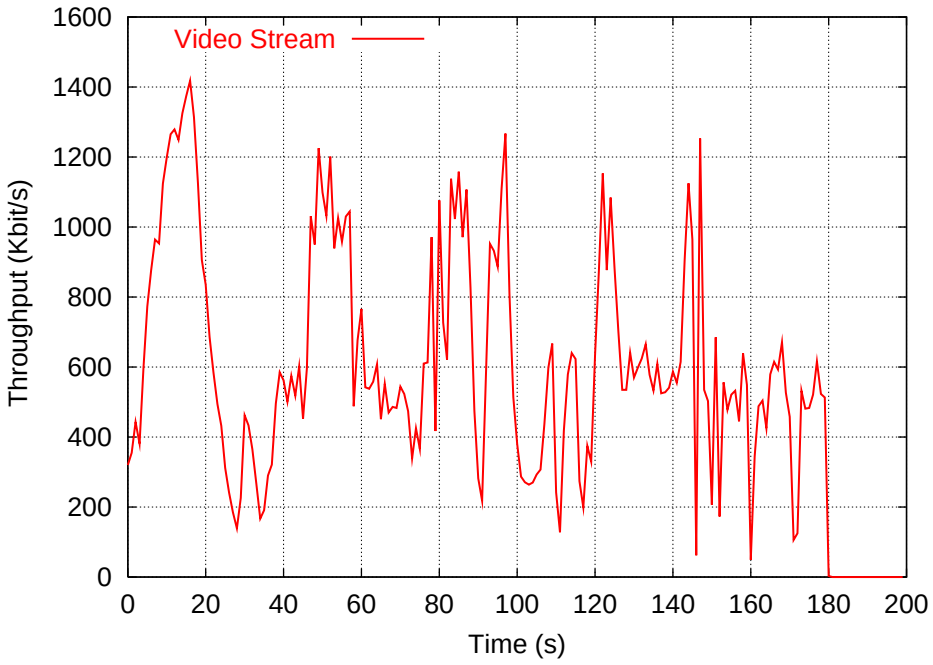


Figure 5-31: Received video traffic

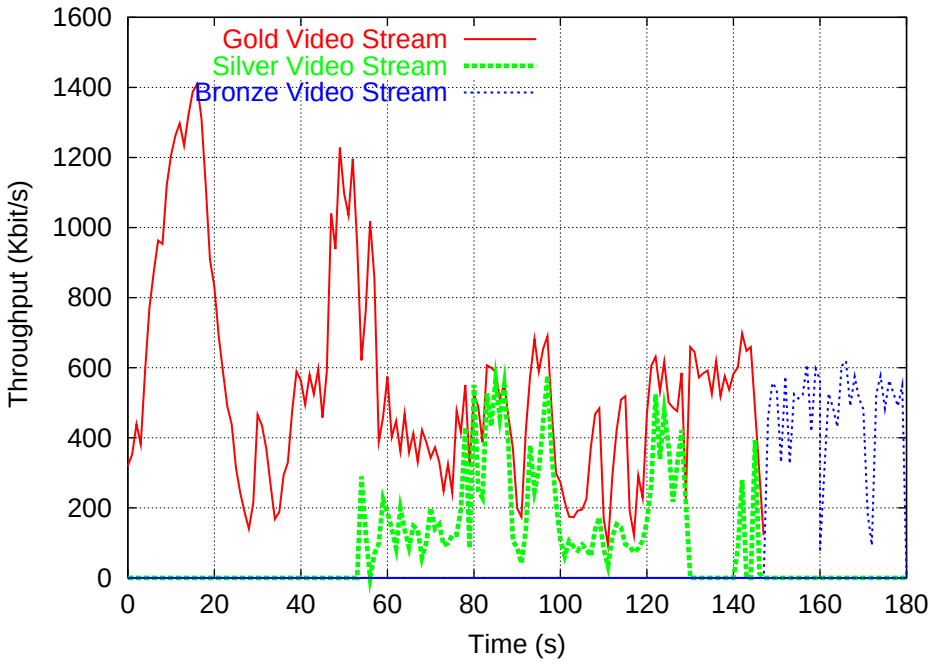


Figure 5-32: Different PHB color of the received video

5.6 Conclusion

In this chapter we have proposed a video packet marking algorithm called DVMA. DVMA is designed and integrated in a QoS effective IP video delivery framework. Performance results have shown that two sensitive QoS parameters have been sensibly improved during network overload: video packet loss and end-to-end video packet transmission delay.

The proposed marking algorithm better takes into account the characteristics and relevance of MPEG-4 sub streams (audio, video, BIFS, OD signaling…) and performs well with Assured Forwarding IP Diffserv PHB. Consequently, sensitive video streams will undergo a privileged processing by the routers using our proposed transport delivery service.

The problem of DVMA is it static behavior. To overcome this limit we proposed a dynamic packet video marking that can be used to mark the video traffic according to network state. Our proposal uses network monitoring feedback and policy decision point. The collected monitoring information is used to manage and to adapt dynamically QoS parameters for user traffic by changing the marking strategy. The example configuration rules described in our testbed clearly demonstrate the advantage of using our proposed algorithm. Our system involves a policy-based management system to achieve a more dynamic network behavior in handling user traffic.

Several issues arise when using dynamic control decisions to handle out-of-profile traffic. One problem is the pricing and charging schemes in use: Who pays for the service (out-of-profile traffic), the sender or the receiver? More work has to be done in order to define accurately the amount of traffic that excesses the profile in order to establish a payment scheme. Also, time-scale measurement of the PDP response is important and should be evaluated in future work.

Chapter 6

6 A SIP/MPEG-4 Multimedia Interworking Signaling Gateway

This chapter discusses technical issues related to delivery and control of IP multimedia services, such as video-conferencing, involving heterogeneous end terminals. In particular, it describes the design and implementation of an experimental multimedia interworking signaling gateway between IETF SIP (Session Initiation Protocol) and ISO MPEG-4 DMIF (Delivery Multimedia Integration Framework) session and call control signaling protocols. This IP videoconferencing interworking system is composed of two core subsystems for supporting delivery of audio-video streams from a DMIF domain to a SIP domain (i.e. DMIF2SIP subsystem) and from a SIP domain to a DMIF domain (i.e. SIP2DMIF subsystem). These subsystems perform various translation functions for transparent establishment and control of multimedia sessions across IP networking environment, including, session protocol conversion, service gateway conversion and address translation.

IP videoconferencing and IP telephony has grown rapidly in the last few years. This rapid expansion and potential underlies the importance of an enabling and unifying standard. Actually, it appears likely that both IETF SIP (Session Initiation Protocol) [140] with SDP (Session Description Protocol) [141] and the ITU-T recommendation H.323 [142] and [143] will be used for setting up Internet multimedia conferences and telephone calls. While the two protocols are comparable in their features, SIP provides a higher flexibility to add new features; a relatively easier implementation; and a better integration with other IP related protocols.

In the other hand, the recent ISO MPEG-4 standards [22] [23] [24], as explained, target a broad range of low-bit rates multimedia applications: from classical streaming video and TV broadcasting to highly interactive applications with dynamic audio-visual scene customization (e.g. e-learning, videoconferencing). In order to reach this objective, advanced coding and formatting tools have been specified in the different parts of the standard (i.e. Audio, Visual, and Systems), which can be configured according to profiles and levels to meet various application requirements. A core component of the MPEG-4 multimedia framework is the "Delivery Multimedia Integration Framework". DMIF offers content location independent procedures for establishing and controlling MPEG-4 sessions and access to transport channels such as RTP/UDP/IP.

We consider an IP videoconference service involving numerous heterogeneous terminals like illustrated in Figure 6-1. IP video-conferencing can serve as a typical multimedia service for testing SIP, H.323 and MPEG-4 DMIF call control signaling interworking.

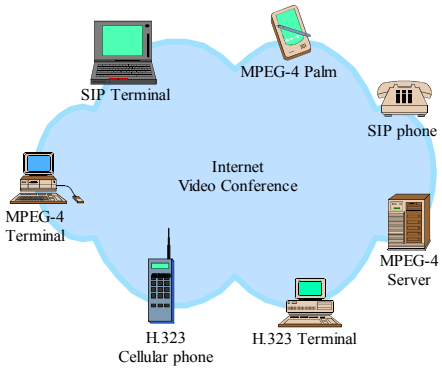


Figure 6-1: Video conferencing between heterogeneous IP terminals

The specific motivation underlying this multimedia service is to help define a framework to enable interoperability between heterogynous terminals which use different standard and to allow each technology to talk transparently to others without knowing what utilized protocol in advance for session establishment.

A videoconferencing session may consist of multiple video or multiple audio streams, addressing multiple codecs with multi-rate requirements.

To permit all connected user to joint conference, one solution consists to use a common protocol for session establishment, which is not obvious. Another solution consists to use new components knows as Media Gateways which essentially allow signaling translation between heterogynous signaling protocols such as SIP, H.323 and DMIF.

The term media gateway is not new. It was proposed first for interconnecting telephone circuits and data packets carried over the Internet or over other IP networks.

In order to achieve universal IP connectivity and seamless IP multimedia service, the interworking between MPEG-4 and SIP terminals is required. Several points of heterogeneity should be addressed for permitting IP multimedia interworking service:

(1) Video and audio formats: digital audio formats will be characterized by many factors such as sampling rates or compression algorithms. Video formats may differ in spatial and temporal resolutions, color depth or compression schemes. This format adaptation issue is addressed by multimedia gateways.

(2) Synchronizing and system encoding: each elementary (video, audio, data) stream should be merged to form a single bit-stream for transmission, and they should be synchronized. In Internet, the Real-time Transport Protocol provides temporal and encapsulation functions.

(3) Application control: users should be able to control the received stream to simulate interactive VCR-like function (e.g. play, pause, fast rewinding, etc.). IETF Real-Time Streaming Protocol (RTSP) [145], DAVIC-S3 [174] and ISO MPEG DSM-CC [175] are examples of signalling protocols developed for that purpose and require interoperability.

(4) Connection control/QoS control: in next generation Internet, QoS could be negotiated by way of different signaling (e.g. RSVP, COPS, MPLS LDP, etc.), while some domains will not provide any QoS guarantee and still perform in best effort, others domains may use different protocols. Thus, there should be a function that translates QoS requests from one domain to another. This can be

performed by many ways like SLA (Service Level Agreement) negotiation between two domains using COPS Protocol for example.

(5) Call/session control: different IP network domains may adopt different method for reserving resources and maintaining session information. There should be a way of managing two independent sessions to form a composite multimedia session (e.g. a SIP compliant phone call and an MPEG-4 DMIF compliant video call).

This later point is addressed in this chapter, in particular it describe service heterogeneity (i.e. call and session control). We depict the design and implementation of an experimental system for IP videoconferencing interworking between ISO MPEG-4 DMIF and IETF SIP signaling protocols. We recall, that this interworking signaling gateway is composed of two core subsystems for supporting two-ways delivery of audio-video streams from a DMIF domain to a SIP domain (i.e. DMIF2SIP subsystem), and from a SIP domain to a DMIF domain (i.e. SIP2DMIF subsystem). The design architecture of the interworking signaling gateway is presented in Section 6.2.

6.1 Related Work

6.1.1 IETF SIP: Session Initiation Protocol

Session Initiation Protocol (SIP) [140] is an application-layer control and signaling protocol for creating, modifying and terminating sessions with one or more participants. These sessions include IP multimedia conferences, IP telephone calls and multimedia distribution. SIP has been approved in early 1999 as an official standard by the IETF for signaling communications services on the Internet. SIP can be used to initiate sessions as well as to invite members to sessions. The SIP architecture includes the following protocols:

- **RTP** (Real-Time Transport Protocol) for transporting real time audio, video and data [110],
- **RTSP** (Real-Time Streaming Protocol) for setting up and controlling on-demand media [145],
- **MGCP** (Media Gateway Control Protocol) and Megaco for controlling media gateways [146],
- **SDP** (Session Description Protocol) for describing multimedia sessions [141],
- **SAP** (Session Announcement Protocol) for announcing multicast session [147],
- **TRIP** (Telephony Routing over IP) for locating the best gateway between the Internet and the **PSTN** (Public Switched Telephone Network) [148],
- Suite of resources management and multicast address allocation protocols.

6.1.1.1 SIP Components

SIP is composed essentially of four logical entities: user agent, registrar, proxy server and redirect server. Figure 6-2 depicts the interaction between these components

1. **UAC** (User Agent Client): it is the caller application that initiates and sends SIP requests.
2. **UAS** (User Agent Server): it receives and responds to SIP requests on behalf of clients, accepts, redirects or refuses calls.
3. **SIP Terminal**: it supports real-time, two-way communication with another SIP entity with both signaling and media handling. SIP terminal contains UA.
4. **Proxy**: it contacts one or more clients or next-hop servers and passes the call requests further. It contains UAC and UAS.
5. **Redirect Server**: it accepts SIP requests, maps the address into zero or more new IP addresses and returns those addresses to the client.
6. **Location Server**: it provides information about a caller's possible locations to redirect servers and proxy servers.

