Mario Marchese

QoS over Heterogeneous Networks

WILEY

QoS OVER HETEROGENEOUS NETWORKS

Mario Marchese

Department of Communications, Computer and System Science
University of Genoa, Italy

John Wiley & Sons, Ltd

QoS OVER HETEROGENEOUS NETWORKS

QoS OVER HETEROGENEOUS NETWORKS

Mario Marchese

Department of Communications, Computer and System Science
University of Genoa, Italy

John Wiley & Sons, Ltd

John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester,
West Sussex, PO19 8SQ, England

Telephone (+44) 1243 779777

Email (for orders and customer service enquiries): cs-books@wiley.co.uk
Visit our Home Page on www.wiley.com

Other Wiley Editorial Offices

John Wiley & Sons Inc., 111 River Street, Hoboken, NJ 07030, USA

Jossey-Bass, 989 Market Street, San Francisco, CA 94103-1741, USA

Wiley-VCH Verlag GmbH, Boschstr. 12, D-69469 Weinheim, Germany

John Wiley & Sons Australia Ltd, 42 McDougall Street, Milton, Queensland 4064, Australia

John Wiley & Sons (Asia) Pte Ltd, 2 Clementi Loop #02-01, Jin Xing Distripark, Singapore 129809

John Wiley & Sons Canada Ltd, 6045 Freemont Blvd, Mississauga, ONT, L5R 4J3, Canada

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Anniversary Logo Design: Richard J. Pacifico

British Library Cataloguing in Publication Data

A catalogue record for this book is available from the British Library

ISBN 978-0-470-01752-4 (HB)

Typeset in 10/12pt Times by Integra Software Services Pvt. Ltd, Pondicherry, India
Printed and bound in Great Britain by Antony Rowe Ltd, Chippenham, Wiltshire
This book is printed on acid-free paper responsibly manufactured from sustainable forestry in which at least two trees are planted for each one used for paper production.

To my mother and father and to my grandparents, for their constant support during my childhood, my boyhood and in the first steps of my career.
To my wife Cinzia and my daughter Michela, for their love, which makes effective my study and research, as well as possible the realization of this book

Contents

Preface

The importance of Quality of Service (QoS) is parallel with the recent evolution of telecommunication networks, which are characterized by a great heterogeneity. On the one hand, many applications require a specific level of assurance from the network. Examples may be assured database access to retrieve information, tele-medicine, remote control of robots in hazardous environments, financial operations, purchase and delivery, tele-learning, applications for emergencies and security. On the other hand, communication networks are characterized by many levels of heterogeneity: portions managed by different Service Providers; different transmission means such as cables, satellites and radios; different implemented solutions such as ATM, IPv4, IPv6 and MPLS. Moreover, a network may be heterogeneous also from the point of view of users, who can require different services and have a different availability to pay for them.

So, even if the QoS is the object of a great number of studies, questions to answer are as follows: Does QoS fit within heterogeneous networks? What is the impact on the performance if information traverses different network portions that use specific QoS paradigms? What happens if the access or another portion of the overall path implements radio or satellite technology?

These questions and corresponding possible answers are the core of this book, which contains both basic material about QoS management and new ideas and proposals, whose aim is to open a discussion with students and scientists about this topic, which, in my view, is of crucial importance for future communications.

Modern telecommunication networks are essentially composed of different portions and technologies: each single portion may implement a different QoS solution, whose algorithms to satisfy performance requirements may change together with the performance parameters. The challenge is to offer end-to-end QoS guarantees over such heterogeneous networks transparently to the users. Essentially, QoS requests should traverse the overall network from the source to the destination through portions that implement different technologies and different protocols; QoS requests should be received and understood by each specific portion where QoS may have a different meaning and interpretation, which depend on used protocols and network features; QoS requests should be managed by control mechanisms suited for the aim; each single QoS solution is composed of layered architectures and each layer must have a specific role in QoS provision.

In my view, the overall problem of QoS interworking may be structured into two different actions, which, together, match the mentioned issues:

- Vertical QoS mapping
- Horizontal QoS mapping

The concept of vertical QoS mapping is based on the idea that a telecommunication network is composed of functional layers and that each single layer must have a role for end-to-end

QoS provision. The overall result depends on the QoS achieved at each layer of the network and it is based on the functions performed at the layer interfaces. The idea is to define an interface between adjacent layers through which to offer a specific QoS service. For example, if Layer 3 implements efficient QoS mechanisms, it is topical that Layer 2 can assure a specific service to Layer 3; otherwise the implementation of complex QoS mechanism at Layer 3 is useless. QoS requirements flow vertically and need to be received, understood and satisfied by the layer below.

The concept of horizontal QoS mapping, even if much linked to the previous concept when implemented in the field, is represented by the need to transfer QoS requirements among network portions implementing their own technologies and protocols.

The implementation of both requires the use of QoS management functions and QoS architectures deeply explained in the book. The idea is that each single network portion which composes the heterogeneous network deserves a peculiar solution. Special tools called QoS gateways implemented through QoS Relay Nodes will take charge of that.

The overall action of QoS gateways heavily depends on the chosen QoS architecture. It may range from no quality when no control is implemented to loose QoS assurance up to hard QoS guarantees. Concerning QoS gateways that operate with hard-QoS guarantees, the final action of the QoS gateways may be modelled as a bandwidth pipe acting as an end to end or, at least, from the first to the last QoS gateway along the network. The bandwidth pipe derives from the co-operation of all layers and bandwidth needs may change passing from a layer to another for the vertical mapping adaptation. It allows a wider interpretation of QoS gateways, which extend their features up to a possible optimization of the overall network performance through a dedicated protocol stack that may be implemented for each specific network portion. The approach may be identified as Performance Optimization Through Layers (POTL) and also referenced as cross-layer interaction.

The book contains many operative examples directly derived from my personal experience and from the studies of my research group at the Department of Communications, Computer and System Science at the University of Genoa, Italy. In particular, I would like to thank Dr Igor Bisio, Dr Tomaso de Cola, Dr Maurizio Mongelli and Dr Giancarlo Portomauro for their precious help. Much material contained in the book, in particular related to Chapters 9, 11 and 12, comes from the research activity co-authored with them.

The book is organized as follows. Chapter 1 is an introduction to the Quality of Service (QoS). It includes the definition, the applications that need it and the main QoS metrics. Chapter 2 introduces the principal issues about the connection of heterogeneous networks and shows a possible example network, which is taken as reference for the entire book. Before entering the main topic of the book, two important issues are explained: the available technology to offer QoS and the control functions necessary to manage it. The former is contained in Chapter 3. Control functions are explained in Chapter 4, which contains many practical examples and results. As said, scenarios and problems of QoS over heterogeneous networks are presented in Chapter 5. The topical concepts of vertical and horizontal QoS mapping are introduced here and are deeply detailed in the following chapters. Chapters 6 and 7 concern the QoS architectures that can be used and the possible signalling protocols to transport information related to QoS management. Chapters 8 and 9 are related to vertical QoS mapping. Chapter 8 presents the reference architecture, the control modules, the interface that separates functional layers, which is crucial for the comprehension of the approach proposed in this book, and the main problems linked to vertical QoS mapping. Just referring to them,

Chapter 9 contains the proposal of a control algorithm to get efficient solutions. Chapter 9 is more oriented to research and should be of interest for scientists working in the QoS management field. Chapter 10 describes the main characteristics of the QoS gateways linked to bandwidth allocation schemes and proposes an extension of the implemented features so as to act also at transport and application layers in peculiar environments such as satellites and radios. Chapters 11 and 12 conclude the book by proposing, respectively, possible algorithms for bandwidth allocation in satellite scenarios and transport layer improvements, as introduced in Chapter 10. The last two chapters have the aim, as Chapter 9, to propose possible studies and research ideas.

List of Abbreviations

6LSA	IPv6 Label Switching Architecture
AAL	ATM Adaptation Layer
ABR	Available Bit Rate
ACK	Acknowledgment
AD	Average Delay
AF	Assured Forwarding
AR	Access Router
ARQ	Automatic Repeat Request
ATM	Asynchronous Transfer Mode
BB	Bandwidth Broker
BE	Best Effort
BER	Bit Error Ratio
B-ISDN	Broadband-Integrated Services Digital Network
BR	Border Router
BSM	Broadband Satellite Multimedia
CAC	Call Admission Control
CAP-ABASC	Constrained Average Probability-Adaptive Bandwidth Allocation over Satellite Channel
CBR	Constant Bit Rate
CCSDS	Consultative Committee for Space Data Systems
CFDP	CCSDS File Delivery Protocol
CK	Complete Knowledge
CK-STP	Complete Knowledge-Satellite Transport Protocol
CLS	Controlled Load Service
CoS	Class of Service
CR	Conformant Rate
CR-LDP	Constrained Routing Label Distribution Protocol
CS	Class Selector
CSF6N	Class of Service Full IPv6 Network
CUMD	Constrained Utopia Minimum Distance
DiffServ	Differentiated Services
DLCI	Data Link Connection Identifier
DoD	Department of Defense

DS	Differentiated Services
DSCP	Differentiated Services CodePoint
DTM	Dynamic Transfer Mode
DVB	Digital Video Broadcasting
DWDM	Dense Wavelength Division Multiplexing
ECN	Explicit Congestion Notification
EF	Expedited Forwarding
EqB	Equivalent Bandwidth
e-QC	external QoS Class
ER	Edge Router
ETSI	European Telecommunications Standards Institute
EXP	Experimental field
F6SN	Full IPv6 Switched Network
FEC	Forward Error Correction
FF	Fixed-Filter
FIFO	First In First Out
FIX	Fixed Allocation
FLSR	Flow Label Switching Router
FLSER	Flow Label Switching Edge Router
FLTT	Flow Label Translation Table
GCRA	Generic Cell Rate Algorithm
GEO	Geostationary Orbit
GIG	Global Information Grid
GPS	Generalized Processor Sharing
GPRS	General Package Radio Service
GS	Guaranteed Service
GSM	Global System for Mobile communications
HEU	Heuristic
IETF	Internet Engineering Task Force
INP	IP Network Provider
IntServ	Integrated Services
IP	Internet Protocol
IPDV	IP Packet Delay Variation
IPER	IP Error Ratio
IPLR	IP Packet Loss Ratio
IPTD	IP Packet Transfer Delay
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
IRS	Internal Resource Signalling
ISDN	Integrated Services Digital Network

ISO	International Organization for Standardization
IW	Initial Window
LAN	Local Area Network
LC	Local Controller
LCT	Last Compliance Time
LER	Label Edge Router
LEO	Low Earth Orbit
l-QC	local QoS Class
LSP	Label Switch Path
LSR	Label Switching Router
LVC	Local Virtual Connection
LVCL	Local Virtual Connection Link
MEO	Medium Earth Orbit
MLPP	Multi Level Precedence and Pre-emption
MOP	Multi-Objective Programming
MOS	Mean Opinion Score
MPLS	MultiProtocol Label Switching
MTA	Mail Transfer Agents
MTSI	Multi Threshold Smoothed Increase
NBS	Nash Bargaining Solution
NCC	Network Control Centre
NM	Negotiation Manager
nrt-VBR	non-real-time-Variable Bit Rate
NSIS	Next Steps in Signalling
NSLP	NSIS Signalling Layer Protocol
NTLP	NSIS Transport Layer Protocol
OAM	Operation And Maintenance
OPWA	One Pass With Advertising
OSI	Open System Interconnection
P	Precedence
PCI	Protocol Control Information
PCN	Pre Congestion Notification
PDU	Protocol Data Unit
PEP	Performance Enhancing Proxy
PHB	Per-hop behaviour
PHOP	Previous hop
PLP	Packet Loss Probability
POP	Pareto Optimal Point
POTL	Performance Optimization Through Layers
P-QoS	Perceived QoS
PRN	Private Relay Node

q-BGP	QoS-Border Gateway Protocol
QCS	QoS Customer Server
QNF	QoS NSIS Forwarder
QNI	QoS NSIS Initiator
QNR	QoS NSIS Responder
QNS	QoS Network Server
QID	Queue Identifier
QoS	Quality of Service
QoS-PRN	QoS-Private Relay Node
QoS-RN	QoS-Relay Node
QPMD	QoS Point Minimum Distance
RCBC	Reference Chaser Bandwidth Controller
RED	Random Early Detection
RL	Relay Layer
RM	Resource Manager
RME	Resource Management Entity
RN	Relay Node
RP	Relay Point
RR	Resource Reservation
RSVP	Resource ReSerVation Protocol
rt-VBR	real-time-Variable Bit Rate
RTO	Retransmission Timeout
RTT	Round Trip Time
SAR	Segmentation And Reassembly
SCFQ	Self-Clocked Fair Queuing
SD	Satellite Dependent
SDH	Synchronous Digital Hierarchy
SDU	Service Data Unit
SE	Shared-Explicit
SFM	Stochastic Fluid Model
SI	Satellite Independent
SIP	Session Initiation Protocol
SI-SAP	Satellite Independent Service Access Point
SLA	Service Level Agreement
SLS	Service Level Specification
SMA	Signalling Management Agent
SME	Service Management Entity
SP	Service Provider
SONET	Synchronous Optical Network
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TCP MTSI	TCP Multi Threshold Smoothed Increase
TD	Technology Dependent

TDM	Time Division Multiplex
TI	Technology Independent
TI-SAP	Technology Independent Service Access Point
ToS	Type of Service
TW	Transmission Window
UBR	Unspecified Bit Rate
UDP	User Datagram Protocol
UMD	Utopia Minimum Distance
UMTS	Universal Mobile Telecommunications System
VALUE	Value Function
VBR	Variable Bit Rate
VC	Virtual Channel
VCI	Virtual Channel Identifier
VCL	Virtual Channel Link
VCo	Virtual Connection
VoIP	Voice over Internet Protocol
VP	Virtual Path
VPI	Virtual Path Identifier
VPL	Virtual Path Link
WAN	Wide Area Network
WF	Wildcard-Filter
WFQ	Weighted Fair Queuing
WF^2Q	Worst-Case Fair Weighted Fair Queuing
WRED	Weighted Random Early Detection
WRR	Weighted Round Robin
WWW	World Wide Web

1

What is QoS?

1.1 QoS Definition

According to ISO 8402, the word quality is defined as "the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs". ISO 9000 defines quality as the degree to which a set of inherent characteristics fulfils requirements. ITU-T (Recommendation E.800 [ITU-TE.800]) and ETSI [ETSI-ETR003] basically defines Quality of Service (QoS) as "the collective effect of service performance which determine the degree of satisfaction of a user of the service". As stated in [ETSI-TR102], IETF considers QoS as the ability to segment traffic or differentiate between traffic types in order for the network to treat certain traffic flows differently from others. QoS encompasses both the service categorization and the overall performance of the network for each category.

Concerning the network viewpoint, QoS is the ability of a network element (e.g. an application, host or router) to have some level of assurance that its traffic and service requirements can be satisfied. QoS manages bandwidth according to application demands and network management settings.

The term "QoS" is used in different meanings, ranging from the users' perception of the service to a set of connection parameters necessary to achieve particular service quality. The QoS meaning changes, depending on the application field and on the scientific scope. Reference [Gozdecki03], starting from the terminology concerning QoS in IP networks, defines some reference points about the QoS issue. The authors, mentioning [Hardy01], identify three types of QoS: intrinsic, perceived and assessed. Intrinsic QoS is directly provided by the network itself and may be described in terms of objective parameters as, for instance, loss and delay. Perceived QoS (P-QoS) is the quality perceived by the users; it heavily depends on the network performance but it is measured by the "average opinion" of the users. Mean Opinion Score (MOS) methods are often used to perform the measure of the quality: users assign an MOS rating to the application they are evaluating as follows: 1 – bad, 2 – poor, 3 – fair, 4 – good, 5 – excellent. The MOS is the arithmetic mean of all the individual scores, and can range from 1 (worst) to 5 (best) [MOS].

QoS over Heterogeneous Networks Mario Marchese

Even if there is a strict connection with the objective metrics provided by the network, the user does not necessarily perceive an objective performance increase (or decrease), in correspondence of an intrinsic QoS variation. References [Adami01] and [Adami02] report a performance evaluation by using MOS values along with measures of objective metrics and show that the correspondence is not always straightforward. The topic is of extreme importance and deserves great attention. Control schemes applied to telecommunications, which will receive a great deal of attention within this book, imply a big economic effort, both in the design and in the implementation phase. If, for example, the application of a specific control algorithm leads to a performance improvement of 20% concerning the performance metric (e.g. a decrease of 20% of the lost packets while traversing a network), it appears as an excellent result and, actually, from an objective viewpoint, it is. But, is it perceived by the real users? Can human eyes and ears appreciate the improvement? Is the economic effort provided to implement the control architecture worthwhile? A practical case study taken from the direct experience of the author within the framework of the activity performed in the 3-year project "Integration of Multimedia Services on Heterogeneous Satellite Networks", funded by the Italian Space Agency (ASI) and carried out by the Italian National Consortium for Telecommunications (CNIT), may help understand. The case study presents an experimental approach to provide a guaranteed QoS over a satellite network based on the Internet Protocol (IP). The aim is to get a proper environment for data, voice and video transmission to be used for tele-working and distance learning through videoconference. Two Local Area Networks (LANs), located in Site A and Site B, are connected through a satellite link at 2 Mbps; the LAN located in Site C, where no satellite station is available, is connected to the network in Site A by using Integrated Services Digital Network (ISDN). The gateways among the LANs and the external parts (satellite stations or ISDN) are represented by routers, which are network devices operating at IP level. The details of the trials are not relevant in this context. The focus is on the relation between objective performance parameters and subjective perception, which is related both to the characteristics of human perception and to the type of application. In short, the videoconference service is experimentally provided through the network both applying two bandwidth reservation control mechanisms within the intermediate routers to protect audio flows and not implementing any bandwidth protection scheme. The three alternatives are identified as Full-Control, Light-Control and No-Control. The implementation complexity is decreasing with the same order. The percentage of lost packets is measured during the tests as well as the MOS of the participants. The bit rate of the video is incremented step by step. Tables 1.1 and 1.2 contain a small portion of the results in [Adami02] but sufficient to have an idea of the problem.

Table 1.1 Objective metrics

Bit/rate	Video packet loss percentage			Audio packet loss percentage		
	No-control [%]	Light-control [%]	Full-control [%]	No-control [%]	Light-control [%]	Full-control [%]
128	0.05	0.40	0.05	0	0.10	0.09
256	21	7.88	6.50	31.45	0.40	0.23
384	34.73	22	7.74	38.70	0.08	0
512	46.05	33.10	12.92	50.98	0	0

Table 1.2 Subjective metrics

Bit/rate	Videoconference MOS			Video MOS			Audio MOS		
	No-control	Light-control	Full-control	No-control	Light-control	Full-control	No-control	Light-control	Full-control
128	3.5	3.75	3.75	2.75	3	3.25	4	4.5	4.5
256	1.75	3.75	3.75	1.75	3.25	3.5	1.5	4.5	4.5
384	1.5	3	3.5	1	2	3.25	1	4.5	4.5
512	1	2.75	3.5	1	1.75	3.25	1	4	4.5

The first table reports the values of Objective Metrics: video and audio packet loss during the videoconference sessions. Table 1.2 shows the Subjective Metric MOS corresponding to the configurations of Table 1.1, divided for media: audio, video and overall videoconference service. Even if the number of users of the system is not sufficient to have precise measures, the values can provide interesting indications. Concerning video it is important to note that if the percentage of lost packets is about 10%, the perception of the users is still quite good (well above 3). If, for example, the video bit rate is set to 256 Kbps, a big effort to implement a Full-Control is not necessary and the implementation of Light-Control is sufficient to provide a satisfying service. It is even clearer if the MOS of the overall service is analysed: at 256 Kbps the evaluation of videoconference is exactly the same for both Full-Control and Light-Control. This measure also allows to make a further comment while looking at the audio packet loss and at the corresponding MOS metrics. The Light-Control can guarantee a low audio packet loss (approximately the same as the Full-Control); there is a very positive perception of the audio (4.5), which compensates the slight performance decrease of the video in the user's perception. Actually, audio performance is dominant in videoconference and most of the overall evaluation is due to it.

Additionally, if video loss is above 20%, the video perception is very low, but the overall videoconference evaluation (supported by the audio result) for Light-Control is still acceptable. Only when the video performance is really damaged (e.g. video bit rate of 512 Kbps and measured loss of 33%), the users of the service are annoyed and the videoconference evaluation is below 3. In this case, the use of a complex and expensive control is fully justified. Concerning the measure of audio, it is sufficient to have a look at the performance in presence of No-Control: a packet loss of about 30% is sufficient to make the overall service completely useless.

Which is the message of this limited example? Not necessarily an objective performance improvement is perceived by the users and the perception is not always the same because it heavily depends on the type of service.

The last type of QoS reported in [Hardy01] concerns assessed QoS. It is referred to the will of a user to keep on using a specific service. It is related to P-QoS and also depends on the pricing mechanism, level of assistance of the provider and other marketing and commercial aspects. For example, a performance decrease may be surely tolerated by a user if a service is free, but the same decrease will raise criticism if the user is paying for it.

At the moment, most of the QoS provision is offered in terms of intrinsic (objective parameters) QoS by using a Service Level Specification (SLS) which is "a set of parameters and their values which together define the service offered to a traffic" [RFC3206]. SLS is

a separated technical part of "a negotiated agreement between a customer and the service provider on levels of service characteristics and the associated set of metrics" [Gozdecki03, ITU-T-Y.1241], which is the commonly adopted definition of a Service Level Agreement (SLA). An example of SLS is represented by the Asynchronous Transfer Mode (ATM) Traffic Contract [ATM-Forum96] that is composed of traffic parameters and descriptors, along with a set of QoS parameters. SLS used in this work includes the type of traffic (e.g. Premium VBR, Mission critical and best effort); traffic description and conformance testing (packet dimension, application peak and average rate, and, if requested, maximum burst and bucket size); and performance guarantees (packet loss rate, packet transfer delay and packet delay jitter). Another possible example of SLS which the author, together with his research group, proposes for military tactical networks is reported in Table 1.3. It includes also the feature of Multi-Level Precedence and Pre-emption (MLPP), which is a peculiar characteristic of military voice switches but, in military environment, is recommendable also for data traffic. It is of topical importance in tactical environment and establishes a level of priority for calls by using 4 bits ([Polk01, Kingston00]). To make an example taken from [Polk01], normal calls are classified as "Routine"; a lower-level command traffic uses "Priority" and "Immediate" levels. Brigade, battalion and division (field grade traffic) is assigned "Immediate" and, in some cases, "Flash". Corps commanders use "Flash". Presidents and Joint Chiefs use "Flash override". It is used to drop a call when a higher priority connection tries entering the network and the bandwidth is not sufficient for both. Actually, military networks are challenging for communication engineers because they include all the services of interest for civil communications but require different performance requirements in terms of loss, delay and jitter. There are also specific applications that concern the military world. In short, SLS specifies the service from the technical viewpoint. The Service Level Agreement (SLA) includes also non-technical issues [Sarangan2006] as well as pricing and device/network capabilities. For example, customers may be available to relax technical constraints in dependence of the price applied; on the other hand, service providers may offer the same service at different prices depending on the current network utilization. Concerning device and network capabilities, the SLA may also indicate the access technology that the customer can use, for example Ethernet, Wireless LANs, GSM, UMTS, Ad-hoc Network, GPRS and Satellite Access. Even if the SLS does not change, SLA should also consider these aspects that have a relevant impact on the customer choices.

Table 1.3 Possible example of SLS

Service Level Specification	Range
Connection type	Constant Bit Rate (CBR)/Variable Bit Rate (VBR)/Best-effort
Scope	End-to-end
Connection identification	Identifier or sequence of identifiers
Traffic description and conformance testing	Peak rate/bucket size for peak rate/maximum burst size
Performance guarantees	Packet loss rate/packet transfer delay/packet delay jitter
Multi Level Precedence and Pre-emption (MLPP)	Not applied/routine/priority/immediate/flash/flash override

Table 1.4 Operative example of SLS

Service Level Specification	Range
Connection type	Constant Bit Rate (CBR)
Scope	End-to-end
Connection identification	Sequence of identifiers
Traffic description and conformance testing	Peak rate = 64 Kbps/bucket size for peak rate = 512 bytes/maximum burst size not applicable
Performance guarantees	Packet loss rate = 1%/packet transfer delay = 250 ms/packet delay jitter = 30 ms
Multi Level Precedence and Preemption (MLPP)	Priority

An operative example applying the SLS defined above is contained in Table 1.4, where a "lower-level command voice call" for military communication is reported.

1.2 Applications

Which applications need QoS? The answer is simple: all the applications that require a specific level of assurance from the network. The answer does not give any idea about the amount of applications that need QoS. Some of them are listed below: basic services for information transfer for both backbone and access networks, assured database access to retrieve information, telemedicine (transmission of clinical tests, X-rays, electrocardiograms and magnetic resonance), tele-control (remote control of robots in hazardous environments, remote sensors and systems for tele-manipulation), bank and financial operations, purchase and delivery, tele-learning, telephony, videoconferences and applications for emergencies and security.

Having very different characteristics, each mentioned application deserves a specific degree of service, defined at the application layer. Several standardization bodies have tried to define service categories (also called "QoS classes", to be intended at application layer).

ITU-T (in Recommendation Y-1541 [ITU-T-Y.1541]) suggests a definition of QoS classes (for the IP world) that is summarized in Table 1.5.

The ETSI Project TIPHON [ETSI-TR102] proposes an alternative QoS class definition, reported in Table 1.6.

Concerning Broadband-Integrated Services Digital Network (B-ISDN), ITU-T defines a set of service categories, which is reported in Table 1.7 [Onvural94].

Table 1.5 ITU-T Y-1541 QoS classes

QoS class	Characteristics
0	Real-time, jitter sensitive, highly interactive
1	Real-time, jitter sensitive, interactive
2	Transaction data, highly interactive
3	Transaction data, interactive
4	Low loss only (short transactions, bulk data, video streaming)
5	Traditional applications of default IP networks

Table 1.6 TIPHON QoS classes (from [ETSI-TR102])

QoS class	Components	QoS characteristics
Real-time conversational (telephony, teleconference, videophony and videoconference)	Speech, audio, video, multimedia	Delay and delay variation sensitive, limited tolerance to loss and errors, constant and variable bit rate
Real-time streaming (e.g. audio and video broadcast, surveillance, graphics)	Audio, video, multimedia	Tolerant to delay, delay variation sensitive, limited tolerance to loss and errors, variable bit rate
Near real-time interactive (e.g. web browsing)	Data	Delay sensitive, tolerant to delay variation, error sensitive, variable bit rate
Non-real-time background (e.g. e-mail and file transfer)	Data	Not delay and delay variation sensitive, error sensitive, best effort

Table 1.7 B-ISDN services

Categories	Applications (examples)
Conversational	Videoconference, video surveillance, high speed document communication (file transfer, fax, images, sound)
Messaging	E-mail and video e-mail, chat
Retrieval	High speed document retrieval (video, still images, sounds, transaction data)
Distribution *without* user-individual presentation control	Document and video distribution, Pay TV, radio
Distribution *with* user-individual presentation control	Full-channel broadcast TV and videography

Derived from the general categories reported in Table 1.7 and consequently to the standardization of ATM as the technology for implementing B-ISDN, the ATM Forum defines five ATM service categories, reported in Table 1.8 [McDysan99].

Each application mentioned at the beginning of this sub-section may be inserted in the classifications reported above but, besides the general definition reported, each application

Table 1.8 ATM forum service categories

ATM service categories	Representative applications	QoS characteristics
CBR (Constant Bit Rate)	Circuit emulation	Low cell delay variation, low loss
rt-VBR (real time Variable Bit Rate)	Video on demand	Moderate cell delay variation, low loss
nrt-VBR (non-real-time Variable Bit Rate)	Packet traffic	Moderate loss
ABR (Available Bit Rate)	Adaptable rate sources	Low loss
UBR (Unspecified Bit Rate)	Best-effort traffic	No requirements

(or, in this context, more exactly, each user) needs a detailed specification in terms of traffic descriptors and intrinsic QoS parameters to allow a proper service provision by the network.

1.3 QoS Metrics

A further step is to associate objective QoS requirements to QoS traffic classes generically defined above. Concerning the IP environment, the QoS objective metrics mostly used [ITU-T-Y.1540] are as follows:

- IPLR – IP Packet Loss Ratio
- IPTD – IP Packet Transfer Delay
- IPDV – IP Packet Delay Variation (known as Jitter)
- IPER – IP Packet Error Ratio.

Another metric often considered is the skew, which is the average value of the difference of the delays measured by packets belonging to different media, as, for example, voice and video within a videoconference service. In this case, if the skew is large, there is no synchronization between voice and video with the general effect of a bad dubbing.

IP Packet Loss Ratio is the performance measure used in the examples in Tables 1.2 and 1.3. Even if, as said, the problem needs to be solved for each specific application, it is also important to give a range of QoS requirements for traffic classes. Possible end-to-end performance-metric upper bounds are reported in Table 1.9, associated with QoS classes in [ITU-T-Y.1541]. This association is introduced in [ITU-T-Y.1541]. The nature of the objective performance parameter is defined in [ITU-T-Y.1541] as follows: IPTD – upper bound on the mean IPTD; IPDV – upper bound on the $1–10^{-3}$ quintile of IPTD minus the minimum IPTD; IPLR – upper bound on the packet loss probability; IPER – upper bound. "U" stands for "Unspecified" (actually meaning "Unbounded", in this context).

The values reported in Table 1.9 apply to IP public networks and are upper bounds to mean quantities. Actually, each single company and service provider may also offer different end-to-end performance. In other words, taking class 0 for example, it means that, for a generic real-time, jitter sensitive, and highly interactive application, the following performance-metric limits should be guaranteed: mean end-to-end transfer delay below 0.1s, mean jitter below 0.05s, mean packet low rate below 10^{-3} and mean packet error rate below

Table 1.9 IP QoS classes and objective performance-metric upper limits

QoS Class	Characteristics	IPTD	IPDV	IPLR	IPER
0	Real time, jitter sensitive, highly interactive	100 ms	50 ms	1×10^{-3}	1×10^{-4}
1	Real time, jitter sensitive, interactive	400 ms	50 ms	1×10^{-3}	1×10^{-4}
2	Transaction data, highly interactive	100 ms	U	1×10^{-3}	1×10^{-4}
3	Transaction data, interactive	400 ms	U	1×10^{-3}	1×10^{-4}
4	Low loss only (short transactions, bulk data, video streaming)	1 s	U	1×10^{-3}	1×10^{-4}
5	Traditional applications of default IP networks	U	U	U	U

Table 1.10 Examples of multimedia applications' QoS requirements

Application	IPTD	IPDV	IPLR
Data acquisition from sensors	100 ms	50 ms	0
Radar traces	20 ms	1–3 ms	10^{-3}
Weapon control	20 ms	10 ms	0
Sensor control	20 ms	10 ms	0
Voice	250 s	30 ms	10^{-2}
Video streaming	5–10 s	U	$2 \cdot 10^{-2}$

10^{-4}. The recommendation does not refer to specific applications but defines upper limits for traffic classes.

Clearly, within the scenario defined above, it is topical to have specific upper-bound indications for each application. Table 1.10 tries to give them for a group of applications concerning IPTD, IPDV and IPLR. The IPER threshold defined in Table 1.9 is still applicable. The IPTD for voice may be extended to 400 ms. Interaction is still possible in conformity with Table 1.9 indications.

1.4 The Concept of Traffic Flow and Traffic Class

There is no unique definition of traffic flow in the literature. If it is not differently indicated in the text, a flow is considered here as a packet stream associated with a precise user service. So, if traffic is differentiated "per-flow", it means that each single user is identified. To better clarify this aspect, the term "user flow" is often used in the book. A traffic class is considered a group of flows associated with a common identifier. A traffic class should contain traffic with similar characteristics and performance requirements. "Per-class" traffic separation is simpler but coarser. It is also called "per-aggregate" granularity, in the text. The identification feature will be the object of deep discussion in Chapters 3 and 4 and a distinction factor for the QoS architectures presented in Chapter 6.

2

QoS-based Networks

2.1 Heterogeneous QoS-based Networks

As said in the previous chapter, the QoS is the ability of a network element (e.g. an application, host and router) to have some level of assurance that its traffic and service requirements can be satisfied. It means that, within the network, suitable mechanisms must be implemented so that the SLS for each traffic class may be monitored and guaranteed for end-to-end communications. The need of QoS is parallel with the evolution of telecommunication networks, whose main characteristic is, now, heterogeneity. In practice, QoS is the object of a great number of tutorials, forums and conferences. Telecommunication companies put effort in driving analysis, specifications and implementations to provide QoS. The real question is, how does QoS fit within heterogeneous networks? In more detail, what is the impact on the performance if information traverses different network portions that use specific QoS paradigms? What happens if the access or another portion of the overall path implements radio or satellite technology? These questions and corresponding possible answers are the core of this book.

The definition of "heterogeneous network" may assume different aspects. Network portions may be managed by different Service Providers, may use different transmission means such as cable, satellite, radio and may implement different protocols such as ATM, IP and MPLS. A network may also be heterogeneous from the point of view of users, who can require different services and have a different availability to pay for them.

Figure 2.1 sketches graphically the mentioned concepts: modern telecommunication networks are essentially composed of different portions and technologies; each single portion may implement a different QoS solution, whose algorithms to satisfy performance requirements may change together with the performance parameters. An example may be an ATM network, where QoS parameters and requirements are completely defined as well as the tools to guarantee them for each specific customer, compared with an IP network supporting the Differentiated Services (DiffServ), widely described and referenced in the reminder of the book, where different customers need to be aggregated within a limited number of flows

QoS over Heterogeneous Networks Mario Marchese

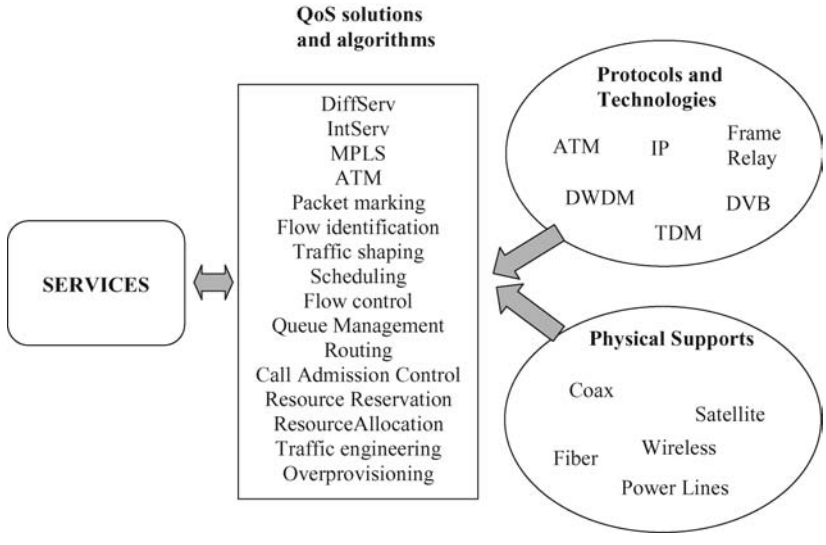


Figure 2.1 Network heterogeneity

for technical reasons. Figure 2.1 includes some examples of protocols and technologies that may be implemented over different physical supports along with a non-exhaustive list of possible QoS solutions and QoS management algorithms. It allows highlighting both the presence of different possible heterogeneities within telecommunication networks and the importance of QoS control algorithms, whose need will be underlined in the following of the book. A more formal relation between services, protocols and physical supports is shown in Figure 2.2. The service, often divided into components such as audio, speech, video and data, is implemented through specific protocols that rely on data bearers acting as physical supports [ETSI-TS-102462]. Actually, data bearers are separated from the upper protocols through a proper formal interface. This important concept will be detailed in Chapter 5.

At this point, it is important to stress the need to have QoS services from the source to the destination, that is directly to the customer. This issue is stressed in Figure 2.3, where a heterogeneous network composed of radio, satellite and cable portions is shown. It is the reference service network for this book.

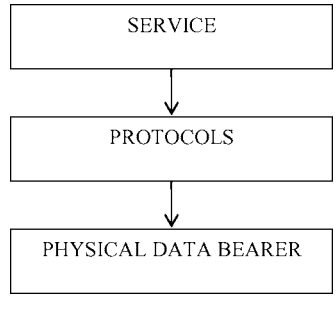


Figure 2.2 Relation between service–protocols–data bearer

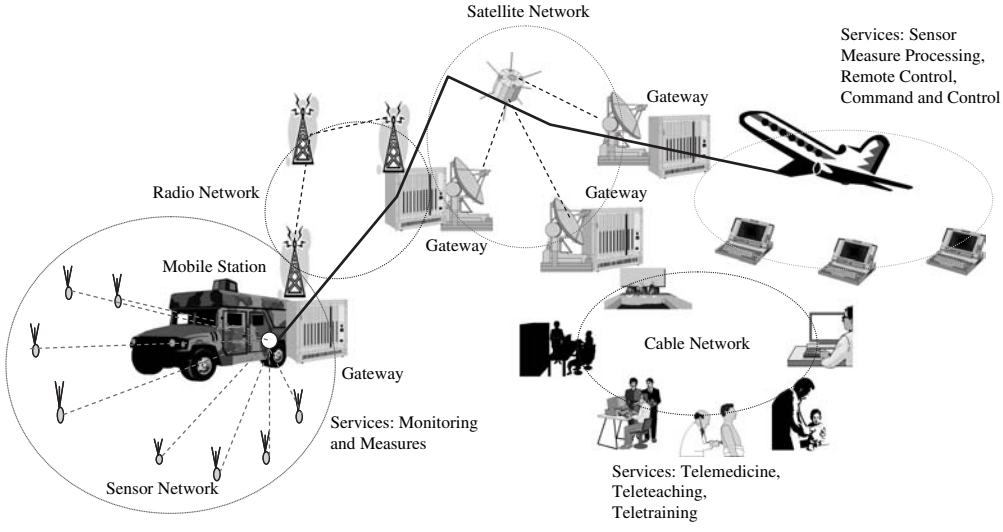


Figure 2.3 End-to-end QoS service through heterogeneous network portions

The line identifies a service that has to be transported end-to-end through the heterogeneous portions composing the network. The service (data acquisition by a sensor network for the analysis of the land) implies the transport of information from the earth to a plane monitoring a portion of the territory through an ad hoc group of radios, a wireless link that connects the ad hoc portion to a satellite backbone and a satellite link to the plane, which contains a suited local network. It is obviously just a challenging example. The service will have a dedicated SLS. Providing a specific service with guaranteed SLS over a heterogeneous network in terms of technology and protocols, where the meteorological conditions and the speed of the plane can also have an important impact, is not simple. The gateways, used in Figure 2.3 for protocol adaptation, will have a role, as should be clearer in the reminder of the book. Actually, any portion of the overall network may deserve a dedicated protocol stack and each single stack needs to be fully involved in the end-to-end QoS provision. In practice, the idea is that the gateways must communicate each other to establish common rules and to exchange control information and data, each single layer will provide a QoS service to upper layers, and end-to-end QoS will spring from the cooperation of each single layer and of each single network portion. These concepts that represent the clue of this book will be explained in detail in Chapter 6 and following chapters, after focusing on main QoS technologies and on available control functions. More formally, the end-to-end chain that connects two end-to-end hosts may be represented as a concatenation of Core Networks (CN) possibly including also border customer networks, commonly indicated as Customer Premise Network (CPN). The interface between CPN and CN is defined as User–Network Interface (UNI) and the interface between two different core networks is defined as Network–Network Interface (NNI). The connection between the defined components is reported in Figure 2.4. The relation between CPN and CN has a general validity but it is integrated in Figure 2.4 by the correspondence with the example network used in Figure 2.3, to give a clear idea of the meaning. The host that originates the information is a special tool (or a group of tools)

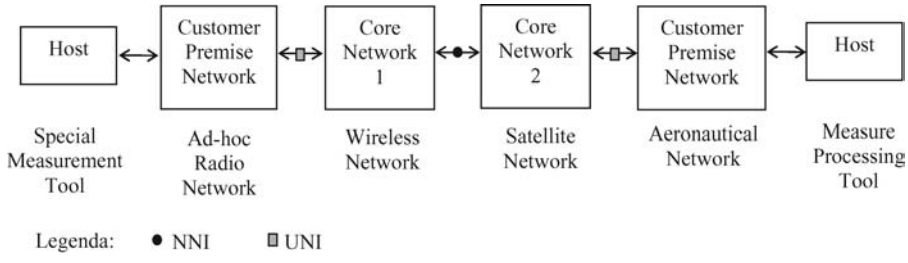


Figure 2.4 Relation between components of end-to-end communication

to take measures in the field. It is connected to the first core network (a wireless portion) through an ad hoc communication (referenced as CPN). Information is then delivered to the satellite network and to the destination through the aeronautical customer network. The end-user is a processing machine suited to elaborate the measure. In more detail, Core Network 2 is a Satellite Network, so the gateways accessing it are two satellite earth terminals, including also bearer service facilities. A simple block diagram is shown in Figure 2.5, where the satellite component is isolated from the rest. Protocols are implemented over the satellite terminals and send data over the satellite bearer.

A so complex network composition perhaps was unthinkable up until few years ago; it now represents the present and the future of telecommunications within the technological framework of the Global Information Infrastructure (GII), where the necessary information may be accessed by the users, anywhere and anytime. Actually,

> Multiple networks composed of different transmission media, such as fiber optic cable, coaxial cable, satellites, radio, and copper wire, will carry a broad range of telecommunications and information services and information technology applications into homes, businesses, schools, and hospitals. These networks will form the basis of evolving national and global information infrastructures, in turn creating a seamless web uniting the world in the emergent Information Age. The result will be a new information marketplace, providing opportunities and challenges for individuals, industry, and governments. [GII]

The need of QoS is outstanding in this environment but its provision is a great challenge for future.

The gateways are a possible way to implement the concept of Network Convergence, where different transmission media, such as satellites, wireless and cables, compose only

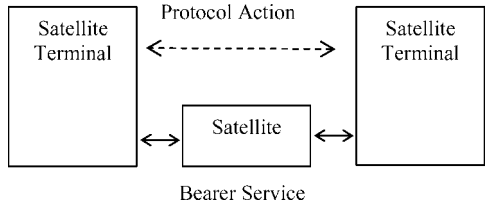


Figure 2.5 Action of satellite terminals

one telecommunication network and operate together transparently to the final users. The gateways should erase the artificial line between different technologies and protocols creating a global information structure for the users themselves.

Chapters 4 and 5 will be dedicated to the description of the main technologies providing QoS and to the importance of control functions and tools. In practice, each single element appearing in Figure 2.1 will be investigated.

2.2 The Concept of Autonomous Systems

In the Internet environment, an Autonomous System (AS) is a group of networks and routers that belong to the authority of a single administration [Farouzan]. An image of AS is reported in Figure 2.6. The entities R and R', indexed with the corresponding AS number, are the ingress/egress devices (called routers in the Internet scenario) to ASs. They act at the IP layer in the TCP/IP stack, which is explained at the beginning of Chapter 3.

Autonomous systems are structured into three categories [Farouzan]:

1. Stub AS, which has only one link to another AS. The traffic traversing a stub AS is either created or terminated in the stub AS. It means that there is no traffic, which traverses the stub AS. For example, the AS1 shown in Figure 2.5 is a stub AS, as well as AS4.
2. Multi-homed AS, which has more than one link to other ASs, but there is no transient traffic, as in the previous case. It may also correspond to an administrative prohibition to traverse the AS.
3. Transit AS, which is a multi-homed AS that admits transit traffic also. An example is represented by backbones.