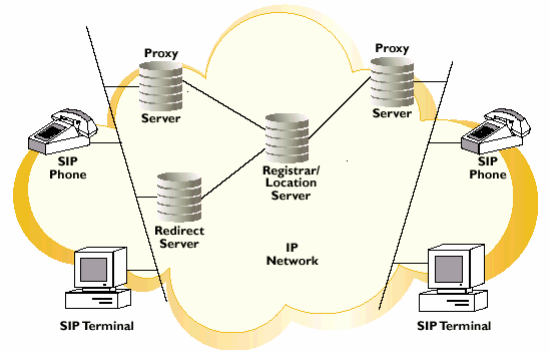


Figure 6-2: SIP network architecture

6.1.1.2 IETF SDP : Session Description Protocol

Session Description Protocol (SDP) [141] is intended for describing multimedia sessions for the purposes of session announcement, session invitation, and other forms of multimedia session initiation. The purpose of SDP is to convey information about media streams in multimedia sessions to allow the recipients of a session description to participate in the session. Therefore terminal capability is described by SDP.

6.1.1.3 IETF SAP: Session Announcement Protocol

Session Announcement Protocol (SAP) [147] is used to assist the advertisement of multicast multimedia conferences and other multicast sessions, and to communicate the relevant session setup information to prospective participants. SAP announces a multicast session by multicasting packets periodically a well-known multicast group. The packets contain a description of the session using SDP.

6.1.1.4 SIP Communication Model

SIP is based on the request-response paradigm. To initiate a session, the UAC sends a request (called an INVITE) addressed to the person the caller want to talk to. The addresses are similar to mailto URL (sip:user@server). The message is not send directly to the called party but rather to the proxy. Then, the proxy delivers the message to the called party. The called party sends a response, accepting or rejecting the invitation, which is forwarded back to the caller in reverse order like undelivered mail in Internet.

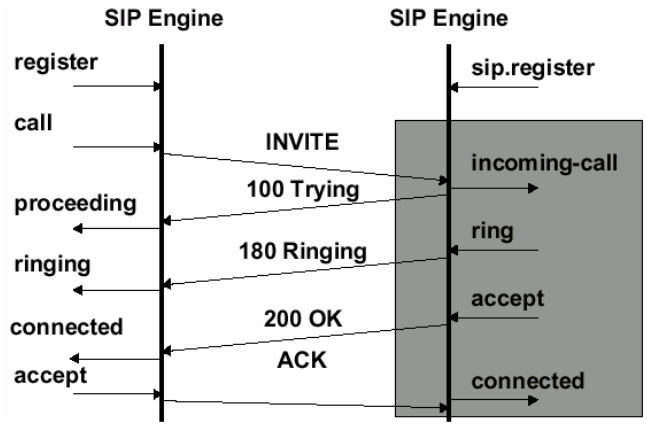


Figure 6-3: A SIP call-setup signaling

Bellow is an example of SIP scenario:

1 User A (a@caller.com) calls User B (b@example.com). User A send INVITE message to the proxy of example.com (INVITE sip: b@example.com), then the proxy of example.com tell User A, that it is trying to connect to b@example.com

2 The proxy locates in which terminal (PC) the User B is logged currently. This task is performed by a REGISTER message send by User B when he turned on his SIP user agent client. This would allow that User B is actually at foo.example.com (sip: b@foo.example.com). The bindings registered are periodically refreshed.

3 The proxy send INVITE message to UAC of User B (INVITE sip:b@foo.example.com), this later replies the proxy with ringing informational message. The proxy informs the caller that is ringing at User B.

4 When B accepts the invitation an acknowledgement message (ACK) is sent, and the session is established, Media can then flow between User A and User B. Others ways, User B can rejects the invitation. Figure 6-3 shows an example of a basic configuration between 2 UAs using SIP protocol.

6.1.2 ISO/IEC 14496-6 (MPEG-4 DMIF)

MPEG-4 DMIF is the control plane of MPEG-4 Delivery layer that allows applications to transparently access and view multimedia streams whether the source of the stream is located on an interactive remote end-system, the stream is available on broadcast media or is located on stored media [144].

As explained, the *Delivery layer* is media unaware but delivery technology aware. It provides transparent access to the delivery of content irrespective of the technologies used (IP, ATM...). The boundary between the *Sync Layer* and the *Delivery Layer* is referred to *DMIF Application Interface* (DAI). It offers content location independent procedures for establishing MPEG-4 sessions and access to transport channels. Also it provides default DMIF signaling protocol which corresponding to the *DMIF Network Interface* (DNI).

6.1.2.1 MPEG-4 DMIF Architecture

DMIF framework covers three major technologies: (1) interactive network technology, (2) broadcast technology and (3) the disk technology. An application accesses data through the DAI irrespectively whether such data comes from a broadcast source, from local storage or from remote server. In all scenarios the Local Application only interacts through DAI primitives. Figure 6-4 clarifies the DMIF aim.

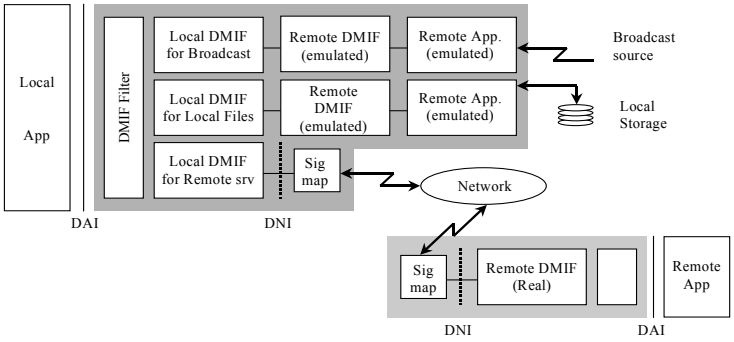


Figure 6-4: MPEG-4 DMIF architecture

6.1.2.2 DMIF Layer

MPEG-4 DMIF allows each delivery technology to be used for its unique characteristics in a way transparent to application developers. Figure 6-5 shows the composition of DMIF stack in conjunction with an IP-based underlying networking environment.

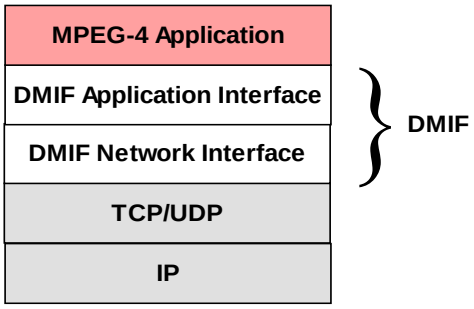


Figure 6-5: DMIF stack protocols with IP networks

DMIF contains functionality needed to establish sessions and connections between an application running at server side and an application running at client side over transport networks.

In general, DMIF provides the following functionalities:

- It assign a value (network session Identifier) to one session. This identifier encodes a unique session instance for network application and for the management of real time, QoS sensitive streams.
- It allows interoperation between the client and the server.
- It hides the delivery technology details from the DMIF User.
- It ensures interoperability between end-systems (in the control plane).

6.1.2.3 DAI: DMIF Application Interface

The DMIF Application Interface (DAI) allows the development of applications to support delivery technologies, regardless of network topology.

The DAI defines the functions offered by the DMIF layer, and comprised the following classes of primitives:

- Service primitives, which deal with the Control Plane, and allow the management of service session (attach and detach).
- Channel primitives, which deal with the control Plane, and allow the management of channels (add and delete).
- Data primitives, which deal with the User Plane, and serve the purpose of transferring data through channels.

6.1.2.4 DNI: DMIF Network Interface

The DMIF Network Interface (DNI) abstracts the signaling between DMIF peers irrespectively of the supported delivery technologies. The parameters conveyed through the DNI are then normatively mapped onto network dependent native signaling when possible otherwise they are carried opaque to the native signaling.

The DMIF Network Interface includes the following classes of primitives:

- Session primitives, which allow the management of sessions (setup and release)
- Service primitives, which allow the management of services (attach and detach)
- Transmux primitives, which allow the management of a transmux (setup, release and config)
- Channel primitives, which allow the management of channels (add and delete)

6.1.2.5 DMIF Communication Model

To hide the delivery technologies to the end-user, DMIF uses a set of primitives (DAI primitives and DNI primitives). Figure 6-6 shows the workflow of message and primitive exchanged between two MPEG-4 DMIF terminals for session establishment.

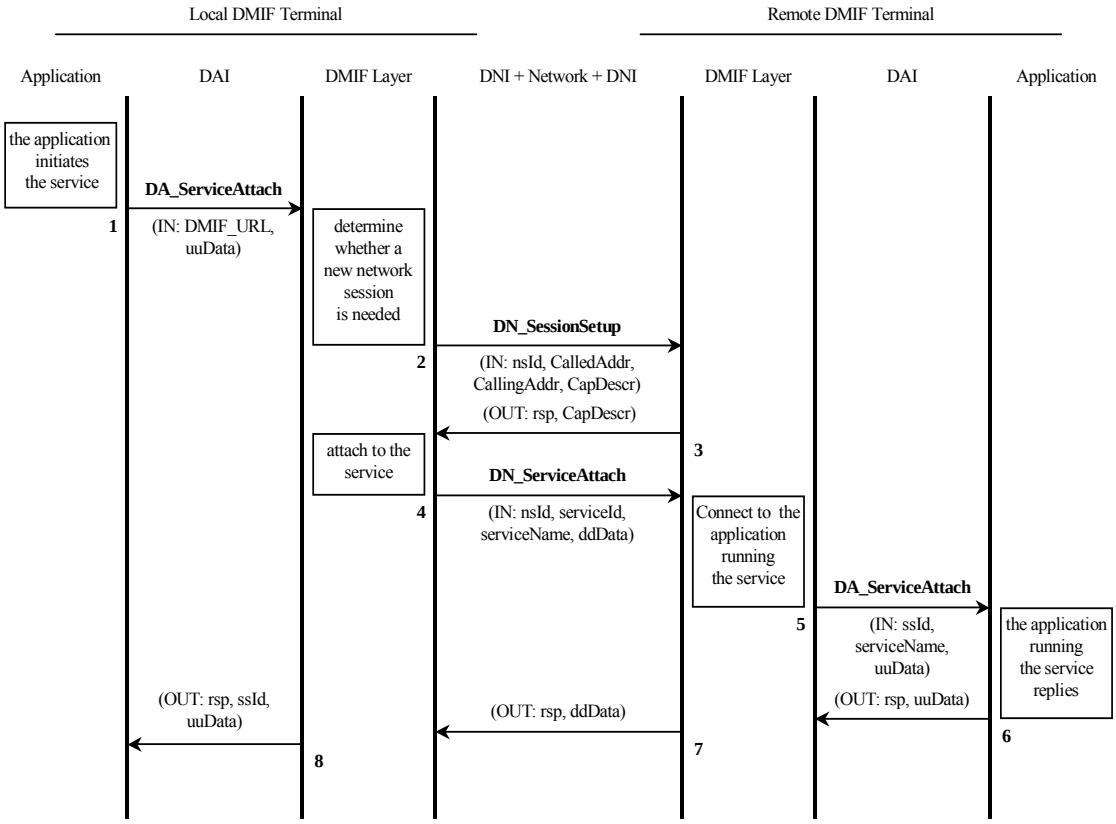


Figure 6-6: A DMIF interactive service initiation

6.2 SIP/MPEG-4 DMIF Interworking Signaling Gateway Architecture

In this section, we propose a functional architecture of logical entity, that performs interworking between MPEG-4 DMIF and SIP. This entity is called **DMIF-SIP IWF** (DMIF-SIP Interworking Function). The Figure 6-7 illustrates our purpose. The DMIF-SIP IWF is a signaling gateway composed of two sides: SIP side and DMIF side performing two-ways signaling translation between SIP and DMIF.

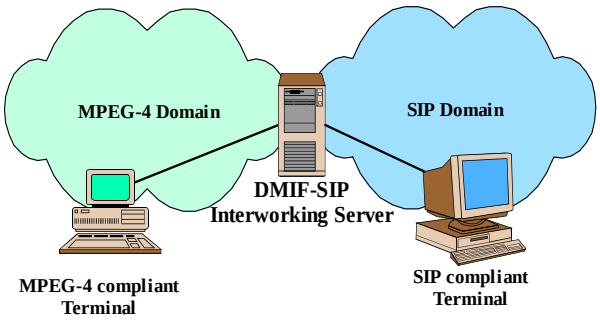


Figure 6-7: Interworking between SIP and DMIF

6.2.1 SIP2DMIF Subsystem

The SIP2DMIF subsystem of DMIF-SIP IWF is part of the IWF that terminates and originates SIP signaling from and to SIP network respectively. We call this function **SIP2DMIF** (SIP to DMIF) signaling. The SIP2DMIF signaling allows a SIP UAC to call MPEG-4 DMIF terminal. SIP UAC talks to DMIF-SIP IWF with SIP specification. When SIP2DMIF IWF receives an INVITE message from SIP UAC, it sends DMIF Signaling Message (DS_Message) to DNI of MPEG-4 Server. When the session and the service are done, the SIP2DMIF IWF sends "200 OK message" back to SIP UAC. An Acknowledgment Message sends by SIP UAC confirms the connection of SIP UAC to the MPEG-4 Server. Figure 6-8 illustrates the messages exchanged between SIP UAC, DMIF-SIP IWF and DNI of MPEG-4 terminal.

The Steps of the call between User A (SIP Terminal) and User B (MPEG-4 DMIF Terminal) is described in what bellow:

Step 1: SIP User A sends an INVITE request to User B. the INVITE request is an invitation to User B to participate to videoconferencing. The INVITE request contains:

a). The identifier (mail address, phone number) of User B is inserted in the Request-URI field in the form of SIP URL

b). SIP User A identifier is recognized as the call session initiator in the From field.

c). A unique numeric identifier is assigned to the call and is inserted in the Call-ID field.

d). The transaction number within a single call leg is identified in the CSeq filed.

e). The Media capability User A is ready to receive is specified via SDP.

Step 2: SIP2DMIF IWF receives INVITE request from User A. it translate the identifier of User B in form of DMIF URL which was obtained when the MPEG-4 DMIF Terminal was turned-on. We suppose that the User B addressee match the DMIF URL. The mapping between SIP addresses and DMIF URL is described later. SIP2DMIF IWF passes DS_SessionSetupRequest message to the remote DMIF peer in order to activate a network session. This message contains SIP User A capability of handling media. The MPEG-4 Capability Descriptor syntax is similar to MPEG-2 syntax. It is defined in [175].

Step 3: DMIF2SIP IWF sends a "100 TRYING" message to SIP User A.

Step 4: DMIF-SIP IWF receives a DS_SessionSetupConfirm message. Both peers (SIP UAC and DMIF Terminal) have knowledge of the others. The received message contains also a common set of User B compatibility in preferred order of choice.

Step 5: SIP2DMIF IWF sends DS_ServiceAttachRequest which contains DMIF URL

Step 6: DNI Layer Informs the MPEG-4 Terminal that User A would to establish a video conferencing with User B.

Step 7: DMIF-SIP IWF receives a DS_ServiceAttachConfirm message indicating that User B is apt to handling videoconferencing.

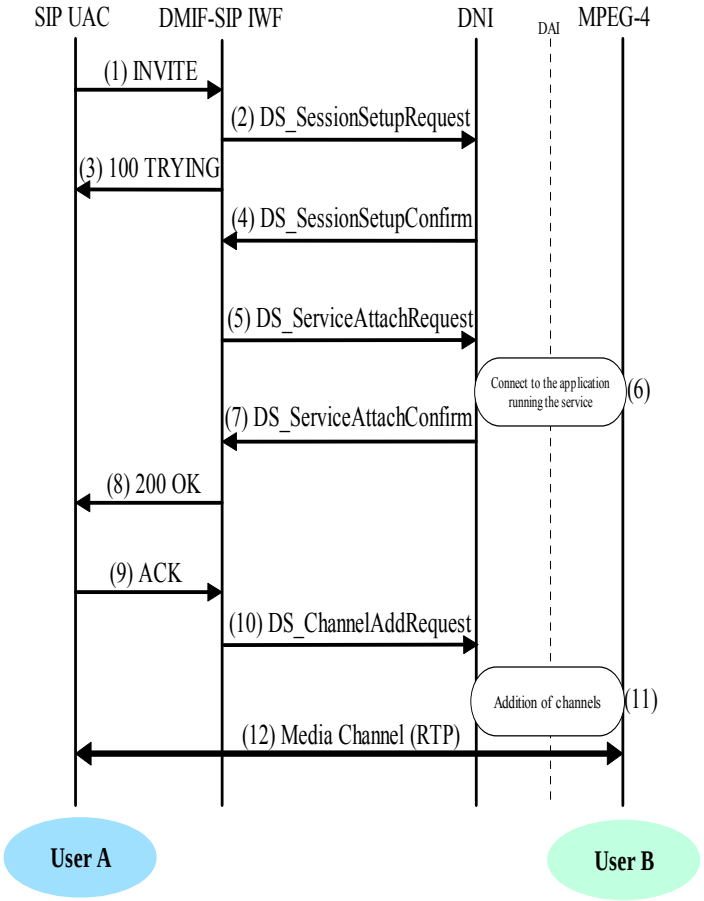


Figure 6-8: SIP2DMIF interworking

Step 8: SIP2DMIF IWF sends a 200 OK message back to SIP User A. The 200 OK Response notifies SIP User A that the connection has been made. This message contains also the intersection of the two terminals capabilities. If there is no support media between the two terminals a 400 Bad Request response with 304 Warning header field is sent.

Step 9: SIP User A sends an ACK to DMIF-SIP IWF

Step 10: By receiving the ACK Media channel must be done. For this fact, a DS_ChannelAddRequest is sent to DNI of MPEG-4 Terminal.

Step 11: The MPEG-4 Terminal notifies the creation of the requested channels.

Step 12: When RTP channel is opened between SIP User A and DMIF User B, media can flows and videoconferencing can begin.

6.2.2 DMIF2SIP Subsystem

The DMIF2SIP subsystem of the DMIF-SIP IWF is the part of the IWF that terminates and originates DMIF signaling from and to DMIF network respectively. We call this function **DMIF2SIP** (DMIF to SIP) signaling. The DMIF2SIP signaling allows a MPEG-4 DMIF terminal to call SIP UAC. Processing steps for establishing connection between an MPEG-4 DMIF terminal and a SIP terminal are illustrated in Figure 6-9 and explained in the following:

Step 1: The MPEG-4 Terminal passes a DA_ServiceAttach() primitive indicating the User B address (email address, phone number, etc.). DNI layer assigns a local session to this request.

Step 2: DNI Layer sends a DS_SessionSetupRequest to DMIF2SIP IWF to establish a network session with SIP terminal.

Step 3: Upon receiving the Session establishment request, the DMIF2SIP IWF sends an INVITE message to SIP Terminal to participate in videoconferencing. The INVITE request contains:

a). The address of User B is inserted in the Request-URI field in the form of SIP URL

b). User A address is recognized as the call session initiator in the From field.

c). DMIF2SIP checks whether its own address is contained in the Via field (to prevent loops), otherwise, it copies it own address in Via filed.

d). DMIF2SIP IWF create a unique numeric identifier which is assigned to the call and is inserted in the Call-ID field.

e). The transaction number within a single call leg is identified in the CSeq filed.

f). The Media capability User A (DMIF terminal) is ready to receive is transformed to form a SDP message.

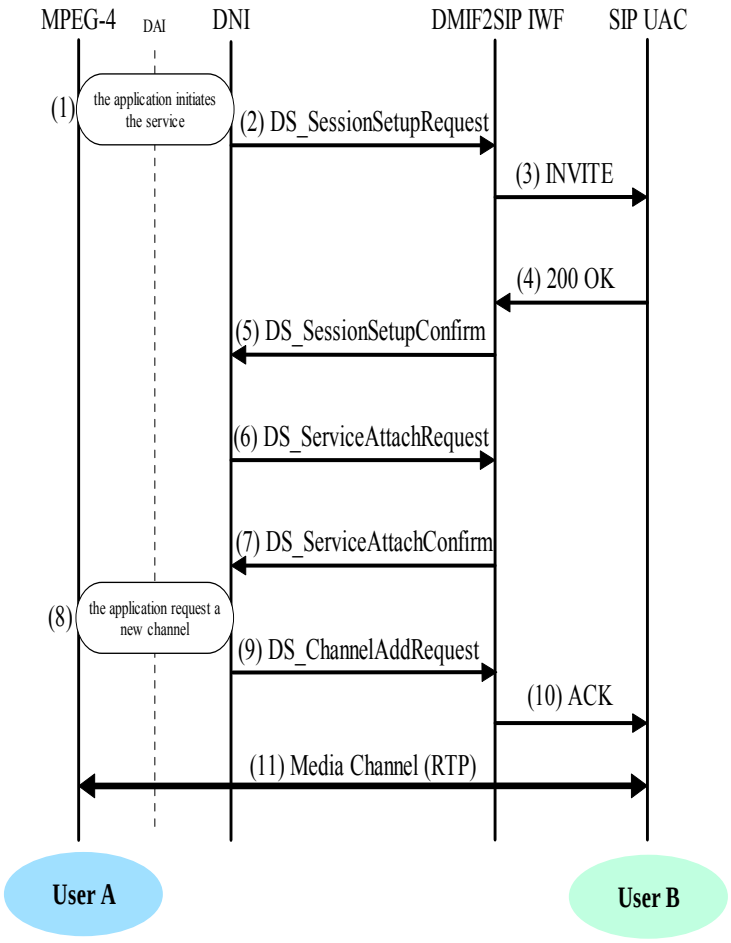


Figure 6-9: DMIF2SIP interworking

Step 4: After sending TRYING and RINGING message, User B (SIP terminal) sends 200 OK Message which notifies DMIF2SIP IWF that the connection has been done. If SIP User B supports the media capability advertised in the INVITE message send by DMIF2SIP IWF, it advertises the intersection of its own and User A's capability in the 200 OK response. If SIP User B does not support the media capability advertised by DMIF2SIP IWF, it sends back a 400 Bad Request response with a 304 Warning h3eader field.

Step 5: Upon receiving 200 OK response, the DMIF2SIP IWF sends a DS_SessionSetupConfirm message back to MPEG-4 DMIF Terminal and specifies the common set of capability advertised by 200 OK response.

Step 6: The MPEG-4 DMIF terminal now is suitable to assign a local significance of the network session and must request a service within the network session. This task is performed by sending DS_ServiceAttachRequest message.

Step 7: DMIF2SIP IWF receives DS_ServiceAttachRequest message which idenfies the services at the DMIF2SIP side.

Step 8: The MPEG-4 DMIF Terminal request the establishment of a new media channel.

Step 9: The DS_ChannelAddRequest message sends by MPEG-4 DMIF Terminal requests the establishment of new media channel.

Step 10: DMIF2SIP IWF sends ACK message to SIP User B and it confirms that the two peers are capable of sending a receiving media.

Step 11: Media channels are now done, media can flows between the MPEG-4 DMIF Terminal and SIP terminal.

6.2.3 SIP/DMIF Terminal Capability Matching

The capability set of a terminal or a user agent refers to the set of algorithms for audio, video and data that it can support. It also conveys information about constraints in the selection of algorithms it may have. For example, due to limited bandwidth, a terminal may indicate that it can use either G.711 without video or G.723.1 with H.261 video [176].

Terminal Capability matching is required to check the ability of two peer and-systems to setup connections between them, and select the possible protocol stack supported. In case of DMIF and SIP, this stage is performing at session setup.

SIP defines SDP as a protocol to describe SIP terminal capability whereas MPEG-4 DMIF uses DSM-CC as a format for capability description.

When SIP UAC initiates the session with DMIF Terminal, it sends an INVITE request to DMIF. The capability descriptor is carried in this INVITE request throw SDP message. When DMIF Terminal initiates the session, it sends a *DS_SessionSetupRequest* to SIP terminal at this stage the capability descriptor is carried in the *comptabilityDescriptor* field part of DS_*SessionSetupRequest* message.

The algorithm to find the common subset of capability descriptor maximal intersection of any two capability sets C1 and C2 is described in [177] and is given in what follows:

1. Set the result C to the empty set.

2. For each pair of capability descriptors (d1, d2), where d1 is from C1 and d2 is from C2, derive the permutations of alternative sets, s1 and s2. For each such permutation, where s1 is from d1 and s2 is from d2, intersect s1 and s2 (written as s=s1 ^ s2) and add s to C.

3. Remove duplicate entries from C.

6.2.4 SIP/DMIF Address Conversion Model

6.2.4.1 MPEG-4 DMIF Address Definition

DMIF address is an URL that allows the DMIF layer at the originating DMIF to parse the network address in order to establish a session with the designated target DMIF and subsequently locate a service entity to enable it. Optionally the DMIF URL can be used to locate the source of an Elementary Stream in MPEG-4. The DMIF URL follows the generic syntax for new URL schemes defined in RFC1738 [182].

The DMIF URL on IP network consists of the following:

xdmif://<user>:<password>@<target dmif>:<dmif port>/<service-entity-path> or <stream-source-path>

6.2.4.2 SIP Address Format

A SIP address can be either a SIP URL or any URI. The BNF of a SIP address is given below for reference:

SIP-Address = (name-addr | addr-spec)

name-addr = [display-name] "<" addr-spec ">"

addr-spec = SIP-URL

SIP-URL = "sip:" [userinfo "@"] hostport url parameters [headers]

userinfo = user [":" password]

hostport = host [":" port]

host = hostname | IPv4address

url-parameters = *(";" url-parameter)

url-parameter = user-param | ...

6.2.4.3 Converting SIP Address to DMIF URL

SIP address can be converted to DMIF URL facially. SIP-URL is copied verbatim to DMIF URL without any additional or supplement information.

6.3 Implementation Issues

Our implementation consists of three applications: an MPEG-4 DMIF Terminal, a DMIF-SIP Interworking Signaling Gateway, and a SIP terminal (i.e. UAC) connected across an IP Network (see Figure 6-10). All the components are developed using Java 2 language.