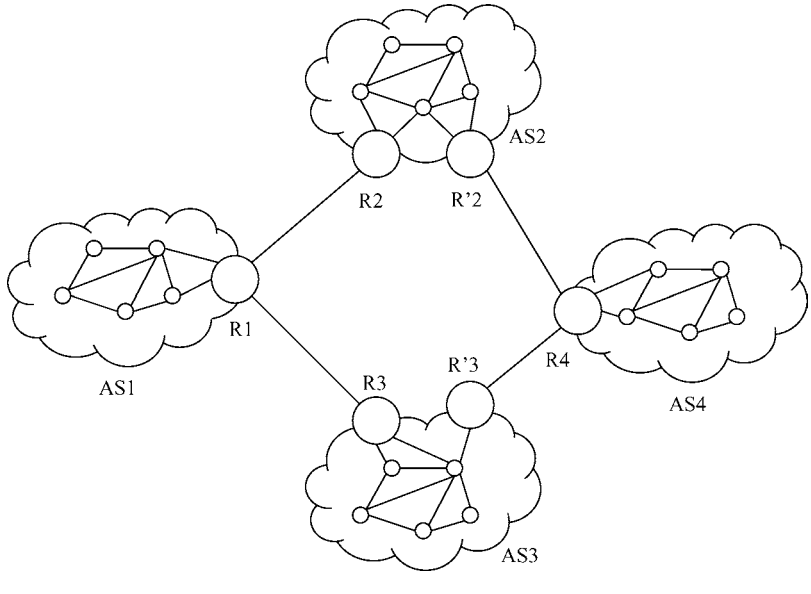


Figure 2.6 Network of autonomous systems

Even if the concept of "autonomous system" is used to classify Internet domains, the correspondence between the definition and the structure of Figure 2.2 is immediate. Each single network portion may be considered an AS. It implies a broader interpretation of AS, not only linked to the Internet but extended to other technologies. In this view, an AS is a group of networks and devices under the authority of a single administration. It totally matches the idea of heterogeneous network and authorizes to consider a heterogenous network as a concatenation of ASs. The concept of AS can be considered, through this extension, equivalent to the concept of network portion. The only subtle difference is that "AS" with respect to "network portion" evokes the administrative management, while "network portion" refers only to technical details. In other words, two different network portions imply the use of different technologies (not necessarily managed by different authorities), while two different ASs imply the existence of two different administrative domains (perhaps not necessarily implementing diverse technologies). Anyway, they are considered to have the same meaning in the reminder of this book. In this broader scenario, the ingress/egress routers of Figure 2.6 may be considered, more generically, gateways as in Figure 2.2. The gateways architecture will be one of the main topics of the book. It will be detailed in Chapter 6.

3

QoS-oriented Technologies

In the following, a list, which should include the main technologies [ChaoGuo02] available in the market to provide QoS, is briefly summarized, with particular attention to the capability of marking traffic, which is the basic function to allow QoS service provision. Before doing it, it is important to remind a fundamental structure to connect remote systems: the layered architecture. It is helpful to better understand the action of each specific technology (ATM, MPLS, IPv4 and IPv6) whose description will follow.

3.1 Layered Architecture and Remote Systems Connection Protocol Stack

The functionalities that must be performed to complete the communication have been structured into functional layers to simplify the communication among remote systems. Figure 3.1 shows the general functional model introduced by International Organization for Standardization (ISO) [ISO94] and known as Open Systems Interconnection (OSI). Figure 3.2 shows the TCP/IP stack, much used in modern telecommunication world, and also taken as reference in this book.

A sub-system is structured into different entities operating in cascade. Two adjacent layers of a sub-system communicate each other through operation tools called "primitives". In practice, each layer asks a service to the layer below through a specific primitive. A primitive is the service offered. Each entity of a remote sub-system communicates with the peer entity within a remote sub-system through a set of rules, called "protocols". Figure 3.3 is aimed at focusing on these points. The set of bits that define the information within one entity (e.g. X-entity) is called "X-SDU" (Service Data Unit of the X-entity). The control information added by the X-entity is defined as X-PCI (Protocol Control Information) and added to the X-SDU either as header or as trailer, or both. The new set of bits composed of X-PCI and X-SDU is called "X-PDU" (Protocol Data Unit of the X-entity) and is transferred to the (X–1)-entity through the primitives made available by the (X–1)-entity itself. The set of such primitives are defined as (X–1)-primitives. X-SDU is rebuilt at the next peer entity met in the end-to-end path, shown in the right part of the figure.

QoS over Heterogeneous Networks Mario Marchese

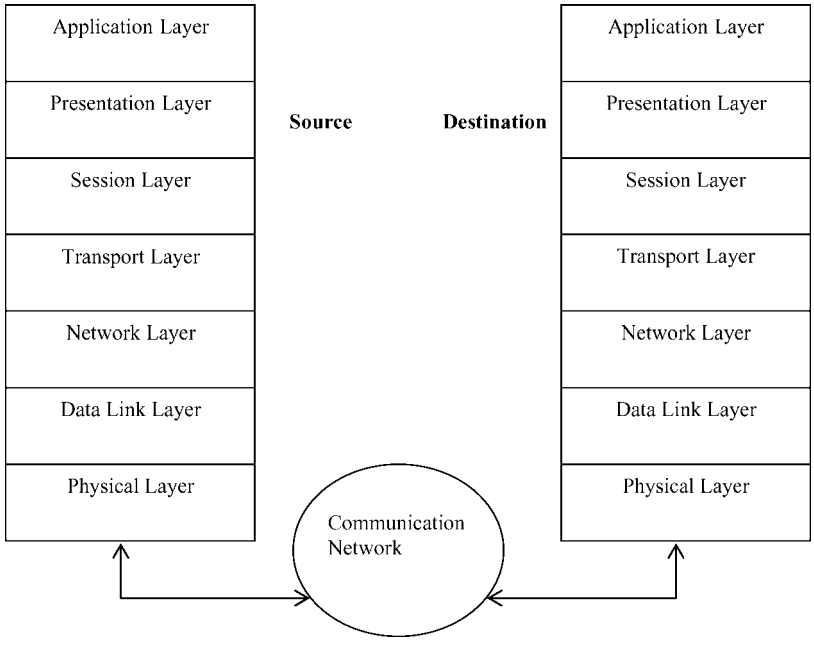


Figure 3.1 OSI layered architecture

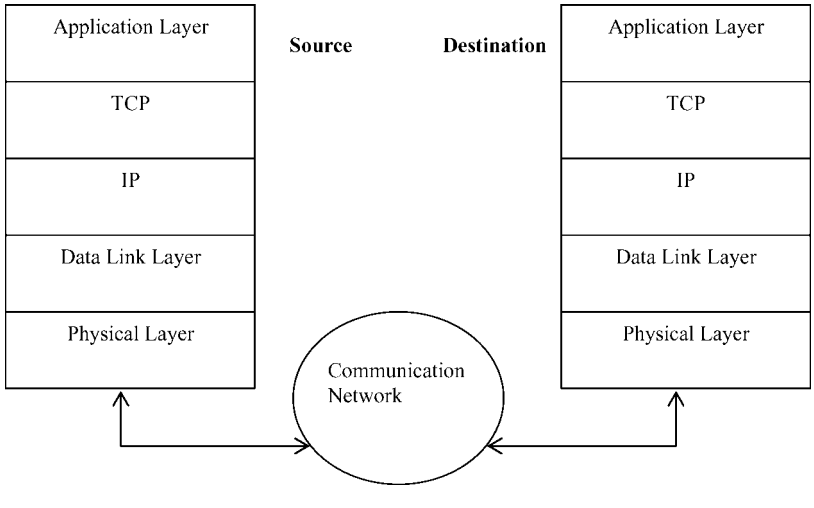


Figure 3.2 TCP/IP layered architecture

The OSI architecture standardizes four basic types of primitives at each level: request, confirm, indication and response. They are used to define the information exchange between the service user at one level and the service provider at the lower layer. A complete description may be found in [Schwartz88]. This reference will be heavily used in the reminder of this paragraph. Not all four types are necessarily used within each layer and in each phase of the communication between peer layers. The basic primitives act within two simple

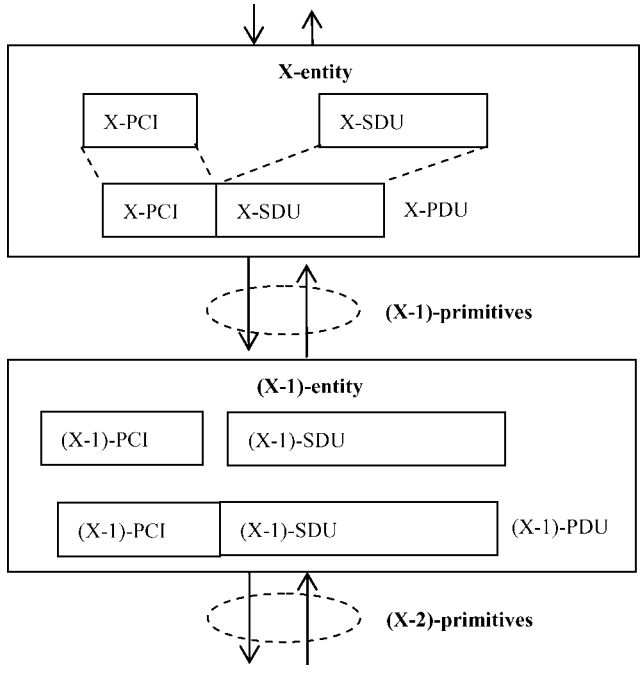


Figure 3.3 Communication between entities within a layered architecture

layered architectures connecting two generic remote systems, called "system A and B" here, where entities X and (X–1) act. A given layer of the system A stack (e.g. the X-entity) issues the "request" primitive to invoke a service offered by the (X–1)-layer below. The request primitive activates an (X–1)-layer message sent to the (X–1)-layer of the system B through an (X–1)-PDU. The receipt of the PDU at the (X–1)-layer of the system B may generate a message internal to system B. It is sent from the (X–1)-layer to the X-layer through the proper primitive called "indication". The "response" primitive is issued by the X-layer (of the system B, referring to Figure 3.4) as a response to the indication primitive. In other words, the X-layer sends an X-PDU to the layer below through the primitive response. After receiving it, the (X–1)-layer builds an (X–1)-PDU that is transmitted to the peer layer at system A. Its reception generates the issuance of a new primitive (called "confirm") by the system A through which the (X–1)-PDU is transported to the X-layer. It completes the "request" procedure invoked by the system A. Each primitive generally carries a set of operation parameters such as entity identifiers, sequence numbers, flags and QoS requirements. A practical example of this process, always taken from [Schwartz88], is represented by the connection of two remote systems implying the following three phases: connection establishment, data transfer and connection release, acting at (X–1)-layer. In practice, the example concerns the communication of two systems that firstly agree about exchanging data, secondly exchange data and eventually communicate that the data exchange is concluded. The systems are again identified as system A and system B. The three phases corresponds to three categories of the four basic primitives. They are identified as CONNECT, DATA and DISCONNECT. The set of primitives defined for the mentioned connection of layered architecture is shown in Table 3.1 together with the service offered by each primitive.

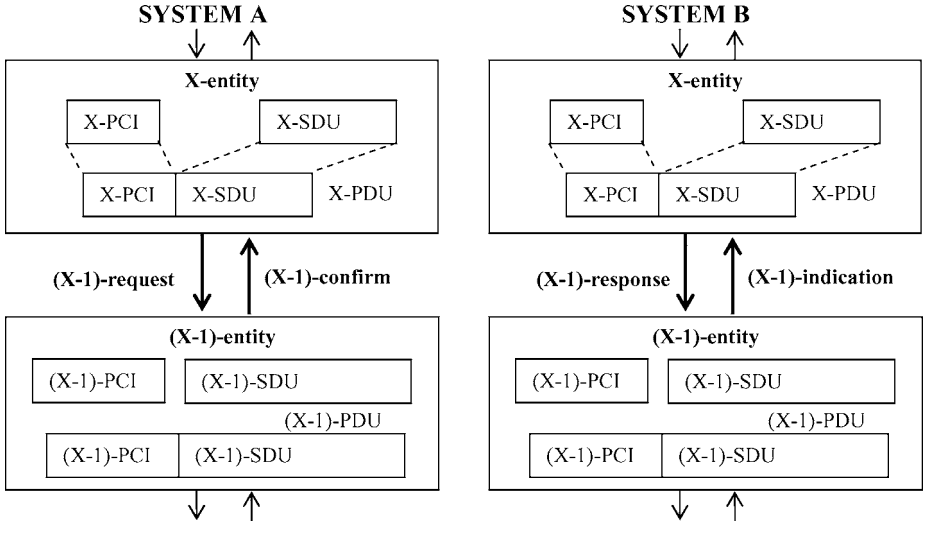


Figure 3.4 Layered architecture basic primitives

Table 3.1 Categories of the basic set of primitives: CONNECT, DATA and DISCONNECT

Primitives	Service offered
CONNECT.request	System A requires to establish a logical connection with system B
CONNECT.indication	System B receives system A logical connection request
CONNECT.response	System B communicates to system A its decision (agreement/disagreement) about the logical connection request
CONNECT.confirm	System A receives system B decision (agreement/disagreement) about the connection logical connection request
DATA.request	System A sends data to system B
DATA.indication	System B receives data from system A
DATA.response	System B sends data acknowledgement to system A
DATA.confirm	System A receives data acknowledgement from system B
DISCONNECT.request	System A requires to release the logical connection established with system B
DISCONNECT.indication	System B receives system A request to release the established logical connection
DISCONNECT.response	System B communicates to system A its decision (agreement/disagreement) about the request to release the established logical connection
DISCONNECT.confirm	System A receives system B decision (agreement/disagreement) about the request to release the established logical connection

Figure 3.5 (divided into three parts: a, b and c, which should be read in sequence) contains a picture of the mentioned primitives use. A logical connection at the (X–1)-layer is established between the two systems before beginning the data transfer (Figure 3.5a). The X-layer of the system A issues a CONNECT.request primitive. The (X–1)-layer below begins establishing the logical connection and sends a (X–1)-layer PDU to its peer layer at remote system B, which, after receiving it, issues a CONNECT.indication primitive to the X-layer above. The X-layer in system B answers through the CONNECT.response primitive that causes the (X–1)-layer to transmit an acknowledgement (X–1)-PDU, received by the peer layer in system A, which issues a CONNECT.confirm primitive. It completes the establishment of the logical connection between the two remote systems and data transmission can begin. It follows the same scheme, which is shown in Figure 3.5b.

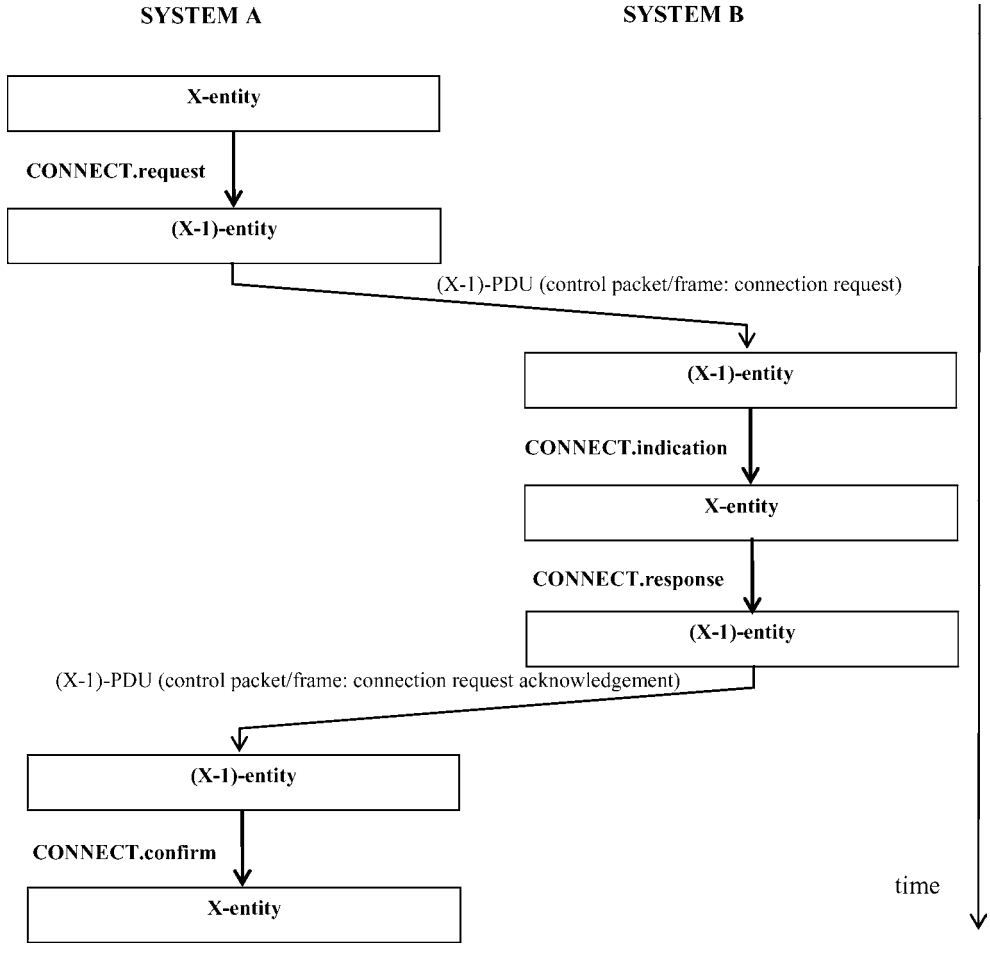


Figure 3.5a Layered architecture, use of the basic primitives and connection establishment

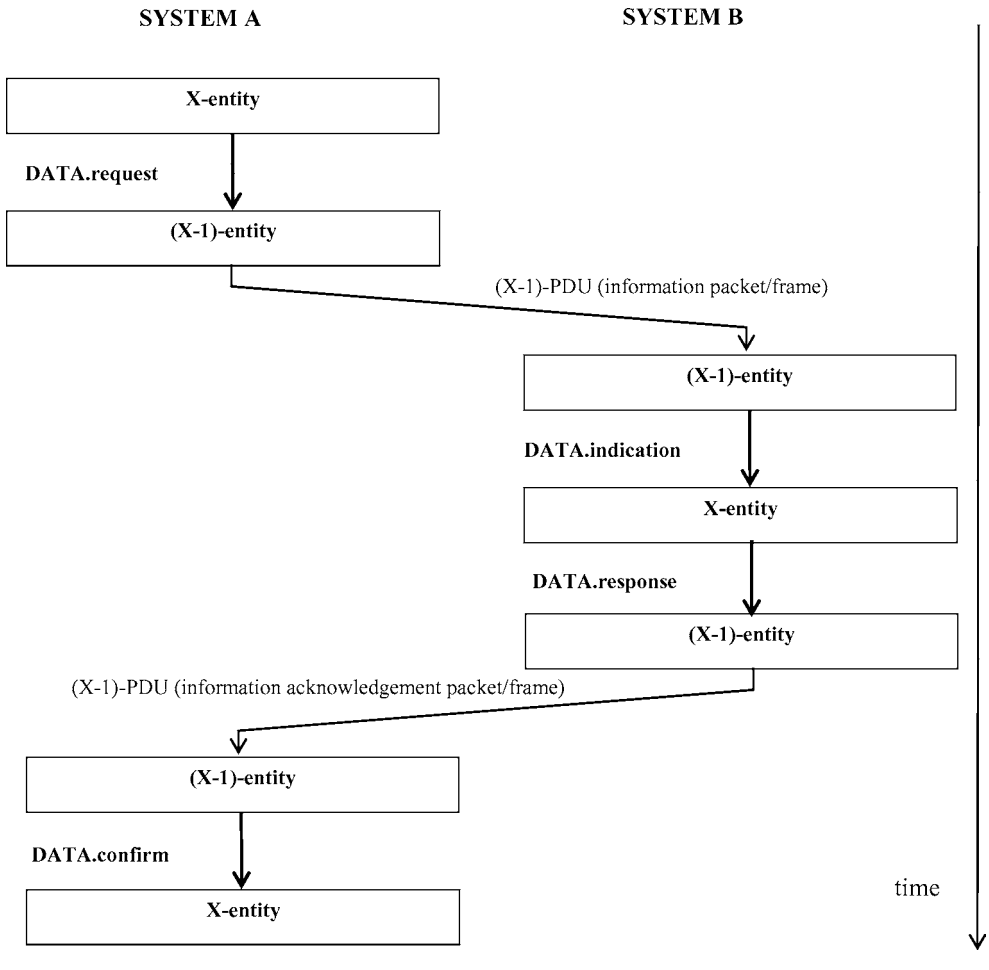


Figure 3.5b Layered architecture, use of the basic primitives and data transmission

Information packets are transmitted through DATA.request primitives carrying the proper data PDUs (the data received from the X-layer in addition to the X-layer PCI), which arrive at the remote system through DATA.indication primitives. Information is acknowledged (if needed) through DATA.response primitives sent at system B and through DATA.confirm primitives received at system A. DISCONNECT.request, indication, response and confirm primitives are aimed at releasing the logical connection at (X–1)-layer. DISCONNECT use is shown in Figure 3.5c. This description about the primitives and the examples about their use will be very helpful in Chapter 8, related to vertical QoS mapping.

Figure 3.6 takes the TCP/IP stack as an example to show the data flow through two remote systems by using an IP switching (e.g. a router). At the top of the suite, there is the Application Layer whose PDU is identified as “Application Data”. It is transmitted via

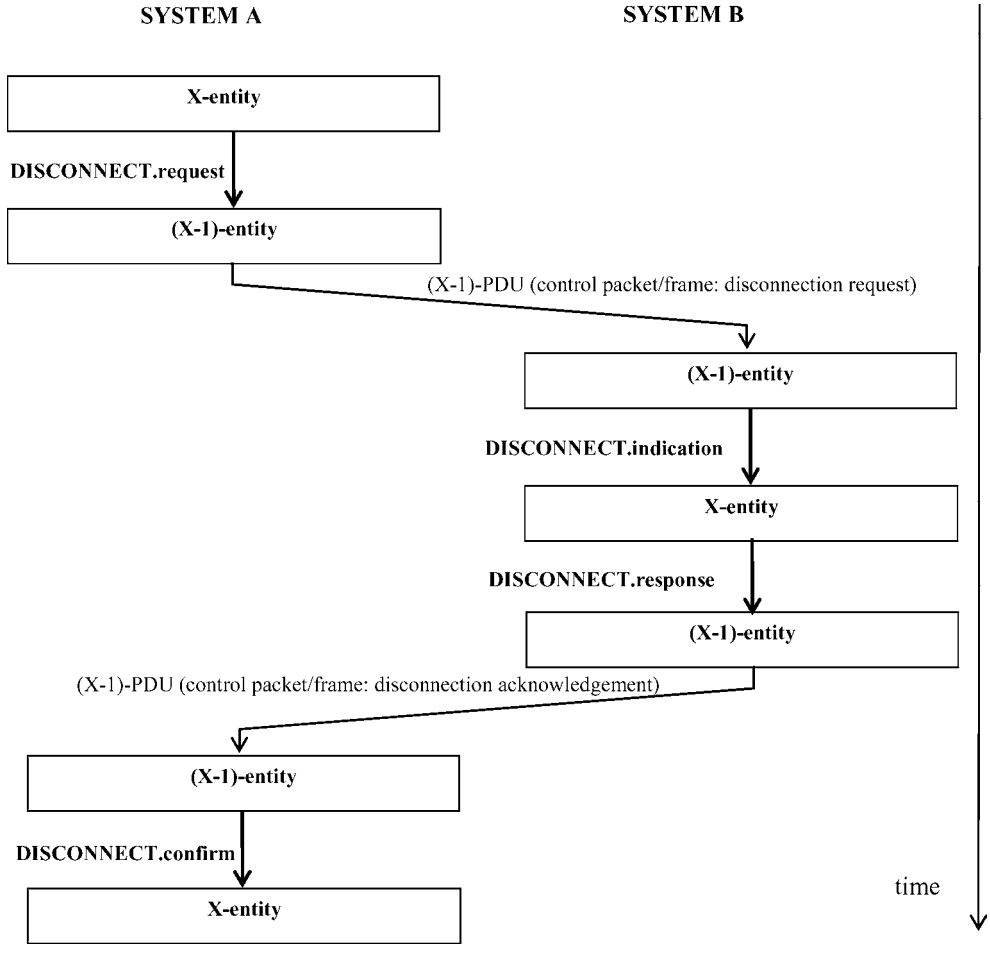


Figure 3.5c Layered architecture, use of the basic primitives and connection release

proper primitives to the Transport Layer whose SDU is the "Application Data" itself. The Transport Layer adds its PCI ("TCP Header") and forms the Transport Layer-PDU, called "TCP datagram", which is transmitted to the IP Layer. It adds the IP Header (i.e. the IP Layer-PCI) and composes the IP packet (i.e. the IP Layer-PDU). The process continues down to the Physical Layer. The original information is rebuilt at each peer layer along the source–destination path.

Entering the details of each layer function and protocol is not within the scope of this book. The aim here is to focus on the role of the functional architecture and on its general meaning: each layer implements a specific protocol and, to reach the aim, adds some information in the form of header and/or trailer. These basic concepts are essential for the comprehension of the concepts contained in the reminder of the book.

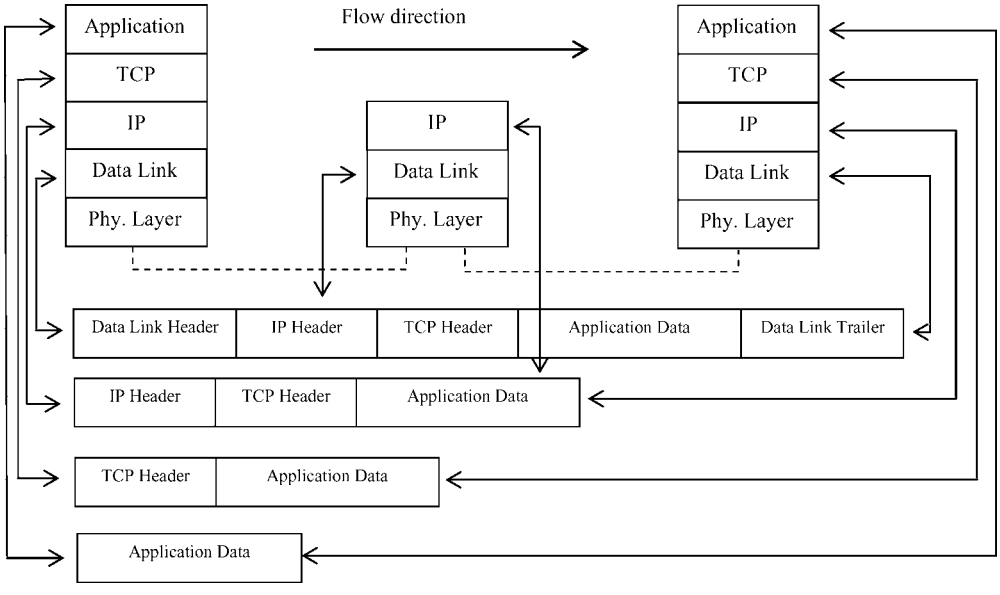


Figure 3.6 Communication between peer entities within a TCP/IP layered architecture: Data flow

3.2 ATM

ATM is a connection-oriented solution. Information is structured in fixed length packets, called "cells", of 53 bytes (5 for header and 48 for information). ATM identifies and assures a single user flow or an aggregate of flows by using two fields contained in each cell header and defined in the ITU-T Recommendation I.113 [ITU-T-I.113]: Virtual Channel (VC) and Virtual Path (VP).

It has been built to guarantee QoS and uses call admission and congestion control schemes properly designed for the aim. The scientific literature of these during the last 15 years is very rich concerning dynamic resource allocation and reservation schemes that use statistical multiplexing gain without penalizing users.

Traffic coming from higher layer is adapted to the ATM environment, whose basic elements are cells, through the AAL (ATM Adaptation Layer). Its composition depends on the higher layer traffic: AAL1 is thought for CBR traffic, AAL2 for multimedia traffic, AAL3/4 for more demanding traffic and AAL5 for data traffic, even if AAL5 is practically used for any type of traffic. The common element for AAL layers is represented by the action of segmentation and reassembly (SAR), where, at the entrance of the ATM network, the cells are generated by segmenting higher layer packets and, at the exit of the ATM network, higher layer packets are composed again by assembling ATM cells.

ATM strength points are listed in Table 3.2, together with possible drawbacks. Many issues have been taken from [Wright01].

The great advantage of ATM is hidden in the capacity of traffic flow identification together with the overall structure (including signalling), completely oriented to QoS provision. ATM has been built to provide QoS and has both the power to identify a single customer deserving a special attention and the flexibility to aggregate traffics with the same performance

Table 3.2 ATM strength points and drawbacks concerning QoS

Strength points	Drawbacks
QoS ATM has been designed for QoS; QoS managing capacity is standardized and checked also in multi-vendor commercial applications; ATM allows requesting quantitative QoS intrinsic parameters, also for each single user	Complexity and overhead for AAL
Diffusion Major carriers and Internet providers use ATM backbones	Recently, ATM use is limited by the increase of IP. It may create compatibility problems in the future
Network Management ATM has several tools, already industrially tested, to manage the network. Monitoring and fault isolation tools are standardized	
Connection oriented ATM can guarantee dedicated bandwidth pipes both between users and network nodes. Bandwidth pipes can be used, priced and sold	
Adaptation options Users can choose among different adaptation layers. For example, AAL1, AAL2 and AAL5 can carry voice in dependence of the customer needs	
Signalling Signalling options are QoS oriented	
Fixed and small dimension cells It is helpful both for statistical multiplexing in switching elements and for inserting a proper error control over errored channels. Moreover, it is important in low bit rate trunks for voice applications because it is simpler to fill the cells avoiding delays in case of long datagrams	

requirements. It is possible because ATM uses two identifiers for a specific flow: VC and VP, contained in the ATM header, shown in Figure 3.7. The meaning of each specific field may be found in specific documents but most important fields are briefly summarized in the following, highlighted by a bold line. The ATM header is slightly different if the cell is generated at the interface between user and network (UNI) or within the network itself (NNI) but the concept is the same: there is a great amount of bits (24 at UNI) to identify a single flow belonging to a customer who has requested a specific SLA/SLS. Being available as a single label (VCI-VPI), it is also possible to aggregate traffic with the same request of SLS. This characteristic, together with the availability of suitable signalling schemes (often designed by single producers), makes this technology so powerful for QoS guarantee. Actually, in early 1990s, ATM was standardized and appeared as the only possible technology for Integrated Service Broadband Networks (ISBNs), which were thought as networks conveying any type of traffic (voice, video and data) together. They were the parents of modern GII. There was the attempt of using ATM for any possible problem that could arise. It brought about a complexity in the layers that have the responsibility of adapting other protocols to ATM (AAL). Then the technology and market evolution took a different direction, imposing the simple TCP/IP suite as the reference for network protocols, but even if now ATM is not the future of telecommunications, it does not mean that the flow identification feature

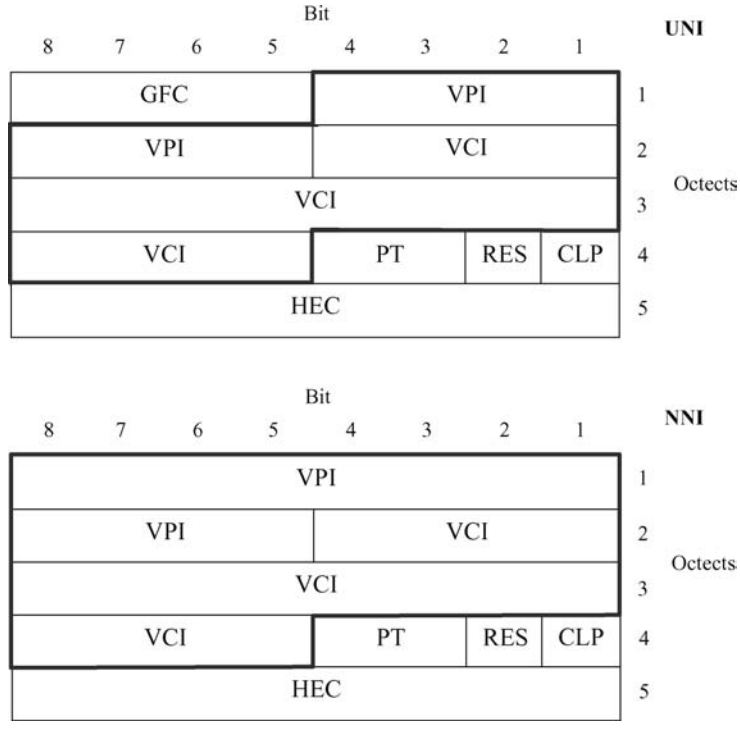


Figure 3.7 ATM header (at UNI and NNI interfaces) and flow identifiers

as well as all the controls embedded within ATM implementation must be trashed. They can (should) be used as technical reference for QoS provision independently of the specific technology used. This concept will be further examined of Chapter 4.

The interaction between users in a Full-ATM network is shown in Figure 3.8. The ATM network is composed of just one ATM switch for the sake of simplicity.

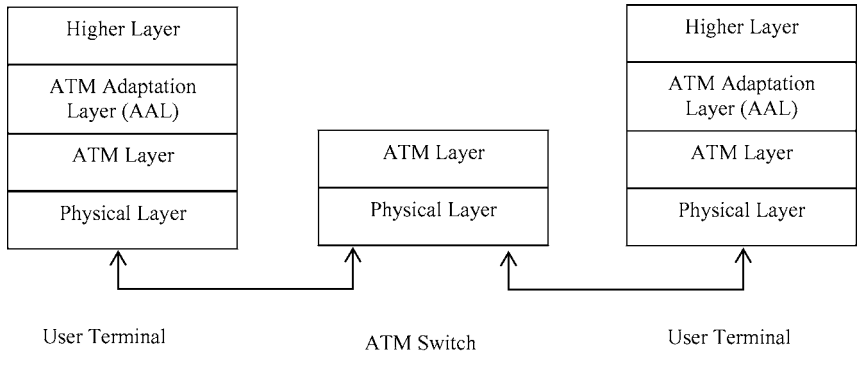


Figure 3.8 User interaction by ATM switch

Each single connection (user flow) is identified by the concatenation of VPI and VCI and is switched through a translation table located within the ATM switch. The values of VCI and VPI are used, together with the incoming physical link to access the translation table. The access result is the outgoing link and a new value of VCI/VPI. The change of identifier allows a full reuse of the available VCI/VPI space.

In more detail [Handel91], ATM provides two hierarchical levels: VC and VP level. They are defined in ITU-T Recommendation I.113 [ITU-T-I.113]:

Virtual Channel : "A concept used to describe unidirectional transport of ATM cells associated by a common unique identifier value." The identifier is called "Virtual Channel Identifier" (VCI).

Virtual Path : "A concept used to describe unidirectional transport of cells belonging to virtual channels and that are associated by a common unique identifier value." The identifier is called "Virtual Path Identifier" (VPI).

Virtual Paths allow grouping different VCs. The relation between VCs, VPs and Transmission Path is shown in Figure 3.9.

Reference [ITU-T-I.113] defines Virtual Channel Link (VCL) as "A means of unidirectional transport of ATM cells between a point where a VCI value is assigned and the point where that value is removed." While a Virtual Path Link (VPL) is a means of unidirectional transport of ATM cells between a point where a VPI value is assigned and the point where that value is removed.

Each connection is identified over a VCL by the identifier composed of the couple VCI/VPI that has only a local meaning. ATM switches are divided into VPI switches (where only the VPI is switched leaving the VCI within the VPI totally untouched, that is only VPL is terminated, so simplifying switching technology) and VCI switches (real ATM switches) where both values are switched (both VCL and VPL are terminated).

A single user flow (a single end-to-end connection) is identified by a concatenation of VCLs and VPLs. In more detail, a concatenation of VCLs is defined as Virtual Channel Connection (VCC) and a concatenation of VCLs is called "Virtual Path Connection" (VPC). In practice, even if the distinction between VC and VP is very important because it increases

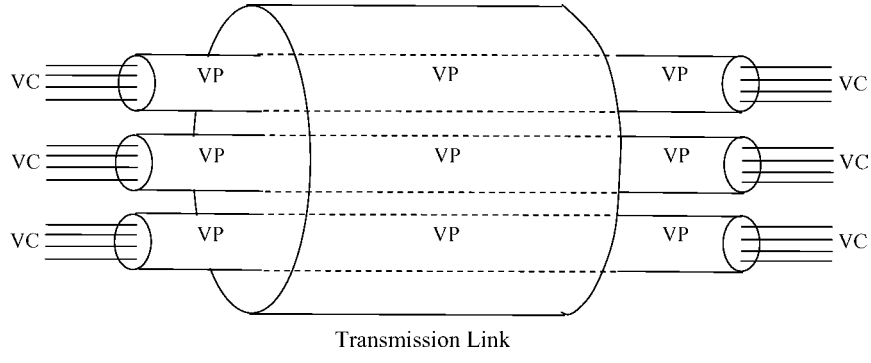
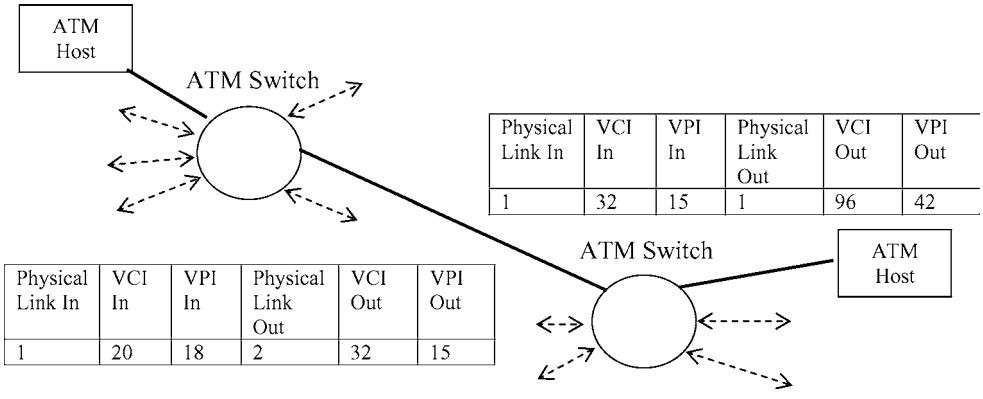


Figure 3.9 Virtual Channels, Virtual Paths and Transmission Link



Physical Link In	VCI In	VPI In	Physical Link Out	VCI Out	VPI Out
1	32	15	1	96	42

Physical Link In	VCI In	VPI In	Physical Link Out	VCI Out	VPI Out
1	20	18	2	32	15

Figure 3.10 ATM switching

the flexibility of QoS provision, it is also possible to think of a single end-to-end connection identified by the concatenation of the couple VPI/VCI values.

Figure 3.10 shows an end-to-end connection over a full ATM network. It is transported through two ATM switches. The first switch uses the incoming physical link interface 1 and the outgoing physical link interface 2, while the second switch uses the incoming physical link interface 1 and the outgoing physical link interface 1. The overall end-to-end connection over the specified physical interface is identified as the concatenation of the following VCI/VPI couples: [20/18; 32/15; 96/42]. In practice, the concatenation of 3 strings of bits in the ATM cell header identifies the traffic of a single user flow. Figure 3.11 shows the data

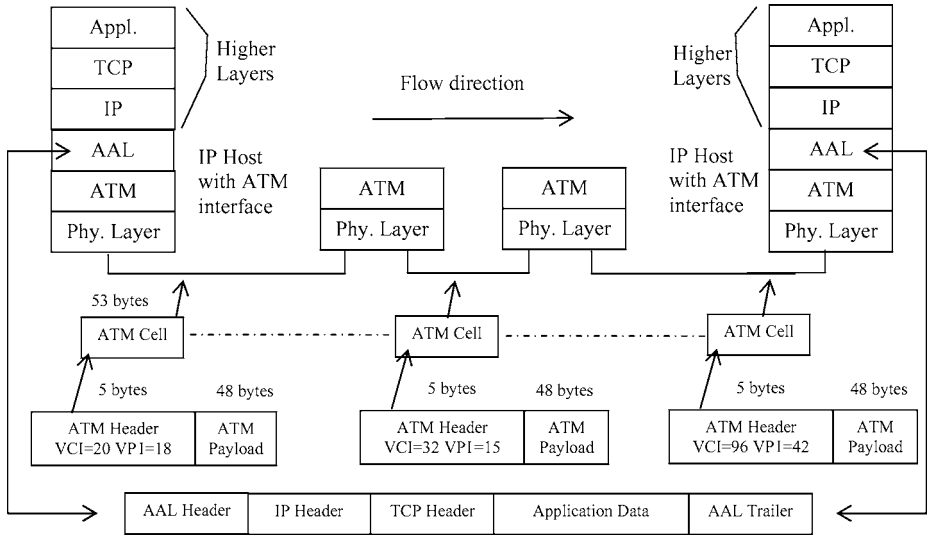


Figure 3.11 Full ATM network: Data flow and switching

flow and the switching action through peer entities as in Figure 3.6 having an IP host both at the source and at the destination. Being the aim of the figure to show a full ATM network, the IP hosts are considered equipped with an ATM interface. Switching is guided by the tables in Figure 3.10. If the specific connection should be removed, it can be identified in any point of the network through its identifiers.

3.3 MPLS

MPLS derives from the convergence between the IP world, which involves open standards and simplicity, and the ATM world, which is completely oriented to the QoS and uses traffic control techniques, as well as QoS aware routing. MPLS would like to "summarize" the best of both technologies. A label of 20 bits identifies the traffic. The label switching mechanism should be associated with control modules that allow guaranteeing a fixed level of quality. In principle, MPLS offers the same capabilities to mark flows as ATM. Figure 3.12 shows the MPLS label format, which is composed of 32 bits structured in the following elements: the already mentioned Label Value; the field Exp, dedicated to future extensions; the bit S, aimed at indicating the presence of more labels; and the Time To Live (TTL).

Entrance and exit of MPLS domains are governed by Label Edge Routers (LERs) that generate and apply the labels when information enters the domain and remove them at the exit. The basic technology inside the domain is represented by Label Switch Routers (LSRs), which switch the traffic in dependence of a specific path, called "Label Switched Path" (LSP), and associated with the MPLS label. An example of MPLS network is shown in Figure 3.13, as well as the path (identified by the dashed line) followed by traffic

Label Value – 20 bits	Exp – 3 bits	S – 1 bit	TTL – 8 bits

Figure 3.12 MPLS label format

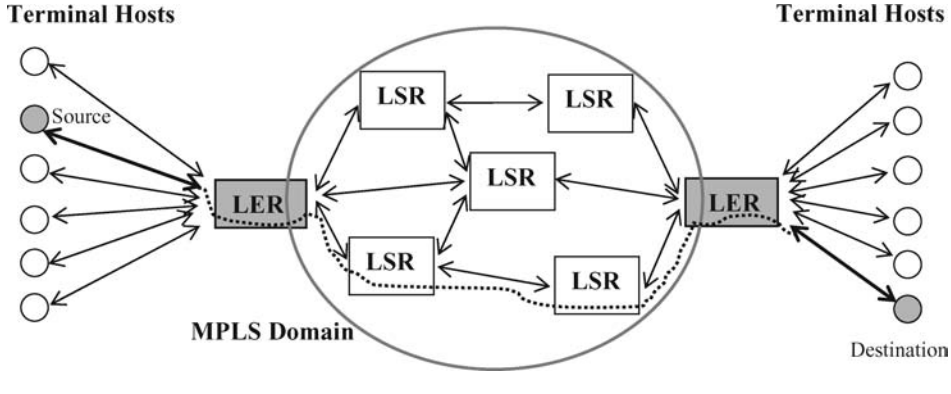


Figure 3.13 MPLS network architecture

flowing from the source to the destination (and vice versa). The LSP defines the sequence of nodes where the traffic of a connection flows within the MPLS domain. The definition is performed through QoS-based traffic engineering techniques [RFC2386] aimed, for instance, at minimizing the number of hops, meeting bandwidth requirements, supporting precise performance requirements, bypassing potential points of congestion, directing traffic away from the default path selected or simply forcing traffic across certain links or nodes in the network. MPLS has a great potential to but it has a great potential to unify and implement QoS-based interworking among network portions implementing different technologies.