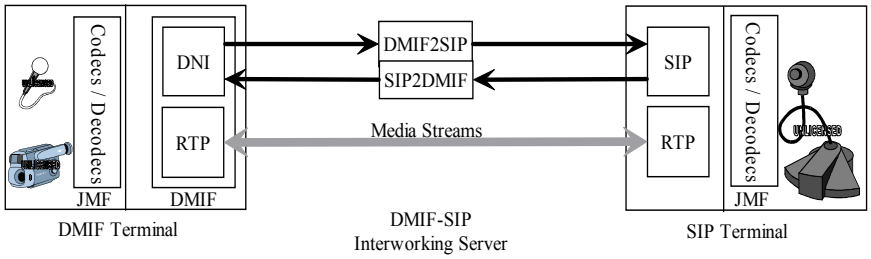


Figure 6-10: A heterogeneous IP videoconferencing service implementation

6.3.1 DMIF-SIP Interworking Signaling Gateway

This signaling gateway is composed of two components SIP2DMIF subsystem and DMIF2SIP subsystem. The implementation is multi-threaded to serve several clients at the same time. The two components are nearly similar.

SIP2DMIF IWF must keep track of all connected client at any time. For this purpose, it maintains a hash table composed of two entries. The entries of the table identify an end-to-end session between SIP UAC and DMIF Terminal. The first entry is composed of the object <SIP_@IP_Address, SIP_Port, IW_@IP_Server, IW_Server_Port> quadruplet (the first part of the communication). The second entry is composed of the object <IW_@IP_Server, IW_Server_Port, DMIF_@IP_Terminal, DMIF_PORT> quadruplet.

The server uses TCP or UDP for both DMIF and SIP signaling. Figure 6-11 gives a screen snapshot of the Interworking Gateway.

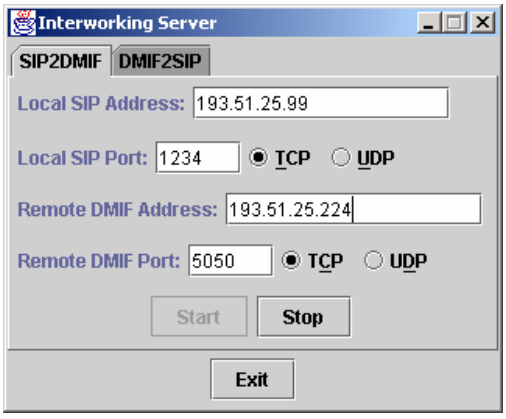


Figure 6-11: The Interworking server snapshot with SIP / DMIF subsystems

6.3.2 MPEG-4 DMIF Terminal

When DMIF operates with IP networks, it shall use a DMIF signaling channel and the default syntax for all primitives (DS_xxx message). Once DMIF daemon started, it listens to a well known

TCP/UDP port, our implementation uses TCP protocol because it is more suitable for signaling. Whenever a new channel is requested, a new TCP, UDP socket is created. In our case, we use RTP socket based on UDP for media transporting.

As example, the session establishment functionality is obtained through the setup of a DMIF Signaling channel, and the exchange of DS_SessionSetup messages. This procedure works as follows in case when DMIF initiates the communication:

1. The target peer is supposed to listen on a well known TCP port number (DMIF_PORT).
2. The originating peer (MPEG-4 DMIF Terminal) creates a TCP socket and connects to the target peer, using the DMIF_PORT port number and TCP protocol.
3. The target peer accepts the connection: this connection carries the DMIF signaling channel.
4. The originating peer sends a DS_SessionSetupRequest message on the just established connection.
5. The target peer replies with a DS_SessionSetupConfirm message.

More details on DMIF implementation in Java can be found in [178].

The MPEG-4 terminal is based on **JMF 2.1 API** (The Java Media Framework API)[1] [179]. JMF API enables audio, video and other time-based media to be added to Java applications and applets. This package, can capture, playback, stream and transcode multiple media formats.

The JMF 2.1 reference implementation can receive and transmit many RTP formats such as (G.711, GSM mono, G.723, MPEG Layer I, II, III, H.261, H.263, etc.). Other formats can be added via plug-ins.

6.3.3 SIP Terminal

We've implemented a simple SIP terminal (see Figure 6-12) which communicates using SIP specification (INVITE, TRYING, RINGING, ACK, CANCEL, BYE) with other terminals.

[1] The JMF 2.0 API was developed by Sun Microsystems, Inc. and IBM. The JMF 1.0 API was developed by Sun Microsystems, Inc., Silicon Graphics Inc., and Intel Corporation.

Figure 6-12: SIP user agent call parameters setup snapshot

Our SIP terminal uses also JMF API to add enable audio and Video. Our SIP implementation uses TCP or UDP protocol for signaling. The Figure 6-12 shows a brief snapshot of SIP Terminal when inviting another user to videoconferencing. We've used also other SIP terminals [180] and [181] to test our Interworking Server and to verify compatibility issues.

6.4 Conclusion

Diversity and heterogeneity of multimedia terminals and services characterize today IP networking environment. Consequently, interworking becomes a critical issue for resolving the differences in these elements and enabling seamless provision of audiovisual applications across networks. Interworking is a way of making different communicating systems cooperate to perform a particular service. Thus, this chapter emphasizes technical issues on call control and session establishment interworking between an ISO DMIF-compliant terminal and IETF SIP-compliant. We described, in this chapter, the design and implementation of an experimental interworking signaling gateway that performs various translation functions between the two signaling protocols, including session protocol conversion, service gateway conversion, and address translation. Multimedia Internet designers may then transparently combine MPEG-4 multimedia content and IP telephony for advance videoconferencing services such as e-learning, e-commerce or collaborative conferencing. Internet users might well have the impression that it is an integrated multimedia service offered by a single Internet site. This is what we call a "seamless IP videoconferencing service interworking". Current implementation supports both translation modes.

Chapter 7

7 Conclusion and Perspectives

This chapter concludes the dissertation by summarizing our major contributions and suggesting some key directions for future work.

IP networks and services are growing exponentially and are impacting every aspect of modern life. Video and audio applications are expected to become a significant portion of World Wide Web, but nevertheless, supporting multimedia applications in the current IP networks is still in its immaturity. The IETF is proposing two QoS management architectures, namely integrated services and differentiated services, to meet the demand of multimedia applications over IP. Besides these technologies are still under development and not yet widely deployed. IP Diffserv model is gaining more and more interests from the telecommunication and networking industry (i.e. Diffserv/ MPLS based backbones).

Therefore in this dissertation, we have explored methods and solutions that can be used for improving user perceived quality of interactive video streaming applications over IP Diffserv. We leverage the characteristics of MPEG-4 standard and IP differentiated service frameworks, to propose an efficient and adaptive cross-layer video delivery system.

7.1 Summary of Key Result

Figure 7-1 depicts an overall view of the proposed end-to-end video QoS management system architecture with the associated functional blocks.

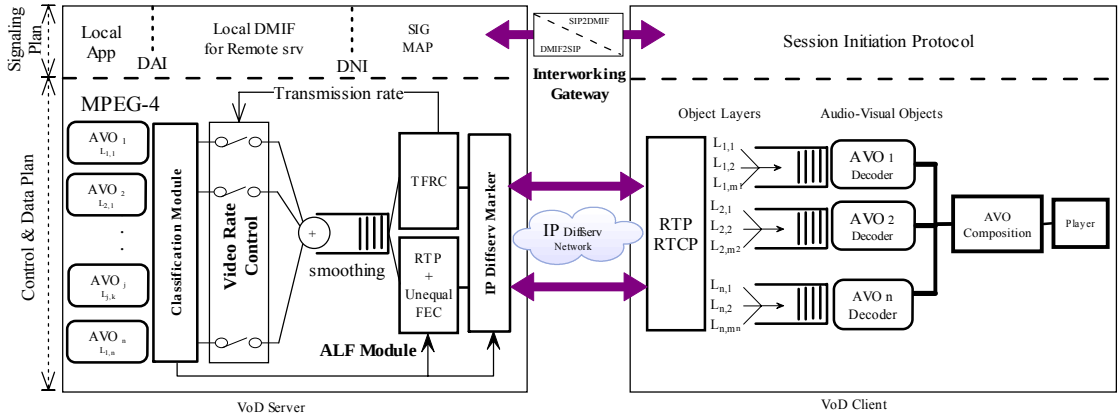


Figure 7-1: General block diagram of our end-to-end architecture

Our cross-layer video streaming system is composed of a video server and a video client communicating using IP Diffserv network. The video server is able to stream audio-visual objects for various heterogeneous clients using RTP protocol. The client decodes and composes the

original MPEG-4 scene. In MPEG-4, each AVO is coded separately so that the decoding process decodes also each AVO separately and then the composition module rebuilds the original scene. We summarize the key results and important components of this system in the following:

- **Content-based video classification component:** that implements the application-level audio-visual object classification model. Since the MPEG-4 scene may be composed of several audio-visual objects (AVO), it is necessary to have a function that allows distinction between important AVO and less important one. This module provides automatic and accurate mapping between MPEG-4 application-level QoS metrics and the underlying QoS-capable network services. Performance evaluation shows a significant enhancing in network QoS metrics for audio-video streams which are marked based on the relative priority score of each AVO.

- **Video Rate Adaptation component:** that implements the fine grained TCP-Friendly video rate adaptation algorithm. It performs video source rate adaptation to deal with varying network resource conditions. Based on end-to-end feedback measurements conveyed by Real Time Transport Control Protocol (RTCP) reports, the video server adapts its sending rate to much the estimated transmission rate and conforms to it. This module adapts a fine adjustment of the video rate by adding/dropping AVOs according to the relative priority score of the AVOs calculated by the Classification module. Performance evaluation shows a constant visual quality for long-time scale.

- **Application level framing component:** that implements the robust and adaptive application level framing protocol with elementary stream multiplexing and unequal error protection. This module uses RTP protocol for preparing audio-video streams to be transported over UDP/IP stack. Performance evaluation shows a significant reduction of the end-to-end transfer delay and the consecutive packet loss of a particular stream. Even if the error protection augments the traffic throughput of the original scene, the gain obtained in term of data recovery is better than with error protection.

- **QoS matching component**. That could be deployed in video streaming servers or on edge routers. It offers packet video marking for IP Differentiated Service. We demonstrate that the interaction and cooperation between application-level (i.e. data classification model) and network-level (i.e. packet marking) mechanisms can improve the efficiency of the overall video delivery system. This module receives input from the classification module, and then it performs dynamic matching between the application-level RPS (Relative Priority Score) of each video packet and the network-level differentiated forwarding mechanism. Performance evaluation shows that applying an intelligent marking allows to better taking into consideration

characteristics and relevance of each stream. As a result, sensitive information will undergo a privileged forwarding processing from the network, and then better QoS. We demonstrate also, that using network state as a feedback mechanism improves considerably the marking strategy.

- **Interworking signaling gateway**. This gateway is a SIP/DMIF Multimedia Interworking Signaling Gateway. It allows SIP and MPEG-4 terminals to use services offered by each other to widely sharing multimedia content between heterogeneous wired and wireless IP terminals. Among the functionalities of this gateway, we cite session protocol conversion, service gateway conversion and address translation.

We believe that the majority of the current video streaming systems over IP are missing at least one of these components. Our main contribution is in combining system-level components (media classification) with transport-level (application level protocol) and network-level components (traffic marking). Combining theses components into a coherent and integrated system architecture lead to a powerful cross-layer video streaming system.

7.2 Open Issues and Future Work

As part of our future work, we plan to pursue our research activity in the following directions:

- **Minimizing the impact of loss**: Loss can be minimized by a real collaboration between the end system (server and client) and network. Packet loss affects the quality of service perceived by the user. So it is necessary to reduce the impact of loss and variation in QoS. At the end system, we have used object prioritization. The traffic can have different levels of priorities, from high to low. The high priority traffic is sent with high protection and with the low drop precedence class and the low priority traffic can be sent over best effort class. It is necessary to a mapping strategy to assure the QoS for each priority and for each class. Using feedback from the network or the end system can ameliorate considerably this strategy. Thus, we can apply high protection only if we detect high loss, and so on.

- **Enhancing video rate control mechanism**: the implemented video rate control mechanism uses TCP-Friendly Rate Control Module. It is necessary to compare this mechanism with other congestion control mechanisms such AIMD and see their impact on the received quality. The quality of the received video depends considerably on the video rate variation. Users want a consistent quality over the time. Hence it is necessary to provide a TCP-Friendly mechanism with low variation in the QoS.

- **Designing protocols for video multicast**: in this thesis, we assumed only a unicast video streaming. We should adapt our mechanisms for multicast video communication. Our video rate control algorithm is designed for unicast communication. It reacts on end-to-end feedback. In case of multicast communication, many users may generate many feedbacks. It is necessary to find schemes that provide a consistent quality for the whole clients.

- **Packet video streaming in wireless ad hoc networks**: One characteristic of the ad hoc networks is that there are many paths between the video source and destination. Thus, a mechanism that takes advantage of these multitude of paths is expected to perform better (i.e., in supporting QoS for real-time traffic) than the classical one-path approach. Moreover, rather than selecting a single path at any time to use for a specific connection, a better scheme would be to always distribute the information among multiple paths, possibly with some correlation between the information on the various paths, so as to protect against failure of some subset of the paths. A possible proposed mechanism may consists of four steps: (1) discovery of multiple paths between the source and the destination nodes and evaluation of the correlation in the paths availability; (2) selection of a subset of the paths into an Active Path Set (APS), based on some "goodness" measures (such as the expected reliability of the path, the capacity of the path, the delay and jitter of the path, etc), and a scheme that allows to evaluate these measures for the network paths; (3) a method of coding and spreading the information among the paths (including matching the paths with the specific requirements of a traffic type); (4) a scheme to monitor the APS paths, estimates their QoS parameters, and updates the APS based on the state of the paths and their correlation.
- **Implementation Issues**: Most of our performance evaluations are done using simulation. We plan to implement the core components in real. This can be done by using an open streaming environment such as MPEG4IP Server developed by the multimedia and mobile networking group of the PRiSM laboratory (University of Versailles). This allows us to perform real world experiment in a local area network as well as on the Internet.

References

[1] Kazaa Media Desktop Home Page, http://www.kazaa.com/.

[2] Gnutella Home Page, http://www.gnutella.com/

[3] iMesh Home Page, http://www.imesh.com/

[4] eDonkey Home Page, http://www.edonkey2000.com/

[5] H. Deshpande, M. Bawa, and H. Garcia-Molina, "Streaming live media over peer-to-peer network.", Technical report, Stanford University, 2001.

[6] V. Padmanabhan and K. Sripanidkulchai, "The case for cooperative networking", In Proc. of 1st International Workshop on Peer-to-Peer Systems (IPTPS '02), Cambridge, MA, USA, March 2002.

[7] V. Padmanabhan, H.Wang, P. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking", In Proc. of NOSSDAV'02, Miami Beach, USA, May 2002.

[8] A. Dutta and H. Schulzrinne, "A Streaming Architecture for Next Generation Internet", In Proc. of ICC'01, Helsinki, Finland, June 2001.

[9] L. Gao and D. Towsley, "Threshold-based multicast for continuous media delivery", IEEE Transactions on Multimedia, 3(4), pp. 405–414, December 2001.

[10] D. Xu, M. Hefeeda, S. Hambrusch and B. Bhargava, "On Peer-to-Peer Media Streaming", IEEE ICDCS 2002, July 2002.

[11] X. Jiang Y. Dong D. Xu B. Bhargava, "Gnustream: A P2p Media Streaming System Prototype", Purdue University, West Lafayette, 2002.

[12] R. Rejaie, A. Ortega, "PALS: Peer to Peer Adaptive Layered Streaming", In Proc. NOSSDAV'03, June 2003.

[13] ITU-T Recommendation G.1010, "End-user multimedia QoS categories", 2001.

[14] ITU-T Recommendation H.261, "Video codec for audiovisual services at p x 64 kbit/s, 1993.

[15] ITU-T Rec. H.262 | ISO/IEC 13818, "Information technology – Generic coding of moving pictures and associated audio information", 1995.

[16] ITU-T Recommendation H.263, "Video Coding for Low Bit Rate Communication", 1996.

[17] ITU-T Recommendation H.263v2 H.263+, "Video Coding for Low Bit Rate Communication", 1998.

[18] ISO-11172 Information Technology, "Coding of moving pictures and associated audio for digital storage media at up to about 1.5Mbit/s."

[19] ISO-14496, "Coding of audio-visual objects", final committee draft, May 1998.

[20] ISO-15938, "Multimedia content description interface (MPEG-7)", December 2000.

[21] ISO-18034, "Digital Audiovisual Framework", 2001.

[22] ISO/IEC 14496-1 "Coding of audio-visual objects, Part 1: Systems", final committee draft, May 1998.

[23] ISO/IEC 14496-2 "Coding of audio-visual objects, Part 2: Visual", final committee draft, May 1998.

[24] ISO/IEC 14496-3 "Coding of audio-visual objects, Part 3: Audio", final committee draft, May 1998.

[25] ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, "Study of Final Committee Draft of Joint Video Specification", February 2003.

[26] P. Amon, G. Bäse, K. Illgner, and J. Pandel, "Efficient Coding of Synchronized H.26L Streams", ITU-T Video Coding Experts Group (VCEG), document VCEG-N35, Santa Barbara, CA, USA, September 2001.

[27] N. Kamaci, Y. Altunbask, "Performance comparison of the emerging H.264 video coding standard with the existing standards", in *IEEE Int. Conf. Multimedia and Expo*, ICME'03, Baltimore, MD, July 2003.

[28] S. Wenger, M.M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer, "RTP payload Format for H.264 Video", Internet Draft, IETF, June 2003, Expires December 2003.

[29] Windows Media Home Page, available at: http://www.microsoft.com/windows/windowsmedia/

[30] Real Video Home Page, available at : http://www.realnetworks.com/solutions/leadership/realvideo.html

[31] K. Shen and E. J. Delp, "Wavelet base rate scalable video compression", IEEE Trans. Circuits Syst. Video Technol., vol. 9, pp. 109–121, 1999.

[32] J. M. Shapiro, "Embedded image coding using zero tree of wavelet coefficients", IEEE Trans. Signal Processing, vol. 41, pp. 3445–3462, 1993.

[33] I. Sodagar, H.-J. Lee, P. Hatrack, and Y.-Q. Zhang, "Scalable wavelet coding for synthetic/natural hybrid images", IEEE Trans. Circuits Syst. Video Technol., vol. 9, pp. 244–254, 1999.

[34] J.-Y. Tham, S. Ranganath, and A. A. Kassim, "Highly scalable wavelet based video codec for very low bit-rate environment", IEEE Trans. Circuits Syst. Video Technol., vol. 16, pp. 12–27, 1998.

[35] S. Wenger, "Video Redundancy Coding in H.263+", Workshop on Audio-Visual Services for Packet Networks, September 1997.

[36] V. Vaishampayan and S. John, "Interframe balanced-multiple description video compression", IEEE Inter Conf. on Image Processing, Oct.1999.

[37] A. Reibman, H. Jafarkhani, Y. Wang, M. Orchard, and R. Puri, "Multiple description coding for video using motion compensated prediction", IEEE Inter. Conf. Image Processing , October 1999.

[38] J. Apostolopoulos, "Error-resilient video compression via multiple state streams", In Proc. International Workshop on Very Low Bitrate VideoCoding (VLBV'99), October 1999.

[39] J. Apostolopoulos and S. Wee, "Unbalanced Multiple Description Video Communication Using Path Diversity", IEEE International Conference on Image Processing, October 2001.

[40] W. Li, "Overview of Fine Granularity Scalability in MPEG-4 video standard," IEEE Trans. Circuits and Systems for Video Technology, vol. 11, no. 3 , pp. 301-317, Mar. 2001.

[41] F. Ling,W. Li, and H.-Q. Sun, "Bitplane coding of DCT coefficients for image and video compression", in Proc. SPIE-VCIP'99, San Jose, CA, Jan. 25–27, 1999, pp. 500–508.

[42] W. Li, "Bitplane coding of DCT coefficients for fine granularity scalability", ISO/IEC JTC1/SC29/WG11, MPEG98/M3989, Oct. 1998.

[43] B. Schuster, "Fine granular scalability with wavelets coding", ISO/IEC JTC1/SC29/WG11, MPEG98/M4021, Oct. 1998.

[44] Y.-W. Chen, H. Radha, and R. A. Cohen, "Results of experiment of fine granular scalability with wavelet encoding of residuals,", SO/IEC JTC1/SC29/WG11, MPEG98/M3988, Oct. 1998.

[45] J. Liang, J. Yu, Y.Wang, M. Srinath, and M.-H. Zhou, "Fine granularity scalable video coding using combination of MPEG4 video objects and still texture objects,", ISO/IEC JTC1/SC29/WG11, MPEG98/M4025, Oct. 1998.

[46] S. C. S. Cheung and A. Zakhor, "Matching pursuit coding for fine granular video scalability", ISO/IEC JTC1/SC29/WG11, MPEG98/M3991, Oct. 1998.

[47] M. Benetiere and C. Dufour, "Matching pursuits residual coding for video fine granular scalability", ISO/IEC JTC1/SC29/WG11, MPEG98/M4008, Oct. 1998.

[48] J. Macnicol, M. Frater, and J. Arnold, "Results on fine granularity scalability", Melbourne, Australia, ISO/IEC JTC1/SC29/WG11, MPEG99/m5122, Oct. 1999.

[49] F. Wu, S. Li and Y.-Q. Zhang, "A framework for efficient progressive fine granularity scalable video coding," IEEE Trans. Circuits and Systems for Video Technology, special issue on streaming video, Vol. 11, no 3, 332-344, 2001.

[50] X. Sun, F. Wu, S. Li, W. Gao, and Y,-Q. Zhang, "Macroblock-based progressive fine granularity scalable video coding", ICME2001, Tokyo, Aug. 22-25, 2001.

[51] S. Li, F. Wu, and Y. Q. Zhang, "Study of a new approach to improve FGS video coding efficiency", ISO/IEC JTC1/SC29/WG11, MPEG98/M5583, Dec. 1999.

[52] J. Hartung, A. Jacquin, J. Pawlyk, and K. L. Shipley, "A real-time scalable software video codec for collaborative applications over packed networks," in Proc ACM Multimedia'98, Bristol, U.K., Sept. 1998, pp. 419–426.

[53] P. Amon, K. Illgner, and J. Pandel, "SNR Scalable Layered Video Coding", International Packet Video Workshop 2002, Pittsburgh, PA, USA, April 2002.

[54] K. Illgner, P. Amon, J. Pandel, and M. Wagner, "Scalable Video Coding and Resilient Transmission over IP", 2002 Tyrrhenian International Workshop on Digital Communications (IWDC 2002), Capri, Italy, September 2002.

[55] S.E. Han and B. Girod, "Robust and Efficient Scalable Video Coding with Leaky Prediction", In International Conference on Image Processing (ICIP 2002), Rochester, NY, USA, September 2002.

[56] Y. He, R. Yan, F. Wu, and S. Li, "H.26L-based Fine Granularity Scalable Video Coding", ITU-T Video Coding Experts Group (VCEG), document VCEG-O60, Pattaya, Thailand, Decembe 2001.

[57] R. Yan, F. Wu, S. Li, and Y.Q. Zhang, "Macroblock-based Progressive Fine Granularity Spatial Scalability (mb-PFGSS)", ISO/IEC JTC1/SC29/WG11, MPEG2001/M7112, March 2001.

[58] S. McCanne and V. Jacobson, "Receiver-driven layered multicast", in Proc. ACM SIGCOMM'96, Stanford, CA, pp. 117–130, Aug. 1996.

[59] P. Amon, and J. Pandel "Evaluation of Adaptive and Reliable Video Transmission Technologies", Packet Video'03, France, April 2003.

[60] X. Sun, S. Li, F. Wu, G. Shen, and W. Gao, "Efficient and flexible drift-free video bitstream switching at predictive frames", ICME 2002.

[61] X. Sun, F. Wu, S. Li, W. Gao, and Y. Q. Zhang, "Seamless Switching of Scalable Video Bitstreams for Efficient Streaming", IEEE International Symposium on Circuits and Systems, ISCAS 2002, Scottsdale, Arizona, USA, 26-29 May, 2002.

[62] Y.K. Chou, L.C. Jian, and C. Wen Lin, "MPEG-4 video streaming with drift-compensated bit-stream switching," in Proc. IEEE Pacific-Rim Conf. Multimedia, pp. 847-855, Dec. 2002.

[63] Real Network, "RealSystem G2: Management and Control of Streaming Media Over Corporate Networks RealVideo", available at: http://docs.real.com/docs/devzone/g2ctrlandmgmt.pdf, Mar. 1999.

[64] S. Wee, J. Apostolopoulos and N. Feamster, "Field-to-Frame Transcoding with Temporal and Spatial Downsampling," IEEE International Conference on Image Processing, October 1999.

[65] E. Amir, S. McCanne, and H. Zhang "An Application Level Video Gateway", In Proc. ACM Multimedia 1995.

[66] E. Amir, S. McCanne, and R. Katz "An Active Service Framework and its application to real-time multime.ia transcoding", in SIGCOMM, symposium on communications architectures and protocols, Septembre 1998.

[67] V. Jacobson, "Congestion Avoidance and Control", ACM SIGCOMM, August 1988.

[68] M. Mahdavi, and S.Floyd, "TCP-Friendly Unicast Rate-Based Flow Control", Technical note sent to the end2end-interest mailing list, January 8, 1997.

[69] S.Floyd, M. Handley, J. Padhye, and J. Widmer "Equation-based congestion control for unicast applications", In Proc. of ACM SIGCOMM, pp. 43–56, 2000.

[70] M. Mathis et al., "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm," ACM Computer Communications Review, July 1997.

[71] J. Padhye et al., "Modeling TCP Reno Performance: A Simple Model and its Empirical Validation," IEEE/ACM Trans. Networking, April 2000.

[72] S. McCanne "Scalable compression and transmission over Internet multicast video", Ph.D thesis University of California, Berkeley. December 1996.

[73] T.V.Lakshman, P.P.Mishra, and K.K.Ramakrishnan "Transporting compressed video over ATM networks with explicit rate feedback control," in Proc. IEEE Infocom 97, pp. 38–47, April 1997.

[74] N.G.Duffield, K.K.Ramakrishnan, and A.R. Reibman "SAVE: An Algorithm for Smoothed Adaptive Video Over Explicit Rate Networks" IEEE/ACM transactions on networking, vol. 6, no. 6, December 1998.