LER and LSR are MPLS-aware routers. The introduction of MPLS is aimed at creating a QoS-guaranteed network suited also for the IP packet format. In other words, MPLS-enabled technology should implement the ATM QoS in a smooth way for other technologies (e.g. IP packets, which should only add a small label). Some literature embeds also ATM in the MPLS world. In this context, the VCI and VPI fields of the ATM header represent the MPLS Label. A similar observation may be done for the Data Link Connection Identifier (DLCI) field of the Frame Relay environment.

MPLS, similarly as ATM, allows identifying a single flow from entrance LER to exit LER. Having in mind two IP hosts as source and destination, an MPLS-switched network involving two LERs and just one LSR, for the sake of simplicity, is summarized in Figure 3.14. In strict dependence on the information embedded in the IP packet and of control information, the first LER adds an MPLS Label (63 in this case) and forwards the packet through the MPLS network. LSR switches the MPLS packet to have a full reuse of the labels themselves (as in ATM) and forwards the packet either to the next LSR or, as in this case, to the outgoing LER which drops the label, restores the original IP packet and forwards it to the destination. LSR switching action, in the shown case, takes all the packets identified by the MPLS label 63 coming from the incoming physical interface 1 and switches them to the outgoing physical interface 1 marking them by using the MPLS label 19. The data flow is shown in Figure 3.15 by using Ethernet as Layer 2 between LSR.

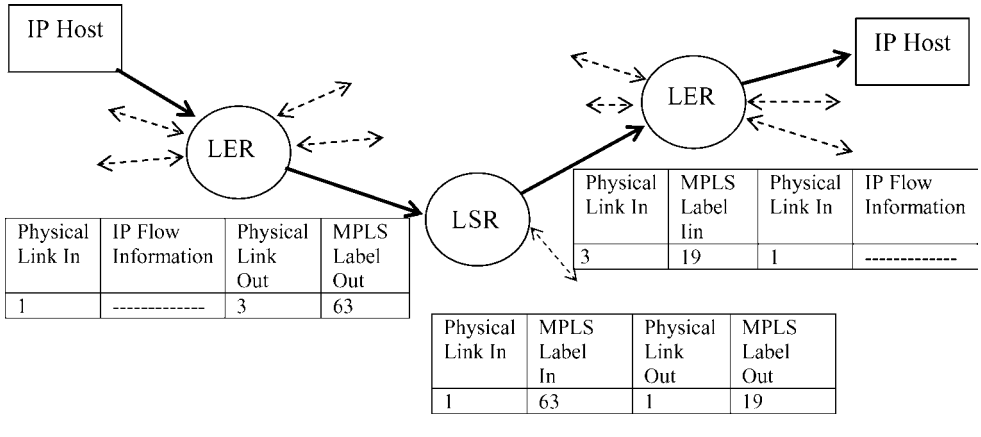


Figure 3.14 MPLS switching

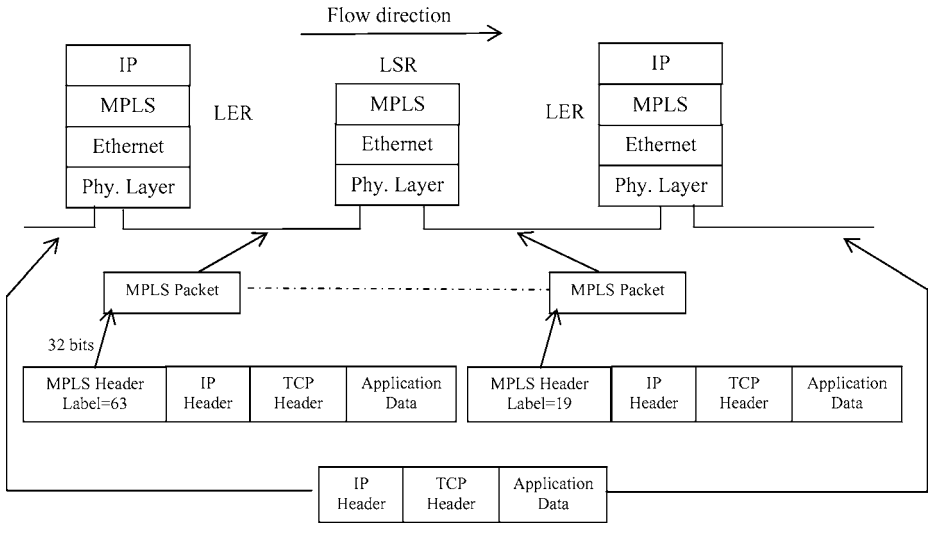


Figure 3.15 MPLS network: Data flow and switching

3.4 QoS-IPv4

Native IP is connectionless and offers best-effort services. The service received by a user depends on the network load. Managing queues within routers is essentially through First In First Out (FIFO) order. Concerning the tools to identify traffic flows, IP needs to be differentiated between version 4 and version 6 (to which the next session is dedicated).

IPv4 offers two ways to mark traffic:

1. A vector composed of the following fields: "IP source address", "IP destination address", "Protocol", all contained in the IPv4 header; "TCP/UDP source port" and "TCP/UDP destination port", all contained in the TCP/UDP header. The option is highlighted in Figure 3.16, which shows the IPv4 header. Each single flow may be identified in this way but, differently from ATM, there is no single label available to identify a group of flows with the same SLS requirements. It implies that, if a control mechanism uses this method to identify a flow, it is not scalable.
2. ToS field (8 bits in the IPv4 header), whose first 6 bits define the Differentiated Services CodePoint (DSCP) field. Packets with the same DSCP identifier need to be treated coherently by each router. The option is focused in Figure 3.17. In practice, having only 8 bits available (actually, only 6 are used) to mark traffic, there is no possibility to identify a single user flow. Groups of flows with possibly similar SLS have to be aggregated together in a traffic class.

Flow identification is only a starting point for QoS guarantee. Two paradigms have been proposed to match the market QoS requests in IP networks: Integrated Services and Differentiated Services.

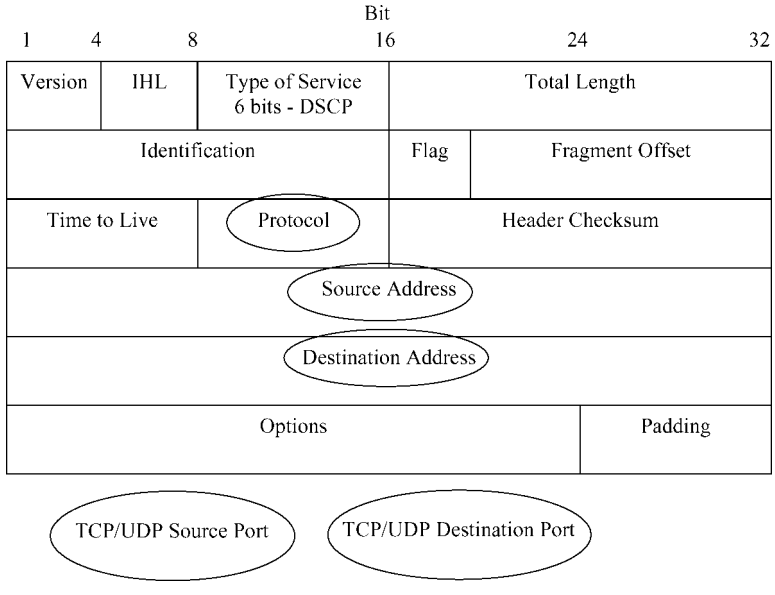


Figure 3.16 IPv4 packet header and single flow identifiers

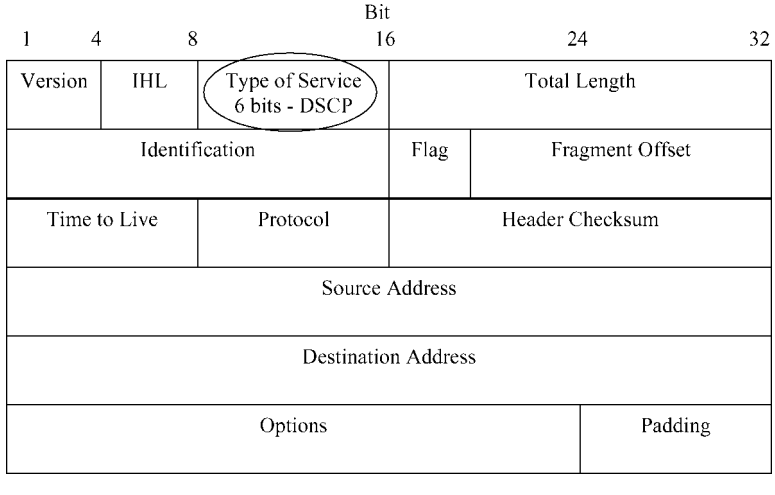


Figure 3.17 IPv4 packet header and group of flows identifier

3.4.1 Integrated Services

The Integrated Services [ChaoGuo02, RFC2205] approach is based on the concept of flow defined as a packet stream that requires a specified QoS level and it is identified by the vector "IP source address, IP destination address, Protocol, TCP/UDP source port, TCP/UDP destination port", as said above. QoS can be reached by an appropriate tuning of different blocks: resource reservation, admission control, packet scheduling and buffer management.

The state of the different incoming flows must be maintained in the routers. It is very different from the best-effort approach provided by the Internet. Information about flows must be periodically updated and a specific resource reservation signalling system (RSVP [RFC2205, RFC2210]) is used for this aim. Details about it will be given in Chapter 7. The IntServ approach defines two service classes: Guaranteed Service (GS) [RFC2212] and Controlled-Load Service (CLS) [RFC2211]. GS is a service characterized by a perfectly reliable upper bound on end-to-end packet delay. As clearly stated in [RFC2212] GS assures that IP packets

- arrive within a specific maximum delivery time;
- are not discarded due to queue overflow,

if the traffic flow respects the declared traffic parameters. The IntServ approach is ideal for applications where delay constraints are mandatory. The service does not control the minimum and average delay of IP packets but only the maximum delay. GS implies the presence of Admission Control. It should result clearer from the following chapters related to QoS control and QoS architectures. CLS has a much looser specification. It is supposed to offer a service that is comparable to best-effort service in a "lightly loaded" network, but it uses admission control and bandwidth allocation to assure that CLS is reached also within an overloaded network. A "lightly loaded" network is assumed as a network that guarantees

- high probability (close to $1 - P_{\text{error rate}}$, where $P_{\text{error rate}}$ is the packet error rate of the transmission medium) that transmitted packets are successfully delivered to the destination;
- end-to-end transit delay measured for high percentage of packets not exceeding the minimum end-to-end transit delay, computed by considering transmission delays and fixed processing delay within network elements.

The drawback of the IntServ approach is scalability. The IntServ needs to detect each single flow and both packet scheduling and buffer management act on per-flow basis. Although IntServ is ATM-like, it has no tools provided by ATM because it is derived from an intrinsically best-effort and connectionless technology. Cost and complexity of the control system increase with the number of flows because there is no single label to identify a flow or a group of flows with similar performance requirements (SLS) and traffic features. Moreover, RSVP signalling does not have any release message, so a periodic refresh message is needed to confirm the resource (bandwidth) request; a connection is dropped only if the refresh message has not been received for some time. The drawbacks are as follows:

- even if the IP flow is terminated, the resources are not immediately released;
- refreshing signalling is bandwidth consuming.

Even if the IntServ approach has clear drawbacks, its features allow assuring a specific QoS to each flow. It cannot be ignored. RSVP is a good example of QoS-provisioning signalling and deserves attention. It will be taken as reference in the signalling chapter.

3.4.2 Differentiated Services

The Differentiated Services (DS or Diffserv) [RFC2475] have been proposed to cope with the scalability problem faced by Integrated Services. The solution uses the DSCP field of

the IP packet header, which is stored in the first 6 bits of the IPv4 ToS field. Also, the IPv6 Traffic Class field has the same aim, as explained in paragraph 3.5. The 6-bit DSCP field specifies the forwarding behaviour (technically called "Forwarding Equivalence Class") that the packet has to receive within the DiffServ domain of each operator. The behaviour is called "Per Hop Behaviour" (PHB) and it is defined locally; that is it is not an end-to-end specification (as for RSVP) but it is strictly related to a specific domain. The same DSCP may have two different meanings in two different domains. Negotiations between all adjacent domains are needed to assure a correct end-to-end forwarding behaviour.

The DiffServ approach does not distinguish each single user flow (a customer) throughout the network. The traffic is classified and aggregated in different traffic classes, each of them individuated by a label provided by setting bits in the DSCP field. The identification is performed at the network edges. Within the network core, packets are managed according to the behaviour associated with the specific identification label. Figure 3.18 shows an example of aggregation: the flows are aggregated depending on their DSCP (light or dark grey) and information about each single user (A, B, C or D) are completely lost after the DiffServ router. The operation shown in Figure 3.18 is performed by a Core Router, that is by a router already within the DiffServ network. In practice, the DSCP is assigned, the traffic is already marked. DSCP is assigned on the base of the TCP/IP header content by a router called "Edge Router", which is the entrance gate to the DiffServ network. In theory, DSCP might also be assigned directly by the source but, in regular DiffServ networks, the operation is performed by Edge Routers whose location and action is sketched in Figure 3.19.

The class selector PHB offers three forwarding priorities:

1. Expedited Forwarding (EF) characterized by a minimum configurable service rate, independent of the other aggregates within the router, and oriented to low delay and low loss services.
2. Assured Forwarding (AF) group, recommended in [RFC2597] for 4 independent classes (AF1, AF2, AF3, AF4) although a DiffServ domain can provide a different number of AF classes. Within each AF class, traffic is differentiated into 3 "drop precedence" categories. The packets marked with the highest drop precedence are dropped with lower probability than those characterized by the lowest drop precedence.
3. Best Effort (BE), which does not provide any performance guarantee and does not define any QoS level.

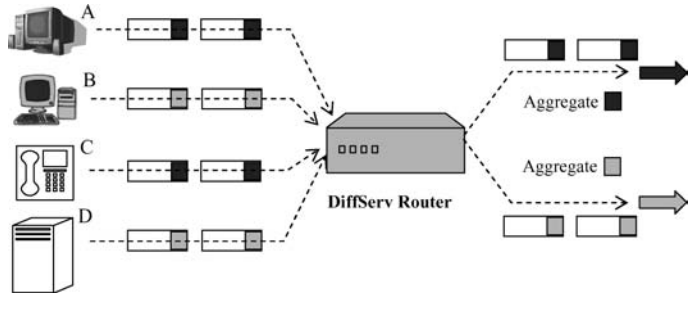


Figure 3.18 DiffServ aggregation behaviour

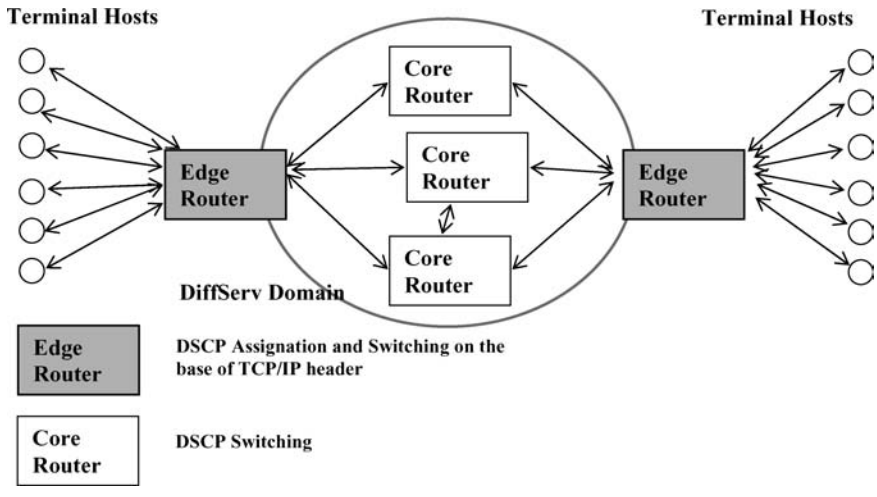


Figure 3.19 DiffServ network

It is clear that the DiffServ paradigm offers a limited set of traffic classes and may be strongly limited by the need of traffic aggregation. Actually, DiffServ implies the need to convey traffic with different SLSs in the same class. "Therefore, while DiffServ architecture solves the scalability problems of QoS provisioning, it fails to be the solution for end-to-end provisioning" [Giordano03]. Actually QoS provision in DiffServ networks depends on traffic handling algorithms. If it is simply limited to traffic priorities, only qualitative QoS indications can be provided but not quantitative values as requested by the SLS. If more complex QoS architectures are used within the DiffServ framework (combining signalling schemes with traffic control issues) as done in Chapter 6, to overcome the limitations of the simple priority action, the small number of bits available to identify a simple flow implies the aggregation of heterogeneous traffic for QoS requirements and consequent bandwidth waste (or, alternatively, QoS decrease), as should be clear from the results in Chapter 4.

3.4.3 Mixed IntServ-DiffServ Approach

A possible improvement of the services provided by DiffServ architectures is represented by a mixed IntServ/DiffServ approach. Figure 3.20 shows an example of it: the two grey circles are the source and the destination of the service; the two bold arrows together with the dashed line show the data flow path through the network; two routers separate the DiffServ domain from the rest of the network. Within the DiffServ domain, the traffic is aggregated in classes, while within the IntServ domain each single customer may be identified and treated separately. The architecture shown in Figure 3.20 relays on the hypothesis that, within small private networks where the bandwidth availability is reduced, each single customer needs to be identified and deserves a special treatment, while the backbone of a network has full bandwidth availability and the differentiation of the traffic through few traffic classes is enough to guarantee the requested SLSs. A parallel with Figure 2.1 is important for a better comprehension of the end-to-end service: the terminal hosts are, respectively, the measurement tool and the device located in the plane, the DiffServ domain may be the

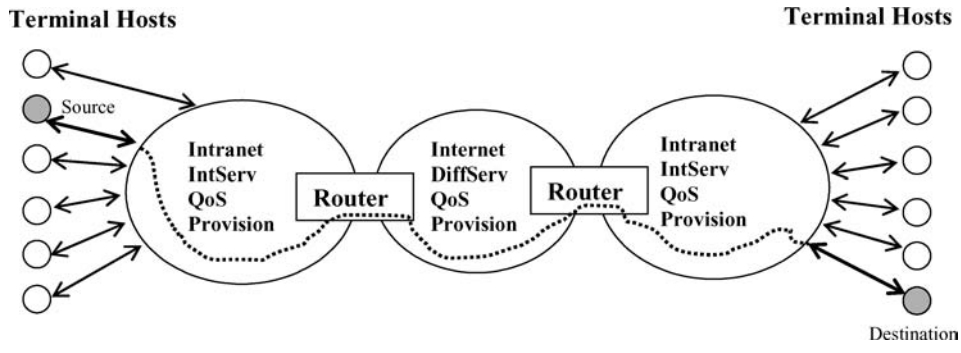


Figure 3.20 IntServ/DiffServ architecture

satellite backbone and the two Intranets are the radio networks accessing the satellite and the LAN in the plane.

Actually, the IntServ approach may be reserved to small private Intranets where there are no scalability problems and the DiffServ approach may be applied to interconnection backbones. The indication perceived in international conferences and magazines is that the Internet community itself recommends IntServ for small private networks and for the access segment. Nevertheless, it means that no quantitative QoS can be provided over the backbone for each customer, also in case of need. It is acceptable having the Internet best-effort service in mind and qualitative service provision but it can be hardly applied to networks offering quantitative SLS-based QoS-guaranteed services in hazardous environments. It is true, in particular, if the application environment includes low speed links and has not great bandwidth availability, as often true for many applications: radio and satellite networks, ad hoc networks and sensor networks. The mentioned networks often carry services that require a very strict SLS: environmental monitoring, military applications, and health services.

A comparison between IntServ and DiffServ is reported in Table 3.3 (derived from [Kota03]). Advantages and drawbacks of QoS-IP solutions are summarized in Table 3.4.

Table 3.3 IntServ versus DiffServ

Feature	IntServ	DiffServ
QoS assurance	Per-flow	Per aggregate
QoS assurance range	End-to-end (application-to-application)	DiffServ Domain (edge-to-edge)
Resource reservation	Controlled by application	Configured at edge nodes based on Service Level Agreement
Resource management	Distributed	Centralized within DiffServ domain
Signalling	Dedicated protocol (RSVP) – Refreshing needed	Based on DiffServ Code Point (DSCP) carried in IP packet header
Scalability	Limited by number of flows	Limited by number of classes
QoS Services	GS, CLS, best effort	EF, AF, best effort
Complexity	High	Low
Availability	Yes	Yes

Table 3.4 QoS-IP solutions strength points and drawbacks

Strength points	Drawbacks
Diffusion IP is widespread: terminal users, LANs, network accesses, backbones, due to the Internet development	QoS IntServ (RSVP) and DiffServ can manage QoS within the IP world but each of them has limitations referring to the performance requirements of users
Simple web integration Web interfaces are much used, both over the Internet and over private networks	IntServ It is considered too complex and not scalable to be used within backbones and multi-user networks. RSVP uses a refreshing mechanism that introduces resource management inefficiencies
Header Compression Header compression is very important to reduce the transmission overhead (in particular the stack RTP/UDP/IP for voice)	DiffServ It is scalable but it can hardly provide end-to-end solutions for environments requiring high quality because of limited numbers of bits to identify a flow

3.4.4 DSCP Assignation

Within an IP DiffServ-based network (i.e. the core DiffServ network), offered QoS is strictly linked to DSCP assignation. The general guidelines for DSCP assignation are contained in [RFC2474]. The DSCP identifies a specific traffic class and implies that all the packets identified with the same DSCP should receive the same treatment but no indication is provided concerning the treatment itself that depends on the implementation choices. DSCP is identified as "xxxxxx" where "x" stands for either "0" or "1". Following RFC2474, the DSCP field can convey 64 distinct codepoints: 32 DSCPs are dedicated to Standard Actions (Pool 1); 16 to experimental and local use (Pool 2) and the other 16 (Pool 3), initially reserved for experimental and local use, are devoted again to Standard Actions when DSCPs of Pool 1 are exhausted. General guidelines are summarized in Table 3.5, taken from [RFC2474].

Respecting the guidelines, a first possibility for DSCP assignation within a DiffServ-based network is reported in Table 3.6, together with possible examples of traffic types taken from [RFC4594]. Codepoint 101110 is recommended for the EF PHB [RFC3246]. The values for AF are taken from [RFC2597]. Table 3.6 follows a regular use of DiffServ: one EF class, four AF classes separated into three precedences and one default BE class, with an additional class for low priority traffic. The DSCP is divided into two parts: the first 3 bits are the Class Selector (CS) and the other 3 bits are the Precedence (P). Table 3.6 contains also the decimal values of the Class Selector, of the Precedence, both computed on the 3 bits defining them, and of the overall DSCP, computed over the 6 bits.

Table 3.5 DSCP assignation general guidelines

Pool Number	DSCP assignation	Policy of assignment
1	xxxxx0	Standard actions
2	xxxx11	Experimental/local use
3	xxxx11	Standard actions/experimental/local use

Table 3.6 Possible DSCP assignation for data within DiffServ network

Service	Traffic class	DSCP assignation	Decimal values CS – P – DSCP	Example of applications
Telephony	EF	101110	5 – 6 – 46	IP Telephony bearer
	AF41	100010	4 – 2 – 34	
Multimedia conference	AF42	100100	4 – 4 – 36	Videoconference
	AF43	100110	4 – 6 – 38	
	AF31	011010	3 – 2 – 26	
Multimedia streaming	AF32	011100	3 – 4 – 28	Streaming video and audio
	AF33	011110	3 – 6 – 30	
Data of low latency transactions	AF21	010010	2 – 2 – 18	Client/server web-based transactions
	AF22	010100	2 – 4 – 20	
	AF23	010110	2 – 6 – 22	
High throughput data	AF11	001010	1 – 2 – 10	Client/server web-based transactions
	AF12	001100	1 – 4 – 12	
	AF13	001110	1 – 6 – 14	
Standard data	Default	000000	0 – 0 – 0	Not specified
Low priority data	CS1	001000	1 – 0 – 8	Best effort
Broadcast video events	CS3	011000	3 – 0 – 24	Broadcast TV
Real-time interaction	CS4	100000	4 – 0 – 32	Interactive applications and gaming

By always following the classification proposed in [RFC4594], some DSCP values may be dedicated to administrative and control traffic (including signalling). A possible proposal is contained in Table 3.7.

An alternative DSCP assignation is proposed by the Department of Defence (DoD) of the United States of America. The proposal is contained in Table 3.8 and it is taken from [DoD].

The assignation proposed in Table 3.8 is much used in the military world within the framework of Global Information Grid (GIG) networks.

Table 3.7 Possible DSCP assignation for control information within DiffServ network

Service	Traffic class	DSCP assignation	Decimal values CS – P – DSCP	Example of applications
Operation and Management (OAM)	CS2	010000	2 – 0 – 16	OAM
Signalling	CS5	101000	5 – 0 – 40	IP telephony signalling
Network Control	CS6	110000	6 – 0 – 48	Routing and control information
Administrative	CS7	111000	7 – 0 – 56	Routing and control information

Table 3.8 DoD DSCP assignation

Service	Traffic class	DSCP assignation	Decimal values DSCP	Example of applications
Continuous Bit Rate (CBR)	EF	101111, 101110, 101101, 101011, 101001, 101000	47, 46, 45, 43, 41, 40	Interactive voice
Variable Bit Rate (VBR)	AF41	100010	34	
	AF42	100100	36	Interactive video
	AF43	100110, 100000	38, 32	
	AF31	011010	26	Streaming multimedia and multicast
Multimedia	AF32	011100	28	
	AF33	011110	30	
	AF21	010010	18	
Mission Critical	AF22	010100	20	Short blocks of interactive data
	AF23	010110	22	
	AF11	001010	10	
Mission Critical	AF12	001100	12	Long blocks of bulk data
	AF13	001110	14	
Best effort	Default	000000, 001000, 010000, 011000	0, 8, 16, 24	Everything else
Control and Management	CS7	111000	56	Network Control
Control and Management	CS6	110000	48	Internetwork Control

Tables 3.6 and 3.8 are two operative examples but, independently of them, each single private DiffServ-based network can perform the preferred assignment choice by using the 6 bits available. It theoretically opens the door to 64 different traffic classes, which is the maximum technical limit for the DiffServ paradigm. What it is important to state is the definition of a common DSCP assignation (a common language) within each DiffServ cloud. Communication and QoS mapping among different network portions is the object of Chapter 6, which also contains the description of possible QoS architectures where the different network components operate.

Possible ranges of QoS performance requirement upper limits associated with DoD assignment may be derived by associating the DSCP class definition of Table 3.8 with the application QoS needs for IP networks specified in Chapter 1 (e.g. Tables 1.8 and 1.9). Table 3.9 contains a possible suggestion about it.

The respect of the SLS for each traffic class heavily depends on the control techniques used within the network, whose importance will be outlined in the next chapter. From this viewpoint, entering the details of the single mechanisms is outside the scope of this book and it is left to the huge amount of literature dedicated to it. The only observation is related to the possible model used to offer a specific service to each single class. The QoS provision is offered at IP layer. In

Table 3.9 SLS and DLCP assignment

Service	Traffic class	DSCP assignation	IPTD Transfer Delay	IPDV Delay Variation (Jitter)	IPLR Loss
Continuous Bit Rate (CBR)	EF	101111, 101110, 101101, 101011, 101001, 101000	100–400 ms	30–50 ms	10^{-2}–10^{-3}
Variable Bit Rate (VBR)	AF41 AF42 AF43	100010 100100 100110, 100000	100–400 ms	30–50 ms	10^{-2}–10^{-3}
Multimedia	AF31 AF32 AF33	011010 011100 011110	5–10 s	Not applicable	10^{-2}–10^{-3}
Mission Critical	AF21 AF22 AF23	010010 010100 010110	20–100 ms	1–50 ms	0
Mission Critical	AF11 AF12 AF13	001010 001100 001110	1–50 ms	Not applicable	0–10^{-3}
Best effort	Default	000000, 001000, 010000, 011000	Unspecified	Unspecified	Unspecified
Control and Management	CS7	111000	0.05–1 s	Not applicable	0–10^{-3}
Control and Management	CS6	110000	1–10 s	Not applicable	10^{-2}–10^{-3}

practice, each router is modelled as a battery of virtual buffers (implemented in the software) that separates the traffic of the different classes supported by the network. Single buffers will be provided with a specific bandwidth so that QoS requirements can be respected. The traffic of a single class receives a dedicated treatment within the DiffServ router (Figure 3.18). QoS mechanisms imply the presence of an overall architecture, including signalling protocols to transfer the QoS needs and possible feedback about the state of the network. Information needs to be transmitted among networks implementing different QoS technologies. It will be the object of Chapter 6. If no QoS architectures, signalling and control mechanisms are implemented, the only possible QoS management strategy is a strict priority scheme where bandwidth is allocated to different queues and the only guarantee is that traffic of less prioritized queues will be forwarded after traffic belonging to more prioritized queues. The limits of it have been evidenced before and should be clarified by the results about control, reported in Chapter 4.

3.5 QoS-IPv6

The envisaged problems of DiffServ may be mitigated by using IPv6, which can take the best of IP technology.

IPv6 may use two fields directly in the IP header to mark traffic:

1. Flow Label field (20 bits).
2. Traffic Class field (8 bits), functionally equivalent to IPv4 ToS field and containing the DSCP field in the first 6 bits leaving the rest for future extensions, as in IPv4.

The two options are shown in Figure 3.21.

The difference between IPv4 and IPv6 is that IPv6 can mark a flow through the Flow Label, as well as an aggregate of flows (similarly to VC/VP in ATM), so keeping full scalability together with the theoretical power to identify a single customer, while IPv4 can identify either a limited number of aggregates through the DSCP field or a specific flow (but not an aggregate of them) through the vector "IP source address, IP destination address, Protocol, TCP/UDP source port, TCP/UDP destination port". IPv6, from the point of view of flow identification, is a very powerful tool, if used with its full features.

Concerning flow identification, IPv6 offers the same capabilities of ATM but, to use all features, it would be necessary to import most of the control functions (part of the ATM approach) within IP. In practice, it would be interesting to keep IPv6 format to simplify interworking, but to use control mechanisms (e.g. CAC, filtering and signalling, bandwidth allocation) for each flow, as done in ATM. This opportunity is still part of the research activity. At the moment, concerning QoS in commercial products, there is no difference between IPv4 and IPv6 [IPJ-June05], which is currently used within the two mentioned paradigms: DiffServ and IntServ. The full use of IPv6 flow identification

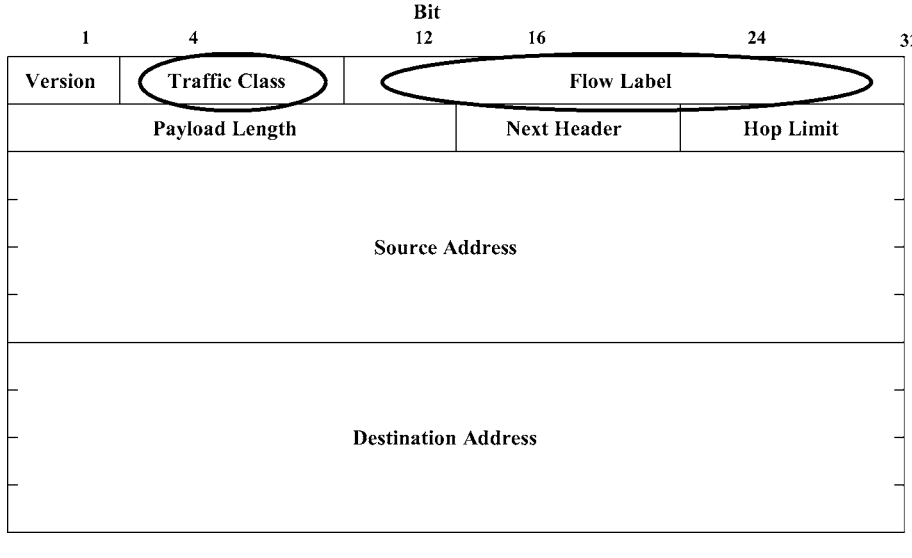


Figure 3.21 IPv6 packet header and flow identifiers

features implies to overcome DiffServ and IntServ paradigms. The idea is to use IP together with all the QoS control schemes (listed in the next section), which characterize the ATM technology. In practice, it allows to keep the simplicity of IP concerning the operational protocol features (e.g. addressing and interfacing) with the QoS management power of ATM. The approach hides several difficulties because implies matching two worlds (IP and ATM), which are always seen as alternatives by the scientific community. Many efforts have been provided to allow operative interfaces between them but the two paradigms have always been conceptually separate, even if operating together. It is also true that IP routers do not have the QoS management features (and the related complexity) of ATM switches. Importing ATM QoS controls over IP implies the solution of implementation problems that are not trivial, but the overall result would provide great advantages and the challenge is worth to play, at least concerning the opinion of the author of this book.

A first step towards Full IPv6 is indicated in [ID-Chakravorty] that contains interesting concepts and practical indications, part of which are much used in the following. The full IPv6 architectural framework is called "IPv6 Label Switching Architecture" (6LSA) in [ID-Chakravorty] and uses the features of the Flow Label in the IPv6 header to establish QoS end-to-end paths, similarly as in ATM and MPLS. Mostly, from [ID-Chakravorty], the 6LSA network may be applied if IPv6 is deployed and where QoS network performance is explicitly required. The main features of 6LSA may be summarized as follows.

IPv6 Label Switching Architecture technology offers a new method that provides a full use of IPv6 Flow Label. It allows QoS and other service delivery, as well as end-to-end IP Transparency because it is a Layer 3 protocol and avoids encapsulation (as in MPLS) and fragmentation (as in ATM) of IP packets. The 6LSA network may be used also with IPSec constraints because it does not use Layer 4 ports. No additional layers (as in MPLS and ATM) are required to add identification labels and no label distribution scheme is necessary. It helps reduce the overhead and allows avoiding special signalling and complex state-machines (as required by ATM). Therefore, 6LSA technology is easy to deploy and can use all Layer 2 protocols with no technological limitation. It can use all IPv6 header features. Since the packet size does not contain additional labels, 6LSA eliminates the potential risk of MTU violations. 6LSA allows also node-local generation of labels; it increases network flexibility and may also considerably enhance security because of a possible totally random or periodically synchronized label choice across the 6LSA domain.

In short, 6LSA provides a significantly efficient Layer 3 mechanism with extensive QoS Label space. The 20-bit Flow Label in addition to the 8-bit Traffic Class field can provide a huge traffic classification space. The full use of the 20-bit Flow Label field can help the exploitation of IPv6 in QoS-based networks, also together with IPv4 and MPLS technologies. The implementation of 6LSA in native IPv6 networks simplifies traffic engineering.

In more detail, a full IPv6 architecture will allow two different visions, introduced in this book. The first one is a Class of Service-oriented DiffServ-like approach; the second one is a Single User-oriented vision (ATM-like). Concerning the former, it means to associate (as done in DiffServ) a fixed string of bits to a specific class of service, that is to a specific Forwarding Equivalence Class, which is the forwarding treatment of an IPv6 group of packets, in this case. Differently from DiffServ, whose ToS field length limits the number of feasible classes, full IPv6 network would allow a huge number of traffic classes. This option is called "Class of Service Full IPv6 Network" (CSF6N). On the other hand, within the Single User-oriented vision, each flow may receive a specific treatment, as in ATM and

MPLS networks. In this view, Flow Labels would be assigned and locally switched, at each IPv6 router (IPv6 Label Switch Router). This option is identified as "Full IPv6 Switched Network" (F6SN)" in the reminder of the book.

Even if, due to the huge number of classes allowed by IPv6, there is no substantial difference about QoS perception between CSF6N and F6SN, it is important to highlight the different philosophy of the two approaches. CSF6N inherits the features of DiffServ, where there is no identification of the single user flow and traffic is switched on the basis of the DSCP value. F6SN heritage is ATM, where each single user flow is identified by a concatenation of VCI/VPI and traffic is switched on the basis of VCI/VPI identifiers that are assigned locally by each ATM switch. Obviously, as in ATM, the traffic characterized by the same features and QoS requirements may be aggregated together and treated in the same way within F6SN routers, so guaranteeing scalability.

3.6 Class of Service Full IPv6 Network (CSF6N)

Traffic is switched as in an extended DiffServ network: each router of a CSF6N needs to share a common definition of the traffic classes with the other routers of the same network. So, similarly as done for DSCP definition in Tables 3.6–3.8, it is necessary to establish a common definition of the Flow Label, having 20 bits available for it and, theoretically, 2^{20} traffic classes. Even if it is difficult to imagine, due to the different number of classes, the structure of the assignment tables used within CSF6N routers would be similar to the structure of the tables applied for DiffServ (from Tables 3.5 to 3.9).

3.7 Full IPv6 Switched Network (F6SN)

The operation mode is connection oriented. The switching architecture may be designed directly from label switched technologies (ATM and MPLS). The reference QoS language is no longer the Traffic Class paradigm but a detailed per-flow SLS (as in Table 1.3), where each single user receives a specific QoS and, in case of need, may be dropped (e.g. MLPP case), as often necessary in military networks. An F6SN switch (also simply called "Flow Label Switch Router" (FLSR), in the following) is shown in Figure 3.22 and may work as follows: information is transported from "n" incoming links $[I_1, I_n]$ towards "m" outgoing channels $[O_1, O_m]$, on strict basis of the IPv6 header Flow Label and of the information contained within the Flow Label Translation Table (FLTT), whose example is reported in Table 3.10. The incoming IPv6 Flow Label and physical link are used to access the translation table. The result of the access will be the physical outgoing link and the new Flow Label value to be inserted in the IPv6 packet header. For example, IPv6 packets coming from the physical interface I_1 and identified by the Flow Label "y" are switched to the outgoing physical interface O_m and take the Flow Label value "g", and packets coming from I_2 with Flow Label "c" are switched to O_1 with Flow Label "a". Table 3.10 allows also understanding the need of changing the Flow Label switch by switch on link basis. If the Flow Label "a" had been kept unchanged along the source – destination path, IPv6 packets coming from both I_1 and I_2, identified by label "y" would have been totally indistinct over the output O_m and the information about the single two flows would have been lost. Changing the Flow Label (technically speaking, switching it) allows keeping the two flows separate over the interface O_m, where one flow is identified by the label "g" and the other one by the label "h".

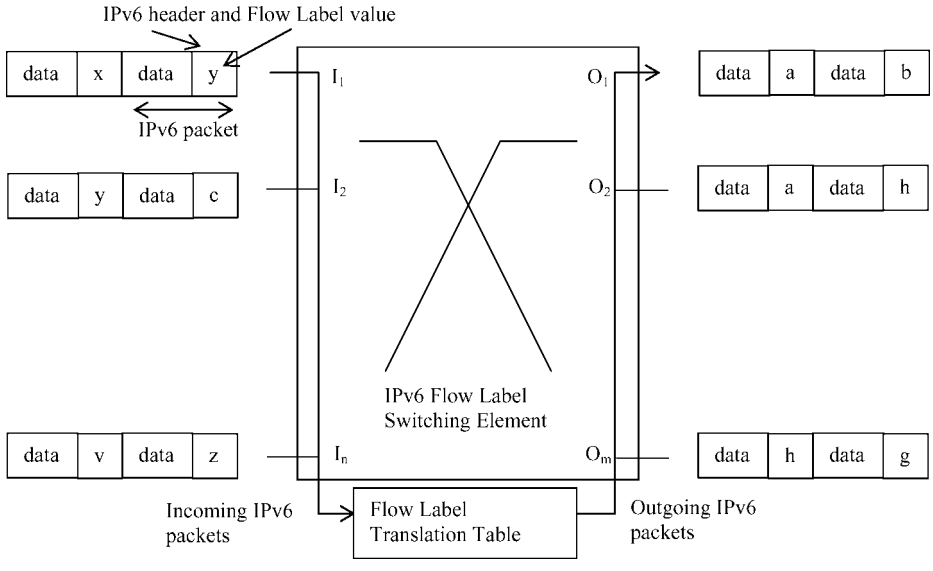


Figure 3.22 F6SN node – Flow Label Switch Router (FLSR)

Table 3.10 F6SN Flow label translation table

Incoming Physical Link	Incoming flow label	Outgoing Physical Link	Outgoing Flow label
I_1	y	O_m	g
I_1	x	O_1	b
I_2	c	O_1	a
I_2	y	O_m	h
. . . .	. . . . .	. . . . .	. . . . .
I_n	z	O_2	h
I_n	v	O_2	a

The number of incoming and outgoing links may be the same (m = n) but it may also vary: if $m < n$ (the number of inputs is larger than the number of outputs) the action of the switch is called either "multiplexing" or "concentration", the former putting the emphasis on the fusion of different streams of IPv6 packets, the latter on the reduction of the number of incoming links to a lower number of outgoing links. The opposite situation ($m > n$) is called either "demultiplexing" or "expansion".

As clear in Figure 3.22, packets coming from different physical interfaces arrive at the node at the same time and may be destined to the same output. They cannot be sent to the outgoing interface exactly at the same time because it would create collision, so a buffer (or a set of buffers) needs to be included in the switching architecture to store the packets that cannot be immediately served and to guarantee switching with no loss. The cross internal to the switching element in Figure 3.22 represents the routing architecture within the switch

itself. In other words, it represents how packets are physically transported from the inputs to the outputs and the hardware/software structure to do it. Routing architecture and strategy are left to the literature about switching theory. Reference [Pattavina98] may be of great help.

A virtual end-to-end connection is identified by a sequence of IPv6 Flow Labels over a specific concatenation of physical interfaces. Always taking ATM as reference, in this context, a IPv6 Local Virtual Connection (IPv6 LVC) may be defined here as "A concept used to describe unidirectional transport of IPv6 packets associated by a common unique identifier value called Flow Label." An IPv6 Local Virtual Connection Link (IPv6 LVCL) may be "A means of unidirectional transport of IPv6 packets between a point where a Flow Label value is assigned and the point where that value is removed." A single user flow (a single end-to-end connection) may be described by a concatenation of Flow Labels, which may be called "IPv6 Virtual Connection" (IPv6 VCo).

Each IPv6 LVC is assigned at the connection set-up. An IPv6 LVC has only a local meaning linked to the physical interface: when the connection finishes, IPv6 LVC values will be released and could be reused by other connections. A possible advantage to exploit the Flow Label is the use of different Flow Labels for multi-mean services. For instance, videotelephony is composed of two components: voice and video. Each of them may be transported over separate Flow Labels. It allows the network to add and remove components during a connection without killing the application: videotelephony service can begin by using only voice (one Flow Label) and the video component may be added later over a separate Flow Label.

Flow Label assignation is a key problem. Even if there is no theoretical obstacle to assign the Flow Label at the source, it appears much better to apply an MPLS-like structure and to use two instruments.

1. Flow Label Switching Edge Router (FLSER), which (1) assigns the label both on the basis of the information contained in the IPv6 header and on the basis of proper signalling algorithms and (2) switches the label towards the F6SN.
2. Flow Label Switch Router, whose action and components have been defined above.

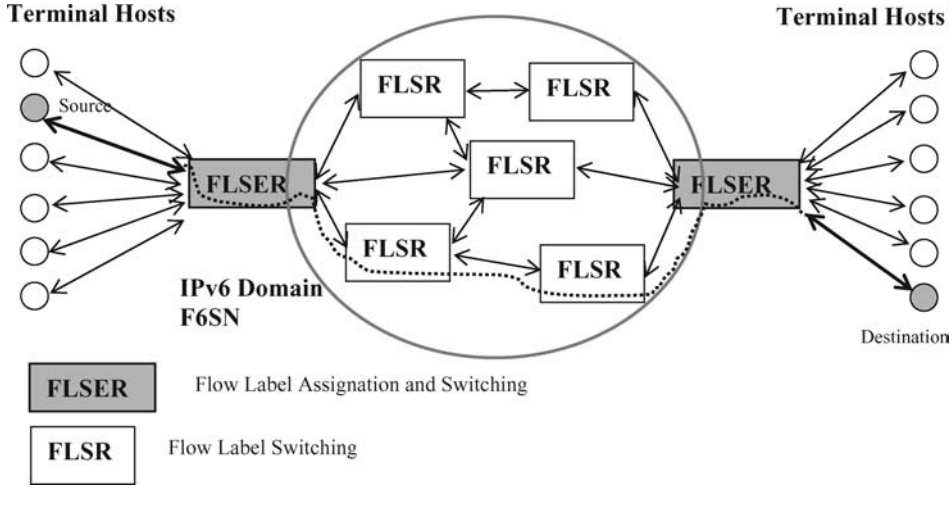


Figure 3.23 Full IPv6 switched network

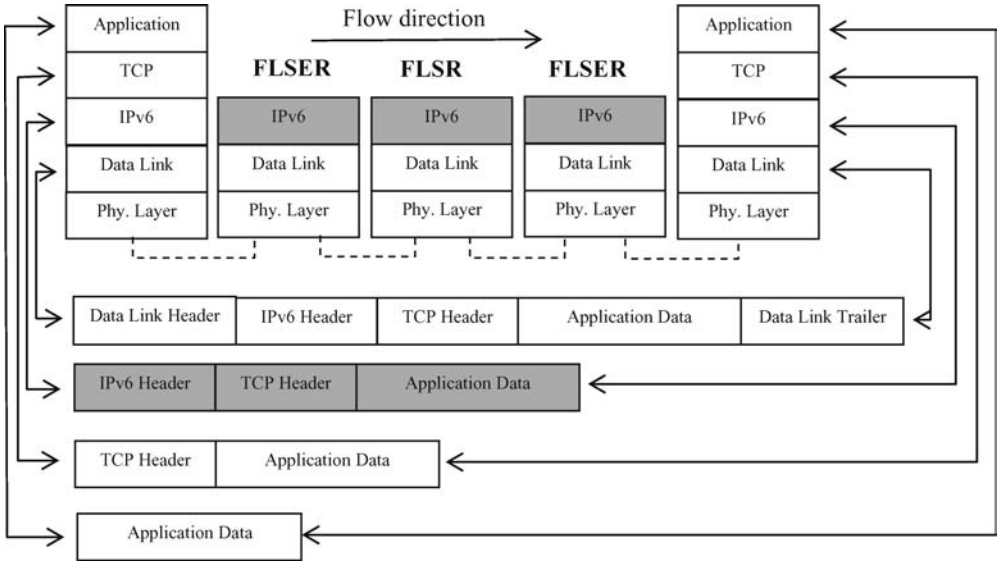


Figure 3.24 F6SN: Data flow and IPv6 header flow label switching

The two entities are reported in Figure 3.23 to compose a F6SN.

From the layered architecture viewpoint, both FLSER and FLSR are two IPv6 routers. The data flow starting from the source and traversing two FLSERs and one FLSR is shown in Figure 3.24. The entities where the IPv6 header Flow Label is switched are shown in grey. The IPv6 packet format is also evidenced.

4

Network Control Issues

4.1 QoS Management Functions

After the analysis of the main QoS technologies available in the market from the point of view of flow identification, it is necessary to highlight the importance to have control mechanisms implemented within the network. Packet identification will have a major role (better explained below) but, alone, it is not sufficient to assure a certain level of service because it is a tool, not a solution. A short description of control mechanisms applied in the communication environment is reported below, starting from the trivial solution of "over provisioning".

4.1.1 Over Provisioning

"There is a common misconception that purchasing an oversupply of bandwidth will solve all service-quality challenges. Throwing bandwidth at the problem is sometimes perceived as a simpler solution than QoS management" [Montanez02]. This approach ignores not only bandwidth optimization but also possible future trends and requirements of new services. It cannot be classified exactly as a solution; QoS management is strongly necessary and QoS management functions are aimed at offering the necessary tools to get a certain level of quality. This is particularly clear for networks acting in hazardous situations as in civil protection and military environments, where the QoS is essential. One of the objectives of this chapter is to show the effect of the various functions and, independently of the used technology, the importance to have specific service functions to satisfy precise performance requirements. The effect of most functions will be evaluated in the results. Meaningful QoS management functions are reported in the following.

QoS over Heterogeneous Networks Mario Marchese

4.1.2 Flow Identification

As already said, the identification of packets so that they may receive a different treatment within the network is topical to guarantee QoS, ranging from a minimum priority-based service to quality assurance for a specific flow. Single QoS-oriented technologies, presented before, show different methods to classify packets: Flow Label and Traffic Class in IPv6, ToS and vector "IP source address, IP destination address, Protocol, TCP/UDP source port, TCP/UDP destination port" in IPv4, VPI/VCI in ATM, Label Value in MPLS. The impact of it on the performance of the network will be evidenced in the following.

4.1.3 Resource Reservation and CAC

An accurate resource reservation to guarantee that traffic flows receive the correct service is strictly needed. The term "resource" means, in this context, bandwidth and buffer. Its allocation is strictly related with Call Admission Control (CAC) mechanisms. The acceptance/rejection of a new connection is performed subject to a check (that may be statistical) about the availability of network resources in consequence of specific requirements. After that, if enough resources are available, they are reserved. Resource reservation may also act on a larger timescale concerning network planning and link dimensioning. The effect of an inaccurate bandwidth and buffer allocation on the performance of a network node will be shown in the following.

Call Admission Control decides whether a new connection request may be accepted or not. It is a powerful tool to guarantee quality because it allows limiting the load entering the network and verifying if enough resources are available to satisfy the requested performance requirements of a new call without penalizing the connections already in progress. A connection provides traffic descriptors (e.g. the set of traffic parameters of an ATM source) and QoS requirements (i.e. a specific SLS). The network evaluates if there are sufficient resources in terms of bandwidth and buffer to match the request and decides whether to accept or reject the connection.

The multiservice structure of broadband networks, where QoS requirements with possibly very different characteristics (see Chapter 1) must be satisfied for traffic flows with different statistical natures, has led to the application of specific control actions whose main goals consist of maintaining the desired QoS levels and maximizing the utilization of network resources (or some revenue function). These features, being connected to service integration, are present in any QoS-oriented telecommunication network. As a matter of fact, bandwidth allocation and CAC problems were also present in the B-ISDN environment and they have been widely developed within ATM. The control strategies in this case are related to QoS at the call level, in terms of both cost or revenue functions and constraints (e.g. on call blocking probability). The statistical multiplexing character of traffic adds another control level to this scenario, because it further requires the satisfaction of packet-level constraints. As said in Chapter 1, SLS is described in terms of packet loss, packet delay and jitter for each flow or group of flows. There is no immediately clear way to identify the amount of resources (buffer space and bandwidth) required by a flow to guarantee performance requirements, except for peak bandwidth assignment. This has given rise to many investigations on the bandwidth allocation problem (and the related CAC strategies). An important notion is represented by the concept of equivalent bandwidth, which is defined as the minimum rate allocation

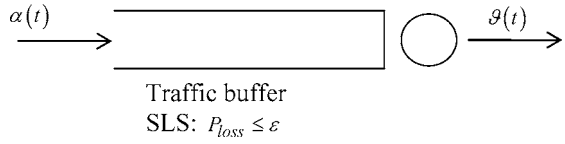


Figure 4.1 Equivalent bandwidth action

necessary to guarantee a specific QoS to a single statistically homogeneous flow. If flows are not homogeneous, either from the QoS requirements or from the traffic sources viewpoint, equivalent bandwidth techniques are hardly applicable and new solutions are necessary. Moreover, the aggregation of heterogeneous traffic flows gives origin to problems in QoS provision highlighted in the following of this chapter and, more specifically, in Chapters 6, 8 (section 8.6.2) and 9 (sections 9.3 and 9.4.2).

Equivalent bandwidth techniques are typically based on the statistical characterization of the traffic by means of descriptors as peak rate, mean rate and maximum burst size. Among the others, an interesting technique is introduced in [Guerin91]. Assuming homogeneity within each traffic class, a possible model is reported in Figure 4.1. A single buffer is dedicated to a specific traffic class, characterized by the inflow process $\alpha(t)$ and by the performance requirement (over the packet loss probability, in this case, $P_{\text{loss}} \leq \varepsilon$). The aim is to determine the service rate over time $\vartheta(t)$ so that $P_{\text{loss}} \leq \varepsilon$. The equivalent bandwidth technique is based on the mean (m) and the standard deviation (σ) of the $\alpha(t)$ process. It uses online measures without assuming any a-priori knowledge of traffic statistics and buffer size. The time instants of the rate reallocations being $k = 1, 2, \ldots n$, $m(k)$ and $\sigma(k)$ the mean and the standard deviation, respectively, of the inflow process measured over the time interval $[k, k+1]$, the bandwidth provision at the instant $k+1$ $(\vartheta(k+1))$, assigned over the time interval $[k+1, k+2]$, is computed as $\vartheta(k+1) = m(k) + z \cdot \sigma(k)$, where $z(\varepsilon) = \sqrt{-2\ln(\varepsilon) - \ln(2\pi)}$ and ε is the limit threshold of the packet loss probability. This technique is well suited for PLP control. The delay control may be reached by properly dimensioning the maximum buffer size [Liu04].

There are other techniques to compute the necessary bandwidth but, in general, the concept of equivalent bandwidth binds the packet-level QoS guarantees and the statistical characteristics of a source with a bandwidth requirement, thus allowing a separation between the packet- and call-level control problems. Among the various approaches, some of them impose a precise structure to allocate resources. Traffic is typically subdivided into classes, which are homogeneous in terms of performance requirements and/or statistical characteristics of the traffic sources, and bandwidth is allocated accordingly, thereby restricting the statistical multiplexing only within each class. This approach (service separation) is formalized and clearly described in [Ross]. This further eases the decomposition of a very complex overall control task, which is generically characterized by very different timescales and requirements, according to the level where the system dynamics is considered (e.g. packet and call level), into smaller and possibly independent problems. For instance, an essential decoupling between packet and call level is obtained through the concept of Feasibility Region (discussed in [BollaDavoliMarchese] for the ATM environment). In other words, it is possible to fix a region in the call space where the requirements at packet level are statistically satisfied. For example, referring to the network node modelled in Figure 4.2, the traffic of two homogeneous traffic

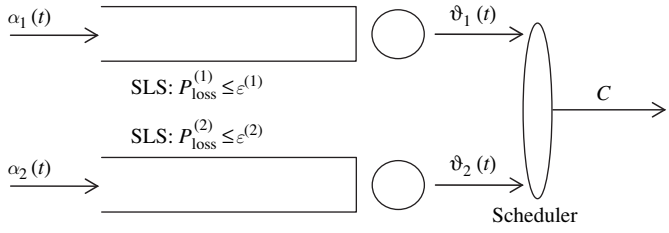


Figure 4.2 Network node conveying two traffic classes

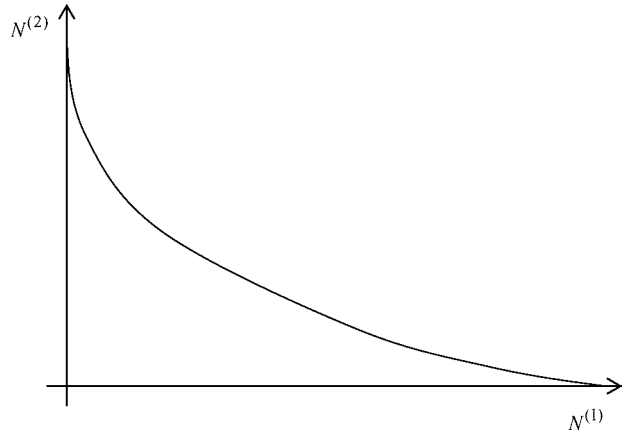


Figure 4.3 Graphical representation of the feasibility region (two traffic classes)

classes, identified as 1 and 2, is conveyed towards two separated buffers. Class 1 and 2 traffic is identified by the corresponding index and described by statistical parameters as mean (m_1 and m_2) and standard deviation (σ_1 and σ_2) of the processes $\alpha_1(t)$ and $\alpha_2(t)$, respectively, as done for the equivalent bandwidth scheme. More completely, traffic can be identified by peak rate, average rate, burst length and by performance requirements (e.g. $P_{\text{loss}}^{(1)} \leq \varepsilon_1$ and $P_{\text{loss}}^{(2)} \leq \varepsilon_2$). In these conditions, it is possible to define a region of the call space where performance requirements are statistically satisfied. The quantity C is total bandwidth available for traffic and, obviously, the constraint is $\vartheta_1(t) + \vartheta_2(t) \leq C$ in any instant of time.

Graphically, the feasibility region may be represented as in Figure 4.3. A maximum number $N^{(2)}$ of Class 2 calls can be accepted in correspondence of a number $N^{(1)}$ of Class 1 connections. Performance requirements are guaranteed by making use of the statistical multiplexing gain. The call space region below the curve is the feasibility region where packet-level requirements are satisfied. The curve is a straight line if the peak bandwidth is given to each single connection. Obviously, in this case, the multiplexing gain is ignored and the corresponding feasibility region (striped in Figure 4.4) is within the feasibility region computed through equivalent bandwidth techniques (dashed in Figure 4.4).

Packet levels and call levels are decoupled by adopting this philosophy. It is also possible to choose a sub-area within the feasibility region (in practice partitioning the feasibility region), so as to allow fixing some objectives over the call blocking probability. The shape

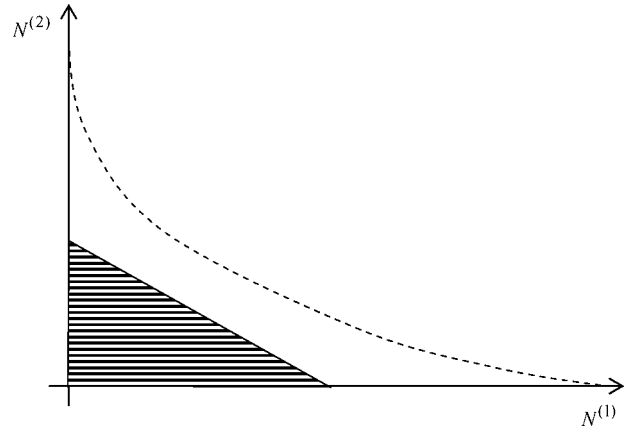


Figure 4.4 Feasibility region: Peak bandwidth allocation

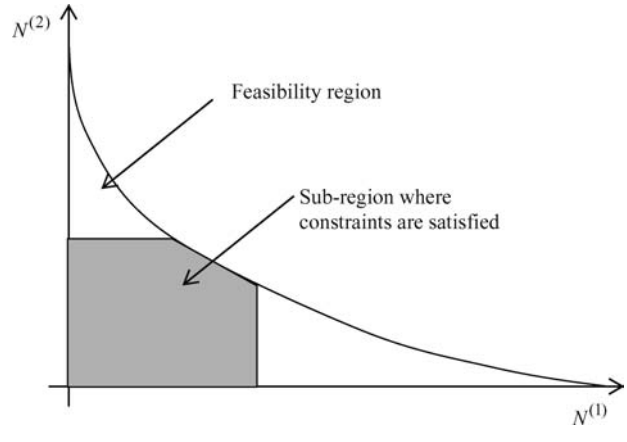


Figure 4.5 Relation between feasibility region and sub-region

of the sub-region depends on the aim of partitioning. Reference [BollaDavoliMarchese] reports some alternatives (related only to ATM and so with fixed packet length) aimed at the following:

- Minimizing the average blocking probability of the overall network node;
- Achieving equalization of the blocking probabilities over the classes;
- Satisfying (or at least approaching) constraints on the call blocking probability such as $P_{\text{block}}^{(i)} \leq \gamma^{(i)}$, where the index i is the traffic class.

Figure 4.5 contains an example of sub-region obtained by partitioning the feasibility region. The call blocking probability is modelled through the Erlang B formula.

Reporting implementation details is outside the scope of the book but extending the results got for ATM to the IP environment is surely an interesting research activity.

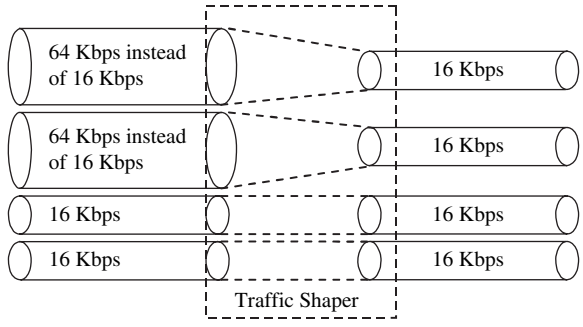


Figure 4.6 Action of traffic shaping

4.1.4 Traffic Control (Shaping)

Shaping policies limit flows to their committed rates (e.g. the flows need to be conform with their traffic descriptors). They are very important to guarantee performance requirements. If flows (or also single connections) exceed their bandwidth consumption specifications, the network, which has dimensioned resources in strict dependence on the declarations, cannot guarantee any specified QoS requirement. An example of traffic shaping action is shown in Figure 4.6, where two, out of four, connections do not conform with their committed rates set to 16 Kbps. The two non-conformant flows that generate 64 Kbps are cut down to the committed rate by the traffic shaper.

There are two basic methods to shape traffic: leaky bucket and token bucket. The former is also called "Generic Cell Rate Algorithm" (GCRA), comes from the ATM world and it was standardized by the ATM Forum in 1996 [ATM-Forum96]. Adaptation is needed because ATM information unit is fixed to 53 bytes while IP packets, for example, have variable length. The basic concept is that incoming flows enters a bucket (virtually located in network edge nodes) and, if the flow exceeds the SLS declarations, either it is discharged just as though there were a hole in the bottom of the bucket or it is tagged for future decisions. In the first case, only conformant flows leave the bucket with the entrance rate. In practice, a specific algorithm implements the described filter operation. It is possible to use more than one algorithm, for example, two schemes acting in tandem and checking two different parameters (e.g. peak and average packet rate). It is called "dual leaky bucket" where there are two algorithms in series. The leaky bucket algorithm also allows giving a precise definition of "conformance" and, as a consequence, to decide if a flow is conformant or not. A possible generic flow-chart is reported in Figure 4.7 to give an idea of the behaviour. The conformance with the quantity Conformant Rate (CR) is checked. To apply the scheme in a real situation, CR would be the rate reported in the SLS. The following definitions are necessary to understand the algorithm:

T: Byte counter that represents the bucket;
T': Auxiliary variable;
t_a^k: Arrival time of the k-th byte;
LCT_a: Arrival time of the last compliant byte (Last Compliance Time);
CR: Conformant rate;
L: Threshold value.

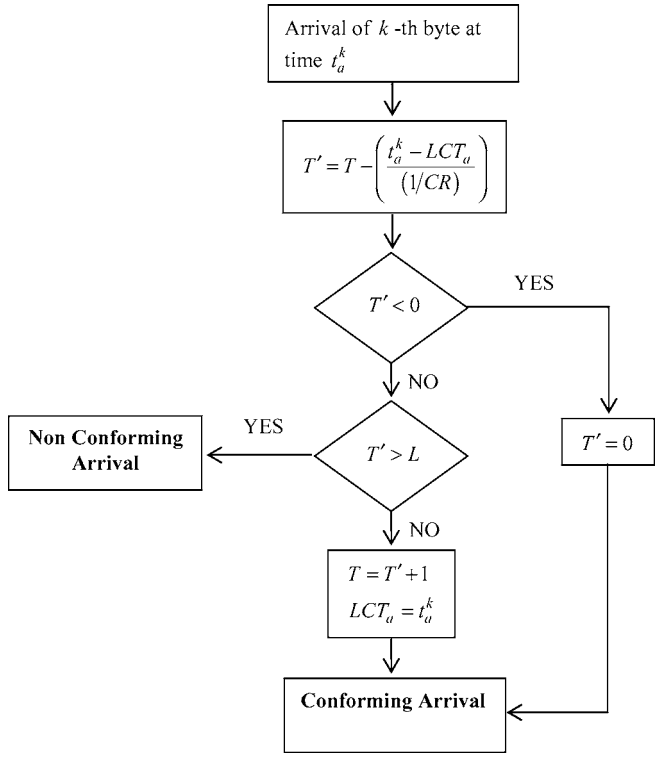


Figure 4.7 Leaky bucket algorithm flow chart

The reported example works in bytes but it may also be adapted for packets. When a new byte (e.g. k-th byte) of a specific flow is entering the network, the "distance" between the arriving byte and the last conformant arrived byte is "measured" in terms of number of attended bytes concerning the conformant rate (CR, under analysis). The obtained object is used to reduce the auxiliary variable. In formula, $T' = T - [(t_a^k - LCT_a)/(1/CR)]$. If the auxiliary variable is below zero, the byte entering the network is surely conformant. Otherwise, it is compared with the threshold value L, which is a sort of tolerance measure. If $L = 0$ there is no tolerance and the first burst, exceeding CR, is immediately identified as non-conformant.

Concerning token bucket, the philosophy is similar to the leaky bucket one but the implementation is different. Tokens are generated with rate CR and put in a bucket. For each byte entering the network, a single token is consumed. A byte is allowed to enter the network only if there is at least one token in the bucket. Otherwise, it waits in the queue until the next token arrives. It can also be either discharged or tagged for future processing. Figure 4.8 reports the essential components. Also, in this case, if there are two quantities to check, two token bucket algorithms in cascade may be used.

4.1.5 Scheduling

Packet scheduling specifies the service policy of a queue within a node (e.g. an IP router, an ATM switch). In practice, scheduling decides the order that is used to pick the packets

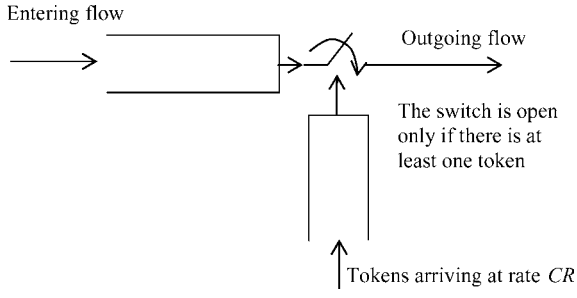


Figure 4.8 Token bucket scheme

out of the queue and to transmit them over the channel. The simplest algorithm is FIFO, where packets are served in the same order as they arrive in the queue. Scheduling is an important issue, concerning QoS. It has a strong impact on different QoS parameters as delay, jitter and loss. The main problem arises from the impossibility of assigning the committed bandwidth to a specific flow at each time instant. Bandwidth is allocated in average and most scheduling policies may guarantee the average bandwidth assignment. Unfortunately, it may not be sufficient to assure the guarantee of QoS parameters, as, for example, the delay. The only scheduler that allocates the overall outgoing bandwidth to all users in progress in strict proportion with the bandwidth allocated, for each time instant, is the Generalized Processor Sharing (GPS). It applies an ideal policy that supposes to serve all users at the same time and to split infinitesimally the bandwidth, which is allocated to the users exactly as requested. Generalized Processor Sharing is often used in simulations by implementing a separate buffer with a dedicated scheduling machine for each flow. Real systems have to use alternative schemes aimed at performing as closest as possible to GPS.

A clear and complete revision of the most interesting schedulers is reported in [ChaoGuo02]. It includes Weighted Round Robin (WRR), Weighted Fair Queuing(WFQ), Virtual Clock, Self-clocked Fair Queuing, Deficit Round Robin, WF^2Q and WF^2Q+, as well as a comparison of the mentioned schemes concerning latency, fairness and time complexity. Another complete description of scheduling algorithms is contained in [Soldatos05], which also includes a classification of control modules.

A practical example related to GPS and to WRR may help understand the importance of scheduling: there are three traffic classes in a network node identified through letter a, b and c; bandwidth assigned to them is, respectively, B_{wa}, B_{wb} and B_{wc}, whose sum equals the maximum channel transmission $B_w(B_{wa}+B_{wb}+B_{wc}=B_w)$. Each traffic class has a separate buffer. The GPS assumption is shown in Figure 4.9. It is an ideal scheduling where, in practice, each single buffer has a dedicated scheduler.

The real situation is shown in Figure 4.10. The channel is shared among the three buffers and there is one scheduler that needs to approach the ideal situation through the quantities $\bar{B}_{wx}^{\text{real}}$, whose average values are equal to the reference values: $\bar{B}_{wx}^{\text{real}}=\lim_{T\to\infty}\frac{1}{T}\int_0^T B_{wx}(t)\mathrm{d}t=B_{wx}$ where $x\in\{a,b,c\}$. The problem is that the equivalence is not guaranteed for any instant of time. A good approximation is represented by WRR. It assigns a weight to each queue and serves the queue in a round robin mode ruled by the weights. The weights for the example under discussion are B_{wa}/B_w, B_{wb}/B_w and B_{wc}/B_w, for buffer a, b and c. A possible way to implement WRR is to keep track of the real bandwidth portion received by a buffer at

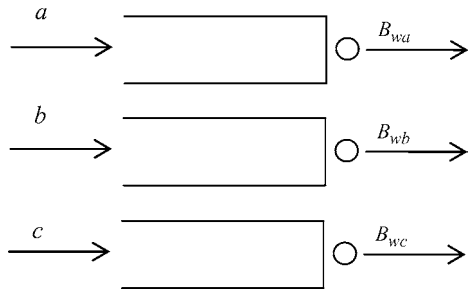


Figure 4.9 GPS scheduling

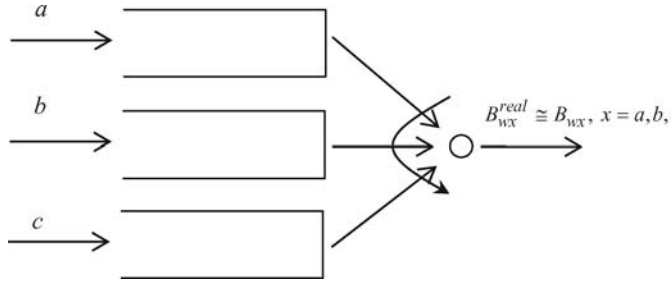


Figure 4.10 Real scheduling

a given decision instant (t_d) and to compare it with the reference value. Following again the example, if $B_{wx}^{real} = \frac{1}{t_d}\int_0^{t_d} B_{wx}(t)dt$, $x \in \{a, b, c\}$ are the real average bandwidths received by buffer a, b and c, the related normalized portions of the overall bandwidth at time t_d are $\frac{B_{wx}^{real}}{B_w}$, $x \in \{a, b, c\}$. When the scheduler must decide, at time t_d, which is the next buffer to serve, a good way to implement WRR is to compare the following quantities $\left|\frac{B_{wx}^{real}}{B_w} - \frac{B_{wx}}{B_w}\right|$, $x \in \{a, b, c\}$, to select the largest one and to serve the corresponding buffer. More formally, at each decision time (t_d), the buffer to be served among the three alternatives of the example is given by $x : \arg\max_x \left|\frac{B_{wx}^{real}(t_d)}{B_w} - \frac{B_{wx}}{B_w}\right|$, $x \in \{a, b, c\}$. In other more simple words, the buffer which is furthest from the percentage of overall bandwidth that should be ideally obtained at any time instant is currently served. The WFQ family of schedulers (WF^2Q, and WF^2Q+) acts in a similar way also trying to minimize the computational effort.

4.1.6 Queue Management

Scheduling is often linked to queue (buffer) management schemes. They are used, for example, to establish the dropping strategy when the buffer is full. There are three possibilities:

1. Tail drop, which discharges the last arrived packets;
2. Front drop, which eliminates the first packet in the queue;
3. Random drop, which discharges a randomly selected packet within the queue.

There are also dynamic schemes to drop packets before saturation. Random Early Detection – RED [Floyd93] – is an algorithm really implemented in commercial routers: it detects incipient congestion by computing the average queue size. If the average queue size exceeds a fixed threshold, RED drops or marks each arriving packet with a certain probability, where the exact probability is a function of the average queue size. The scheme is thought to help transport layer protocols such as TCP, to avoid entering severe congestion. There are some modifications of the original scheme. Weighted Random Early Detection (WRED) drops packets selectively based on their priority (typically IP Precedence) by following the rule that packets with a higher IP precedence are less likely to be dropped than packets with a lower precedence. Flow-based WRED is a particular feature of WRED, which forces it to afford greater fairness to all flows depending on how packets are dropped [Ciscodocumentation].

4.1.7 Flow Control

4.1.7.1 Introduction to Flow Control and to TCP

In some cases, the bit rate entering the network may be ruled according to a congestion notification (ECN – Explicit Congestion Notification). Some protocols (e.g. TCP) consider packet loss as a congestion indication. Generally, flow control is implemented end-to-end at the transport layer (but some mechanisms are implemented at the application layer); even if it may help avoiding network saturation, it cannot guarantee, if used alone, a specific QoS requirement. It may be used mostly to improve the performance of best-effort traffic. The action at this layer by modifying algorithms and tuning parameters has a big effect on the performance in radio and satellite networks. Its study is very important for the definition of the gateway introduced in Chapter 2. Details about it will be given in Chapter 10. As said, if the application requires a safe transmission, TCP/IP-based networks typically use Transmission Control Protocol – TCP – to guarantee a safe and transparent transmission to the application layer. TCP is a connection-oriented, end-to-end reliable transport protocol working between hosts in packet-switched networks, and between interconnected systems of such networks.

TCP assumes that it can obtain a simple, potentially unreliable service from the lower level protocols and, in principle, the TCP should be able to operate above a wide spectrum of communication systems ranging from hard-wired connections to wireless and satellite networks. Nevertheless, its use within a satellite environment, even if it does not affect the correct working of the protocol, heavily affects the performance, as it will be shown in Chapter 12. It is set just above the IP, which offers a service to the TCP to send and receive information segments of variable length called “datagram” in the IP terminology. Being a network layer, the IP manages also issues of addressing and routing of the information. It may fragment TCP segments, which have to traverse portions of networks with different characteristics at the data link layer. The primary purpose of the TCP is to provide reliable service between pairs of host computers (actually their processes running on the hosts because TCP is supposed to be a module within an operating system but the term host simplifies the comprehension). To match the requirement on top of a less reliable Internet communication, it uses facilities in the following areas: basic data transfer, reliability, flow control, multiplexing, connections, precedence and security. TCP provides a means to rule the amount of data sent by the sender, applying an Automatic Repeat reQuest (ARQ) system.

It assigns a sequence number to each segment and uses acknowledgement packets that flow from the destination to the source to be sure the information is arrived at the destination. More exactly, that it is arrived at the TCP process by the destination host computer. Conceptually, each segment is assigned a sequence number. The sequence number of the segment is transmitted with the segment itself. When the TCP transmits a segment, it puts a copy on a retransmission queue and starts a timer called "Retransmission Timeout" (RTO). When the acknowledgement for that segment is received, the segment is deleted from the queue. If the acknowledgement is not received before the timer runs out, the segment is retransmitted. The flow control mechanism employed is explained in detail in Chapter 12. In general, TCP uses a "window", which specifies the number of data that can be sent. Flow control may also be efficiently applied at application layer. Two examples about it may also be provided to individuate possible future research activities. They are identified as CCSDS and application layer coder control in the following.

4.1.7.2 CCSDS

In special environment, TCP/IP protocol architectures have great limits when applied in hazardous environments characterized by long propagation delays, intermittent, asymmetric links and frequent transmission errors. The Consultative Committee has designed a protocol architecture alternative to TCP/IP stack for Space Data Systems (CCSDS), taking the satellite technology as reference for its clear benefits concerning the wide area coverage and its inherent capability of broadcast/multicast communications. CFDP (CCSDS File Delivery Protocol) protocol has been designed to run at the application layer of a full CCSDS protocol stack and to manage the transmission of data over satellite networks in space missions. In practice, the aim of TCP about guaranteeing a safe and transparent transmission to the application layer may be moved to the application itself, which, possibly relying on unsafe transport layers, takes charge of safety of the transmission.

CFDP protocol splits the data to be sent into blocks (namely CFDP PDUs), composed of a payload, carrying a maximum information amount of 65,536 bytes, and of a header whose length is assumed to be equal to 20 bytes. Furthermore, the protocol is responsible of segmenting the CFDP PDU into smaller data units if the maximum dimension of packets allowed in the network is lower than the size of the blocks.

The protocol works both in reliable and in unreliable mode. Concerning the latter, the delivery of data has to be guaranteed by other entities, typically resident in underlying layers (e.g. transport and data link layers), if necessary. Concerning the former, the reliability request of the communication is guaranteed by a recovery mechanism based on negative acknowledgement notification located within CFDP itself. Nevertheless, even if efficient, acknowledgement-based schemes are not always feasible in real environments and need to be substituted by alternatives that, on one hand, improves the communication reliability but, on the other hand, require no explicit requests of retransmissions.

CCSDS protocols may be integrated with coding schemes implemented at the application layer, adopted over the highly asymmetric and intermittent link satellite environment mentioned above. In order to cope with the impairments introduced by such links, it is preferable to adopt error correction codes implemented within CFDP (CCSDS File Delivery Protocol) specification, rather than ARQ schemes because the high propagation delays along with the intermittence of transmission links would imply very long recovery periods. The

trade-off is represented by the implementation cost and by the waste of network resources (namely channel bandwidth) because of the encoding operations.

The adoption of the CFDP protocol combined with the use of erasure coding schemes may be applied to data communication achieved between inter-planetary satellite platforms.

4.1.7.3 Application Layer Coder Control

Essential information is that the rate entering the network may be adapted in dependence of the network state by reducing/increasing rate not only at the transport layer but also at the application layer in connection with a non-interfering transport layer protocol (e.g. UDP, which does not perform any control). Commonly used applications send voice and video. In these cases the coder represents an important component. The control mechanism may receive indications about the network performance and adapt the source rate by changing either the coder parameters or the entire coder [BollaMarcheseZap]. In this framework the Perceived-QoS (e.g. the real user perception) is used to perform the best coder choice. The communication scheme is depicted in Figure 4.11. It is divided into different operating blocks, of which the topical ones are the "coding agents" and the "congestion controllers". The "congestion controller" detects a congestion situation and decides the bit rate to use for the transmission. The choice is based on feedback information got from the receiver and is aimed at reducing future congestion. Feedback information should concern the packet loss rate and the jitter. The ideal bit rate for the outgoing flow is stated at fixed time intervals. The selected bit rate is communicated to the "coding agent": this block chooses an audio–video coder with transmission characteristics as closer as possible to the ideal outgoing bit rate. As already stated, it is possible to use a subjective evaluation of the quality based on the Mean Opinion Score (MOS), explained in Chapter 1, to select the coder. Actually, many coding factors have a role in the possible improvement of user perception. For example, in a video service, objective parameters that directly influence the user perception are summarized in Figure 4.12. They are classified in the following categories: Application Type, Network Performance and Application Tunable Parameters which can be controlled through the coder implementation. Similar comments may be given concerning audio transmission. The set of controllable parameters is identified as a "configuration" and a specific MOS may

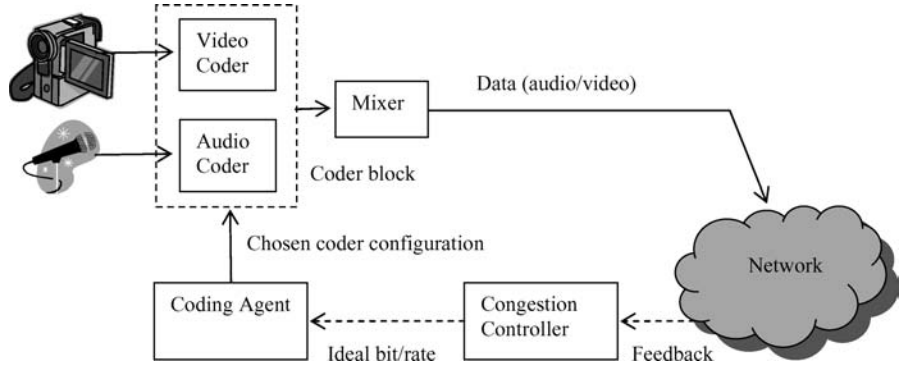


Figure 4.11 Application layer coder control

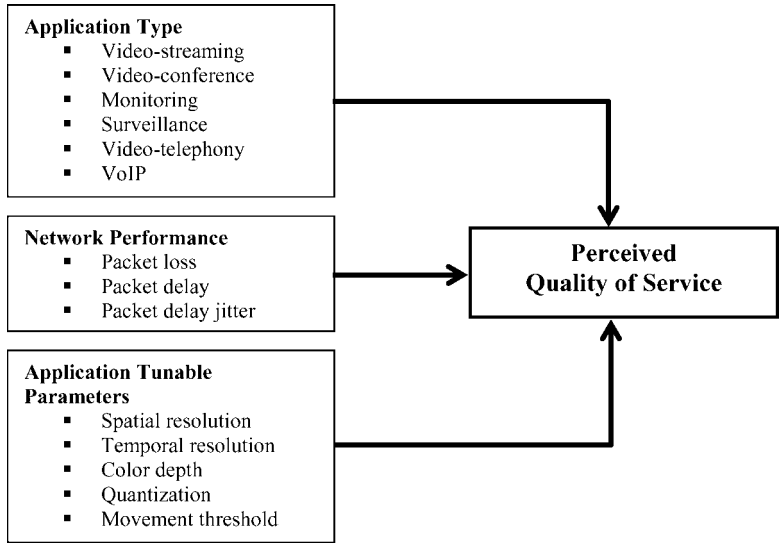


Figure 4.12 Parameters influencing user perception (video transmission)

be associated with it. The configuration table can be computed off-line and used by the application. The "coding agent" block of Figure 4.11 has the role to choose the configuration with the highest MOS among the feasible ones. The set of feasible configurations depends on the congestion of the network, measured by the "congestion control" whose feedback is sent to the "coding agent".

4.1.8 QoS Routing

Packet routing decisions are often taken with little or no awareness of network state and resource availability. This is not compatible with QoS provision. With the progress in standardization and the increasing deployment of switching nodes, the issue of routing in QoS-oriented networks has found its place among the various controls that are necessary to maintain QoS requirements to the different traffic types. Typical controls are CAC, shaping and other algorithms reported above. Actually, the routing problem cannot be considered as separated from the other controls, but it should be integrated with them. Moreover, the particular nature of QoS-oriented traffic-engineered networks, where the dynamic structure of statistical multiplexing must be combined with resource allocation techniques, allows getting suggestions from the vast previous experience in routing from both circuit- and packet-switched networks (see, e.g., [Girard], [Bertsekas] and [Ross]). Among the various control approaches (especially concerning CAC), the schemes seen in this chapter concerning CAC may be applied within each node and extended to get a QoS-oriented routing policy. It is important to highlight that QoS routing needs to be used in conjunction with resource reservation to guarantee necessary network resources along the path, so that QoS requirements can be satisfied. It means to apply traffic engineering techniques.

QoS routing needs to identify end-to-end paths where there are enough available resources to guarantee performance requirements in terms of metrics as loss, delay, call blocking,

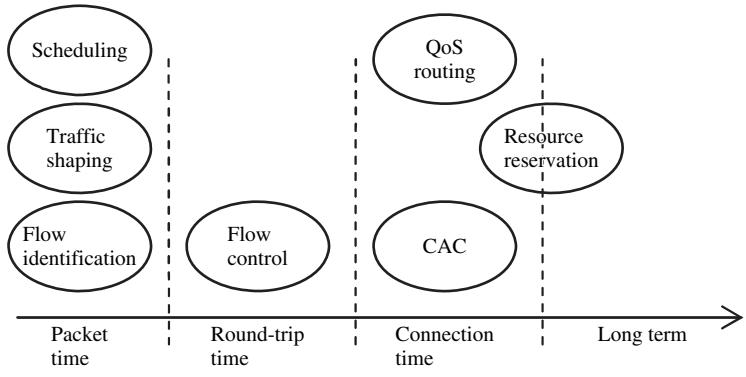


Figure 4.13 QoS management functions versus time

number of hops, reliability, as well as bandwidth optimization. The path selection process involves the knowledge of QoS requirements and information about the availability of network resources, which varies frequently. The knowledge about network state is conveyed through proper signalling protocols that need to be distinguished from the protocols necessary to transmit QoS requests, well defined in Chapter 6. Compared with shortest-path routing that chooses the best routes based on either hop count or cost, QoS routing exploits complex path computation and signalling traffic.

The intervention timescale of the mentioned functions is different. Figure 4.13 shows a possible time mapping for them.

4.2 The Risk of No Control

The aim of this section is to show the effect of the presence of some specific management functions. More than privileging a technology versus another one, it is important to know which functions a solution may use, which performance limits the functions have and what the user expectations can be. The following issues will be investigated by showing some performance results concerning Traffic Aggregation (CAC), Traffic shaping and Buffer and Bandwidth Allocation. The real limitations are not imposed only by the technology, whose features may be changed and extended, but by the application of control functions that can guarantee a degree of service, which supports the application needs in terms of QoS. The discussion has been carried out without any reference to current fashion trends (as, for instance, ATM some years ago and IP now). In other words, even if the choice of a specific technology may help simplify QoS provision, the focus of this section is about the need to implement control mechanisms in the network. There is no guaranteed QoS provision without control, independently of the technology chosen. The reminder of the chapter should allow also individuating what can be done to improve QoS provision in future. To get the results shown in the following, each single network node is modelled as a battery of buffers that receive the traffic entering the node as shown in Figures 4.9 and 4.10. An ideal GPS scheduler is supposed. Most of the results reported concern a single node. It does not affect the generality of the presentation and of the approach because the overall network may be considered as a chain of nodes (buffers) and the qualitative observations derived from the

results of a single node may be extended to the overall network. Obviously, the quantitative values change but, quantitatively, the comments have a general validity.

Additionally, it is true that, often, the overall end-to-end path is also modelled as a single buffer-server scheme. Even if it is an approximation of the real behaviour, it seems acceptable over label-switched networks, where the concatenation of identifiers establish the identity of a single connection and where resources are allocated from the point where the label is assigned to the point where it is removed. This discussion, even if very interesting, is outside the scope of this book. The same comments may be applied to buffer dimension. It will be the object of tests concerning resource allocation. The author, in this chapter, would like to raise an alarm concerning the risk of no control. The number of buffers, their dimension, how the packets entering the node are forwarded within the node and how traffic flows are associated with the different buffers concern the management schemes of the nodes and the chosen resource allocation algorithm.

4.2.1 Flow Identification

The impact of flow identification over performance is linked to the possible need, in case of limited bandwidth availability for each single connection in progress, to have a detailed traffic differentiation. In other words, in particular cases, often linked to applications in hazardous environments, it is necessary to identify each single user flow or, at least, to aggregate traffic in strictly homogeneous classes where all the flows contained in a class are characterized by the same traffic descriptors and by the same Service Level Specification (SLS). If the technology can identify only a limited number of traffic classes, the risk is the aggregation of non-homogeneous traffic. The following examples should allow a deeper comprehension.