[75] N.Wakamiya, M.Miyabayashi, M.Murata, and H.Miyahara "MPEG-4 Video Transfer with TCP-friendly Rate Control" in IFIP/IEEE MMNS 2001, pp 29-42 October 2001.

[76] T. Kim and M. H. Ammar, "Optimal quality adaptation for MPEG-4 Fine-Grained Scalable video", In IEEE INFOCOM 2003.

[77] D.Sisalem, and H.Schulzrinne "The Loss-Delay Adjustment Algorithm: A TCP-friendly Adaptation Scheme, Network and Operating System Support for Digital Audio and Video" (NOSSDAV), Cambridge, UK, 8-10, July 1998.

[78] R. Rejaie, M. Handley, and D. Estrin, "RAP An End-to-End Congestion Control Mechanism for Realtime Streams in the Internet". In Proc. IEEE Infocom '99, New York, NY, March 1999

[79] D. Bansal and H. Balakrishnan, "Binomial Congestion Control Algorithms", In Proc. IEEE Infocom '01, Anchorage, AK, April 2001.

[80] R. Rejaie, M.Handley, and D. Estrin "Layered quality adaptation for Internet video streaming" IEEE Journal of selected areas in communications, vol. 18, No. 12, December 2000.

[81] R. Rejaie, M. Handley, and D. Estrin, "Quality Adaptation for Congestion Controlled Video Playback over the Internet", In Proc.ACM Sigcomm '99, Cambridge, MA, September 1999.

[82] S. jin, L. Guo, I. Matta, A. Bestavros, "A Spectrum of TCP-Friendly Window-Based Congestion Control Algorithms", in IEEE/ACM Transaction on Networking, Vol. 11, No.3, pp.341-355, June 2003.

[83] Q.Zhang, W. Zhu, Y. Zhang, "Resource Allocation for Multimedia Streaming over the Internet", Multimedia, Vol 3, No. 3, pp. 339, September 2001.

[84] M.Handley, S.Floyd, J.Padhye, and J.Widmer "RFC 3448, TCP Friendly Rate Control (TFRC): Protocol Specification" Request for Comments, IETF, Jan. 2003.

[85] B. Wang, S. Sen, M. Adler, and D. Towsley, "Proxybased distribution of streaming video over unicast/multicast connections," Technical Report UMASS TR-2001-05, University of Massachusetts, Amherst, 2001.

[86] S.-H. G. Chan and F. A. Tobagi, "Caching schemes for distributed video services," in Proceedings of the IEEE International Conference on Communications (IEEE ICC), Vancouver, Canada, June 1999.

[87] J. Chuang, "Distributed network storage service with quality-of-service guarantees," Journal of Network and Computer Applications, 2000.

[88] J. Kangasharju, F. Hartanto, M. Reisslein, and K. W. Ross, "Distributing layered encoded video through caches," in Proceedings of the Conference on Computer Communications (IEEE Infocom), Anchorage, Alaska, Apr. 2001.

[89] Z.L. Zhang, Y. Wang, D. H. C. Du, and D. Su, "Video staging: A proxy-server-based approach to end-to-end video delivery over widearea networks," IEEE/ACM Transactions on Networking, vol. 8, no. 4, Aug. 2000.

[90] R. Rejaie and J. Kangasharju, "Mocha: A Quality Adaptive Multimedia Proxy Cache for Internet streaming",Proceedings of NOSSDAV'01, Jun. 2001

[91] Simon S. Lam, Simon Chow, and David K. Y. Yau, "A Lossless Smoothing Algorithm for Compressed Video", IEEE/ACM Transactions on Networking, 4(5), October 1996.

[92] J. D. Salehi, Z.-L. Zhang, J. F. Kurose, and D. Towsley, "Supporting stored video: Reducing rate variability and end-to-end resource requirements through optimal smoothing", IEEE/ACM Trans. Networking, vol. 6, pp. 397--410, August 1998.

[93] Ng, J.K.-Y.;, Shibin Song, "A video smoothing algorithm for transmitting MPEG video over limited bandwidth", Proc. of the 4th International Workshop on Real-Time Computing Systems and Applications (RTCSA '97)

[94] Y. Mansour, B.Patt-Shamir, and O. Lapid, "Optimal Smoothing Schedules for Real-Time Streams",. In ACM Principles of Distributed Computing (Portland, OR, July 2000).

[95] S. V. Anastasiadis, K. C. Sevcik, and M. Stumm, "Server-based smoothing of variable bit-rate streams", In Proceedings of the ninth ACM Multimedia Conference, pp.147--158, Ottawa, October 2001.

[96] X. Li, S.Paul, and M. Ammar "Layered Video Multicast with Retransmission (LVMR): Hierarchical Rate Control Schemes", In Proc. of Infocom 98, San Francisco, CA, March 1998.

[97] R. Marasli,P. D. Amer, and P. T. Conrad, "Retransmission-Based Partially Reliable Transport Service: An Analytic Model", INFOCOM, pp. 621-629, 1996.

[98] B. J. Dempsey "Retransmission-based Error Control for Continuous Media Traffic in Packet Switched Networks" Ph.D Dissertation, University of Virginia, may 1994.

[99] I. Rhee, "Error control techniques for interactive low-bit rate video transmission over the Internet", In Proc. ACM SIGCOMM, September 1998.

[100] C. Papadopoulos and G. Parulkar, "Retransmission-based error control for continuous media applications", In Proceedings of the Sixth International Workshop on Network and Operating System Support for Digital Audio and Video, pp. 5–12, 1996.

[101] B. W.Wah, X. Su, and D. Lin. "A survey of error-concealment schemes for real-time audio and video transmissions over the Internet". In Proc. International Symposium on Multimedia Software Engineering, pp. 17–24, Taipei, Taiwan, December 2000.

[102] S. Lin, D.J. Costello, "Error Control Coding: Fundamentals and Applications", Prentice Hall, 1983.

[103] J. Rosenberg, and H. Schulzrinne "RFC 2733, RTP Payload Format for Generic Forward Error Correction", Request For Comments, IETF, Dec. 1999

[104] J-C. Bolot, T. Turletti, "Adaptive Error Control for Packet Video in the Internet", in Proc. of IEEE ICIP '96, pp. 232-239, Lausanne, Sept. 1996.

[105] G. Carle and E. W. Biersack, "Survey of error recovery techniques for IP based audio-visual multicast applications", IEEE Network, vol. 11, no. 6, 1997, pp. 24-36.

[106] I. Rhee and S. Joshi, "Error recovery for interactive video transmission over the Internet", IEEE J. Selected Area Comm., vol. 18, no. 6, pp. 1033-49, June 2000

[107] B. Girod, K. Stuhlmuller, M. Link, and U. Horn, "Packet Loss Resilient Internet Video Streaming", in Proceedings of SPIE Visual Communications and Image Processing, San Jose, CA, January 1999, pp. 833-844.

[108] G. Liebl, M. Wagner, J. Pandel and W. Weng "An RTP Payload Format for Erasure-Resilient Transmission of Progressive Multimedia Streams" draft-ietf-avt-uxp-04.txt, work in progress, expires: May 2003

[109] Adam H. Li, et al. "An RTP Payload Format for Generic FEC with Uneven Level Protection" Internet draft, draft-ietf-avt-ulp-07.txt, work in progress, expires: May 4, 2003.

[110] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson "RFC3550 RTP: A Transport Protocol for Real-Time Applications, obsolete 1889", Request for Comments, IETF, July 2003.

[111] T. Turletti, and C. Huitema, "RFC: 2032 RTP Payload Format for H.261 Video Streams", Request for Comments, IETF, Oct. 1996

[112] D. Hoffman, G. Fernando, and V. Goyal "RFC: 2038 RTP Payload Format for MPEG1/MPEG2 Video", Request for Comments, IETF, Oct. 1996

[113] C. Zhu "RFC: 2190 RTP Payload Format for H.263 Video Streams", Request for Comments, IETF, Sep. 1997

[114] Y. Kikuchi, T. Nomura, S. Fukunaga, Y. Matsui, and H. Kimata "RFC: 3016 RTP Payload Format for MPEG-4 Audio/Visual Streams", Request for Comments, IETF, Nov. 2000

[115] N. Laoutaris, I. Stavrakakis, "Intrastream Synchronization for Continuous Media Streams: A survey of Playout Schedulers", IEEE Network Magazine, Vol.16, No.3, May 2002

[116] K. Fujimoto, S. Ata, and M. Murata, "Statistical analysis of packet delays in the Internet and its application to playout control for streaming applications", IEICE Transactions on Communications, vol. E84-B, pp. 1504–1512, June 2001.

[117] K. Fujimoto, S. Ata, and M. Murata, "Playout control for streaming applications by statistical delay analysis," in Proceedings of IEEE International Conference on Communications (ICC 2001), vol. 8, (Helsinki), pp. 2337–2342, June 2001.

[118] E. G. Steinbach, N. F¨arber, and B. Girod, "Adaptive Playout for Low Latency Video Streaming," Proc. International Conference on Image Processing (ICIP-01), Oct. 2001.

[119] M. Kalman, E. Steinbach, and B. Girod, "Adaptive playout for real-time media streaming, in Proc. IEEE Int. Symp. on Circ. and Syst. May 2002.

[120] E. Crawley, R. Nair, B. Rajagopalan, H. Sandick "RFC: 2386 A Framework for QoS-based Routing in the Internet", Request for Comments, IETF, August 1998.

[121] S. Shenker, J. Wroclawski, "RFC: 2216 Network Element Service Specification Template", Request for Comments, IETF, sep. 1997.

[122] J. Wroclawski, "RFC: 2211 Specification of the Controlled-Load Network Element Service", Request for Comments, IETF, sep. 1997.

[123] S. Shenker, and J. Wroclawski "RFC: 2215 General Characterization Parameters for Integrated Service Network Elements", Request for Comments, IETF, sep. 1997.

[124] S. Shenker, C. Partridge, R. Guerin "RFC: 2212 Specification of Guaranteed Quality of Service", Request for Comments, IETF, sep. 1997.

[125] L. Georgiadis, R. Guerin, V. Peris, and R. Rajan, "Efficient support of delay and rate guarantees in an internet", In Proceedings of ACM SIGCOMM, Stanford University, CA, August 1996.

[126] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss "RFC: 2475 An Architecture for Differentiated Services", Request for Comments, IETF, December 1998.

[127] K. Nichols, S. Blake, F. Baker, D. Black "RFC:2474 Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", Request for Comments, IETF, December 1998.

[128] J.Heinanen F.Baker, W. Weiss, and J. Wroclawski "RFC: 2597 Assured Forwarding PHP Group", Request for Comments, IETF, June 1999.

[129] V. Jacobson, K. Nichols, K.Poduri, "RFC: 2598 An Expedited Forwarding PHB", Request for Comments, IETF, June 1999.

[130] W. Fang, Seddigh, B. Nandy "RFC: 2859 A Time Sliding Window Three Colour Marker (TSWTCM)", Request for Comments, IETF, June 2000.

[131] J. Heinanen , R. Guerin, "RFC: 2698 A Two Rate Three Color Marker (TRTCM)", September 1999.

[132] Jitae Shin, JongWon Kim and C.-C. Jay Kuo, "Content-Based Packet Video Forwarding Mechanism in Differentiated Service Networks", IEEE Packet Video Workshop PV'00, May 2000.

[133] M. Albrecht, M. Köster, P. Martini, M. Frank, "End-to-end QoS Management for Delay-sensitive Scalable Multimedia Streams over Diffserv", In Proc. of the 25th Annual Conference on Local Computer Networks, LCN'00, Tampa-FL, 314-323, November 2000.

[134] Huai-Rong, Wenwu, and Ya-Qin Zhang, "Scalable Object-Based Video Multicasting Over The Internet. International Conference on Image Processing", September 2000.

[135] E. Rosen, A. Viswanathan, and R. Callon, "RFC: 3031 Multiprotocol Label Switching Architecture", Request for Comments, IETF, January 2001.

[136] R. Braden, et al. "RFC: 2205 Resource ReSerVation Protocol (RSVP)", Request for Comments, IETF, Sep. 1997.

[137] D. Durham, Ed., J. Boyle, R. Cohen, S. Herzog, R. Rajan, A. Sastry, "RFC: 2748 The COPS (Common Open Policy Service) Protocol", Request for Comments, IETF, January 2000.

[138] S. Herzog, Ed., J. Boyle, R. Cohen, D. Durham, R. Rajan, A. Sastry, "RFC: 2749 COPS usage for RSVP", Request for Comments, IETF, January 2000.

[139] K. Chan, J. Seligson, D. Durham, S. Gai, K. McCloghrie, S. Herzog, F. Reichmeyer, R. Yavatkar, A. Smith "RFC: 3084 COPS Usage for Policy Provisioning (COPS-PR)", Request for Comments, IETF, March 2001.

[140] M. Handley H. Schulzrinne, E. Schooler J. Rosenberg "RFC: 2543 Session Initiation Protocol (SIP)", Request for Comments, IETF, Mar. 1999.

[141] M. Handley, V. Jacobson, "RFC: 2327 SDP Session Description Protocol", Request for Comments, IETF, Apr. 1998.