Figure 4.14 shows two traffic classes (identified as black and grey) distinguished by using the traffic identifier in the packet header and forwarded to two different buffers within a network node. Each class is characterized by a specific SLS, called “black SLS” and “grey SLS”, and it is composed of homogeneous traffic sources. Obviously, the two aggregates are supposed to be carried through a technology that allows a separate identification of the two classes. Each traffic class will receive a specific bandwidth provision so that the SLS is respected. On the other hand, Figure 4.15 shows the same situation but using a technology that does not allow to implement two separate traffic classes. They are totally aggregated and no distinction can be done even if the original performance requirements

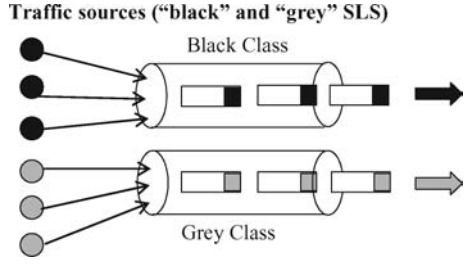


Figure 4.14 Distinguished traffic classes

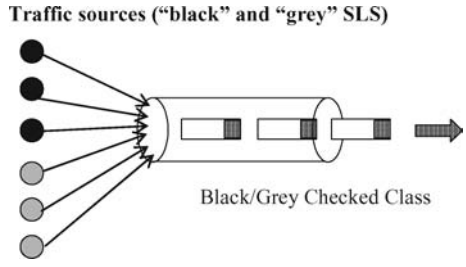


Figure 4.15 Aggregate flows

are different. A checked label identifies all the traffic and the distinction between them is lost. Several important questions arise. Which SLS will be assigned to the "checked class"? The most severe between the "black" and the "grey" SLS? The less severe? An SLS in the middle between the two? How much bandwidth should be assigned to the "checked class"? Obviously, if the choice for the "checked" SLS is the most severe between "black" and "grey", both SLSs will be respected, but how much bandwidth will be wasted for that? On the other hand, fixing the SLS to the less severe one, what is the measure of the SLS violation for the user flows belonging to the other class?

In general, if traffic requiring different performance is joined in one class, it is necessary to investigate the additional bandwidth required to keep the same performance level. The environments that use a limited number of classes with respect to the technologies where a very large number of traffic classes are available may represent an example. In practice, due to the limited number of traffic classes, QoS-non-homogeneous traffic classes (i.e. composed of flows requiring diverse QoS) need to be aggregated and conveyed together. The example of Figures 4.14 and 4.15 is related to two traffic classes but, obviously, the more relevant QoS problems arise where there is a great number of aggregate flows requiring different SLSs.

In this perspective, the following simulation results regard the aggregation effect on performance for traffic requiring different QoS constraints, in terms of packet loss, packet delay and delay jitter. They allow having also a measure of the bandwidth necessary to guarantee the required performance and, in particular, a precise idea of the mentioned traffic aggregation problem.

If traffic needs to be aggregated, the choice of the bandwidth to be assigned to guarantee the QoS is topical. It will be the object of a specific section in the following. The relevant metric, in this case, is the measure of bandwidth increase (or reduction) necessary to keep the same level of service when traffic flows are aggregated in comparison with a complete separation. The parameter used in this work is the gain, defined as the percentage difference between the overall bandwidth necessary to satisfy the requirements if the flows are kept separated (Figure 4.14) and the bandwidth needed by the complete aggregation (Figure 4.15). For example, if the flows belonging to the black class need 1.0 Mbps to satisfy the requirements and the flows belonging to the grey class need 2.0 Mbps, when kept separated, if the aggregation of the two flows requires 4.0 Mbps to offer the same performance, the defined gain is $100 \cdot \frac{(1+2)-4}{(1+2)} = -33.33\%$. It means that, in this example from scratch, the aggregation of non-homogeneous traffic is not convenient and that 33% more bandwidth is necessary to guarantee the fixed requirements. A deeper performance investigation is reported in the

following. Many studies confirm the efficiency of aggregating QoS-homogeneous traffic but the performance of QoS-non-homogeneous trunks is still a very interesting open research issue.

Concerning the tests reported, performed by a simulator studied and built for the aim, the application level generates on–off sources whose traffic descriptors are peak bandwidth (Mbps or Kbps), mean burst duration (s) and mean silence duration (s). Burst and silence durations are both Pareto distributed. The packet length is fixed to 53 bytes. It implies the use of ATM but, in this case, the technology is not an issue. Independently of it, it is important to check the effect of traffic aggregation on the performance. Obviously, if the transport technology changes (e.g. IP instead of ATM), the specific numerical values change but the concepts are still the same. A buffer of 5.3 KB for each single traffic class has been chosen to model the bandwidth pipes in Figure 4.14. When the flows are aggregated, the buffer length is scaled up (e.g. in case of two flows, as used in these tests, is set to 10.6 KB) to keep the same storage capacity. The buffer and the server, whose service rate is the assigned bandwidth to the buffer, may represent both a single network node and the overall end-to-end path, where the QoS requirements are satisfied, as highlighted above.

Different simplified SLSs have been used in the tests. The first part of them has been performed with the traffic flows appearing in Tables 4.1 and 4.2 and by supposing that the two traffic classes need to be aggregated because there are not enough classes to be assigned. They differ only for the Packet Loss Rate (PLR) parameter.

The result heavily depends on the composition of the aggregate trunk.

Table 4.1 Black class SLS, packet loss test

Service Level Specification	Range
Premium VBR	Variable Bit Rate (VBR)
Traffic description and conformance testing	Packet dimension: 424 bit; Peak rate: 1.0 Mbps; Average Rate: 500 Kbps;
Performance guarantees	Packet loss rate: 10^{-4}; Packet transfer delay: not specified; Packet delay jitter: not specified.

Table 4.2 Grey class SLS, packet loss test

Service Level Specification	Range
Premium VBR	Variable Bit Rate (VBR)
Traffic description and conformance testing	Packet dimension: 424 bit; Peak rate: 1.0 Mbps; Average rate: 500 Kbps;
Performance guarantees	Packet loss rate: 10^{-2}; Packet transfer delay: not specified; Packet delay jitter: not specified.

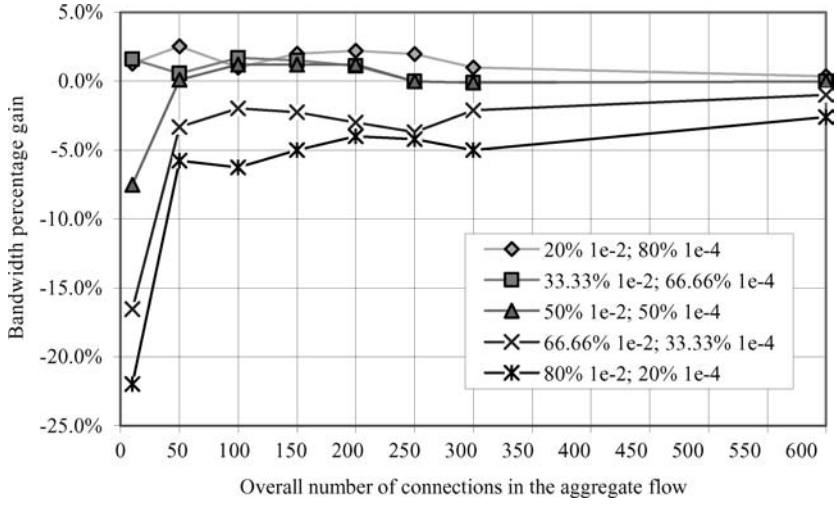


Figure 4.16 Bandwidth percentage gain in traffic aggregation: The Packet Loss Rate case

Figure 4.16 contains the aforementioned bandwidth gain by varying

- the number of connections within the aggregate trunk;
- the percentage of connections belonging to the two traffic classes requiring, respectively, a PLR of 10^{-2} and 10^{-4}. For instance, the percentage 33.33% and 66.66% stand for 1/3 and 2/3, respectively, so as to get 100% of traffic and so on.

Obviously, the relative amount of black and grey traffic is topical to understand the problem.

The SLS value (i.e. the PLR performance requirement) for the aggregate trunk is set to 10^{-4} (the most severe between the two) in order to assure that the overall trunk is guaranteed. It is worth noting again that the packets of the two traffic classes are no longer distinguished within the trunk.

In this case, non-homogeneous aggregation is often convenient (see the "20% 10^{-2}; 80% 10^{-4}" "33.33% 10^{-2}; 66.66% 10^{-4}" cases) but, if traffic is unbalanced towards the less restrictive traffic (the "66.66% 10^{-2}; 33.33% 10^{-4}" and "80% 10^{-2}; 20% 10^{-4}" cases), a bandwidth portion is wasted to guarantee the specified performance. The results reported above not only arise an alarm concerning the possible risk, which is the main aim of the paragraph, but also give an operative solution to operate bandwidth dimensioning, that is how much bandwidth needs to be allocated to the aggregate class (the checked class in Figure 4.15) so that the users are not penalized.

The trend is even clearer if the QoS differentiation stands in the Packet Transfer Delay (PTD) constraint (Tables 4.3 and 4.4). The performance constraints fix the PLR to 10^{-2} and differentiate traffic flows by imposing a network node (buffer) delay transfer of 10 and 50 ms.

Even in this case, the more restrictive constraint is chosen for the overall trunk (i.e. 10 ms, requested by the black class). Figure 4.17 contains three cases: (1) 20% of traffic requires 50 ms of PTD and 80% requires 10 ms; (2) traffic requiring 10 and 50 ms equally shares the

Table 4.3 Black class SLS, packet delay test

Service level specification	Range
Premium VBR	Variable Bit Rate (VBR)
Traffic description and conformance testing	Packet dimension: 424 bit; Peak rate: 16 Kbps; Average rate: 8 Kbps;
Performance guarantees	Packet loss rate: 10^{-2}; Packet transfer delay: 10 ms; Packet delay jitter: not specified.

Table 4.4 Grey class SLS, packet delay test

Service level specification	Range
Premium VBR	Variable Bit Rate (VBR)
Traffic description and conformance testing	Packet dimension: 424 bit; Peak rate: 16 Kbps; Average rate: 8 Kbps;
Performance guarantees	Packet loss rate: 10^{-2}; Packet transfer delay: 50 ms; Packet delay jitter: not specified.

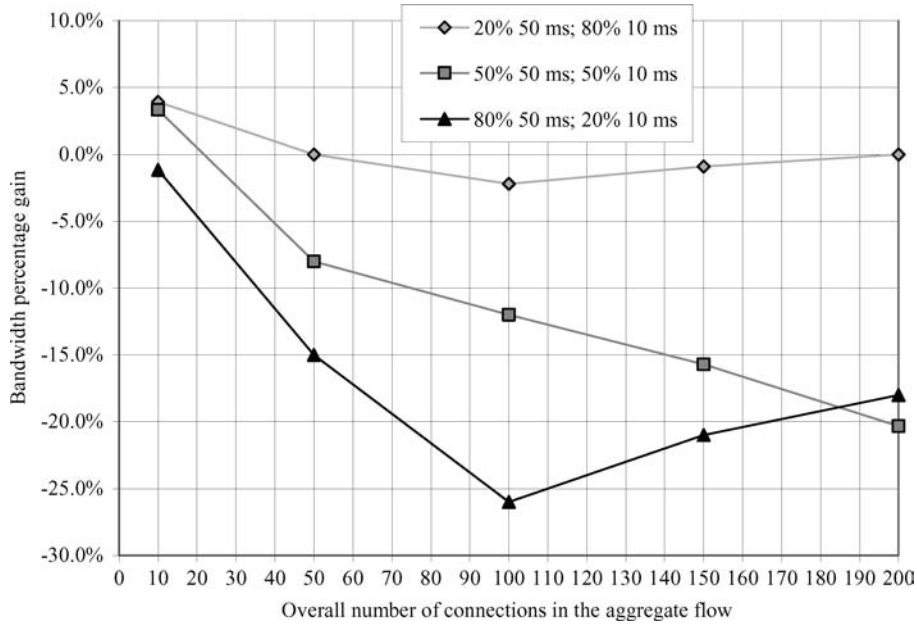


Figure 4.17 Bandwidth percentage gain in traffic aggregation: The Packet Transfer Delay case

Table 4.5 Black class SLS, jitter test

Service level specification	Range
Premium VBR	Variable Bit Rate (VBR)
Traffic description and conformance testing	Packet dimension: 424 bit; Peak rate: 16.0 Kbps; Average rate: 8.0 Kbps;
Performance guarantees	Packet loss rate: 10^{-2}; Packet transfer delay: 50 ms; Packet delay jitter: 5 ms.

Table 4.6 Grey class SLS, jitter test

Service level specification	Range
Premium VBR	Variable Bit Rate (VBR)
Traffic description and conformance testing	Packet dimension: 424 bit; Peak rate: 16.0 Kbps; Average rate: 8.0 Kbps;
Performance guarantees	Packet loss rate: 10^{-2}; Packet transfer delay: 50 ms; Packet delay jitter: 9 ms.

buffer and (3) traffic whose constraint is 50 ms is dominant representing 80% of the load. The percentage of additional bandwidth required to guarantee the fixed performance for the aggregate class is relevant in the last two cases: also when most demanding (10 ms) traffic is 50% of the overall load, more than 10% of additional bandwidth is required when there are 100 connections, more than 15% with 150 connections and 20% with 200 connections.

Similar comments may be made if a constraint on jitter is used (Tables 4.5 and 4.6) to differentiate traffic. A packet loss of 10^{-2}, a transfer delay of 50 ms and a delay jitter of 9 and 5 ms are imposed. Figure 4.18 reports the results by changing the percentage of 9 and 5 ms traffic in the overall load.

The behaviour is similar to the transfer delay case but it is worth noting that, in this case, the quality differentiation (5 and 9 ms) is really minimal. It means that jitter is a very sensible parameter for bandwidth reservation and a very critical issue for aggregation.

All the results reported have value of examples, but they should help understand which the real problem of traffic aggregation is.

4.2.2 CAC

Other operative examples may help point out the importance of CAC function and the effect of implementing (or not) CAC. The following three data traffics are used. Traffic model and buffer dimension are the same as used in the traffic aggregation analysis. There are three traffic classes, identified as 1, 2 and 3. Class 1 imposes a constraint only on PLR of 10^{-4}.

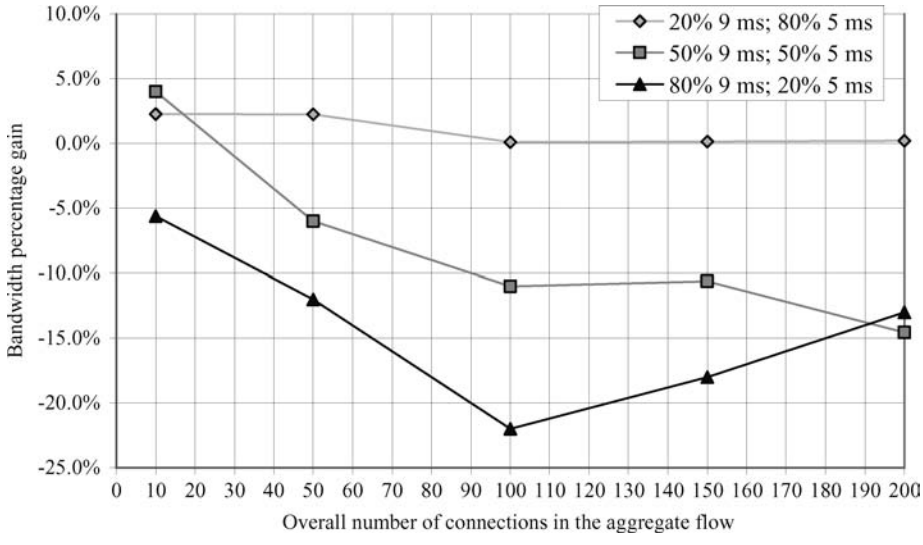


Figure 4.18 Bandwidth percentage gain in traffic aggregation: The packet delay jitter case

Class 2, additionally to packet loss (10^{-2}), puts a constraint also on transfer delay, set to 10 ms. Class 3 constraints the delay jitter below 5 ms, as well as loss and delay, set, respectively, to 10^{-2} and 10 ms. The SLS of Classes 1, 2 and 3 is reported in Tables 4.7–4.9, respectively. Traffic of each class enters a separate buffer.

Table 4.7 Class 1 – SLS

Service level specification	Range
Mission critical data	Variable Bit Rate (VBR)
Traffic description and conformance testing	Peak rate: 1 Mbps Average rate: 500 Kbps
Performance guarantees	Packet loss rate: 10^{-4}; Packet transfer delay: not specified; Packet delay jitter: not specified.

Table 4.8 Class 2 – SLS

Service level specification	Range
Mission critical data	Variable Bit Rate (VBR)
Traffic description and conformance testing	Peak rate: 16 Kbps Average rate: 8 Kbps
Performance guarantees	Packet loss rate: 10^{-2}; Packet transfer delay: 10 ms; Packet delay jitter: not specified.

Table 4.9 Class 3 – SLS

Service level specification	Range
Mission critical data	Variable Bit Rate (VBR)
Traffic description and conformance testing	Peak rate: 16 Kbps Average rate: 8 Kbps
Performance guarantees	Packet loss rate: 10^{-2}; Packet transfer delay: 10 ms; Packet delay jitter: 5 ms.

The results show what happens to the performance guarantees if no CAC is implemented, if the requests of new connections to enter the network are not limited. The reference "0%" in the figures from 4.19 to 4.25 identifies the situation where traffic entering the network is blocked by CAC and, consequently, the bandwidth has been properly dimensioned and the quality guaranteed. The vertical value in the figures identifies the percentage of traffic above the acceptance threshold (i.e. the percentage of connections that would have been dropped in case of CAC) up to 50% of unbalance. The number of connections ranges from 20 to 300 for Class 1 tests and from 20 to 120 for Classes 2 and 3 tests.

Figure 4.19 reports the values of the measured PLR for "Traffic Class 1", as well as Figures 4.20 and 4.21, which contain the same values but show a limited portion of the results: 0–20% for Figure 4.20 and 0–5% for Figure 4.21. Restricting the portion of overloading has the effect of magnifying lens and allows understanding the effect also with small overloading.

Figures 4.22 and 4.23 show the tests for "Traffic Class 2", focusing, respectively, on 0–50% and 0–30% of non-controlled traffic and reporting the measured PTD.

Figure 4.24 concerns "Traffic Class 3" and shows the Packet Delay Jitter (PDJ), as well as Figure 4.25 that contains the "magnifying lens" for the 0–20% range.

The numerical values reported are outstanding and any additional comment would be probably redundant: small variations of the number of accepted calls with respect to the reference value (that guarantees the required QoS) have a heavy effect on the traffic flow

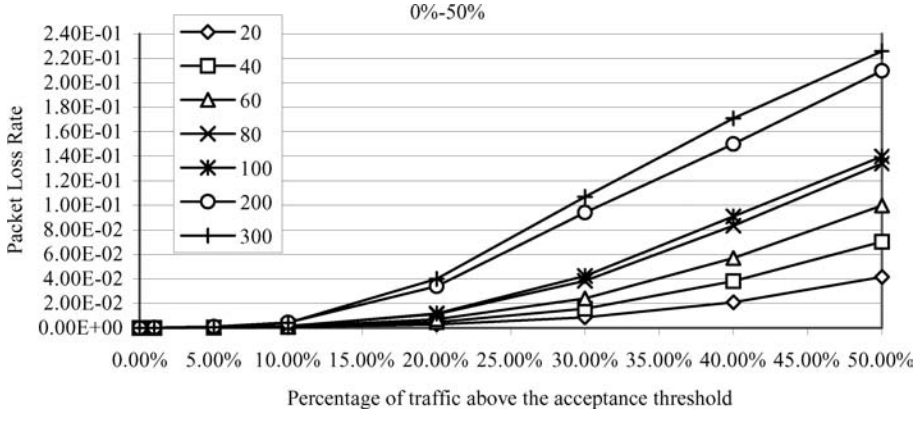


Figure 4.19 Traffic Class 1, 0–50%

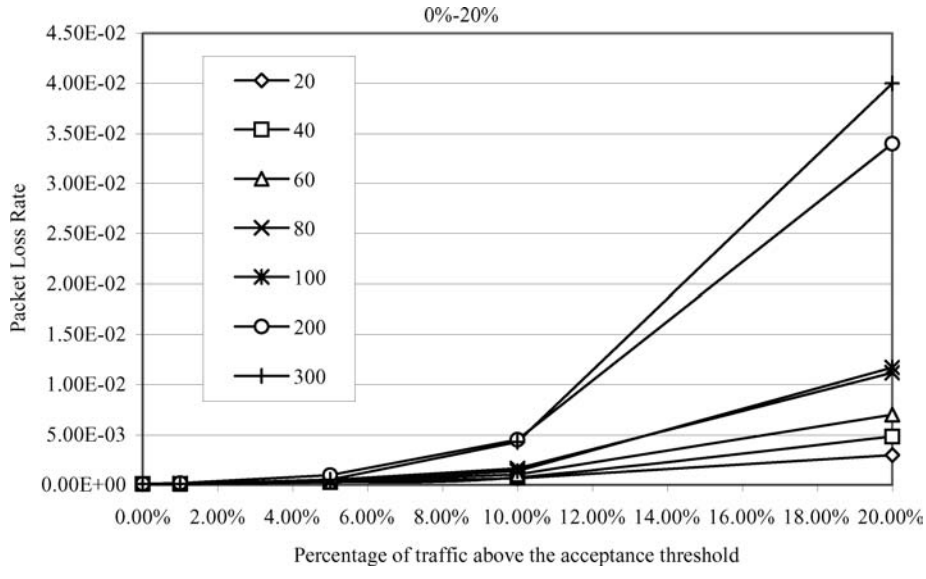


Figure 4.20 Traffic Class 1, 0–20%

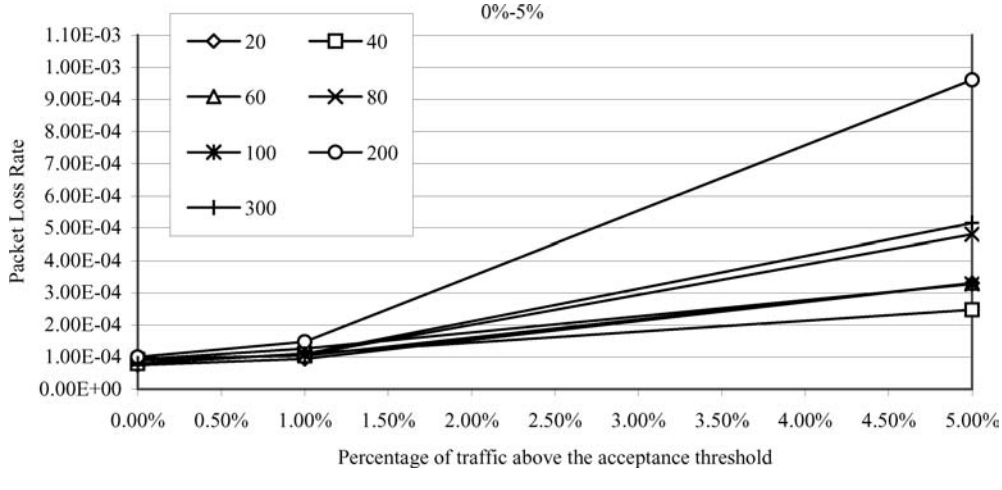


Figure 4.21 Traffic Class 1, 0–5%

performance. The reported values allow measuring the performance decrease. The conclusion is that no guarantee is possible if no CAC is implemented. It should bring new light also on the problems that a pure priority mechanism, not joined to any QoS architecture and without CAC control and, at least, per-class bandwidth allocation, can have (as said during the discussion about DiffServ).

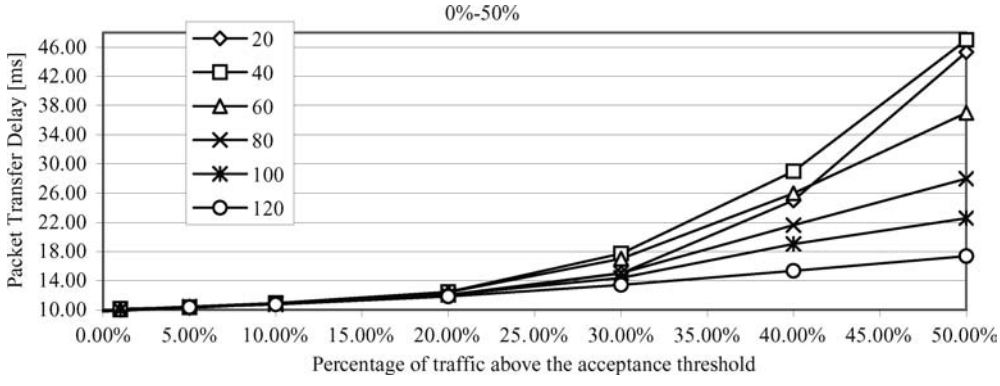


Figure 4.22 Traffic Class 2, 0–50%

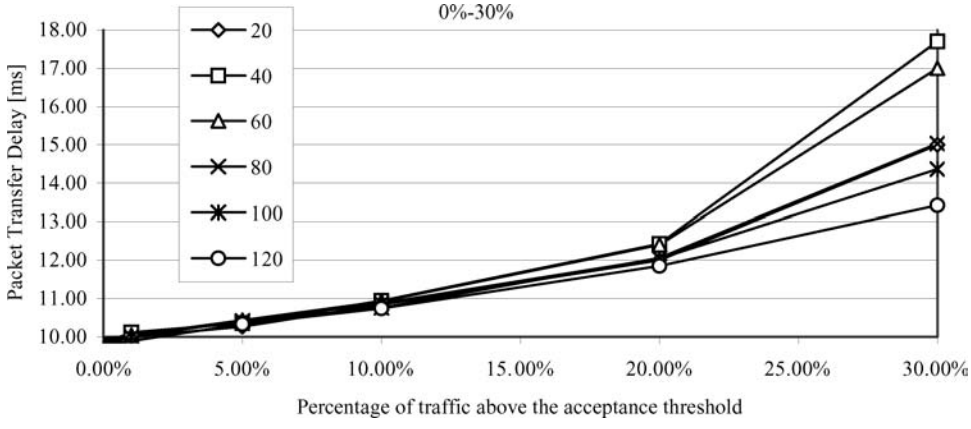


Figure 4.23 Traffic Class 2, 0–30%

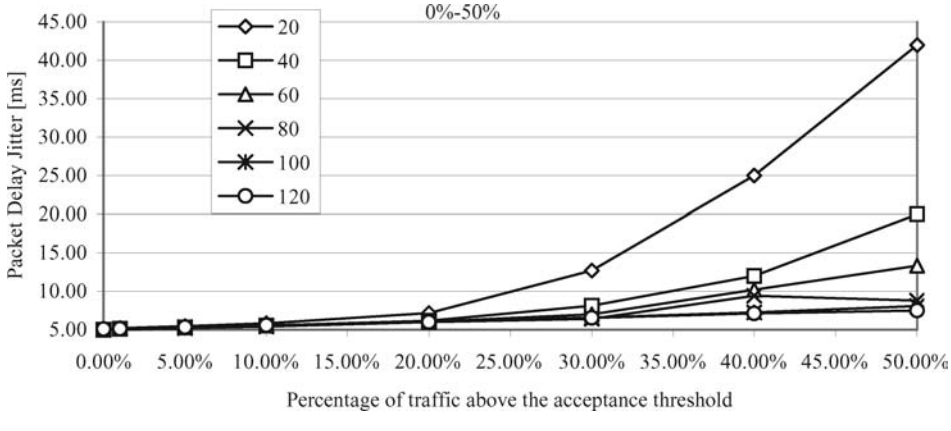


Figure 4.24 Traffic Class 3, 0–50%

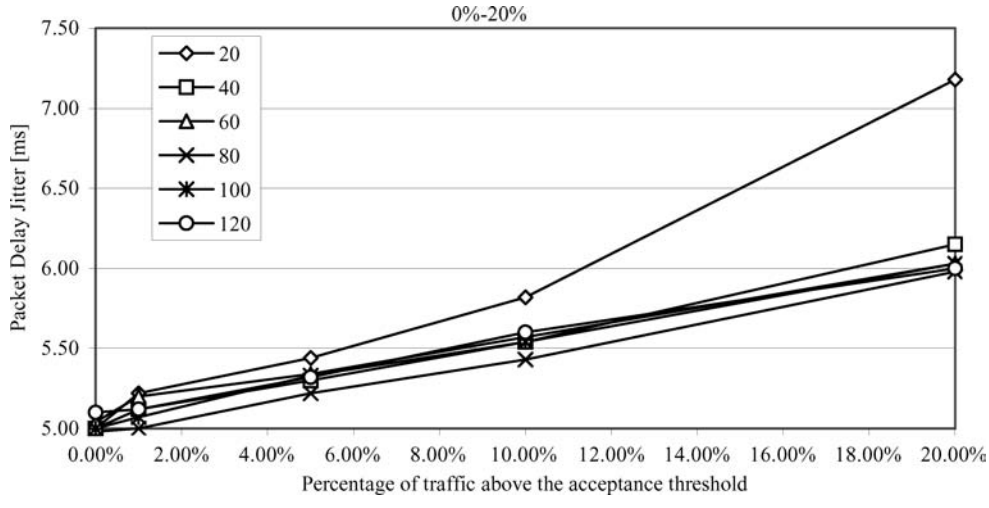


Figure 4.25 Traffic Class 3, 0–20%

4.2.3 Shaping

Using the same method based on numerical examples, it is important to emphasize the role of traffic shaping to keep the traffic conformant with the traffic contract. The reference architecture is Figure 4.6, except for the number of flows, which is set to 100 in the numerical results in the following.

The example includes a *Voice over IP* (VoIP) traffic flow as in Table 4.10. Taking [Byungsuk02] as a reference, each VoIP source is modelled as an exponentially modulated on–off process. The mean on and off times, as in ITU-T recommendation P.59 ([ITU-T-P.59]), are 1.008 s and 1.587 s, respectively.

All VoIP flows are modelled as 16 Kbps voice over RTP/UDP/IP. Without header compression, RTP+UDP+IP headers impose 40 bytes of overhead to each VoIP packet. The 40-byte

Table 4.10 VoIP *Service Level Specification* (SLS) ([ITU-T-P.59])

Service Level Specification	Range
Premium VBR for *Voice over IP*	Variable Bit Rate (VBR)
Scope	End-to-end
Connection identification	Sequence of identifiers
Traffic description and conformance testing of VoIP	Peak rate: 16 kbps; Mean rate: 6.22 kbps; Packet size: 80 bytes; Maximum burst size: 1.008 s Bucket size for peak rate: 16 kbps;
Performance guarantees	Packet loss rate: 2%; Packet transfer delay: 50 ms; Packet delay jitter: not applicable

overhead may be reduced to 4 bytes by using header compression. An IP packet size of 80 bytes is taken as a reference. The required performance objectives of a VoIP flow at network node level are PLR below 2% and PTD below 50 ms.

The shaper guarantees that the peak rate does not exceed the declared threshold (i.e. 16.0 Kbps). The example of aggregate traffic is composed by 100 VoIP flows and the buffer size is set to 8 KB (100 VoIP packets). If traffic filtering is not active, not expected "non-conformant traffic" may significantly affect the overall performance similarly as the CAC case presented before. The measure of the performance degradation is the aim of the reported results.

Tables 4.11–4.13 depict the QoS degradation as a function of the percentage of non-conformant traffic, taking the PLR and the PTD as metrics. Three different degradations scales are considered, since the non-conformant traffic is supposed to generate packets at the following rates: 20.0, 30.0 and 50.0 Kbps.

The ideal situation (active shaper) corresponds to the "0%" case, where non-conformant connections are shaped before entering the network and reduced to the committed rates of 16 Kbps. In this case, the QoS requirements are guaranteed because the bandwidth is supposed to be correctly dimensioned. Numerical values depicted in Tables 4.11–4.13 highlight the performance decrease in case of no shaper availability. If non-conformant traffic peak rate is 20 Kbps, 10% of not expected connections (e.g. 10 more connections at 20 Kbps peak rate, corresponding to a maximum of additional unexpected 200 Kbps out of 1600 Kbps maximum expected) are sufficient to violate the PLR performance requirement. If not expected connections are more than 20% of the overall load, the impact on performance is dramatic, both for packet loss and for delay. Twenty per cent of the not expected connections generates

Table 4.11 VoIP traffic, active shaper versus non-active shaper, non-conformant peak rate: 20 Kbps

Non-conformant connections	PLR	PTD [ms]
0%	1.86E-02	41.366
10%	2.38E-02	45.815
20%	4.25E-02	54.622
50%	7.72E-02	68.181
80%	1.23E-01	78.395
90%	1.45E-01	79.464

Table 4.12 VoIP traffic, active shaper versus non-active shaper, non-conformant peak rate: 30 Kbps

Non-conformant connections	PLR	PTD [ms]
0%	1.86E-02	41.366
10%	6.26E-02	61.715
20%	1.11E-01	74.831
50%	2.73E-01	89.064
80%	3.83E-01	89.648
90%	4.11E-01	89.136

Table 4.13 VoIP traffic, active shaper versus non-active shaper, non-conformant peak rate: 50 Kbps

Non-conformant connections	PLR	PTD [ms]
0%	1.86E-02	41.366
10%	1.44E-01	77.447
20%	2.69E-01	87.998
50%	4.92E-01	93.221
80%	6.16E-01	93.189
90%	6.51E-01	93.003

a performance decrease of an order of magnitude, for PLR, and of 25 ms, for "per-hop" PTD, out of a constraint of 50 ms, if non-conformant traffic has a peak rate of 30 Kbps. A 10% of non-conformance decreases the performance, concerning PLR, of more than an order of magnitude and, concerning PTD, of 27 ms (over the given per-hop threshold), when non-conformant peak rate reaches 50 Kbps.

It is simple to figure out the effect on a voice call that requires a packet loss of 10^{-2} as well as a delay below 50 ms, if it receives a loss of 10^{-1} and a delay of about 80 ms per hop. In practice, there is no service.

4.2.4 Resource Allocation

To be completely clear, the role of bandwidth (and, implicitly, of buffer) can be seen also by the figures already presented in this chapter, concerning both CAC and shaping. The increase of the packet loss, of the packet delay and jitter due to the non-controlled traffic entering the network is due to missing bandwidth. The performance decrease caused by not shaped connections is again due to missing bandwidth. In both cases, the unexpected traffic entering the network has created a consequent need of additional bandwidth.

On the other hand, the results presented up to now were not focused on bandwidth dimensioning because the aim was to show the effect of not controlled load. The reminder of the chapter is aimed at showing the need of correct bandwidth and buffer dimensioning to get specific performance requirements.

The same data of the previous examples about "shaping" are used. The chosen traffic is VoIP. The packet loss threshold is fixed to $2 \cdot 10^{-2}$. The number of calls (flows) is 100 and the buffer dimension is 5300 bytes. To keep the packet loss under threshold, a minimum bandwidth of 656 Kbps needs to be allocated to the buffer server. The computation has been performed through simulations and a bandwidth step of 3.2 Kbps has been used.

Figure 4.26 shows what happens to the packet loss rate if the allocated bandwidth is varied of a percentage with respect to 656 Kbps but the load is kept fixed to 100 flows. The origin is set to the reference case (100 calls, 656 Kbps allocated), where the mean packet loss is 1.86E-02.

The values reported in the figure allow focusing on the importance of a correct bandwidth dimension. If it is varied of only 10% more, the effect over packet loss is outstanding (2.08E-03). In practice, there is one order of magnitude difference. The same comment may be reported if bandwidth less than 656 Kbps is provided. Ten per cent less (identified as

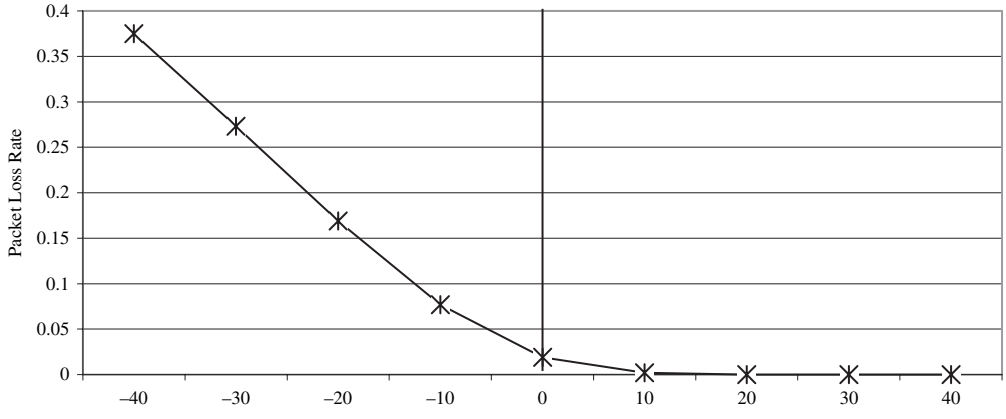


Figure 4.26 Packet loss rate versus varied percentage of allocated bandwidth

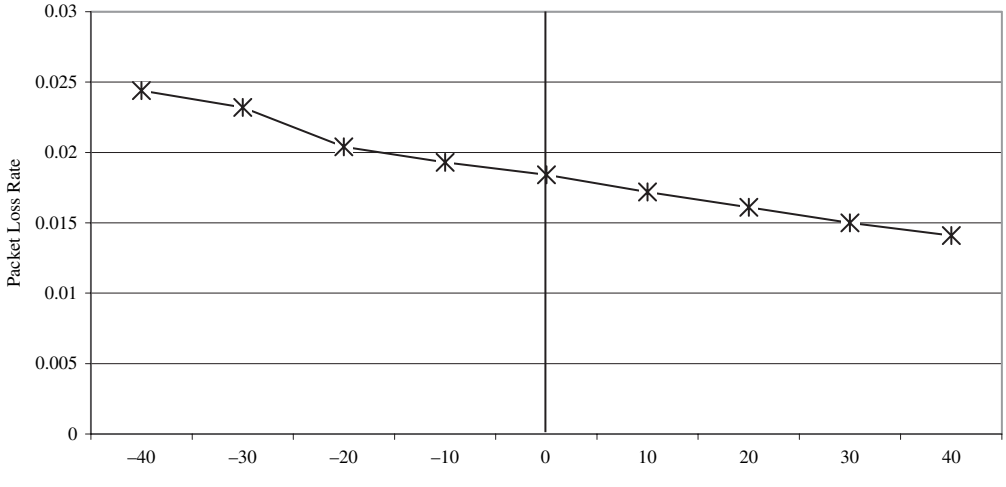


Figure 4.27 Packet loss rate versus varied percentage of buffer dimension

−10% in Figure 4.26) is enough to make the packet loss jump to 7.68E-02, well above the threshold. Resource tuning is essential for QoS provision. It is true not only for bandwidth but also for buffer. Figure 4.27 shows the packet loss rate by varying the buffer length of a percentage with respect to the reference (5300 bytes used for the origin). Buffer tuning is a delicate matter. Even if the effect of the variations are less evident than for bandwidth, it is important to note that decreasing the buffer from 5300 to 4240 bytes (−20%, in Figure 4.27) is sufficient to overcome the loss threshold.

5

QoS Interworking in Heterogeneous Networks

5.1 Scenarios and Problems

The concept of heterogeneity has been introduced in Chapter 2 and a possible future network shown in Figure 2.3. Even if the final aim is to provide end-to-end QoS to each single customer or to groups of customers, the problem of QoS over heterogeneous networks may be divided into different steps:

1. QoS requests (SLSs) should traverse the overall network from the source to the destination through portions that implement different technologies (e.g. cable, wireless and satellite) and different protocols (e.g. TCP/IP stack, ATM stack and DVB stack).
2. QoS requests should be received and understood by a specific portion (QoS may have a different meaning and interpretation depending on the used protocol and network).
3. SLS requests should be mapped on the specific technology stack of a portion by activating control mechanisms suited for the aim.
4. Each single stack is composed of layered architectures and each layer must have a specific role in QoS provision.

The idea is that each single portion deserves a specific QoS-oriented solution but this concept should be completely transparent to the final users.

Theoretically, the overall problem of QoS interworking may be structured into two different areas:

1. Vertical QoS mapping, which includes steps 3 and 4.
2. Horizontal QoS mapping, which includes steps 1 and 2.

QoS over Heterogeneous Networks Mario Marchese

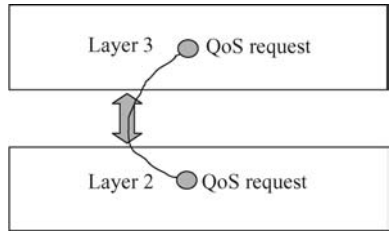


Figure 5.1 Vertical mapping

The concept of vertical QoS mapping is based on the idea that a telecommunication network is composed of functional layers and that each single layer must have a role for end-to-end QoS provision. The overall QoS depends on the QoS achieved at each layer of the network and it is based on the functions performed at the layer interfaces. Figure 5.1 shows the idea focusing on interface between Layers 2 and 3, evidenced by a double arrow: even if Layer 3 implements efficient QoS mechanisms, it is topical that Layer 2 can assure a specific service to Layer 3, otherwise the implementation of complex QoS mechanism at Layer 3 is useless. QoS requirements flow vertically and need to be received, understood and satisfied by the layer below.

The concept of horizontal QoS mapping, even if much linked to the previous concept when implemented in the field, is depicted in Figure 5.2. The need is to transfer QoS requirements among network portions implementing their own technologies and protocols. Three different networks are represented in Figure 5.2. They use three different QoS technologies: IP IntServ, ISDN and ATM.

A host generating a connection must receive a specific quality based on the SLS. The connection, depicted with the grey curve in Figure 5.2, traverses three different network portions which are, actually, Autonomous Systems (ASs) in the wider meaning assumed in Chapter 2. When the different ASs meet (at the arrows), it is important to map the request of quality from one network to another so that the final user can receive the negotiated quality transparently. The task is difficult and its solution implies the introduction of additional hardware and/or software and signalling protocols. It implies an efficient

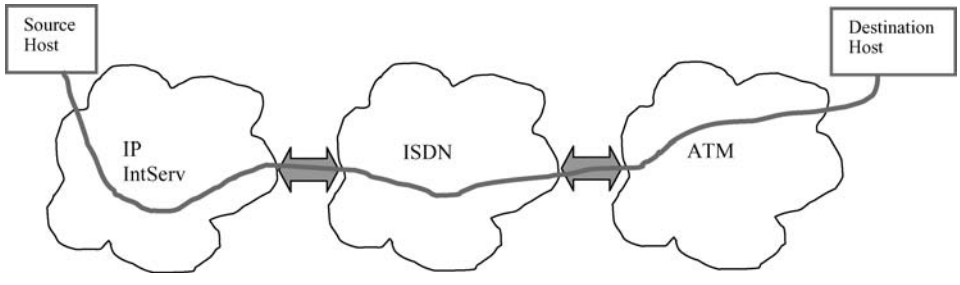


Figure 5.2 Horizontal Qos mapping

implementation of vertical mapping and also additional features. It will be treated in detail in the following.

5.2 Vertical QoS Mapping

5.2.1 Information Transport Technologies

A QoS-based service derives from reliable physical layers (including, in this case, Layers 2 and 1) that can offer specific transport service to the upper layers. "It is up to the service provider to determine the Layer 2 encapsulation and layer 1 infrastructure" [Meddeb].

Figure 5.3 reports a graphical model of the relation between lower (physical) layers and higher layers. The flows (or bundles of them) are forwarded down to physical interfaces that transport the information along the channel. Some transport technologies, whose short description is reported in the following, also appear in Figure 5.3. Some of them, as TDM in telephone networks, may also be applied on end-to-end basis but this brief and not exhaustive summary is aimed at concentrating on link-by-link use, where the techniques are exploited only to physically connected network devices.

5.2.1.1 Time Division Multiplexing (TDM)

Time Division Multiplexing infrastructure was designed for telephone networks. The channel resource is divided into portions of time called "frames" and each of them is structured in smaller portions called "slots". Each slot (or group of slots) is assigned to a user who maintains the assignation for the entire duration of the connection. The resource allocation is static but it should not be confused with the equivalent end-to-end transfer mode, where TDM is used to install a circuit between the source and the destination, which will be dropped at the end of the communication. The focus of this book, as said above, is on link-by-link technology: using TDM means essentially having a time-slotted channel.

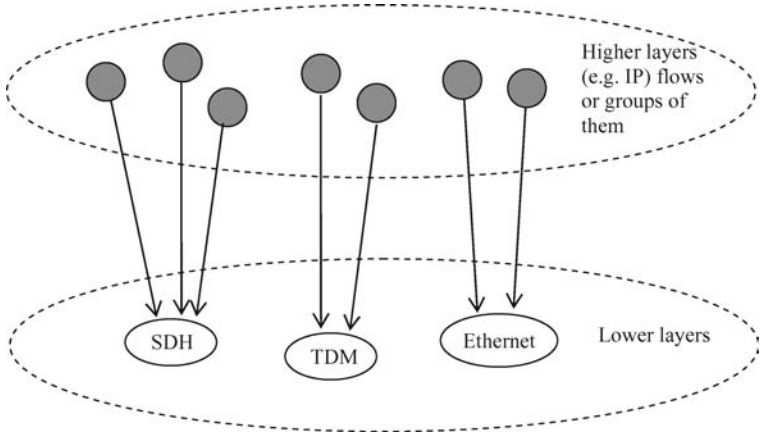


Figure 5.3 Higher layers over transport technologies

5.2.1.2 Dynamic Transfer Mode (DTM)

The concept is the same as TDM: the channel is divided into frames (one each 125 microseconds) and each frame in slots (of 64 bits). The slots may be either "control slots", located at the beginning of the frame, or "data slots". The main difference with TDM is that time slots may be dynamically allocated even during a communication, either automatically or upon an explicit user request issued by an involved party over the signalling channel. In other words, while TDM time slots are allocated in the call set-up phase and maintained in number and identity until call clearing, DTM allows dynamic bandwidth tuning during communications.

5.2.1.3 Synchronous Digital Hierarchy / Synchronous Optical Network (SDH/SONET)

Synchronous Digital Hierarchy (SDH) is a hierarchy of synchronous time-multiplexed transmission. In North America, an equivalent technology is called "SONET". SDH has been standardized in [ITU-T-G.707]. The basic SDH transmission signal is called "STM-1", whose bit rate is 155.520 Mbps; higher interfaces are obtained by scaling forward the basic rate n-times ($n = 4, 16, 64$), so obtaining STM-4 (622.080 Mbps), STM-16 (2488.320 Mbps) and STM-64 at 9953.280 Mbps. The SDH format is the standardized transport frame for ATM at physical layer: the 53-byte cells are transported over the STM payload.

5.2.1.4 Ethernet

The main interest concerning Ethernet (a technology designed for LANs and characterized by a pure FIFO queue management) is the recent possibility of implementing classes of service (CoS) at the Layer 2. CoS (being focused on traffic differentiation and related priorities) does not directly imply QoS (that requires more control functions), but it is very important. The extension of the basic Ethernet standard for CoS is defined in IEEE 802.1Q [IEEE802.1Q] and 802.1p [IEEE802.1p]. In addition to the standard Ethernet frame, 802.1Q adds a 4-byte TAG in the header, which includes a priority field of 3 bits. The 3 bits of the priority guarantee a minimum traffic differentiation whose default priorities are defined in IEEE 802.1p. Due to speed increase, from 10 Mbps to 100 Mbps (Fast Ethernet), up to 1 Gbps, 10 Gbps and 100 Gbps (Gigabit Ethernet), this solution is increasingly being used for transportation technology. Recently [Allan Bragg] [Elengonov], Ethernet has been presented as carrier transport networking infrastructure enhancing scalability and capability by using larger address spaces to provide backbone bridged networks [IEEE802.1ah], where, similarly as in MPLS, frame tagging mechanisms are used to distinguish different flows or groups of flows over a common shared channel. A recent special issue of the *Communications Magazine* [EthernetTransport] was totally dedicated to the present and future of Ethernet Transport over WAN.

5.2.1.5 ATM and Frame Relay

ATM (and also MPLS) technology has not been defined to be a physical layer transport technology (see Figure 5.3) but, sometimes, it is seen as a Layer 2 because it is much used in core networks [Bocci03] to transport other technologies (e.g. IP). Frame Relay, similarly to ATM, identifies virtual circuits through Data Link Channel Identifier (DLCI).

5.2.2 *Formal Relation Among the Layers*

An example of formal relation among layers and a clear example of vertical QoS mapping is represented by the protocol architecture proposed by ETSI [ETSI-TR101] for the access points to a Broadband Satellite Multimedia (BSM) network portion. The architecture is reported in Figure 5.4. The reference stack for the upper layers is the TCP/IP suite. The physical layers (i.e. satellite physical, MAC and Link Control, strictly satellite dependent) are isolated from the rest by a Satellite Independent Service Access Point (SI-SAP), which offers specific QoS services to the upper layers. Using a satellite network is only an example and the architecture can be generalized to include different physical supports, both wired and wireless. It is formalized for satellites but the general idea is not limited to them. The SD layers receive a service request from the SI layers through the SI-SAP. The request may be considered a small SLS and can be defined in objective terms by packet loss probability, delay probability and packet jitter. The service offered by the SI-SAP must be accurately defined to assure a QoS-based service. To keep in touch with a real technology development, the reference to Satellite Dependent (SD) and Satellite Independent (SI) layers will be often kept for the book but more general identifiers as Technology Dependent (TD) and Technology Independent (TI) layers will substitute them when more generalized approach is necessary for the treated topic.

The ETSI definition implies the involvement of management, control and user planes within the framework of horizontal mapping and QoS architectures. Moreover, it should act together with resource allocation schemes. Some details will be given in the following

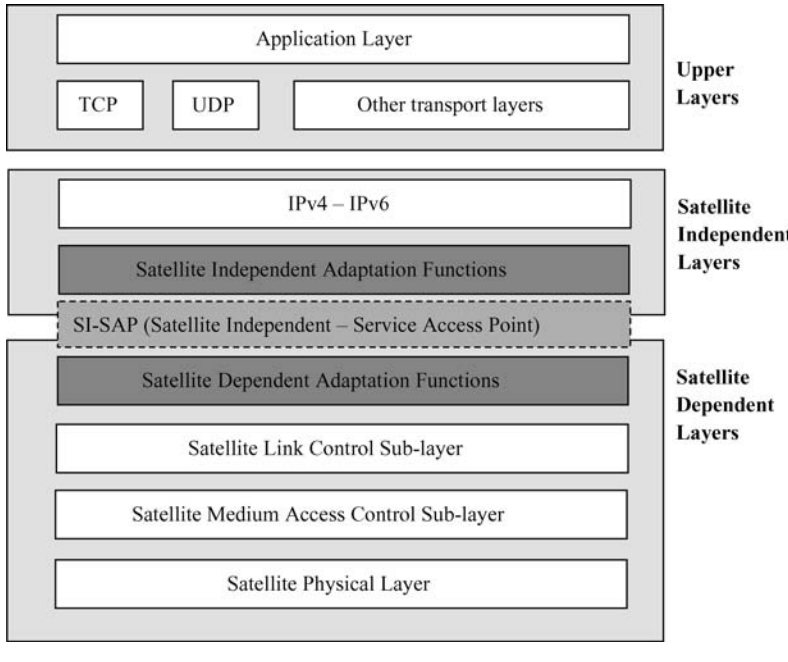


Figure 5.4 ETSI BSM protocol architecture

sections. Now it is important to focus on the vertical QoS mapping, that is on the service the lower layers can offer to the upper layers through the SI-SAP.

In such a context, the interworking between SI and SD layers (or, more generically, TI and TD layers) reveals to be a hot topic of research. There are two main problems: (1) the change of information unit (encapsulation) and (2) the need to aggregate traffic. In case of a wireless/satellite network, there is a third serious problem: the presence of fading conditions and the need to counteract it. Actually, for wireless communication, the bandwidth assigned to the traffic at SI layer for QoS reason (e.g. to keep the packet loss probability of SI buffers below a given threshold) is not the real physical bandwidth seen by the buffer server because of the high variability of the physical medium state (i.e. the bandwidth of the air interface). An example concerning problem (1) is the transport of IP traffic over a BSM portion that uses a different transport technology at the SD layer (e.g. ATM [Schmitt03], as often implemented in industrial systems [Lutz99], DVB, whose application still involves standardization details). A special algorithm to solve these open issues will be detailed in Chapter 9.

Concerning traffic aggregation, as clearly indicated in [ETSI-TS102],

> it is accepted in the BSM industry that at the IP level (above the SI-SAP interface) between 4 and 16 queues are manageable for different IP classes. Below the SI-SAP these classes can further be mapped into the satellite dependent priorities within the BSM which can be from 2 to 4 generally.

Therefore, even if all the QoS paradigms described in Chapter 3 could be used, including IntServ, DiffServ approach is, in practice, the business and implementation choice for the BSM SI layer. Concerning SD layer, the due association of IP QoS classes to SD transfer capabilities (e.g. ATM and MPLS queues, DVB request categories) is limited by hardware implementation constraints. This limit may be overcome in the future but it is a fact for now.

A queue model describing the QoS mapping operation performed at the SI-SAP interface, as similarly used in [ETSI-ETR003] and [Combes04] (for DVB at the SD layer), is reported in Figure 5.5. The composition of Satellite Independent and Dependent Adaptation Functions is ignored for the sake of simplicity. Some more detail will be given in Chapter 8.

The first block (IP traffic classifier) assigns the arriving IP packet to a queue in dependence of the DSCP value. The example reports one Expedite Forwarding (EF) class, four Assured Forwarding (AF) classes and one BE class. Below the DiffServ scheduler that picks the packets up from the queues, an AAL5 adapter block encapsulates the IP packet within the AAL5 frame and creates the ATM cells. The ATM portion is SD and, in this example, implements three ATM queues acting at MAC layer: one queue is dedicated to EF traffic, one to all the AF and one to BE. Four AF flows are aggregated into a single SD queue, thus composing a heterogeneous aggregated trunk, possibly deriving from different traffic categories (e.g. voice, video and data). The problem of traffic aggregation analysed in Chapter 4 now rises from a technological architecture not for theoretical reasons but for practical implementations. The problem to be solved is how much bandwidth must to be assigned to each SD queue so that the SI IP-based SLS (i.e. the service expected, which flows from SI to SD through the SI-SAP) is guaranteed. It is the essence of vertical QoS mapping. Figure 5.6 sketches the bandwidth assignment problem by using the simple cascade

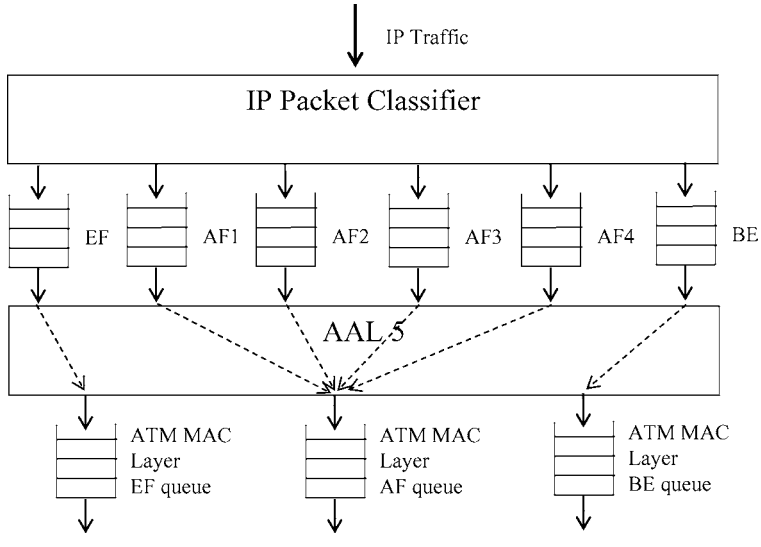


Figure 5.5 SI–SAP interface. SI layer (e.g. DiffServ) over SD layer (e.g. ATM)

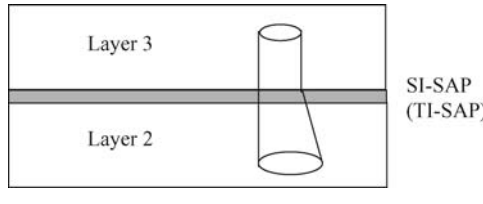


Figure 5.6 Bandwidth assignment problem

of two layers. If a certain amount of bandwidth is assigned to Layer 3 (e.g. to a traffic class of the Layer 3) so creating a bandwidth pipe, it is important, to keep at least the same level of quality to assign a bandwidth pipe also at Layer 2. Due to encapsulation and aggregation problem, the bandwidth pipe shall be enlarged.

As said, vertical QoS mapping is much linked to horizontal QoS mapping. Actually, each single element that will compose the end-to-end QoS architecture will include the concept of vertical QoS mapping, as should be clear from the next section.

5.3 Horizontal QoS Mapping

The main idea is to have a special gateway entering each single network portion (in practice each single Autonomous System) as defined in Chapter 2. It allows matching the QoS mapping problem but it also opens the door to a possible optimization of the overall network performance because a dedicated protocol stack may be implemented for a specific network portion. It will be the object of Chapter 10, in the final part of the book, and it will be called "Performance Optimization Through Layers" (POTL).

The gateway's position has been shown in Figure 2.3, where a general heterogeneous network is represented. In more detail, a possible composition of network portions (called "AS" here) connecting single LANs and Wide Area Networks (WANs) is shown in Figure 5.7. The technology chosen to guarantee services in AS A may be ATM, while AS B may be IP-based. AS C may implement an ISDN-based plain telephony backbone and AS D may have chosen MPLS. Single ASs may be also identified with single nations, as done in Figure 5.7. It is of main interest for military networks where the choice of a specific technology is made nationally but QoS strongly needs to be provided from the source to the destination. It was one of the objectives of the TACOMS Post 2000 group [Sorteberg05].

The problems, essentially, are (1) establish a proper interface among the network portions; (2) transfer the QoS needs for each end-to-end connection across the heterogeneous network and (3) once transferred the QoS requests among the network portions, it is topical to map the performance requests over the peculiar technology implemented within each AS. The latter is the link with vertical QoS mapping.

The interconnection interface between two different network portions is defined as Relay Point (RP) and it is reported with dark circles in Figure 5.8. At each RP the mentioned problems are solved by the gateway, called, from now on, "QoS gateway". The specific implementation of QoS gateways dedicated to horizontal QoS mapping will be called "QoS-Relay Node" (QoS-RN) or "QoS-Private Relay Node" (QoS-PRN), as better specified below. Anyway, even if the stress of the term "QoS gateway" is more on the architecture composition and "QoS-RN/-PRN" on implementation, the two words may be simply exchanged and they

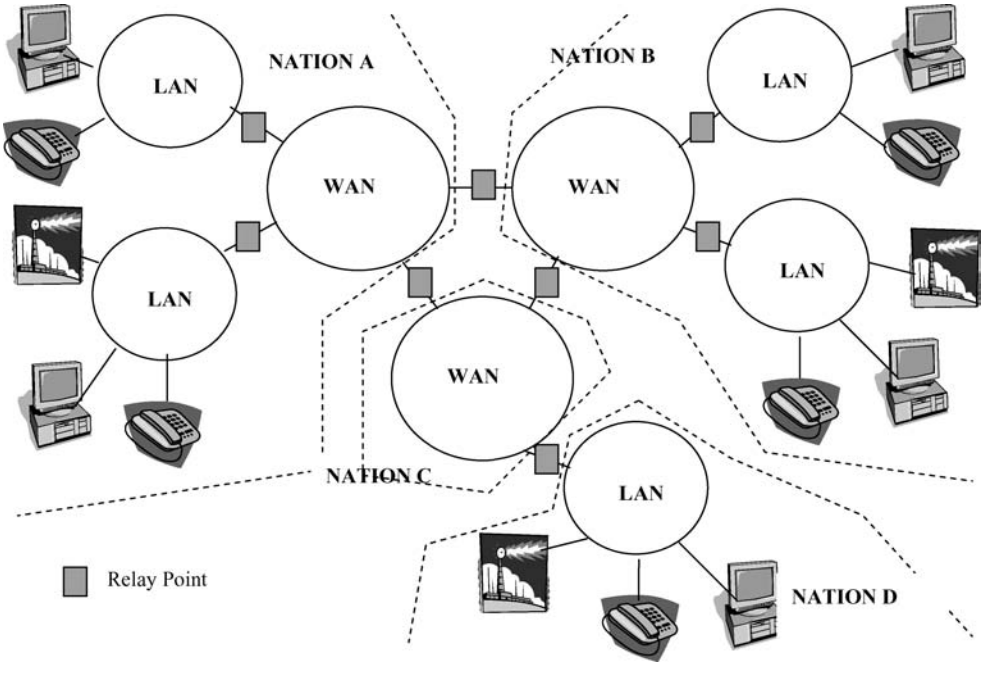


Figure 5.7 Interworking scenario among network portions (Autonomous Systems)

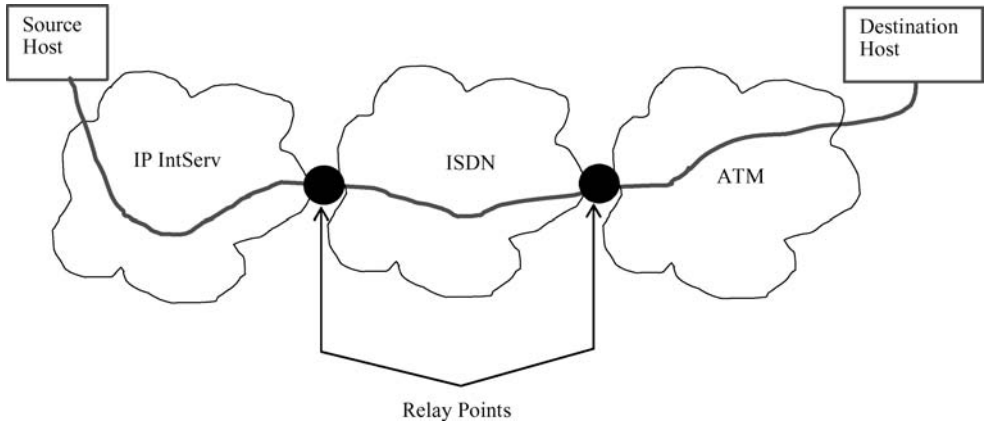


Figure 5.8 Formal definition of Relay Points

are used in the book with the same meaning. The need to keep a more general concept for QoS gateway is due to the larger applicability of it. As already said, having a gateway between two different network portions allows changing the entire protocol stack to get performance optimization. It is not the case here. The focus is only on QoS mapping for now. Chapter 10 will match this point.

It is topical now to focus on the possible architectural composition of QoS gateways. The interconnection may be composed either by single machine or by two machines connected through a simple physical support. The former is shown in Figure 5.9, for an IP/ATM Relay Point. It implies the definition of a special communication node as QoS gateway, the QoS-RN (Quality of Service – Relay Node), which should be defined in common by the two involved networks. It is not a problem when the portions are located within the same company or in similar environments but, when the ASs represent, for instance, different nations in a military network, a special hardware may be hardly shared. This case shown in Figure 5.10 implies the presence of a couple of communication nodes located within

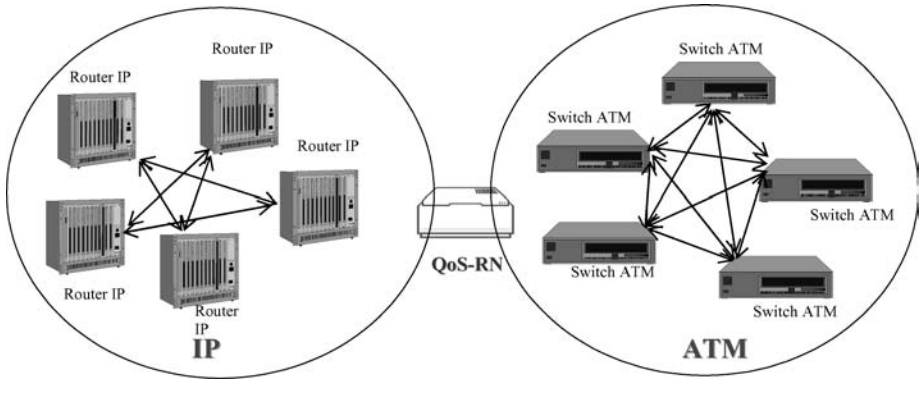


Figure 5.9 QoS-RN

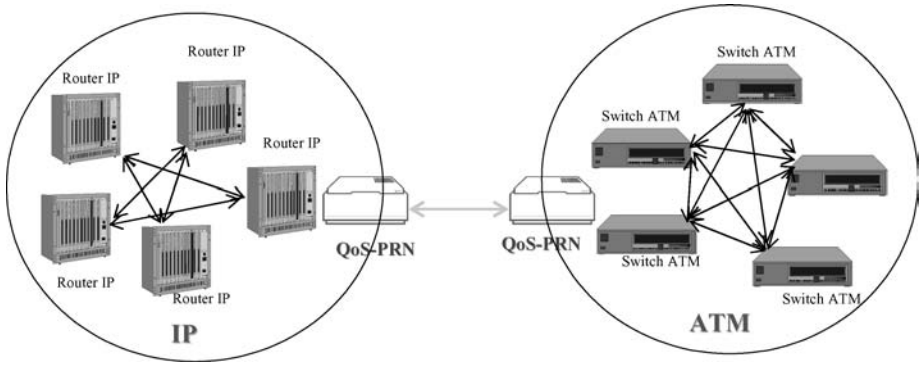


Figure 5.10 QoS-PRNs

the proprietary networks and defined as QoS-PRNs (Quality of Service – Private Relay Nodes). Even if the proprietary networks own the QoS-PRNs, it is always necessary to have common definitions, concerning protocols and interfaces. Only the property of the tool is preserved.

Figure 5.11 contains a possible architecture for QoS-RN. Again, the choice of IP and ATM is only an example and any technology may be used. An important element has been added: the Relay Layer (RL). Figure 5.12 contains an example of a possible architecture dedicated to QoS-PRN. In the case reported, an ATM-based subnet and an IP-based subnet are again interconnected, as in the previous case. The ATM subnet provides QoS through the ATM traffic contract and technology, while the IP-based technology bypasses the best effort service inherently offered by IP by using either the Integrated Services technique or the Differentiated Services approach, as seen in Chapter 3.

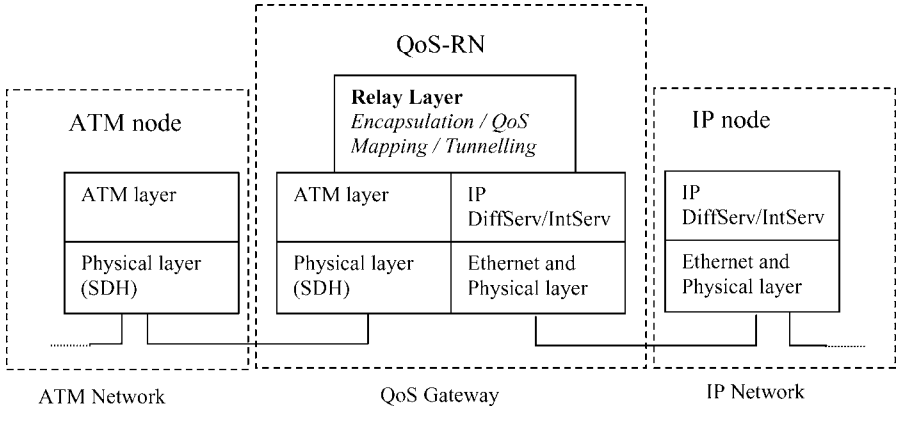


Figure 5.11 QoS-RN layered architecture

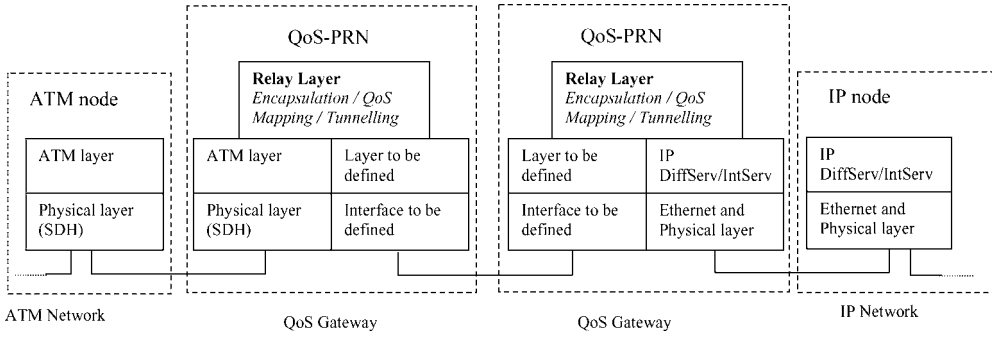


Figure 5.12 QoS-PRN layered architecture

The Relay Layer, in both cases, has the role of encapsulating information, establishing a common format and language to exchange information about QoS requirements, establishing tunnelling and implementing the other required functionalities (e.g. QoS routing if there are more than two QoS-RNs and QoS-PRNs attached together). Supposing that the ATM subnet requires a specific service for a connection in terms of peak and average bandwidth, packet loss and delay, the Relay Layer has to transfer this information through the interface to the IP-based subnet so that it can provide the required service by using, for example, Integrated Services. The two environments do not work with the same quantities and a "mapping service" is required. A relevant problem, for instance, is the effect of the packet/cell length on the bandwidth requirements. QoS-RNs and QoS-PRNs have essentially the same features which, in QoS-RNs, are composed in one machine. QoS-PRNs having a wider application that also includes military networks, they will receive deeper consideration in the book even if QoS-RNs will be constantly referenced. Once Relay Layers and their functionalities are defined, it is also important to define the interface layers for QoS-PRNs. They have been left without identification in Figure 5.12. Some possible alternative solutions, acting both at Layer 2 ("Layer to be defined" of Figure 5.12) and at Layer 1 ("Interface to be defined" in Figure 5.12) may be SDH/ATM, Ethernet/IP and MPLS. The latter is a very interesting solution. The possible reasons for such a choice are detailed in Chapter 6. Anyway, any Layer 2 transport technology and protocol [Meddeb] may be used.

QoS-RNs (-PRNs) are real QoS gateways because they act as routers but with additional functionalities. It is worth noting that, in the case shown, the two network portions operating together apply different technologies. So, not necessarily the information coming from one portion and directed to a specific destination is encapsulated in the other technology. There may be a pure change of protocol where the information of the upper layers (e.g. from transport layer onwards) acting only end-to-end is extracted at the Relay Layer and encapsulated again. It depends on the agreement about the possible recognized hosts. For example, if valid recognized hosts must have an IP address, the IP packet needs to be encapsulated within each technology when it traverses the different portions because it owns the address information to get to the destination. On the other hand, other possible hosts may be represented by ISDN, plain telephone and ATM addresses, among the others. In this case, a proper Relay Layer must implement a cross interchange among this technologies so as to assure correct routing of information.

Figure 5.13 shows the QoS-PRN architecture by using two generic proprietary technologies of WANs 1 and 2 and by referencing to generic WANs, as indicated in Figure 5.7. The proprietary choices are simply indicated as WAN-dependent technology layer (tech, in the figure) and WAN-dependent physical layer (phy, in the figure). The generic figure for QoS-RN is shown in Figure 5.14.

In practice, what is proposed here using Relay Layers is a Meta-network. The Relay Layers communicate inter-peers through a common signalling protocol carrying the QoS requests contained in the SLS/SLA. The QoS requests will be mapped onto the dependent technologies giving origin to a vertical QoS problem. The data traffic flowing in the Relay Layer will be encapsulated into the different technologies.

The mentioned problem of host definition is not trivial because it heavily depends on the specific investments made by the AS providers. It is true in particular in the military environment, where, actually, each single AS corresponds to a nation. It is obvious that, if a nation has made heavy investments in the ATM technology, it would like to have ATM as reference choice also concerning hosts. It is also true for IP and ISDN, to mention the most

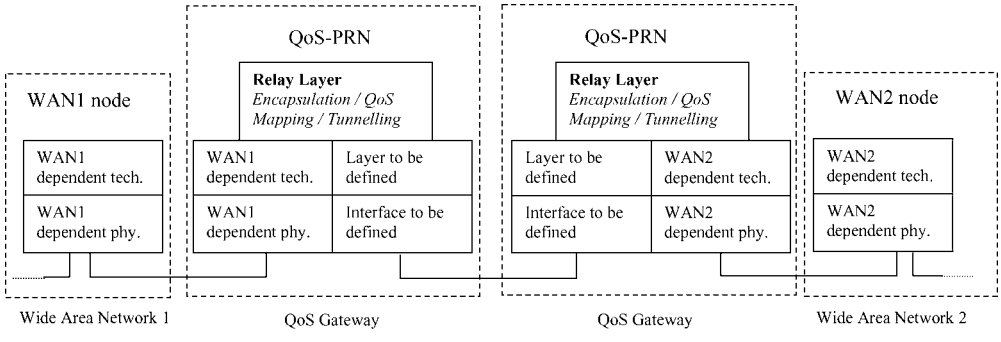


Figure 5.13 QoS-PRN architecture for generic WANs

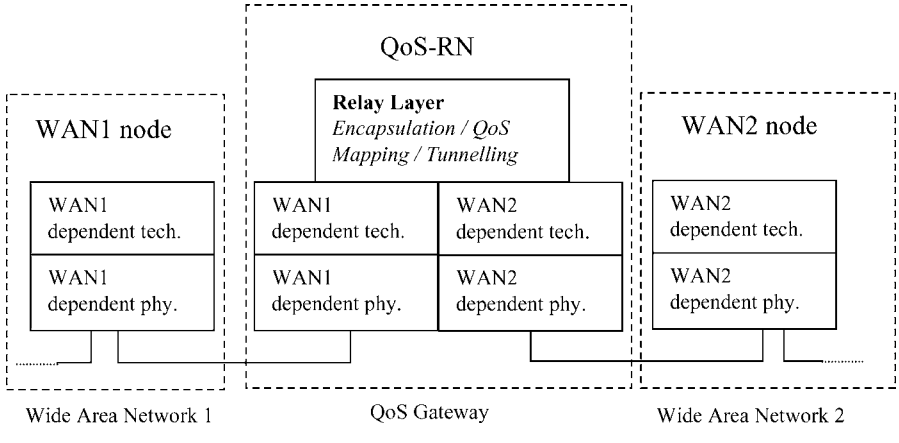


Figure 5.14 QoS-RN architecture for generic WANs

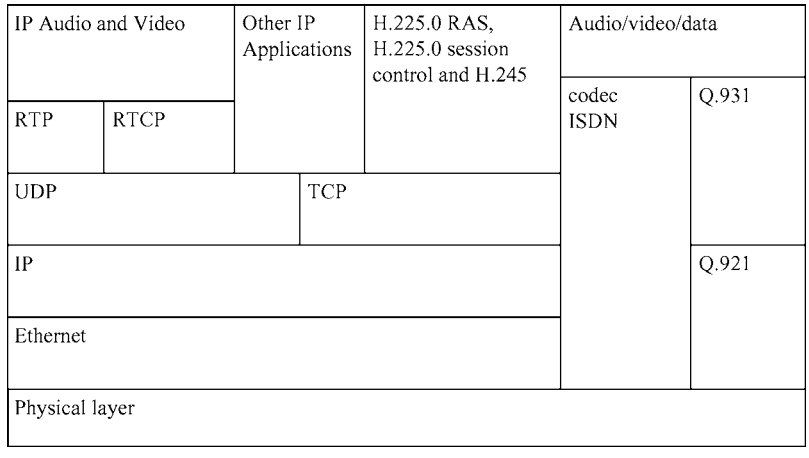


Figure 5.15 Example host stack

used technologies for military networks in Europe. It is also true that, in the civil world, the dominant technology is IP, as well as telephone networks, still much used for personal communication. Therefore, in this book, IP is considered the reference host but also ISDN will be referenced where needed. The choice, here, is due to the need of having real hosts to show how the encapsulation can work at the Relay Layer. It should not be seen as a technological advice because, as said above, the choice is often guided by not technological reasons. The host stack chosen is shown in Figure 5.15.

The main problem of QoS mapping concerns the transfer of QoS requirements among different network portions. It will imply the involvement of an end-to-end QoS architecture, proper signalling protocols and efficient vertical QoS mapping algorithms. The choice of a specific technology at the Relay Layer may have a strong impact on the overall performance of the network. The tests reported in Chapter 4 concerning the aggregation of the flows are also applicable here: what happens if the technology applied at the Relay Layer cannot identify the flows as requested by the users. If, for example, a single user flow needs to receive a specific service, what happens if the Relay Layer cannot identify the single source but only a group of flows? Many QoS architectures have been investigated in the literature; each single service provider company offers a proprietary and specific solution for that. This book tries summarizing some of the solutions proposed and suggests possible alternatives for future research.

6

QoS Architectures

6.1 End-to-End Quality of Service: State-of-the-Art

Support of end-to-end QoS over heterogeneous networks is a topic treated in the literature, in particular if applied to the IP world. Reference [Giordano03] states that "… the network operators are willing to open up their network resources to innovative new service providers, which include mechanisms for supporting end-to-end QoS guarantees (across multiple domains), and for the flexible and dynamic creation of new services". Within the same context, the European Union (EU) has founded projects in this area. In particular, three of them had the aim of generating proposals to provide IP premium services (IP QoS within the DiffServ environment): AQUILA [Engel03], TEQUILA [Mykoniati03] and CADENUS [Cortese03]. In particular, resource control for QoS over IP is managed by AQUILA, which assumes the presence of admission control agents managing the QoS requests and operating within the Edge Routers of a DiffServ Domain. Admission Control Agents communicate with Resource Control Agents (acting intra domain) to get information about available resources. Four traffic classes are admitted, having in mind IPv4 and DiffServ: Premium constant bit rate, Premium variable bit rate, Premium multimedia and Premium mission critical. Similarly, [Fineberg02] uses QoS network server (QNS) to manage QoS information and to check resource state over an IP WAN, but introduces the use of MPLS and RSVP-TE to transport QoS SLS, again within IP DiffServ world.

Recent interesting results come from the Project MESCAL [MESCAL], Management of End-to-end quality of ServiCe Across the Internet at Large, which proposed solutions for the deployment and delivery of inter-domain QoS across the Internet. MESCAL was partially funded by the EU as part of the IST Programme and ran from November 2002 to August 2005. It has defined a Business Model identifying actors of QoS provision over an IP-based network.

QoS over Heterogeneous Networks Mario Marchese

The business model actors are Service Provider (SP), IP Network Provider (INP), Physical Network Provider and Users. INPs are the key point for MESCAL because they are responsible for QoS provision over their own domain and they need to communicate with other INPs to guarantee QoS defined through SLS over end-to-end paths. To do that, MESCAL establishes a common vocabulary to define QoS-based services through the Internet, across multiple provider domains, so extending the model proposed in TEQUILA and dedicated to the domain of only one provider: SLS between customer and INPs is called "cSLS" (customer-SLS); SLS between INPs is called "pSLS" (Peer SLS). The model is fully described in [Howarth06]: Different INPs communicate in cascade through a QoS peering model where quality is defined through pSLSs. It means that INPs establish agreements with neighbours (only one-step away) to allow the extension of QoS guarantees over different domains. The concept of QoS class (QC) is used to define the QoS transfer capability within each single domain in terms of objective parameters (loss, delay and jitter). In more detail, QoS provided locally within each single domain is defined as l-QC, while the QoS transfer capability extended over more domains is identified as e-QC. The concepts are summarized in Figure 6.1 (partially from [Howarth06] and [MESCAL]). There are two domains: cSLS acts between the user (customer) and SLS Management entity of Domain 1; pSLS between SLS Management entities of the two domains; l-QC1 and l-QC2 are the QoS classes acting on Domains 1 and 2, respectively; e-QC is the extended QoS class, which acts across the two domains, so enlarging the scope of l-QC1 and l-QC2. The extended QoS class is provided by combining two local QCs. The Access Routers (ARs) and Border Routers (BRs) may hide the features of QoS-PRNs, defined in Chapter 5. The explicit reference to routers is kept for now because MESCAL Project is mainly related to IP environment.

To provide inter-domain QoS-based services a detailed functional architecture is proposed in [Howarth06]. It includes the following functional groups structured into Management, Control and Data Plane. SLS Management and Traffic Engineering functionalities are shared between Management Plane and Control Plane.

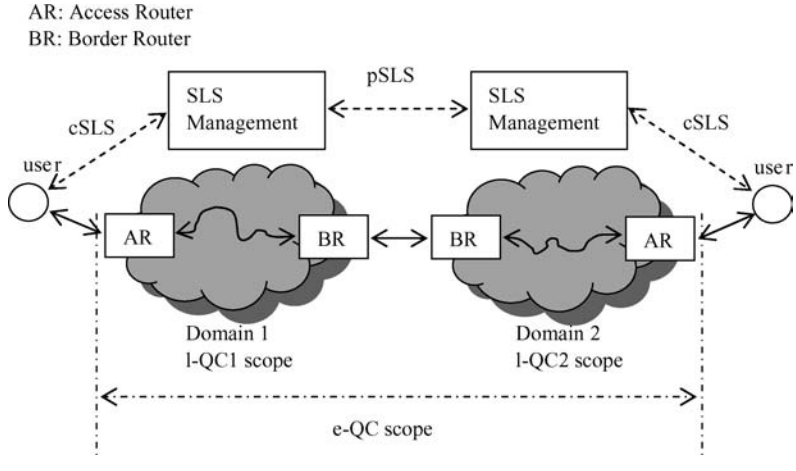


Figure 6.1 Cascaded end-to-end QoS management model

1. Management Plane
   (a) Service Planning and QoS Capabilities Exchange Group
      (i) QoS Capabilities Advertisement, to promote offered services
      (ii) QoS-based Service Planning, to define a service from the business activities viewpoint
      (iii) QoS Capabilities Discovery, to discover QoS capabilities to different destinations and their costs
   (b) Monitoring and Assurance Group
      (i) SLS Assurance, to monitor contracted SLS state
      (ii) Intra/Inter-domain Monitoring, for node and network monitoring and communication with other functional groups
   (c) SLS Management
      (i) pSLS Ordering, receiving and negotiating new pSLS requests
      (ii) pSLS Invocation, to manage pSLS invocating requests
      (iii) SLS Order Handling, to implement admission control at subscription level
   (d) Traffic Engineering
      (i) Traffic Forecast, to forecast traffic requests
      (ii) Offline Inter-domain Traffic Engineering, to built potential e-QC meeting service requirements
      (iii) Offline Intra-domain Traffic Engineering, to select some e-QCs to optimize intra-domain and inter-domain network resources
   (e) Network planning, to plan offline type and quantity of resources required by INPs
2. Control Plane
   (a) SLS Management
      (i) SLS Invocation Handling, to receive customer requests and implement CAC control
   (b) Traffic Engineering
      (i) Dynamic Intra-domain Traffic Engineering, to dynamic resource control (routing, bandwidth allocation, load control) within domains
      (ii) Dynamic Inter-domain Traffic Engineering, for dynamic routing between domains

3. Data Plane

   (a) Traffic Conditioning and QC Enforcement, for packet identification, classification, traffic shaping in accordance with SLS
   (b) PBH Enforcement, to proper traffic schedule and management
   (c) IP Forwarding, to send traffic forward along the end-to-end path

The functional structure of MESCAL is also very interesting because it represents a classification and implementation of the traffic controls reported in Chapter 4.

Two other recent Research Projects are important for the book's scope: AGAVE [AGAVE] and Intermon [Intermon]. Concerning the former, it is based [AGAVE] on Internet/telecommunications convergence and on the fact that IP is forecast to be the ubiquitous multi-service delivery platform. The supposed trend is separating service and network concerns, so giving distinct roles to the two key players: SPs and IP INPs. Within the context of QoS-guaranteed services over the IP networks, AGAVE aims at solving the problem of end-to-end service provisioning by studying, developing and validating a new inter-domain architecture based on Network Planes that will allow multiple INPs to provide Parallel Internets tailored to service requirements. AGAVE specifies an open connectivity service-provisioning interface to allow SPs to interact with underlying INP network resources for the provision of end-to-end IP-based added-value services. A lightweight QoS approach will be developed, based on the principles of differentiated routing with inherent load balancing and resilience, without requiring universal deployment of differentiated forwarding. The proposed solution will be deployable with small incremental additions to the existing best effort Internet.

As far as Intermon is concerned, the following are the main objectives [Intermon]:

- To monitor and analyse Inter-domain QoS to validate, plan, forecast and integrating different components for automated Internet inter-domain structure analysis, QoS and traffic monitoring, measurement-based modelling, simulation and visual data mining using data base with policy control.
- To analyze QoS and SLS of applications in inter-domain environment.
- To monitor BR traffic flow monitoring, modelling and simulation for inter-domain capacity planning and traffic engineering (inter-domain traffic matrix).
- To collect policy controlled data.
- To distribute data base with automated information processing.
- To model and forecast advanced toolkit for inter-domain QoS and traffic.
- To model and simulate scalable and efficient inter-domain measurement-based environments considering fluid and hybrid-packet technologies.

MESCAL architecture and QoS requirements are always referred to the IP DiffServ paradigm, as well as AGAVE and Intermon approaches, but, as should be clear below, the cascaded model may be considered also valid for generic QoS-provisioning approach and to match QoS over technologically heterogeneous networks.

The expressed ideas, widely shared in the scientific community, are also applied to military communications: [Jia01] focuses on end-to-end QoS provision over DiffServ networks by using Bandwidth Brokers (BBs) communicating each other for inter-domain information and managing intra-domain resources. A specific signalling is forecast for end-to-end

communication. Also in this case, BBs act in strict connection with ingress/egress routers of the different IP domains.

It is important to note again that the scope of the mentioned scientific and technical works is IP. Now, modern military networks implement heterogeneous technologies. It is a widespread perspective that " …capital expenditure constraints in both service providers and enterprises will mean that MPLS will evolve in the carrier core network first, with ATM remaining for some time to come as the primary technology for multiservice delivery in bandwidth-limited edge and access networks" [Bocci03]. "Today the ATM network are located in the heart of the network and IP in the periphery, but in the future only one network will be used. The best of IP and ATM will provide to develop Computer Telephony Integration applications, which take into account the convergence of data and telephony networks" [Lorenz00]. It means to provide a universal network integrating the best of packet and circuit-switched networks (which was exactly the aim and motivation of standardizing ATM) but considering the IP importance and diffusion. It implies that IP should be the recognized technology to identify a host (without forgetting plain and ISDN telephony) but it does not imply the use of IP currently implemented everywhere, also concerning QoS managing. It is just the idea on which this book is based. The main point is to build an overall end-to-end architecture offering full support to QoS, independently of the single portion in the heterogeneous network. It involves, as said, a common language for QoS definition, QoS interworking, vertical QoS mapping, signalling protocols use and control mechanisms implementation. Additionally, as often remembered in previous chapters, because of a general application, it would be recommendable to have QoS for each specific user (i.e. to possibly identify each single end-to-end flow having an SLS each or, at least, to identify a very large number of SLSs). Finally yet importantly, there can be the need of dropping some connections already in progress because other calls with higher priority require the access to the network and there is no sufficient bandwidth for all of them. It is often part of SLSs (see Chapter 1) and is much used in military communications identified as Multi Level Precedence and Pre-emption (MLPP).

In summary, the following are the important requirements, in the author's view:

- Designing a QoS-based interworking, providing the QoS specified by the SLS and allowing the definition of a large number of SLSs (one for each flow, if needed).
- Providing interworking of network portions implementing different technologies (still an open research problem, see, e.g. [Trimintzios03]) independently of the technology commonly used within the portions.
- Providing MLPP capabilities for each specific flow according to the required priority levels ([ID-Polk01]).