[142] IUT-T Recommendation H.323 (11/96), "Visual telephone systems and equipment for local area networks which provide a non guaranteed quality of service", Nov. 1996.

[143] IUT-T Recommendation H.323 Draft v4 (11/2000), "Packet-based multimedia communications systems", Nov. 2000.

[144] ISO/IEC 14496-6, "Coding of audio-visual objects -- Part 6: Delivery Multimedia Integration Framework (DMIF) final committee draft", May 1998.

[145] H. Schulzrinne, A. Rao, R. Lanphier, "RFC: 2326 Real Time Streaming Protocol (RTSP)", Request for Comments, IETF, April 1998.

[146] Media Gateway Control (megaco) Working group available at: http://www.ietf.org/html.charters/megaco-charter.html

[147] M. Handley, C. Perkins, E. Whelan, "RFC: 2974 Session Announcement Protocol", Request for Comments, IETF, Oct. 2000.

[148] J. Rosenberg, Schulzrinne "RFC: 2871 A Framework for Telephony Routing over IP", Request for Comments, IETF, Jun. 2000.

[149] S. Iyer, R.R. Kompella, and A. Shelat **"**ClassiPI: An Architecture for Fast and Flexible Packet Classification", IEEE Network special issue on Fast IP Packet Forwarding and Classification for Next Generation Internet Services (March-April 2001).

[150] Pankaj Gupta and Nick McKeown "Algorithms for Packet Classification", IEEE Network Special Issue, vol. 15, no. 2, pp 24-32, (March /April 2001).

[151] D. Heckerman et al. "Real-World Applications of Bayesian Networks", Communications of the ACM volume 38 (1995).

[152] ISO/IEC JTC1/SC29/WG11, "MPEG-7 overview," N4980, July 2002

[153] Werner Almesberger: Differentiated Services on Linux Home Page http://icawww1.epfl.ch/linux-Diffserv/

[154] W. Almesberger, J. H. Salim, A. Kuznetsov: "Differentiated Services on Linux". Work in progress. (June 1999).

[155] ITU-T Recommendation Y.1541 "Internet Protocol Communication Service – IP Performance and Availability Objectives and Allocations", (Oct 2001).

[156] K. Yoshihiro al., " RFC: 3016 RTP Payload Format for MPEG-4 Audio/Visual Streams", the Internet Engineering Task Force (IETF), November 2000.

[157] Basso, Civanlar, P. Gentric, Herpel, Lifshitz, Lim, Perkins, Van Der Meer, "RTP Payload Format for MPEG-4 Streams", the Internet Engineering Task Force (IETF), July 20, 2001, work in progress, expires January 2002.

[158] D.Curet, E.Gouleau, S.Relier, C.Roux, P.Clement, G.Cherry; "RTP Payload Format for MPEG-4 FlexMultiplexed Streams", Internet Draft ,IETF, Work in progress, November, 8 2001.

[159] J. Van Der Meer, D. Mackie, V. Swaminathan, D. Singer, P. Gentric, "RTP Payload Format for Transport of MPEG-4 Elementary Streams", Internet Draft ,IETF, draft-ietf-avt-mpeg4-simple-08.txt, Work in progress, August 2003, Expires February 2004.

[160] C.Guillemot, P.Christ, S.Wesner, and A. Klemets, , "RTP Payload Format for MPEG-4 with Flexible Error Resiliency", Internet draft, IETF, Work in progress, Mars 2000.

[161] A. Klemets, "Common Generic RTP Payload Format", draft-klemets generic-rtp-00, March 13, 1998.

[162] T. Ahmed, G. Buridant and A. Mehaoua, "Encapsulation and Marking of MPEG-4 Video over IP Differentiated Services", in Proc. Of IEEE ISCC 01, pp. 346-352, July 2001.

[163] A. Nafaa, T. Ahmed, Y. Hadjadj Aoul, and A. Mehaoua "Rtp4mux: A Novel MPEG-4 Rtp Payload For Multicast Video Communications Over Wireless IP" in Proc. Of Packet Video PV'2003, April 28-29, April 2003.

[164] S. Floyd, , V. Jacobson, "Link-sharing and Resource Management Models for Packet Networks", in proc. of IEEE/ACM Transactions on Networking, Vol. 3, No. 4, pp. 365-386, August 1995.

[165] David D. Clark and Wenjia Fang, "Explicit Allocation of Best Effort Packet Delivery Service", ACM Transactions on Networking, pp.362-373, August 1998.

[166] S. Floyd and V. Jacobson "Random early detection gateways for congestion avoidance", IEEE/ACM Transactions on Networking. Vol. 1, no. 4, pp. 397-413, August 1993.

[167] S. Floyd, and K. Fall "Promoting the Use of End-to-End Congestion Control in the Internet" Proc. of IEEE/ACM Transaction on Networking, 7(4), pp.458-472, August 1999.

[168] A. Albanese, J. Bloemer, J. Edmonds, M. Luby, and M. Sudan, "Priority Encoding Transmission", IEEE Trans. on Inf. Theory 42, pp. 1737-1747, Nov. 1996.

[169] J. Stuart Hunter. "The Exponentially Weighted Moving Average" J Quality Technology, Vol. 18, No. 4, pp. 203-207, 1986.

[170] W.Lai, B.Christian, R.Tibbs, S.Berghe "Framework for Internet Traffic Engineering Measurement" Internet draft, Work in progress, November 2001.

[171] V. Paxson, G. Almes, J. Mahdavi, M. Mathis "RFC: 2330 Framework for IP Performance Metrics", May 1998.

[172] "OpenLDAP software" available at http://www.openldap.org/

[173] L. Georgiadis, R. Guerin, V. Peris and R. Rajan "Efficient Support of Delay and Rate Guarantees in an Internet" in ACM SIGCOMM, volume 26, number 4, October 1996.

[174] H. Yasuda and H.J.F. Ryan, "DAVIC and Interactive Multimedia Services", IEEE Comm. Mag; vol. 36, n°9., Sept. 1998, pp. 137-143.

[175] ISO/IEC 13818-6:1998 "Generic coding of moving pictures and associated audio information -- Part 6: Extensions for DSM-CC", October 1998.

[176] Singh, Schulzrine "Interworking between SIP/SDP and H.323", the Internet Engineering Task Force (IETF), Work in progress.

[177] H.Agrawal,R.R Roy, V.Palawat, A.Johnston, C.Agboh, D.Wang, H.Schulzrinne, K.Singh, J.Maeng, "SIP-H.323 Interworking", the Internet Engineering Task Force (IETF), w ork in progress 2002.

[178] T. Ahmed, "MPEG-4 DMIF Implementation in Java" Technical report, Master of CS, University of Versailles, France, Sept. 2000. Available at: http://www.prism.uvsq.fr/~tad/publication.

[179] Sun Microsystems "Java Media Framework API" http://java.sun.com/products/java-media/jmf/index.html

[180] "Netdynamics Ressources available » at: http://www.dynamics.com/resources/downloads

[181] Columbia University Resources: SIP User Agent Client available at: http://www.cs.columbia.edu/~kns10/ software/sipua/

[182] T. Berners-Lee, L. Masinter, M. McCahill "RFC: 1738 Uniform Resource Locators (URL)" Request for Comments, IETF, December 1994.

[183] N. Bjorkman, A. Latour-Henner, A. Doria "The movement from monoliths to Component-Based Network Elements", IEEE communications magazine, pp. 86-93, January 2001.

[184] J. Huang "Parlay System and Network Management Working Group" tutorial at www.parlay.org, 2002.

[185] Biswas J., Lazar A. A., Huard J-F, Lim K.,Mahjoub S., Pau L.-F, Suzuki M., Tortensson S., Wang W. and Weinstein S. "The IEEE P1520 Standards Initiative for Programmable Network Interfaces", IEEE Communications Magazine, Vol. 36. No. 10, pp.64-70, October 1998

Appendix A

A Appendix A: Overview of the MPEG-4 Framework

A.1 Introduction

MPEG-4 standard [22][23][24] is an emerging digital multimedia standard with associated protocols for representing, manipulating and transporting natural and synthetic multimedia content (i.e. audio, video and data) over a broad range of communication infrastructures.

The MPEG-4 standard introduces a new technique of coding multimedia scenes called "object-based compression". This technique allows the encoding of different audio-visual objects in the scene independently.

The original characteristic of MPEG-4 is to provide an integrated object-oriented representation of multimedia content for the support of new ways of communication, access, and interaction with digital audiovisual data, and offering a common technical solution to various telecommunications, broadcast, and interactive services. MPEG-4 addressed a broad range of existing and emerging multimedia applications such as video on the Internet, multimedia broadcasting, content-based audiovisual database access, games, audiovisual home editing, advanced audiovisual communications and video over mobile networks.

An MPEG-4 scene consists of one or more Audio Visual Objects (AVO), each of them is characterized by temporal and spatial information. The hierarchical composition of an MPEG-4 scene is depicted in Figure A-1. Each Video Object (VO) may be encoded in a scalable (multi-layer) or non scalable (single layer) form. A layer is composed of a sequence of a Group of Video-Object-Plane (GOV). A Video Object Plane (VOP) is similar to the MPEG-2 frame. VOP supports intra coded (I-VOP) temporally predicted (P-VOP) and bi directionally predicted (B-VOP).

To reach the defined objective of MPEG-4 standard, a set of tools was defined in several recommendations of the standard. First, the media compression schemes are defined in the Visual [22] and the Audio [23] parts of the MPEG-4 framework. Every multimedia element to be compressed is presented as individual AVO. The combination of these elements is assured by the scene description language called *Binary Format for Scenes* (BIFS) defined in MPEG-4 Systems document [22]. The delivery of the media is defined in the *Delivery Multimedia Integrated Framework* (DMIF) [144], which is the part 6 of MPEG-4 specification. DMIF is actually the control plane of MPEG-4 *Delivery layer*. It allows applications to transparently access, retrieve, and view multimedia streams whether the source of the stream is located on a remote or local end-system.

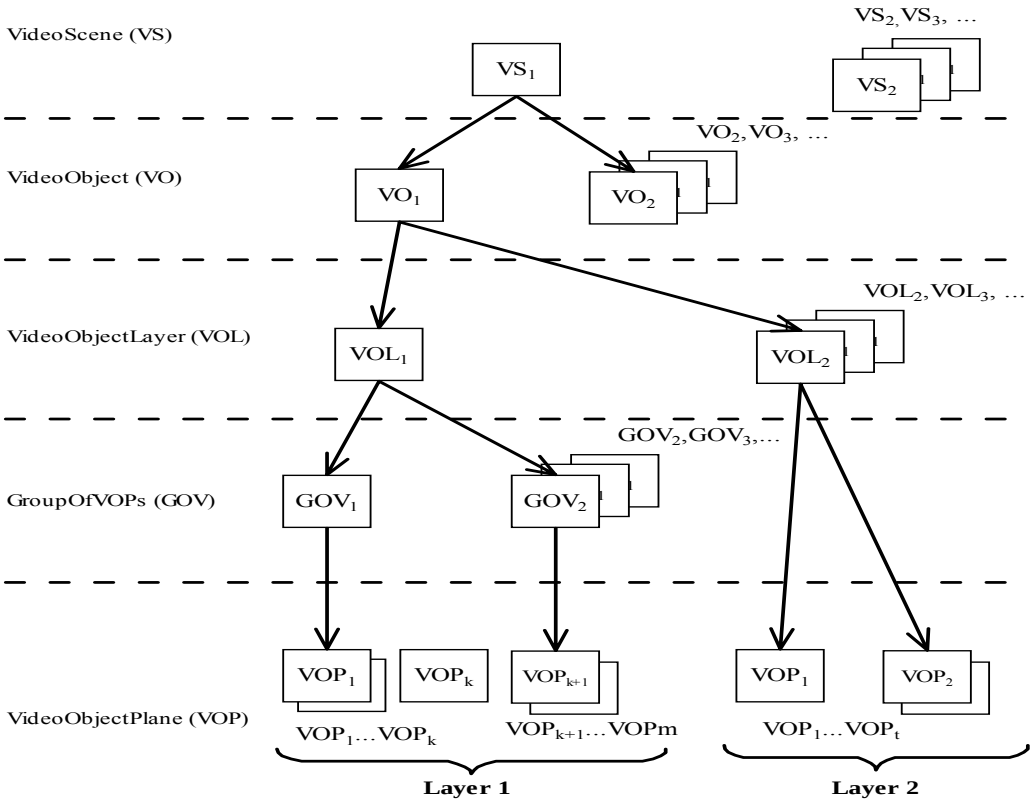


Figure A-1: Hierarchical composition of an MPEG-4 scene

Figure A-2 depicts an MPEG-4 scene, along with the associated object tree structure. The scene shows news broadcast that contains four objects: logo, speaker, background and speech. The final scene is obtained by composition.