It is worth noting that IP-centric approaches not always solve the main objectives listed below, even if they represent an important reference for present networks. Therefore, by retaining many important details of the IP-centric mentioned solutions (as, e.g. the functionalities of BBs within each specific portion), it is important to propose new alternative solutions (as done in the following of this book). The aim is to solve the mentioned problems without penalizing national and industrial peculiar choices and privileging QoS provision, also opening the door to future needs and applications. Before detailing the different technology-based solution it is important to identify the control blocks that compose architectures oriented to QoS

provision. Even if the architectures reported in the next session have their origin within the IP environment, they can assume here a general meaning. In other words, they classify in operative blocks the concepts introduced in this section and describe overall end-to-end architectures to control QoS expanding the objects graphically reported in Figure 6.1.

6.2 Architectures for QoS Control

QoS architectures have been the object of many research projects and scientific works, as seen in section 6.1. Some of them are strictly related with the topic of this book and fit very well with the QoS-PRN (RN) framework proposed here. They are summarized in the following trying to make an effort to interpret them in the QoS approach carried on in the book.

Reference [Jia01] proposes a popular QoS architecture. It introduces a distinction between data forwarding and management plane. The former uses a differential treatment of individual packets at each network node, as allowed by packet service disciplines and queue management. The latter defines the resource allocation techniques and requests. In more detail, each single network portion (called "domain", in [Jia01]) implements a BB that negotiates the SLS with neighbour domains, implements Call Admission Control and QoS management. Control and data planes are completely decoupled: there is one centralized BB for each domain, which communicates with the routers separating the different network domains. The end-to-end QoS architecture proposed applies directly in the framework proposed in this book. Edge devices (the QoS-PRNs, identified also as ingress and egress routers as in [Jia01] in strict dependence on the IP origin of the end-to-end architecture) perform data encapsulation. Resource management functions (bandwidth allocations, SLS transfer and mapping), logically hidden in QoS-PRNs, are implemented by virtually separate tools called "Bandwidth Brokers" (BBs, in the following), which communicate each other. Figure 6.2 summarizes the network model fully introduced in [Jia01] and applies it to the general QoS environment. Only the link between QoS-PRNs and BBs is shown in Figure 6.2, but BB needs to communicate also with the QoS architectures, if any, which control the quality within domains, as shown in detail in section 6.7.

The intra-domain clouds may use a private technology. It will be the care of QoS-PRNs to implement vertical QoS mapping over each single intra-domain technology, as introduced in

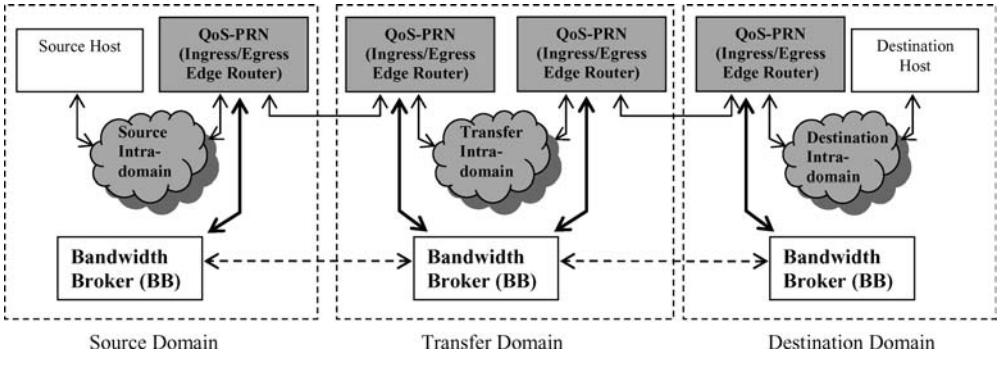


Figure 6.2 End-to-end QoS architecture [Jia01]

section 5.2 and detailed in Chapters 8 and 9. This function is well modelled by the QoS-PRN concept, which decouples the technological implementation problem.

Reference [Jia01] implies also the use of signalling to communicate SLS requests and related actions (e.g. acceptance or not). Signalling flows follow the direction indicated by the dashed lines connecting BBs for inter-domain signalling and by the bold lines between BBs and Edge Routers, for intra-domain signalling. The action steps, taken from [Jia01], are summarized as follows: the sender emits a resource allocation request to the BB of its own domain. The request contains also information about traffic (as profile and destination). The BB performs resource control based on the resource state of its domain: if a call can be accepted the request is forwarded to the neighbour BB, otherwise the call is rejected, notifying the source. The operation is iterated BB after BB up until the destination. If the end-to-end call is accepted, the source is notified. During the backward propagation, BBs perform resource reservation within their domain through intra-domain signalling, which can be completely different from inter-domain signalling. The edge routers (QoS-PRNs, in this book) are configured in this phase. No information is given about the encapsulation format of the signalling protocol. Only the functionalities are envisaged. Figure 6.3 (also based on the contents of [Jia01]) sketches the interaction among different agents within the management plane when the Source domain is connected with the Transfer domain. The interaction goes on up to the destination. The SLS agent is responsible for the contract between neighbour domains. Based on the contract, the SLS agent controls the configuration agent to set the forwarding path in the edge router and the related controls (such as classifier,

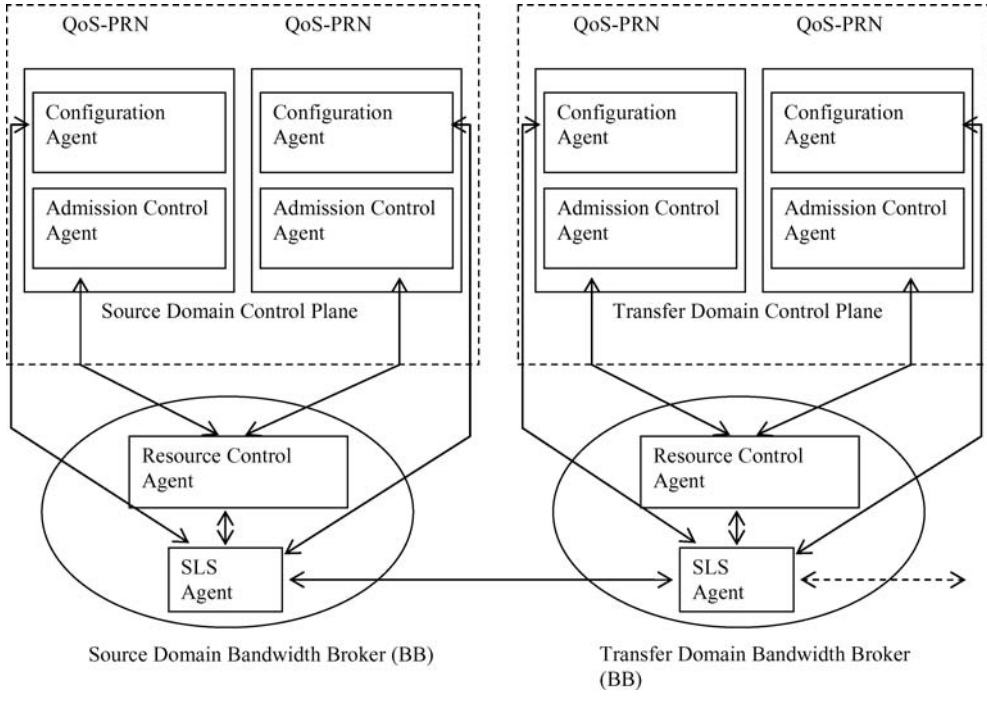


Figure 6.3 Agent interaction in the management plane [Jia01]

maker and scheduler). If the contract between neighbour domains includes both technical and commercial information, it is possible to speak of SLA agent.

The relation with the QoS-PRN approach is immediate: the SLS agent and Resource Control Agents may be part of the Relay Layer, while the check of the state of the single portions may be left to the portions themselves, as done in Figure 6.3.

Another very interesting end-to-end QoS architecture to which the concept of QoS-PRN may be directly applied is contained in [Fineberg02]. The implemented concepts are similar to the approach just introduced but the intra-domain quality is guaranteed through MPLS. It is close to the concept introduced in this book, where, even if the QoS is forwarded and implemented through a specific solution (IP, MPLS, IPv6), each single sub-portion may choose the preferred technology. Only entities represented as boxes are specified here from [Fineberg02] because the focus is on architectures. Also, this architecture has been thought for IP but it is generalized in Figure 6.4 because it has a wider application. As in the [Jia01] case, this architecture also uses separate QoS management entities, taking the role of QoS-PRNs.

Reference [Fineberg02] also makes some suggestions about the important topic of signalling, by envisaging two possible alternatives to transmit QoS requirements within the MPLS domain: RSVP tunnelling extensions (RSVP-TE) and Constrained Routing Label Distribution Protocol (CR-LDP). Some details about signalling will be reported in the next chapter, even if the solution is presented for the MPLS domain, it can be extended and applied in a more general QoS environment.

QoS architecture has been investigated and designed also within the Project AQUILA [Engel03]. It is part of three projects funded by the EU in the area of QoS support over IP networks: AQUILA, TEQUILA [Mykoniati03] and CADENUS [Cortese03], already mentioned at the beginning of this chapter. The latter mainly concerns the service including a definition of a business model. TEQUILA concentrates on service management and traffic engineering (as dimensioning and resource management). AQUILA defines a DiffServ architecture for the delivery of QoS services but in this case also a generalization may be applied.

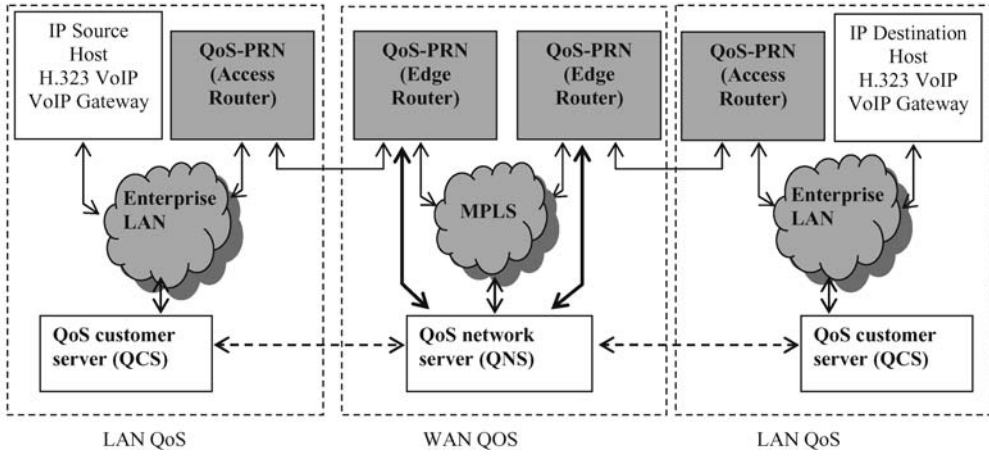


Figure 6.4 End-to-end QoS architecture [Fineberg02]

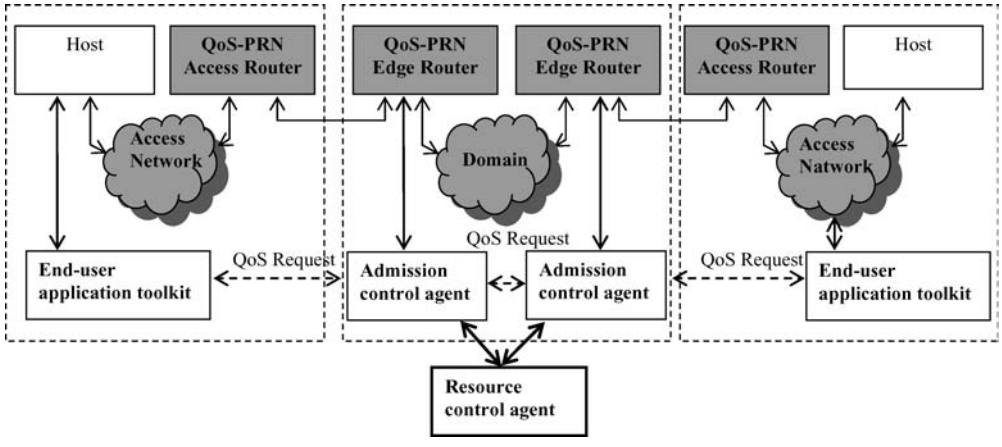


Figure 6.5 AQUILA end-to-end QoS architecture

QoS-PRNs separate the private domains from the rest. Admission control agents coordinated by a Resource control agent, as in [Jia01], control them. AQUILA architecture, extended to QoS-PRNs, is shown in Figure 6.5.

The "Domain" hides a group of core routers. The QoS requests flow through "Admission control agents" that tune the "QOS-PRNs". As in the previous cases, control blocks may physically located within QoS-PRNs.

Actually, the QoS architecture composed of QoS-PRNs (RNs) presented here is totally conformant to the architecture existing in the literature but generalizes the possible application because it does not make any hypothesis concerning the technology implemented in the core (e.g. the different portions). It conveys the encapsulation functions and the flow treatment carried out within the "QoS-PRNs" within the Relay Layer and, possibly, also the control functionalities performed by the control agents as well as BBs, call admissions and configuration agents. Obviously, the Relay Layer may also work together with control agents related to private network sub-portions, as done within other architectures in the literature and explained in Chapter 8 referring to vertical QoS mapping. Figure 6.6 shows a possible abstraction of the QoS end-to-end architecture at control and management plane. It is proposed in this book as framework, which theoretically summarizes the architectures already in the literature. The protocol architecture at the control and management plane, in practice, is the same as in Figure 6.1. Figure 6.6 highlights the presence of a signalling protocol for service negotiation. It has been left unidentified and called, generically, "QoS-Signalling". The signalling protocol path is sketched in Figures 6.7 and 6.8, over the QoS-PRN and QoS-RN architectures for the generic WAN interconnection shown in Figures 5.13 and 5.14.

As stated in Chapter 3, QoS provision may be reached only in the presence of proper QoS architectures, including control mechanisms and signalling protocols to carry QoS request. Actually, signalling protocols are also explicitly used for different control mechanisms (e.g. QoS Routing scheme) to transmit network conditions. Nevertheless, in this book, related to QoS management over heterogeneous network, the stress is on the communication between

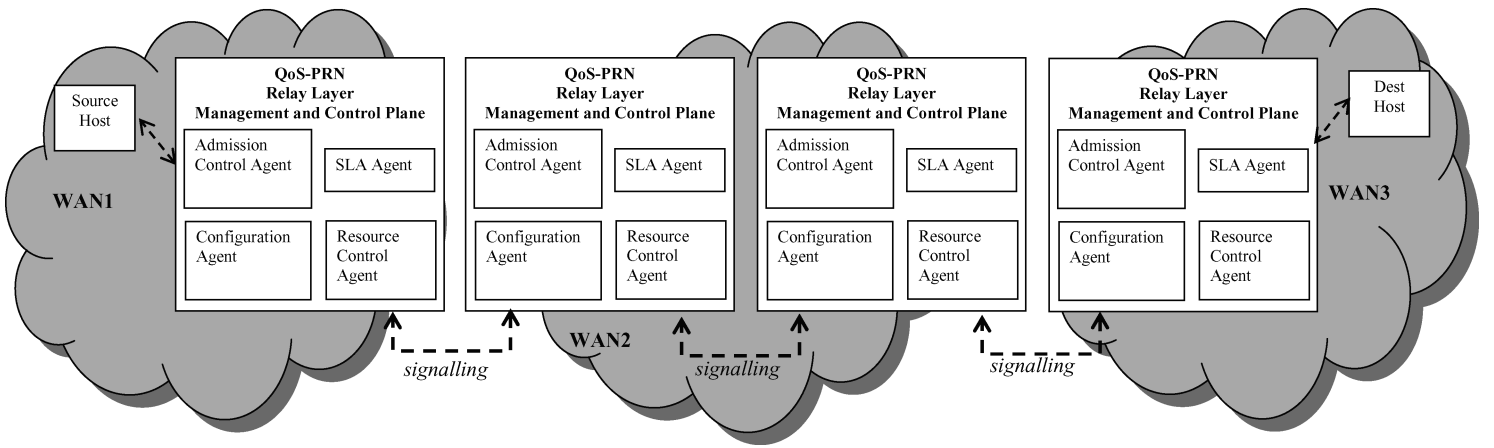


Figure 6.6 QoS-PRN end-to-end QoS architecture: Management and control plane

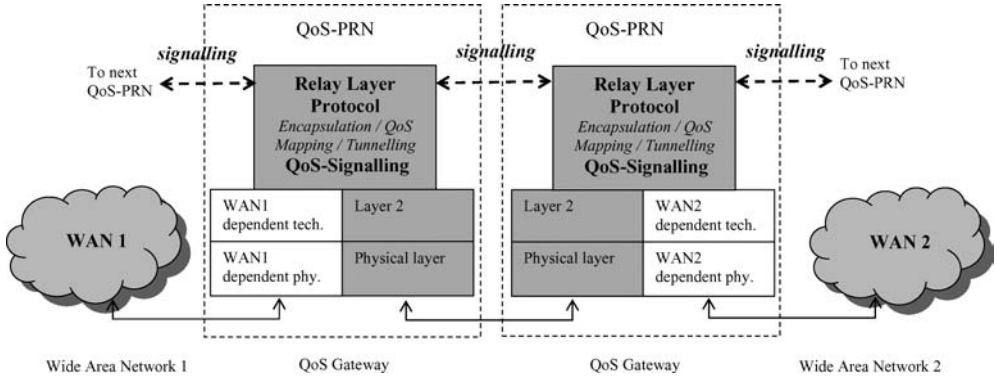


Figure 6.7 QoS-PRN signalling architecture

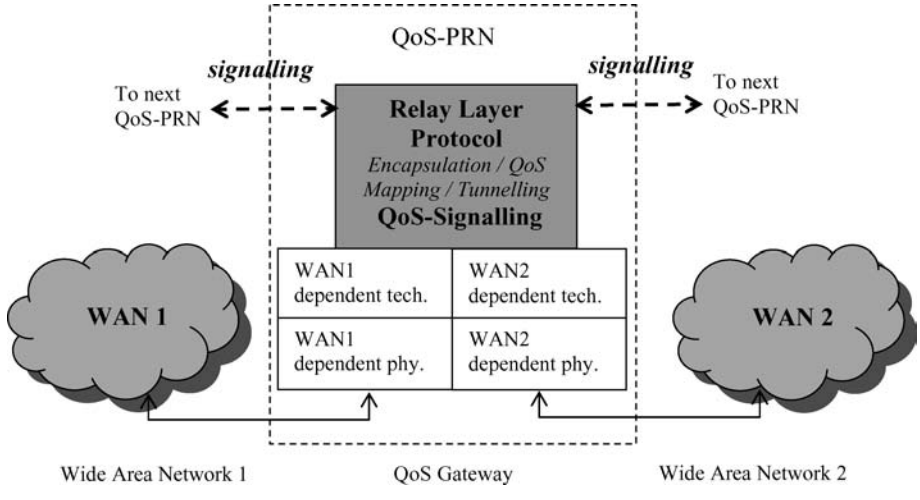


Figure 6.8 QoS-RN signalling architecture

networks. If not indicated differently, "signalling protocols" will carry QoS requests and related management actions (e.g. CAC decisions) among QoS-PRNs (RNs)). Signalling will be discussed in the following and explicitly treated in Chapter 7.

Before giving some indications about signalling, just to follow up what was said up until now, it is important to focus on different technology-centric approaches. The generic encapsulation environment is presented in the next section having Figures 6.7 and 6.8 as references. After that the first presented approach is IP-centric, the second one is MPLS-centric and the third one is based on IPv6. MPLS allows using a large number of traffic classes, without penalizing the scalability, while IPv6 reaches the same aim if all the features of IPv6 are used, overcoming the DiffServ paradigm.

6.3 "Technology"-centric QoS Architecture

This section details the encapsulation characteristics of the protocol implemented at Relay Layer within the QoS-PRNs. The encapsulation format is the same for QoS-RNs. The description should allow a deeper comprehension of the specific "technology"-centric approaches presented in the reminder of the chapter, where, in short, the word "technology" is substituted by a specific existing solution: IP, MPLS and IPv6. As said, the reference Figures are 6.7 and 6.8. The indication "technology"-centric architecture implies that, within QoS-PRNs and QoS-RNs, only the packets of that "technology" have meaning over the Relay Points. The data flow is shown in Figure 6.9 for QoS-PRNs. The considered end terminal is an IP host, as defined in section 5.3. The ISDN terminal is not considered for now. The IP host generates packets forwarded to the first QoS-PRN of the overall end-to-end path. The data packet is encapsulated at the Relay Point interface by using Layer 2 encapsulation and a proper physical solution. No specific protocol is indicated in this generic environment. On the other side of the QoS-PRN, the IP packet is encapsulated in the private WAN dependent technology, identified in Figure 6.9 as "WAN dep. tech.". The "technology" implemented at the Relay Layer is the reference and when it is embedded in another private transport technology, it is topical to map the reference QoS requirements over the WAN transport technology. This concept will be repeated for each specific "technology"-solution because of its extreme importance. In practice, it is the concept of typical vertical QoS mapping problem, detailed in Chapter 8.

Figure 6.10 reports the data flow through the equivalent Relay Node (RN). The reference with the RN will be reported when strictly needed. Otherwise the PRN will be taken as reference.

6.4 IP-centric QoS Architecture

6.4.1 Architectures and Data Encapsulation

The first suggested solution is IP-centric. It means that the common language is IP concerning both QoS definition and interworking architecture. IP-centric QoS solution may be implemented through both IntServ and DiffServ. It has no impact on the architecture because the IP layer, if properly configured, can support both solutions, but it has a great impact on the definition of the SLS because the IP DiffServ model assures a service only to aggregates of flows as clearly stated in Chapter 3, while IntServ can distinguish each single flow. There is also no difference from the encapsulation viewpoint because IP packets have exactly the same format but there is a big difference concerning signalling (detailed in the next chapter) and related QoS architecture specified in the following. Five IP-centric alternatives will be presented. As said in Chapter 3, due to scalability problems, even if IP IntServ QoS solution can guarantee a peculiar treatment for each single customer in the network, it is not much considered in the recent literature. Actually, most of the QoS architectures reported in this chapter and taken from the literature have origin within the DiffServ environment; nevertheless, the IntServ solution should not be forgotten and will receive attention in the reminder of the book, in particular, concerning signalling schemes applied to QoS architectures.

A picture of IP-centric architecture is shown in Figure 6.11 by following the scheme introduced in the previous chapters for PRNs. Figure 6.12 contains the same solution for QoS-RNs.

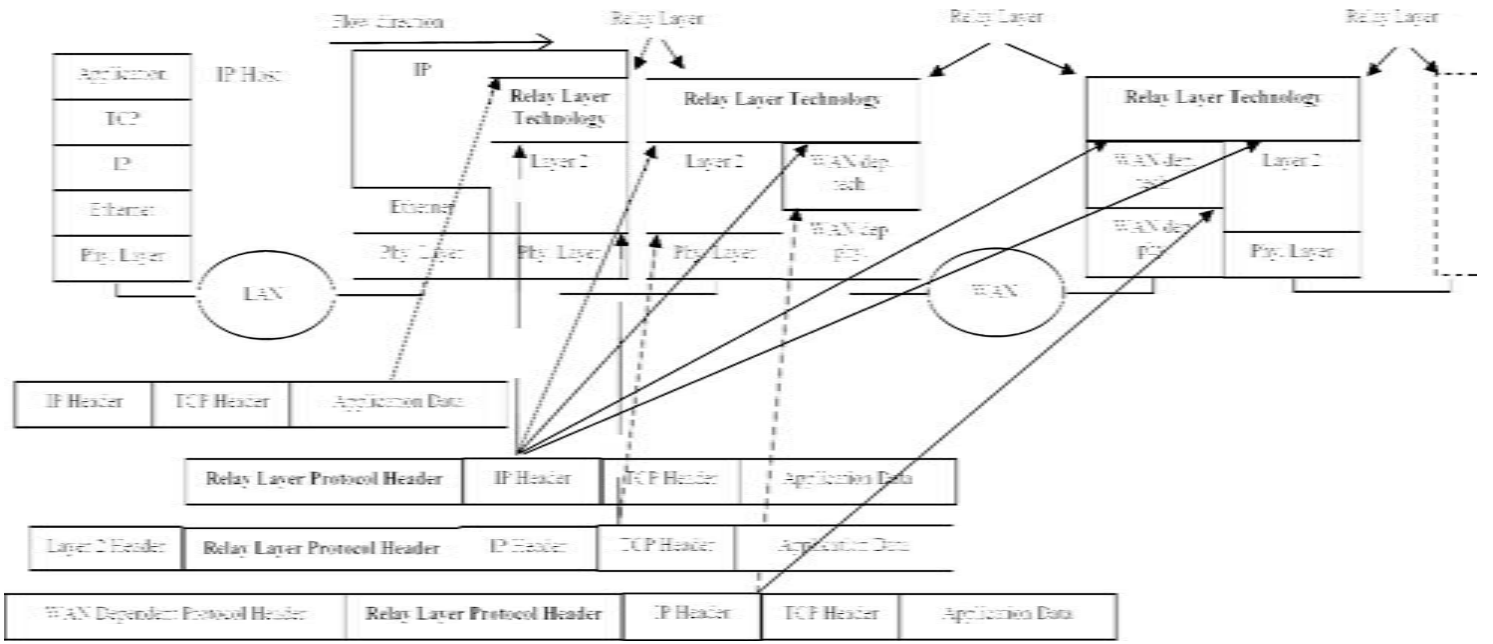


Figure 6.9 "Technology"-centric QoS-PRN connecting an IP host to a WAN-dependent technology portion: Data flow

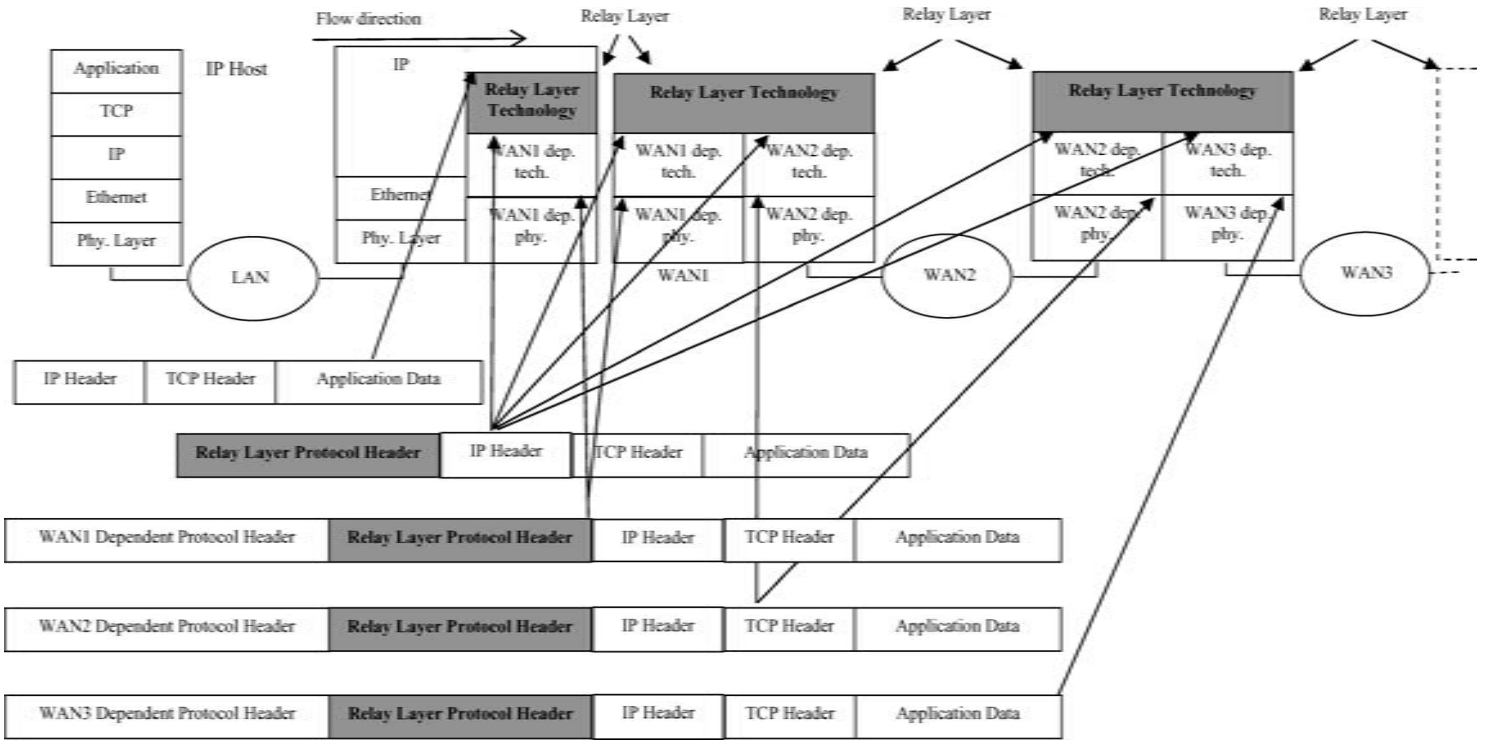


Figure 6.10 "Technology"-centric QoS-RN connecting an IP host to a WAN-dependent technology portion: Data flow

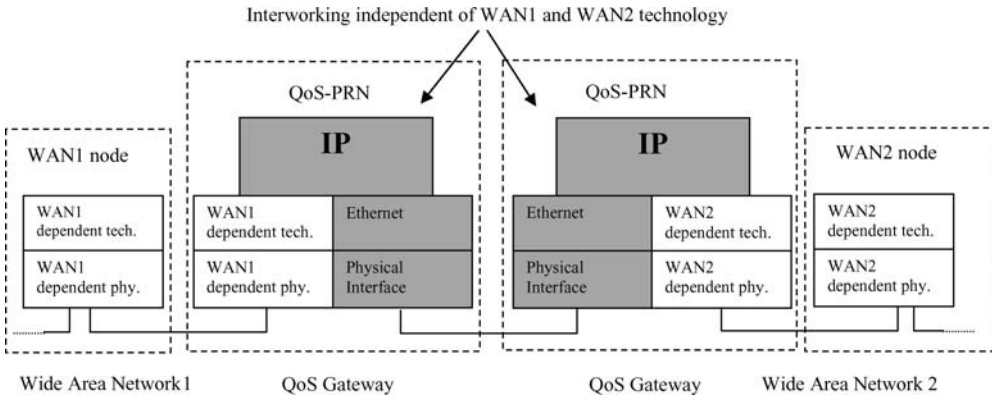


Figure 6.11 IP-centric QoS-PRN

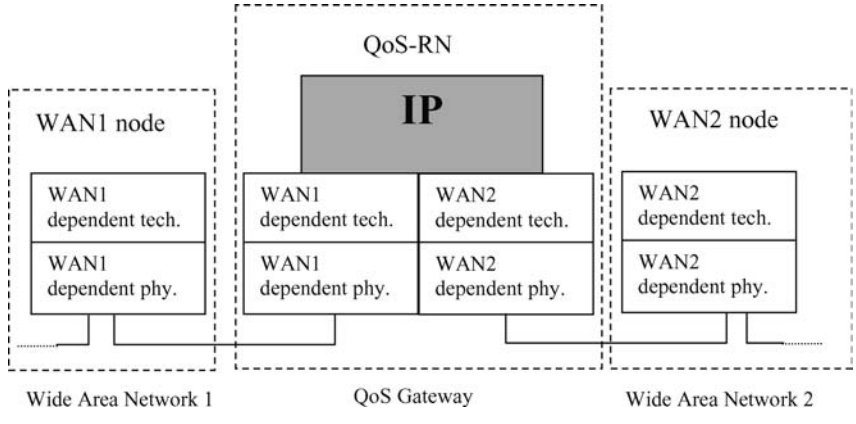


Figure 6.12 IP-centric QoS-RN

IP-centric QoS-PRNs and QoS-RNs act as IP routers concerning all aspects (routing, encapsulation and processing) and can simply be addressed by using an IP address. This is a great advantage also concerning signalling because the entire interface is simplified. The reference language is IP also concerning QoS: if the DiffServ paradigm has been chosen, for example, a proper common definition of the DiffServ Code Point (DSCP) is necessary. It must be the same for the overall network. If the WAN technology connected to the QoS-PRN (-RN) is IP-based, vertical QoS-mapping of IP requirements over IP is very simple because there are two IP networks in cascade. If the WAN uses the DiffServ paradigm too, QoS mapping is immediate, unless the specific WAN does not use a proprietary DSCP assignment. In this case, the common DSCP SLS definition (as well as all the related parameters) needs to be mapped on the proprietary DSCP used in the single WAN. However, if the WAN dependent technology is, for example, ATM, a precise vertical QoS mapping is necessary. The same case applies if the WAN technology is ISDN. The general problems of vertical mapping have been introduced in Chapter 5 and will be detailed in Chapter 8. Indications how vertical QoS mapping may be implemented will be provided in Chapter 9. This chapter is focused on QoS architectures.

IP-centric architectures imply that, at Layer 3, only IP packets have meaning in the QoS-PRNs. Some practical examples concerning different traffic flows at the QoS gateway may help understand. Figure 6.13a shows the encapsulation of data information in an IP-centric QoS-PRN connecting an IP host to an ATM network portion by directly applying the generic format encapsulation of Figure 6.9. The IP host generates packets forwarded to the first QoS-PRN of the overall end-to-end path through a LAN. From IP packets viewpoint, QoS-PRN is a simple router. Actually, it will also have signalling functionalities for QoS management but, concerning the pure data flow, it is a router. It is important to note that, being the overall end-to-end network IP-based, the first encapsulation at the interface between the LAN and the QoS-PRN may be simplified. In other words, referring to Figure 6.9, a double IP header would be formally necessary because there is a Relay Layer Protocol Header just before the IP Header. Anyway, being the "Relay Layer Protocol Header" correspondent to an "IP Header" and having the QoS-PRN router functionalities, the "IP header" identifier appearing in Figure 6.9 may be dropped so simplifying the encapsulation as shown in Figure 6.13b. The same QoS reference is supposed in this case between the LAN and the QoS-PRN as, for example, the same DSCP assignation. The assumption is very reasonable for a LAN where, typically, there are no problems to guarantee QoS. The data packet is then encapsulated over the Relay Layer interface by using Ethernet at Layer 2 (it is a reasonable choice even if not the only one) and proper physical coding. On the other side of the QoS-PRN, the IP packet is encapsulated in the private WAN technology. It is ATM in this case. It is also interesting to see what happens in case of interface with another WAN IP backbone where the QoS approach used is different (e.g. a private DSCP assignation). The case is shown in Figure 6.14, the IP header transporting the relay information is encapsulated in the WAN IP packet and transported up to the next QoS-PRN, where the Relay Layer IP header will be decoded. If the Relay Layer IP packet is embedded in another transport technology, it is strongly necessary to map the reference QoS requirements over the WAN transport technology. The next chapter is fully dedicated to vertical QoS mapping problems.

The data flow traversing the network is similar if the hosts are not IP. For example, if the host is an ISDN phone, the data flow over an IP backbone is as shown in Figure 6.15. The information encapsulated and transported from the source to the destination is guided by ISDN. Due to the presence of IP hosts and ISDN terminals, the interoperability of the terminal tools implementing the two technologies is of special relevance. The transport technology has been defined above but, if the two terminals implement different protocol stacks, a conversion is needed to operate together. The solution reported here in Figure 6.16 is IP-driven, that is the conversion should be done when the ISDN flow enters the first QoS-PRN but alternatives, technically equivalent, are also possible as, for example, source-type-driven. In this case, the last QoS-PRN over the end-to-end path should include specific interfaces described below:

- VoIP–ISDN interface: If a VoIP phone, managed by an IP network, needs to communicate with an ISDN phone, managed by an ISDN network, IP signalling and traffic will be terminated and translated over ISDN signalling and traffic.
- ISDN–VoIP interface: If an ISDN phone, managed by an ISDN network (connected or not with an ATM network), needs to communicate with a VoIP phone, managed by an IP network, ISDN signalling and traffic will be terminated and translated over VoIP signalling and traffic.

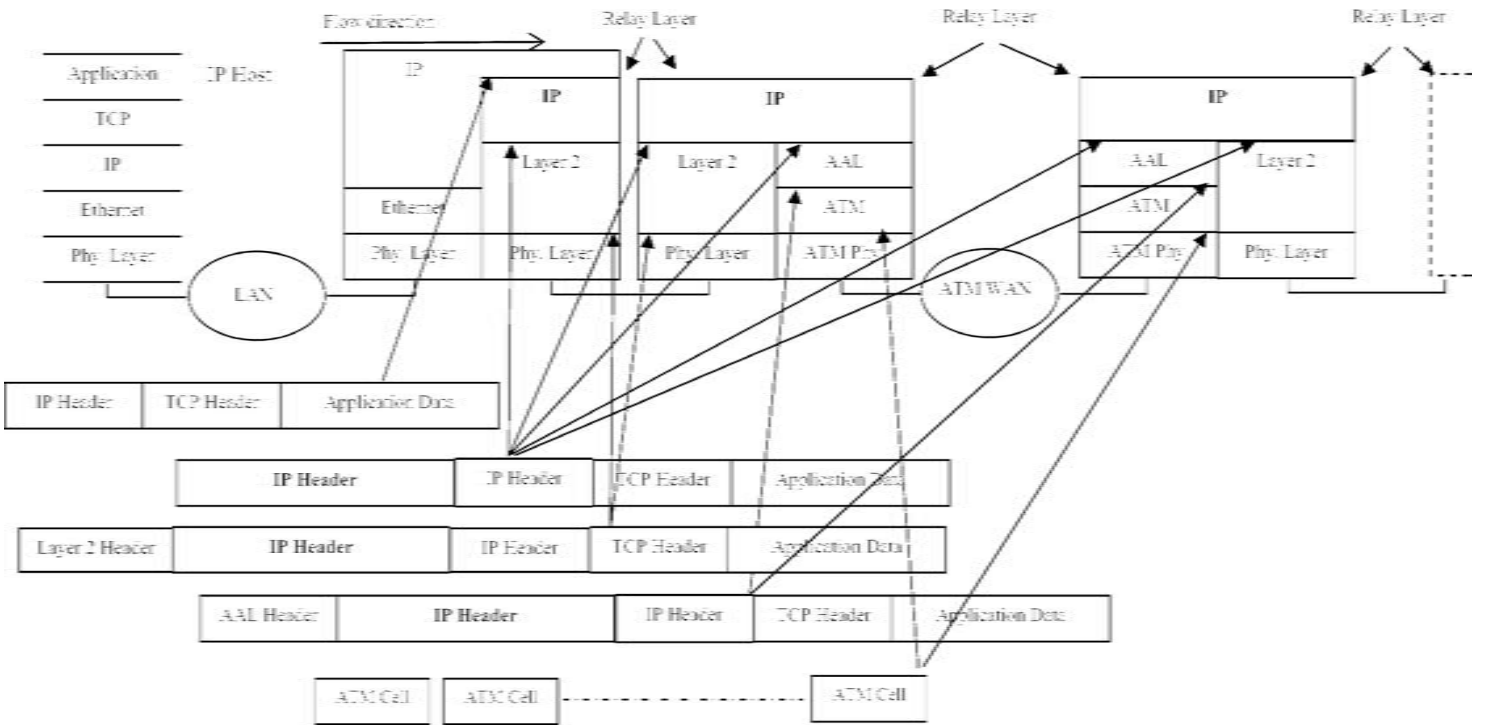


Figure 6.13a IP-centric QoS-PRN connecting an IP host to an ATM network portion: Data flow with IP source encapsulation

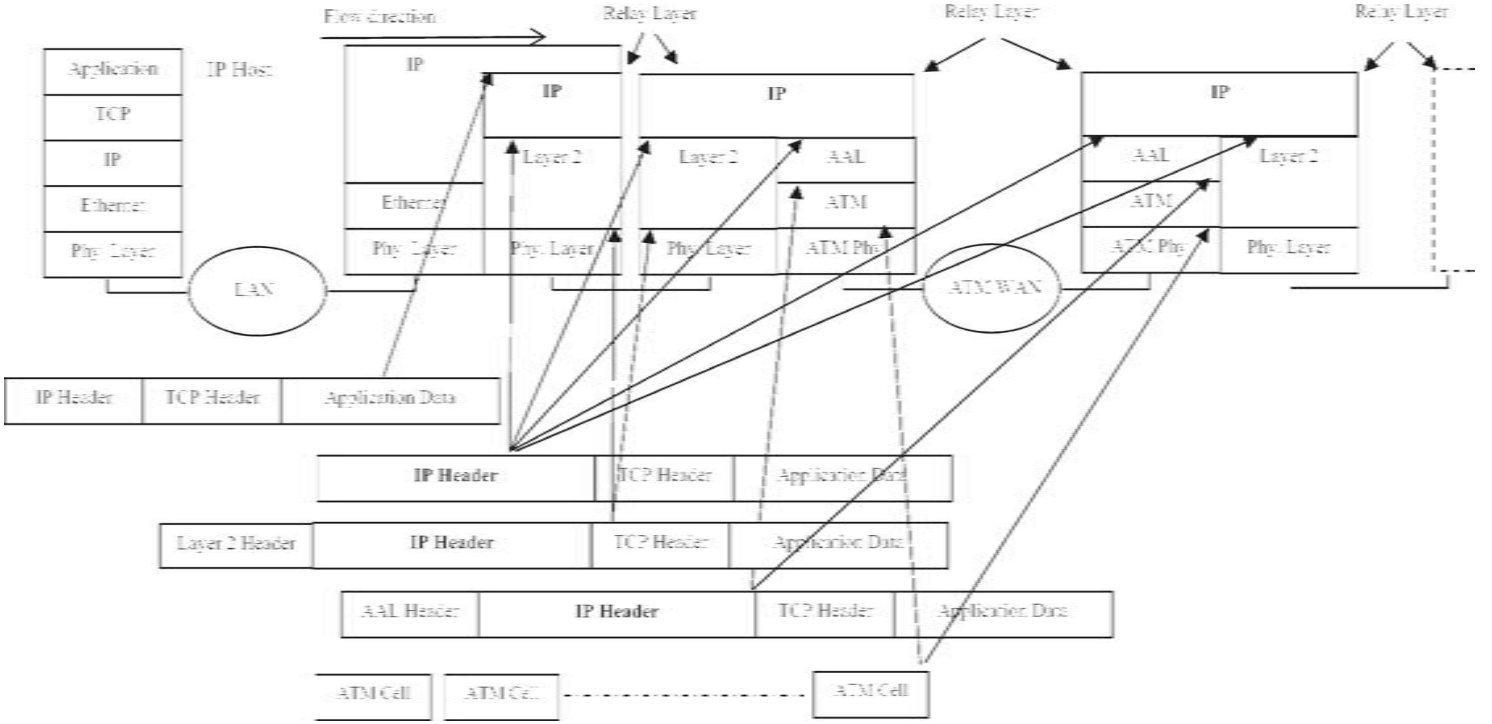


Figure 6.13b IP-centric QoS-PRN connecting an IP host to an ATM network portion: Data flow