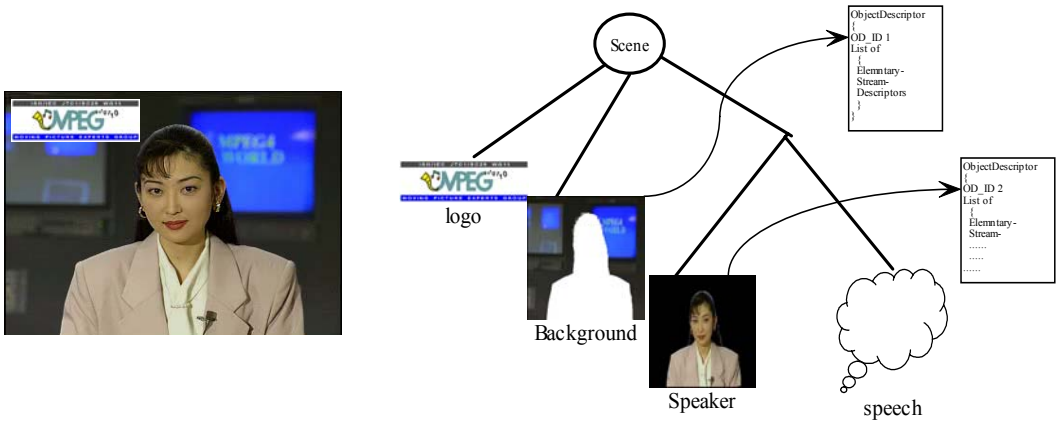


Figure A-2: Simple composite MPEG-4 scene using "Akiyo" video sequence

A.2 MPEG-4 Layered Architecture

MPEG-4 terminal architecture is composed of three layers (see Figure A-3): the *Compression Layer*, the *Sync Layer* and the *Delivery Layer* [22].

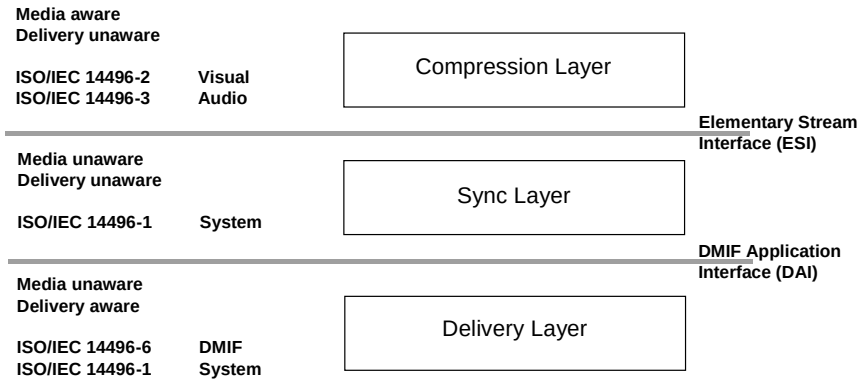


Figure A-3: MPEG-4 framework

Compression layer generates representation of content data called *Elementary Streams* (ES), the hierarchical relations; locations and properties of ESs in a representation are described by dynamic set of *Object Descriptors* (OD). ESs are the basic abstraction for any data source. ODs are themselves conveyed through one or more ESs. MPEG-4 scene is described by *Scene Description* (SD) information that addresses the organization of AVOs within a scene, in terms of both spatial and temporal location. SD is performed using BIFS, which is a VRML-based language. ESs are partitioned into *Access Units* (AU) which are defined by MPEG-4 general framework for the framing of semantic entities in ES. An MPEG Access Unit (AU) is the smallest data entity to which timing information is attributed. In case of audio an Access Unit may represent an audio frame and in case of video a picture. For examples, a valid MPEG-4 ES could be an MPEG-1 video, labelled with MPEG-4 system information in its headers. An AU would then be one video frame I, B or P. Those AUs will be labelled with priority information, timing information, and others. An AU is the smallest data entity to which timing information can be assigned.

On the *Sync layer*, AUs are conveyed into a sequence of packets called SL-PDUs. A SL-PDU consists of SL packet header and SL packet payload. The header provides means of continuity checking in case of data loss, carries the time stamps, and associated control information.

The SL-PDUs are subsequently transmitted to the *Delivery Layer* for multiplexing and generating a FlexMux stream. The FlexMux tool is one of the components of the MPEG-4 DMIF. FlexMux is a flexible and simple data multiplexer that accommodates interleaving of SL-PDUs. Two different modes of operation for FlexMux are defined. The first mode is called "Simple Mode" in which one SL-PDU is encapsulated into one FlexMux packet, and the second mode is called "MuxCode Mode" in which one or more SL-PDU are encapsulated into a single "FlexMux" packet. The interface between the *Sync layer* and the *Delivery layer* is referred to DAI (*DMIF Application Interface*) [144]. It offers content location independent procedures for establishing MPEG-4 sessions and access to transport channels such as RTP/UDP/IP. DMIF is primarily an integration framework. It provides default DMIF signaling protocol which corresponding to DNI (*DMIF Network Interface*).

A.3 Object Description Framework

The purpose of the MPEG-4 object descriptor framework is to identify, describe and associate elementary streams with the various components of an audiovisual scene [22]. A "bootstrap" stream describes the streaming resources used in a virtual world. A media object is associated to its elementary stream resources via an object descriptor. The Figure A-4 shows the principle of MPEG-4 Object Descriptor.

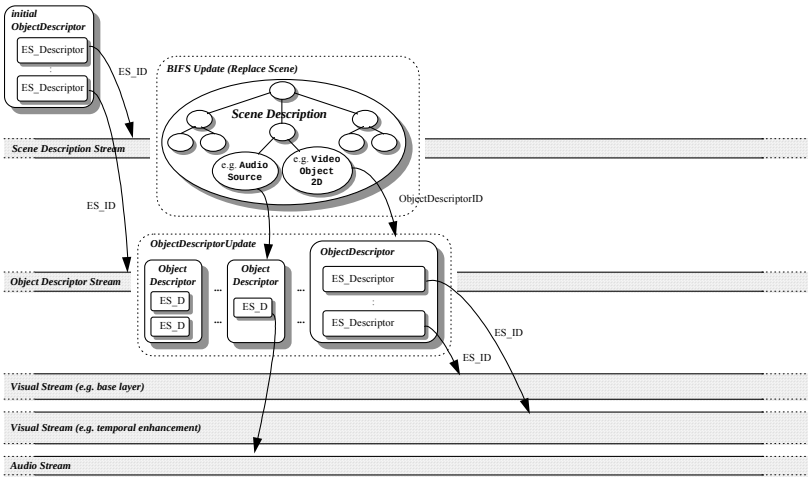


Figure A-4: MPEG-4 object descriptor structure

A.3.1 Object Descriptor

An object descriptor is a collection of one or more Elementary Stream descriptors that provide configuration and other information for the streams that relate to a single object (media object or scene description. Each object descriptor is assigned an identifying number (Object Descriptor ID), which is unique within the current session. This identifier is used to associate media objects in the Scene Description with a particular object descriptor, and thus the elementary streams related to that particular object.

Elementary Stream descriptors include information about the source of the stream data, in form of a unique numeric identifier (the Elementary Stream ID) or a URL pointing to a remote source for the stream. ES IDs are resolved to particular delivery channels at the transport layer. ES Descriptors also include information about the encoding format, configuration information for the decoding process and the Sync Layer packetization, as well as QoS (quality of service) requirements for the transmission of the stream and intellectual property identification. Dependencies between streams can also be signalled, for example to indicate dependence of an enhancement stream to its base stream in scalable AVO representations, or the availability of the same speech content in various languages.

A.3.2 Initial Object Descriptor

Initial object descriptors (IODs) serve to access MPEG-4 content. The IOD is an object descriptor that does not only describe a set of elementary streams, but it also conveys the set of profile and level information that is needed by a receiver to assess the processing resources needed for that content.

A.3.3 Scene Description Streams

Scene description addresses the organization of AVOs in a scene, in terms of both spatial and temporal positioning. This information allows the composition and rendering of individual AVOs after their respective decoders reconstruct them. This specification, however, does not mandate particular composition or rendering algorithms or architectures since they are implementation-dependent.

A.3.4 ES_Descriptor

`ES_Descriptor` conveys all information related to a particular elementary stream and has three major parts.

1. The first part consists of the identifier `ES_ID,` which is a unique reference to the elementary stream within its name scope, a mechanism to group elementary streams within this `ObjectDescriptor` and an optional URL string.
2. The second part is a set of optional extension descriptors that support the inclusion of future extensions as well as the transport of private data in a backward compatible way.
3. The third part consists of the `DecoderConfigDescriptor`, `SLConfigDescriptor,` `IPI_Descriptor` and `QoS_Descriptor` structures, which convey the parameters and requirements of the elementary stream.

Appendix A gives a structure example of an `ES_Descriptor`.

A.3.5 DecoderConfigDescriptor

`DecoderConfigDescriptor` provides information about the decoder type and the required decoder resources needed for the associated elementary stream. This is needed at the receiver to determine whether it is able to decode the elementary stream. A stream type identifies the category of the stream while the optional decoder specific information descriptor contains stream specific information for the set up of the decoder in a stream specific format that is opaque to this layer.

A.3.6 DecoderSpecificInfo

Decoder specific information constitutes an opaque container with information for a specific media decoder. Depending on the required amount of data, two classes with a maximum of 255 and 2^{32}-1 bytes of data are provided.

A.3.7 SLConfigDescriptor

This descriptor defines the configuration of the Sync Layer header for this elementary stream.

A.3.8 IP Identification Data Set

Intellectual Property Identification Data Set is used to identify content. All types of elementary streams carrying content can be identified using this mechanism. The content types include audio, visual and scene description data. Multiple `IP_IdentificationDataSet` may be associated to one elementary stream. The IPI information shall never be detached from the ES_Descriptor.

A.3.9 IPI_DescPointer

`IPI_DescPointer` class contains a reference to the elementary stream that includes the IP Identification Data Set(s) that are valid for this stream. This indirect reference mechanism allows conveying `IP_IdentificationDataSet` elements only in one elementary stream while making references to it from any `ES_Descriptor` that shares the same IP Identification Data.

A.3.10 QoS Descriptor

In our studies, QoS_descriptor is the important parameters of the classification. QoS_descriptor conveys the requirements that the ES has on the transport channel and a description of the traffic that this ES will generate. Table A-1 shows QoS parameters used in the OD of an ES. The unit of each metric is specified in Table A-2.

QoS Metric	Synthetic Description
MAX_DELAY	Absolute maximum end to end delay for the stream
PREF_MAX_DELAY	Preferred maximum end to end delay for the stream
LOSS_PROB	Allowable loss probability of any single AU.
MAX_GAP_LOSS	Maximum allowable number of consecutively lost Aus
MAX_AU_SIZE	Maximum size of an AU
AVG_AU_SIZE	Average size of an AU
MAX_AU_RATE	Maximum arrival rate of Aus
PRIORITY	Priority for the stream

Table A-1: List of defined QoS metrics in MPEG-4 object descriptor

QoS_QualifierTag	Type	Unit
PRIORITY	Byte	Higher values mean higher priority
MAX_DELAY	Long	Microseconds
AVG_DELAY	Long	Microseconds
LOSS_PROB	Double	Fraction (0.00 – 1.00)
MAX_GAP_LOSS	Long	Integer number of Access Units (AU)
MAX_AU_SIZE	Long	Bytes
MAX_AU_RATE	Long	AUs/second
AVG_AU_SIZE	Double	Bytes

Table A-2: Metrics data syntax

A.3.11 ExtensionDescriptor

Additional descriptors may be defined using the syntax in the class definitions above. Depending on the value of the class tag, the maximum length of the descriptor can be either 2^8-1, 2^{16}-1 or 2^{32}-1 byte.

A terminal that conforms to this specification may ignore these descriptors. The available class tag values for extension descriptors allow ISO defined extensions as well as private extensions.

A.3.12 RegistrationDescriptor

The registration descriptor provides a method to uniquely and unambiguously identify formats of private data streams.

This appendix presents structure example of MPEG-4 Object Descriptor containing H263 video and G723 audio elementary streams. Each stream is described by an ObjectDescriptor, this later, contains a QoS_Descriptor.

A.4 Example of an Object Descriptor

This example presents an MPEG-4 scene composed of an H263 video and G723 audio elementary streams. Each stream is described by an ObjectDescriptor record, this later, contains the following QoS_Descriptor.

```
ObjectDescriptor {
       objectDescriptorID 2
       es_descriptor ES_Descriptor {
          es_Number 1
          decConfigDescr DecoderConfigDescriptor {
             streamType 5    // VisualStream
             bufferSizeDB 16000
             specificInfo H263
          }²
       alConfigDescr ALConfigDescriptor {
             useAccessUnitEndFlag TRUE
             useRandomAccessPointFlag TRUE
             useTimeStampsFlag TRUE
             timeStampResolution 1000
             timeStampLength 10
          }
       QoS_Descriptor {
             streamPriority 0
             max_delay   150
             pref_max_delay 100
             loss_prob  0.001
       }

    }
}

ObjectDescriptor  {
       objectDescriptorID 3
       es_descriptor ES_Descriptor {
          es_Number 1
          decConfigDescr DecoderConfigDescriptor {
             streamType 6    // AudioStream
             bufferSizeDB 500
             specificInfo G723
          }
          alConfigDescr ALConfigDescriptor {
             timeStampResolution 1000
             accessUnitRate 30
          }
          IPI_Descriptor {
             IPI_ES_Id 0
          }
          QoS_Descriptor {
             streamPriority 0
                              max_delay   350
                              pref_max_delay 250
                              loss_prob  0.01

          }
       }
```