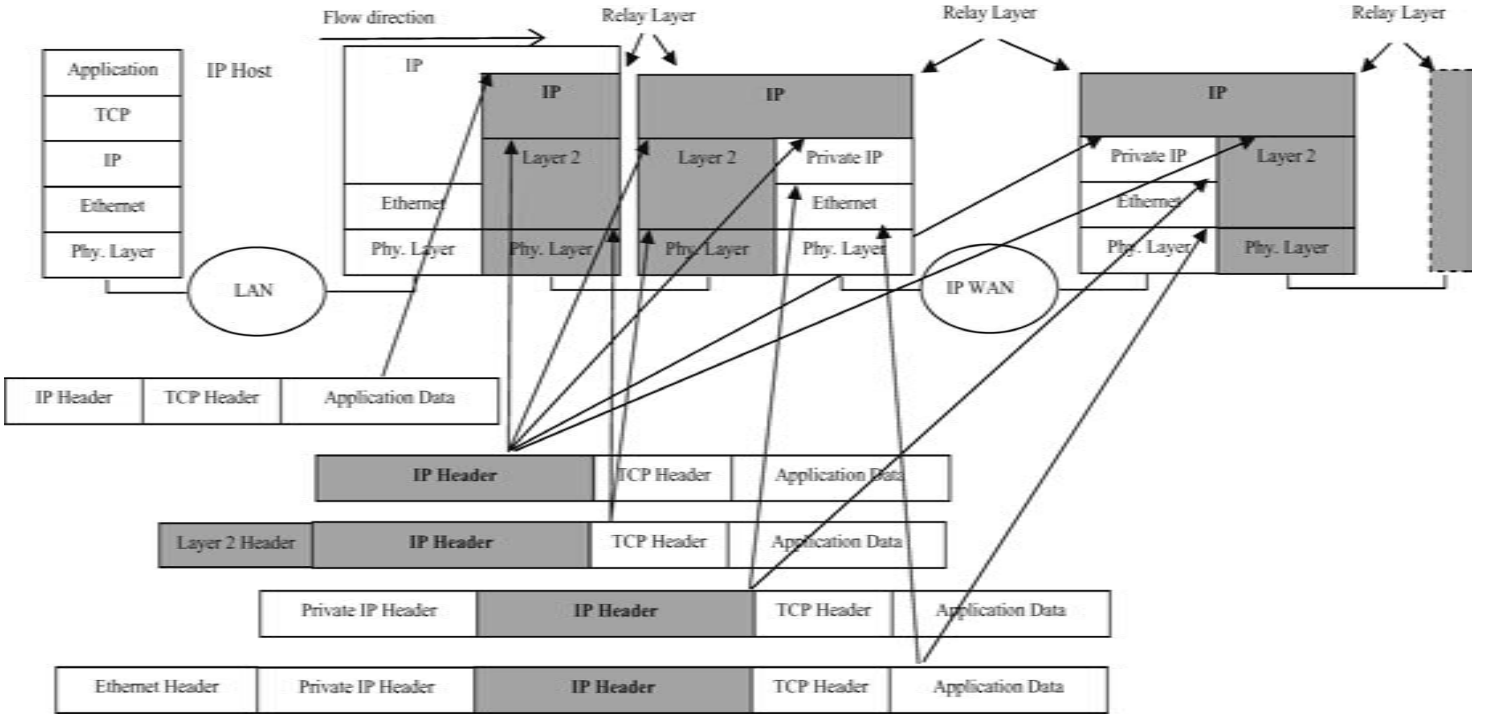


Figure 6.14 IP-centric QoS-PRN connecting an IP host to a private-IP network portion: Data flow

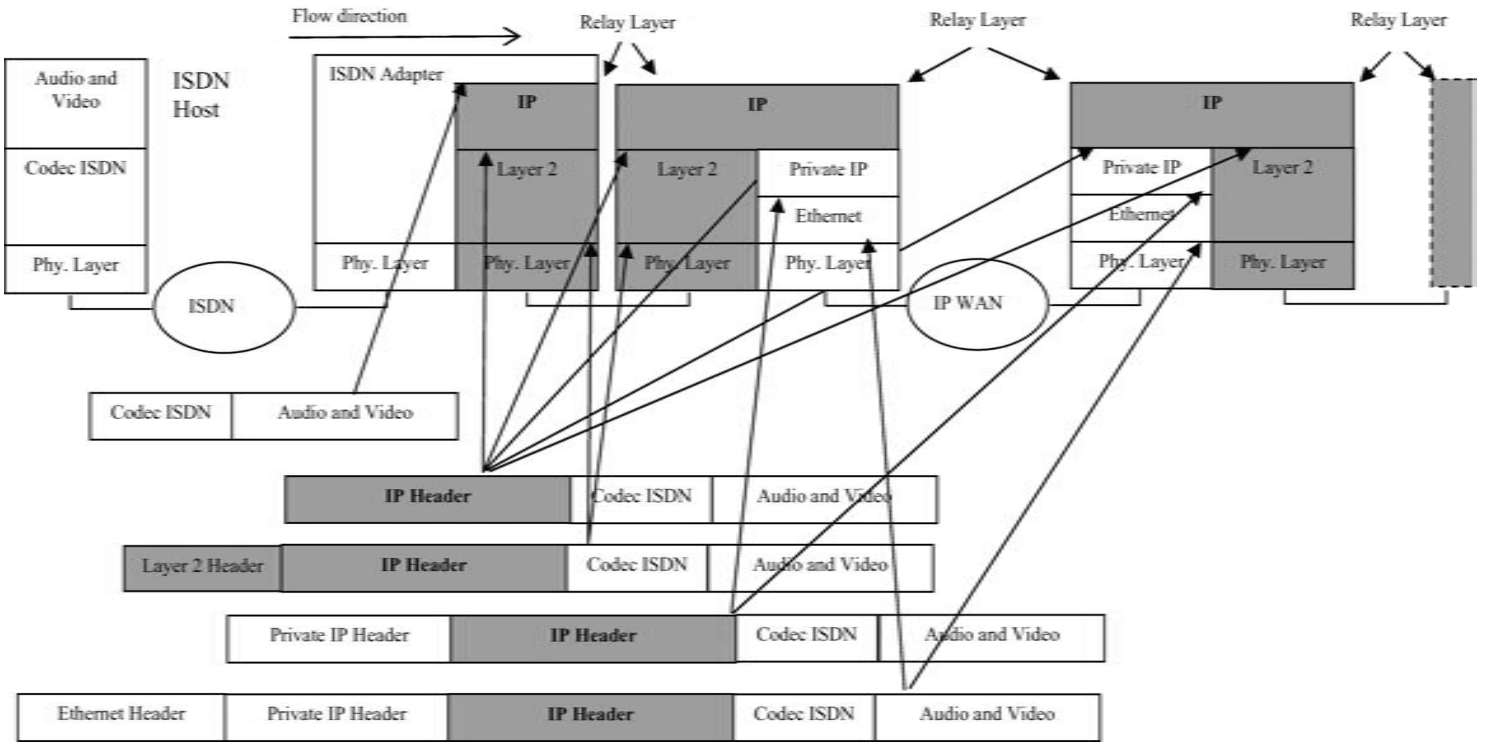


Figure 6.15 IP-centric QoS-PRN connecting an ISDN host to a private-IP network portion: Data flow

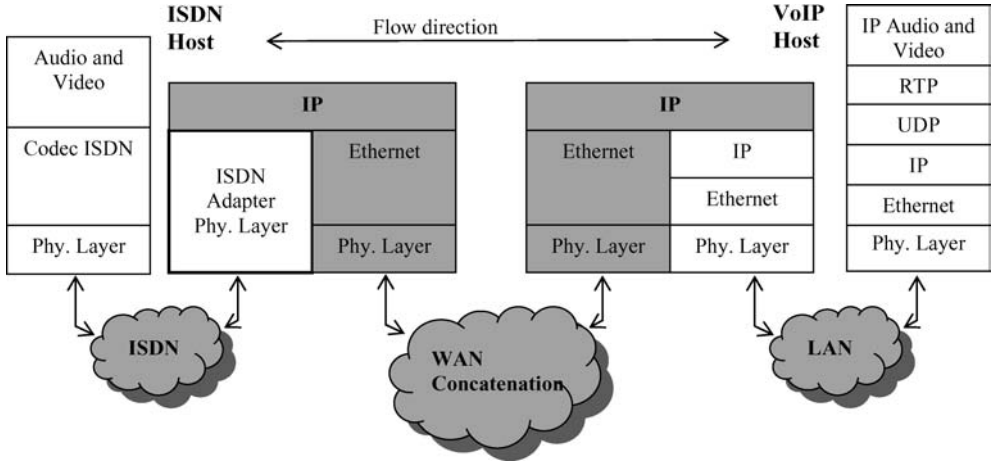


Figure 6.16 IP-centric QoS-PRN, VoIP and ISDN interoperability

The topic is very important and deserves a great deal of attention. Anyway, being strictly related to technology is beyond the scope of this book.

Once chosen the IP-centric approach, it is essential to match another requirement to offer end-to-end QoS services: the definition of a common SLS. The choice heavily depends on the IP version chosen and on the QoS paradigm adopted. Five solutions that use IP paradigm are briefly summarized here: four adopts DiffServ, one exploits IntServ. As already said, in case IPv6 is used, the overall architecture would be the same (as well as the data flow encapsulation) but the QoS management is very different. For example, IPv6 can identify each single flow, as clearly stated in Chapter 3, by using F6SN paradigm or a large number of SLSs by using CSF6N. SLS definition and flow treatment within QoS-PRN (and –RN) is different. Due to the importance of this choice over the QoS provision, a special section of this chapter is dedicated to IPv6-centric approach.

6.4.2 IntServ-IP-centric QoS Architecture

Concerning encapsulation and data flow, it perfectly matches the generic IP architectures presented above. In particular, IntServ-IP, QoS-PRN and QoS-RN architectures are the same as in Figures 6.11 and 6.12, respectively, where the generic reference to IP is substituted by IntServ-IP. It does not affect data flow encapsulation. The schemes reported in Figures 6.13–6.16 are still valid. IntServ has a great impact on QoS management and signalling schemes. Concerning management and control plane, IntServ-IP is the direct application of the QoS-PRN end-to-end QoS architecture shown in Figure 6.6. The signalling protocol is already established for this architecture. It is called “RSVP” and will be presented in the next chapter where it will receive a great deal of attention because it is of basic importance also for other QoS models. The IntServ-IP-centric QoS approach offers powerful tools to manage QoS and allows extending at end-to-end level all the features already seen at single network portion level in Chapter 3. In short, a flow from the first QoS-PRN to the last one is defined as a packet stream that requires a specified QoS level and it is identified by the vector “IP source address,

IP destination address, Protocol, TCP/UDP source port, TCP/UDP destination port". QoS is reached by implementing the following control blocks within the control entity presented in section 6.7 and in the next chapter because it is related to signalling: resource reservation, admission control, packet scheduling, traffic shaping and buffer management. Information concerning the different incoming flows is maintained in the routers and periodically updated through proper entities available within a specific resource reservation signalling protocol (RSVP). There are two service classes: guaranteed service (GS) and controlled-load service (CLS), as already said in Chapter 3 at single network portion level. Also at QoS-PRN level, the drawback of IntServ approach is scalability because IntServ identifies each single flow and has no tools to aggregate traffic with similar features and requirements.

6.4.3 DiffServ-IP-centric QoS Architecture

6.4.3.1 Encapsulation and Data Flow

DiffServ is a widely diffused solution. In this case, the use of either IPv4 or IPv6 has no impact because there is no difference in traffic management after fixing DiffServ QoS paradigm. As said in Chapter 3, there are 14 traffic classes characterized by a common assignation of the DSCP field. This choice is topical because it affects totally the QoS provision of the overall end-to-end communication. Each single data packet arriving at the IP layer of each QoS-PRN (and -RN) needs to be treated in conformance with the DSCP assignation performed for all traffic. Being defined for IP-centric approach, the common language, both for addressing/signalling and for data encapsulation, is IP.

Possible choices concerning DSCP assignation within the DiffServ paradigm have been presented in Chapter 3, from Table 3.6 to Table 3.9, whose definition may now be applied for the overall end-to-end network.

All the architectures presented in section 6.2 may be thought within the DiffServ paradigm. Figure 6.5 shows the QoS architecture investigated and designed within the Project AQUILA [Engel03]. Actually AQUILA defines a DiffServ architecture for the delivery of QoS services. In line with it, AQUILA defines 4 traffic classes: premium constant bit rate (for real time applications with low bit rate variability); premium variable bit rate (for streaming real time applications with high bit rate variability), premium multimedia (for elastic long-lived applications with TCP or TCP-like bit rate adaptation) and premium mission critical (for transaction oriented data applications). The definition of the DSCP value is simpler than the model used in the tables in Chapter 3 because the application framework is less demanding, but the base concept is the same. The proper DSCP values are set within QoS-PRNs-Edge Routers in Figure 6.2. ERs separate the DiffServ domain from the rest. The same action may be taken, if necessary, in ARs. Admission control agents are coordinated by a Resource control agent, as in [Jia01]. DSCP values are set in the Routers also in the QoS architecture shown in Figure 6.4. Two LANs are connected through a WAN: the AR implements a class-based queuing; the common language of the architecture is DiffServ; Edge Routers implement Weighting Fair Queuing. The cloud containing MPLS hides a set of Core Routers where MPLS is implemented. It opens the door to a particular use of MPLS, which will be investigated in the MPLS-centric approach explained in the next section. QoS is managed through QCS and QNS that communicate each other. The architecture perfectly matches the flow encapsulation of Figure 6.17. Figure 6.2 defines a complete approach including the control plane, shown in Figure 6.3.

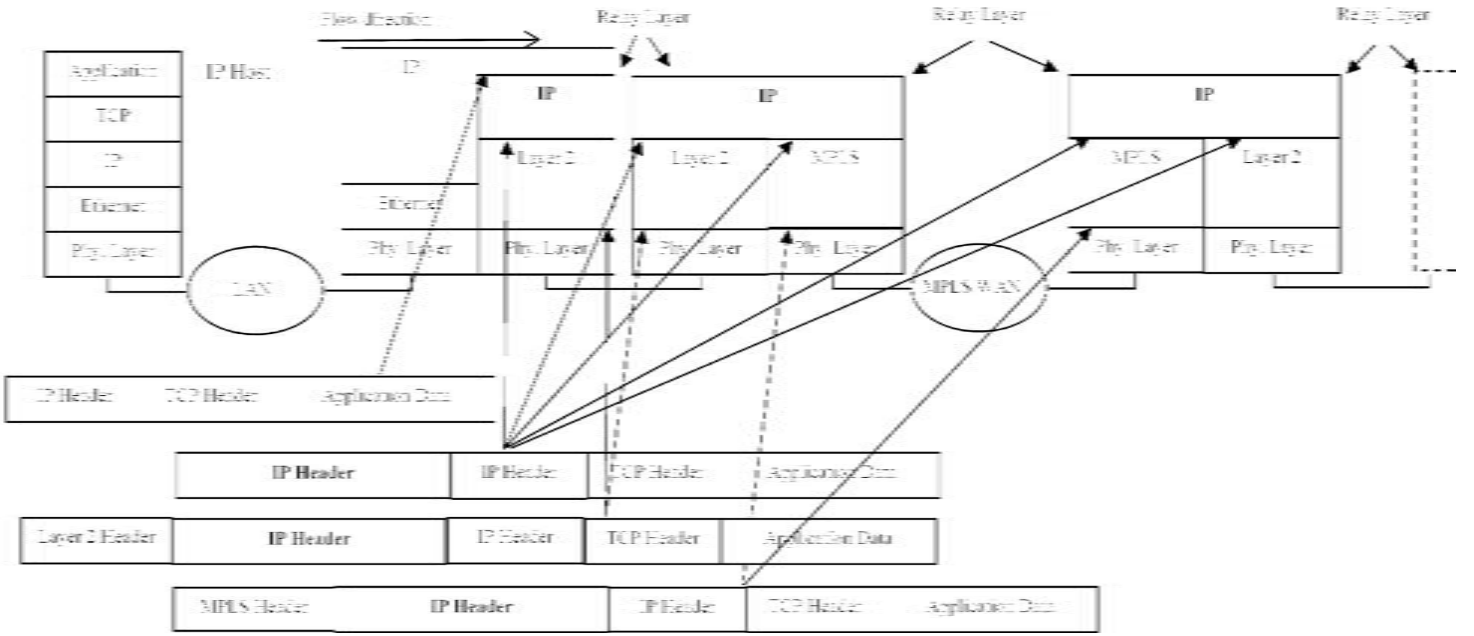


Figure 6.17 IP-centric QoS-PRN connecting an IP host to a private-MPLS network portion: Data flow

Actually, after fixing the encapsulation functions, defining the SLS (the DSCP settings, in this case) and showing end-to-end QoS architectures, one more feature is left to be defined: the way to transport QoS requirements between QoS agents in the network (e.g. the signalling protocol). DSCP values have been fixed but no indication (except for [Fineberg02] approach) has been given for the protocol itself. As said at the beginning of the section about IP-centric approach, the big advantage is that the Relay Layer interface of the QoS-PRN is addressed through IP. In short, the signalling protocol may be managed directly within the Relay Layer.

QoS implementation implies the presence of a signalling protocol to transfer the content of inter-domain SLS and other useful information for QoS control. The signalling protocol triggers resource allocation for incoming connections along the end-to-end path. Resource reservations are managed through proper signalling actions, private to each single network portion. Obviously, not necessarily a signalling protocol providing full QoS support is implemented. A simpler solution but with less QoS guarantees may be preferred. The simplest choice, for example, is to avoid signalling completely. A step forward is to have a light signalling protocol. The choice will have a deep impact on the QoS provision and on the QoS architecture. Some alternatives will be reported in the following.

One of the most recent solutions for IP QoS-PRNs (-RNs) is QoS-Border Gateway Protocol (q-BGP) [Howarth06]. It is more a routing than a signalling protocol but it may be regarded as a first step towards QoS-signalling. Some more detail will be given in Chapter 7. q-BGP-based QoS-PRNs (-RNs) communicate the reachability of specific destinations with a fixed degree of service based on the established SLS and associated with each DSCP value. More information about the available degrees of service may be transferred by exchanging extended payloads. The key point is that no assurance is given on the guaranteed QoS along the e2e path. q-BGP is an instrument to communicate reachability without any explicit resource planning. It is the common language to let the RNs exchange the required information only after the QoS has been installed within the different network portions. The QoS architecture that uses only q-BGP may be considered a light-QoS assurance solution.

Another important signalling is RSVP. It will receive a great deal of attention in the next chapter because it is a full QoS-oriented protocol. Modifications to the original protocol support information transfer about CAC (RSVP-Pre Congestion Notification, PCN) [Briscoe CAC, BriscoePCN, RFC3168] and call priority [BriscoeCAC]. Within IP-centric QoS-PRNs (-RNs), RSVP can be used for resource control as detailed below. It may be utilized both by using all its features, as in IntServ architectures, and by using only a portion of its features, as often done within DiffServ paradigm.

The IETF Next Steps in Signalling (NSIS) working group [RFC4080] is considering protocols to transfer signalling information about data flow along network paths. NSIS adoption is compliant with IP-centric solutions and may support dynamic control as summarized below. NSIS framework will be detailed in the next chapter.

The general architecture including signalling is reported in Figure 6.18. The signalling protocol is generically referenced as QoS signalling.

Even if resource control reveals to be the ultimate key to assure QoS, as evidenced in Chapter 4, four approaches may be implemented within DiffServ IP-centric QoS-PRNs (-RNs): no control, static trunks, DiffServ-Pre Congestion Notification and BBs. It is true also

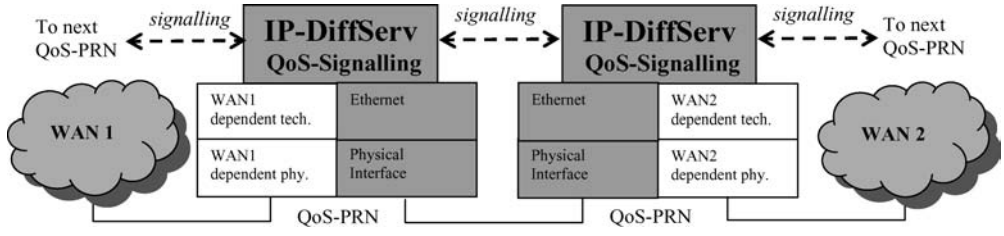


Figure 6.18 DiffServ-IP-centric QoS-PRN end-to-end architecture

for every IP network, including the network portions that implement IP, not only for QoS-PRN and QoS-RN. The choice about resource management also influences the signalling protocol both between QoS-PRNs and internally to different portions.

6.4.3.2 No Control

In this case there is no signalling at all. It is a trivial solution but, in some case, may be efficient. The DSCP classification within QoS-PRNs is used just as a priority mechanism. The model associated may be thought as a battery of buffers to which a specific priority is associated. Traffic is served in conformance with the priority. The only guarantee is that the traffic characterized by higher priority is served before the less prioritized traffic. The rationale under this choice is always over-provisioned. SLS, in this case, should be interpreted as a loose indication about the desired request of service of the users.

6.4.3.3 Static Trunks

No specification to dynamically reserve resources or to receive indications of network resource availability is implemented. User traffic is obviously distinguished through DSCP and served by static trunks. The term ‘static’ denotes the manual management of network resources over large timescales, as, for example, the pre-allocation of a bandwidth trunk for a specific traffic class. Neither traffic prediction nor real time reaction to congestion including Call Admission Control (CAC) is provided. For this reason, static trunks are typically over-provisioned. Obviously, the same static trunks provided within QoS-PRNs need to be “vertically” mapped on private network portions (also called “Autonomous Systems”(AS)), located below, through which the traffic is transported to the next QoS-PRN. This issue is part of chapter 8. The QoS protocol that can be used in this case is just the q-BGP, mentioned before. Also in this case, it cannot be surely defined as a QoS-signalling as reported in Figure 6.18 but, at least, there is an indication about the reachability of the destinations with a fixed SLS-driven degree of service, which depends on the different DSCP. It is not exactly a QoS-oriented solution, at least, following the meaning used in this book, but it is much used in real commercial networks.

6.4.3.4 DiffServ-Pre Congestion Notification (DiffServ-PCN)

It is presented in [BriscoeCAC] and in other documents of the related IETF working group. It supports both CAC and MLPP within the IntServ over DiffServ framework, defined in

section 3.4.3. It constitutes a decentralized and more flexible resource management than static trunks. Each QoS-PRN (Edge Router, in the IP language) is responsible of the QoS maintenance in the AS. It receives specific alerts concerning the congestion state of the traffic classes within the AS. The alerts are build by IP-based AS interior routers, or may be generated by other devices for non-IP-based ASs, by properly setting the 2-bits Explicit Congestion Notification (ECN) field of IP packets. In other words, some bits of DSCP are reserved for PCN. Two alerting states are defined: one for updating CAC and one for triggering MLPP. This approach guarantees a decentralized control because each device independently communicates its congestion state without centralized monitoring. The congestion information is reported to the QoS-PRN in order to shape the incoming traffic from the other network portions. In turn, the first QoS-PRN alerted about congestion may perform its specific CAC and MLPP decisions or may automatically forward the congestion state to the upstream QoS-PRNs of the path. The mentioned RSVP-PCN Protocol must be installed within each QoS-PRN to carry information about the actual rate of a given traffic class and about the priority of the calls. The key idea of the control structure is fixing sustainable rate limits for each traffic class, whose violation is reported towards the closest QoS-PRN before uncontrollable congestion could generate QoS degradation. A continuous monitoring of rate fluctuations must be reported from internal devices to the QoS-PRN, without explicit portion's internal signalling. In other words, vertical QoS mapping is much simplified and there is no guarantee that the QoS requirements transported through the signalling protocol are really mapped on the single portions. The monitoring information is embedded in regular traffic packets, properly ECN marked. QoS-PRNs update the current estimates of each traffic level, perform CAC and MLPP decisions, and forward the information through RSVP to upstream QoS-PRNs of the end-to-end chain.

Even if, from the architectural viewpoint, this approach perfectly map on Figure 6.18, DiffServ-PCN contains some drawbacks that affect its QoS provision.

- The mentioned sustainable rate limits are statically configured, unless a signalling scheme is used within DiffServ, for instance considering future NSIS results.
- In case the RSVP-PCN and Emergency RSVP messages build by QoS-PRN traverse a connectionless core (the different portions where there is no QoS-guarantee), the time interval necessary to implement control reactions cannot be dimensioned precisely.
- The mentioned monitoring mechanism needs a continuous real time estimation of the actual rate of each traffic class. It is implemented for each couple of QoS-PRNs along the path. It may be computationally expensive and requires some time to reach convergence.

If CAC, MLPP and regular IP fault tolerant re-routing must guarantee QoS with tight time constraints, such disadvantages make the planning of the network very difficult. It is intuitive that these services cannot be fast enough if unexpected congestion takes place.

The mentioned QoS signalling may be integrated by q-BGP.

6.4.3.5 Bandwidth Broker

This approach totally maps the QoS architecture presented in [Jia01] and reported in Figures 6.2 and 6.3. A BB is an entity responsible for the resource control of a network portion. It represents the Network Management Entity. It maintains a complete real time

knowledge of traffic and resource states. Specific protocols may be used to monitor states (e.g. OSPF-TE, IS-IS-TE) and to allocate resources (e.g. RSVP, NSIS). Each single QoS-PRN may implement a centralized Network Management Entity by receiving information by the Network Portion Management Entity, for example, the local BB of that domain, as clarified in section 6.7. The BB negotiates traffic contracts to support SLSs with neighbour network portions and may periodically allocate network resources as a function of the traffic matrix. Traffic contracts may be renegotiated following predefined timescales. BBs also implement CAC and QoS management. The QoS-PRN user plane performs data encapsulation, while the control plane implements resource management functions as well as bandwidth allocations, SLS transfer and mapping through the BB entity. The data sender emits a resource allocation request to the QoS-PRN. The request contains some information about traffic profile and destination. The request is transferred to the BB that performs resource control based on the state of its domain. Each network portion may have a local BB, the Network Portion Management Entity, which dialogues with the BB located at QoS-PRN, as shown in section 6.7. If a call can be accepted, the request is forwarded to the neighbour QoS-PRN; otherwise the call is rejected, notifying the source. The operation is iterated QoS-PRN after QoS-PRN through BBs up until the destination. If the inter-domain end-to-end call is accepted, the source is notified. During the backward propagation, local BBs perform resource reservation within their domain.

The signalling protocol used by local BBs and QoS-PRN BBs may be RSVP-PCN, or, if hard and not loose QoS provision is requested, as explained just below, some evolution taken from the NSIS suite, as detailed in the next chapter. In this last case it is possible to have a full respect for SLS requirements.

6.4.3.6 Hard versus Loose QoS Guarantees

Except for the possible evolution of the BB approach, the solutions presented above offer loose guarantees QoS, following the terminology in [Howarth06], or Controlled Load (CL) QoS, as for IntServ, defined in section 3.4.1. It means they support neither real time management of network resources to follow traffic variations and topology changes nor resource optimization to minimize network deployment and maintenance costs. Together they can be considered the key support elements for hard guarantees QoS [Howarth06], that is QoS delivery with tight constraints and full conformance to SLS.

Hard guarantees QoS can be approximated within the IP DiffServ-PCN and Static Trunks architectures through appropriate planning and accurate resource tuning, often together with over-provision. In theory, the BB paradigm may support hard guarantees QoS, exploiting the future evolutions of the NSIS suite. So, even if, at the best of the author's knowledge neither standards nor pilot networks of BB DiffServ schemes have been deployed up to now – it is one of the solutions that are more promising for future and will be taken in deep considerations in the reminder of the book.

The central point is that the DiffServ paradigm application may be not completely satisfying. It (i) looses the reference to single flows because DSCP is the only tool to identify them; (ii) does not guarantee sufficient SLS flexibility, because traffic may need to be aggregated, as stated in Chapter 4, sections 4.1.2 and 4.2.1, concerning Flow Identification and (iii) does not implement any signalling to perform dynamic resource control and optimization. Actually, points (i) and (ii) are true also including the BB solution.

Concerning point (ii), and recalling attention to the services reported in Tables 3.6–3.9, it is remarkable that each SLS is expressed in terms of (1) traffic descriptors; (2) conformance testing parameters; (3) required performance guarantees as packet loss rate, delay and delay jitter and, when necessary (4) priority preemption, as stated in Chapter 1, Tables 1.3 and 1.4. Additionally, connection protection [Pongpaibool04] levels may also be part of the SLS. The proposed composition is derived from different standards and literature sources, each emphasizing different objective parameters for the QoS definition in inter-AS [Tacoms05, Sorteberg05, Howarth06, Gozdecki03, Pongpaibool04]. All this information should be encoded in the DSCP of the packets.

Moreover, in future, customer needs may increase due to new applications. Enlarging the granularity of QoS constraints may be mandatory for Service Level Specifications (SLSs). If MLPP and connection protection classification are also required, the 6 bits of the DSCP may reveal to be insufficient for SLS categorization, especially if some parts of the DSCP itself are used for control purposes as in Tables 3.6–3.9.

Concerning point (iii), a dedicated signalling to control resources along end-to-end path is needed if real time reaction to network congestion is pursued. It covers both inter-AS and intra-AS resource control scopes, as envisaged in previous sections and detailed in section 6.7.

The adoption of an alternative technology matching the previous points and supporting both intra-AS-QoS [Awduche99] and inter-AS-QoS [RFC4655] may be of great interest. This is the rationale behind the proposal of the MPLS-based architecture, as well as behind the IPv6-centric QoS approach, introduced in section 6.6. MPLS has been already partially used in [Fineberg02], even if only as an intra-AS hard QoS solution, as shown in Figure 6.17.

6.5 MPLS-centric QoS Approach

IP traffic at the QoS-PRNs is encapsulated within the MPLS frame. The MPLS shim header is added at the first QoS-PRN of the end-to-end path, which acts as a regular Label Edge Router (LER) by identifying traffic, classifying the SLS and applying the label. The opposite operation is implemented at the last QoS-PRN before the destination. Intermediate QoS-PRNs act as conventional Label Switch Routers (LSRs). Since the network portions (the ASs) are not necessarily MPLS capable, the labelled packets will be tunnelled within proprietary technologies and forwarded to the next QoS-PRN.

The MPLS-based solution may be structured into two consecutive steps.

6.5.1 MPLS-integrated QoS Approach

It can be seen as a support for QoS provision offered to DiffServ IP networks. In this case, MPLS is used for packet switching and for the establishment of intra-AS, as in [Fineberg02] and Figure 6.17, explicitly routed tunnels.

The architecture is shown in Figure 6.19, for QoS-PRN, and in Figure 6.20, for QoS-RN. The data flow encapsulation is shown in Figure 6.21 for a generic WAN encapsulation. If the WAN should be implemented in MPLS as in Figure 6.17, the encapsulation may be simplified so avoiding one MPLS header but only if the MPLS label meaning is the same both within QoS-PRN and within the single WAN. The concept is the same as expressed in section 6.4.1, Figures 6.13a and 6.13b, for IP in cascade.

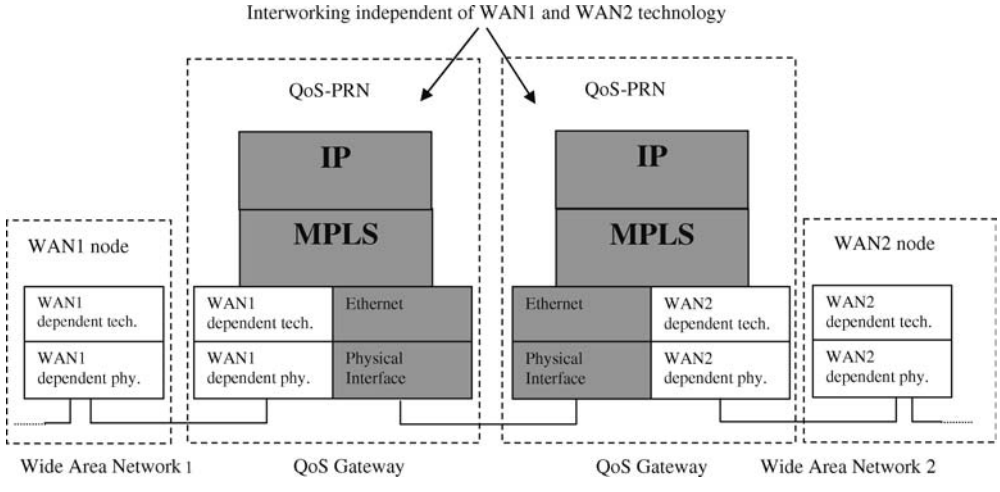


Figure 6.19 MPLS-Integrated QoS-PRN

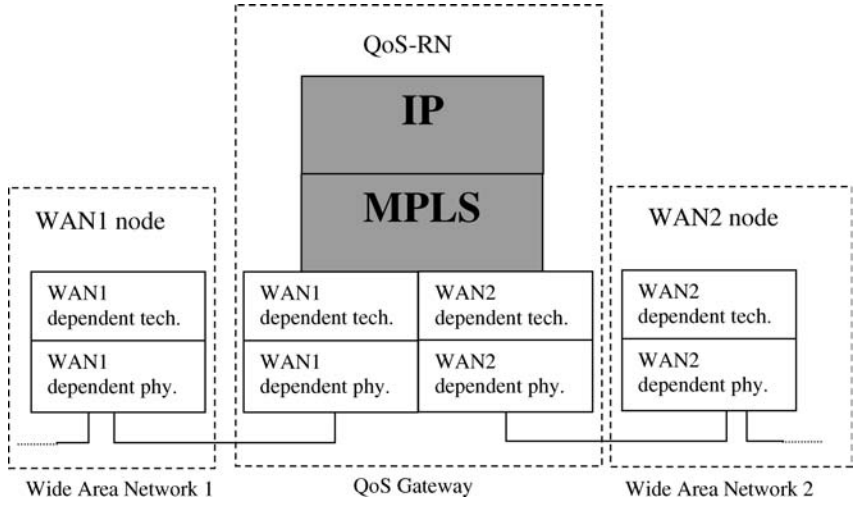


Figure 6.20 MPLS-Integrated QoS-RN

Within the same MPLS-Integrated framework, two different implementations are feasible. In these cases, the MPLS layer is added below IP so that hard guarantee QoS may be offered without the adoption of IP non-standard solutions based on BB. Actually, it is the solution commonly used in real deployed networks because it allows having guaranteed quality by using network components available in the market. It may be applied both as QoS-PRN solution and as a solution within each AS. The common SLS language is IP, as well as for IP-centric approaches. If applied for QoS-PRN, the solution may be defined as MPLS integrated QoS-PRN. It may also be seen as a special QoS-oriented case of DiffServ-IP-centric QoS solutions.

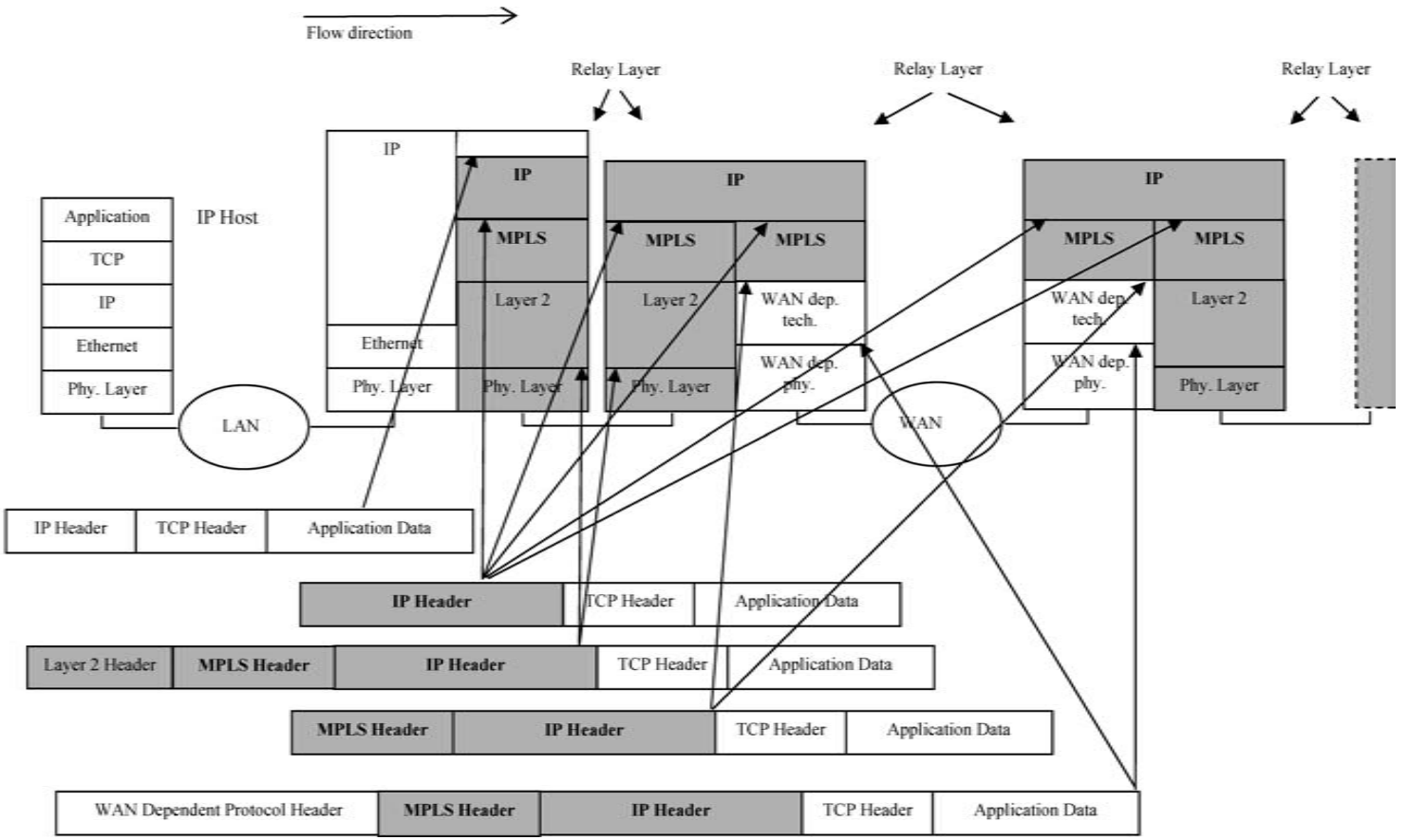


Figure 6.21 MPLS-Integrated QoS-PRN connecting an IP host to a WAN-dependent technology portion: Data flow

6.5.1.1 EXP-Inferred MPLS-Integrated QoS Approach

The information concerning routing is inferred from the 20-bit MPLS label and the traffic class, that is scheduling and drop precedence/packet discarding, the SLS in the DiffServ terminology, is inferred from the 3 bits of the Experimental (EXP) field of the MPLS header. The approach is called EXP-inferred "Label Switch Path" (LSP). Note that in EXP-inferred LSPs, up to only 8 SLSs can be defined, but it may be still acceptable within many DiffServ networks. Figure 6.22 shows the EXP-inferred mapping concept.

6.5.1.2 Label-Inferred MPLS-Integrated QoS Approach

Another option is also possible, called "label-inferred LSP", where both routing and scheduling treatments are inferred from the MPLS label and only the drop precedence is inferred from the EXP field. If the drop precedence concept is not used, as often done in DSCP definitions, packet treatment depends on the MPLS label. In practice, the DiffServ SLS is ruled by the MPLS label. Figure 6.23 shows the label-inferred mapping concept not using drop precedence concept.

6.5.2 Full-MPLS-centric QoS Approach

The evolution of MPLS-integrated QoS-PRNs implies a full use of MPLS, which acts as both Layer 2 and Layer 3, except for QoS-PRN addressing at control plane. In other words, the presence of IP above MPLS is necessary to route signalling packets. It will be better

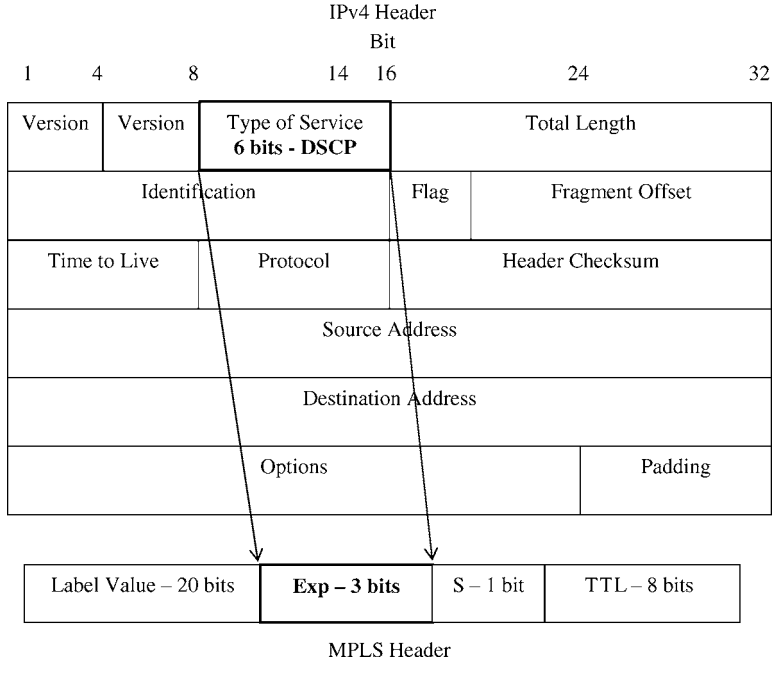


Figure 6.22 EXP-inferred DSCP mapping

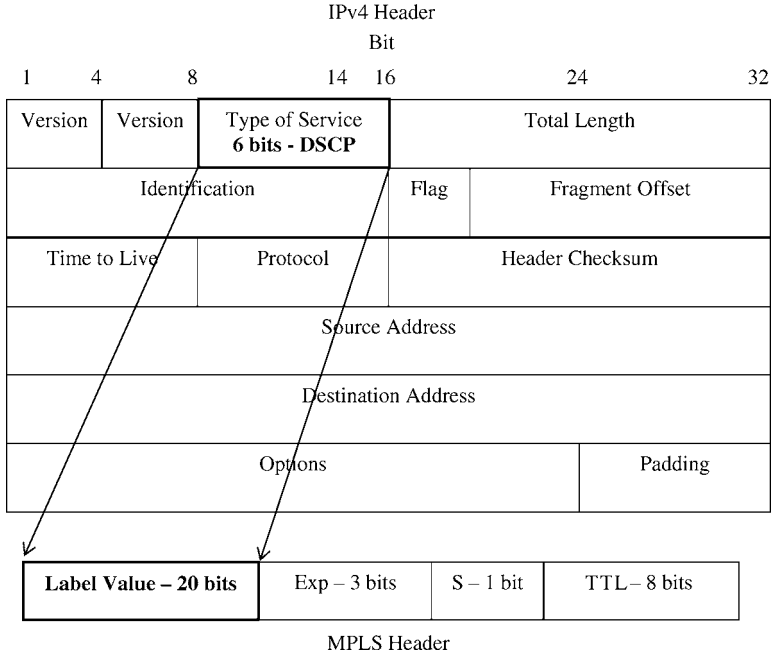


Figure 6.23 Label-inferred DSCP mapping

defined just below. SLSs are based on the label value and may include MLPP and fault tolerance guarantees. In this view, MPLS QoS-PRN defines a generalized version of regular label-inferred LSPs. The solution is called "Full MPLS QoS-PRN". The number of possible SLSs is hugely extended. The protocol architecture for data traffic is reported in Figure 6.24, for QoS-PRN, and in Figure 6.25, for QoS-RN. The concepts expressed in Figure 5.12 (and 5.14) are detailed: Layer and Interface to be defined are now indicated as Physical Interface and MPLS acts both as Relay Layer and as Layer 2. The QoS-PRN architecture at the Relay Points works as an LER if the source/destination host is directly attached. LER functionalities include assignation/dropping and forwarding to next LSR/LER and to single LAN. The architecture works as a LSR if the source/destination host is not directly attached. LSR functionalities include Label switching and forwarding to next LSR/LER.

MPLS-centric architectures implies that only MPLS packets have meaning in QoS-PRNs and that the common language is MPLS, at least from first QoS gateway to the last one in the end-to-end path. This is, essentially, the topical difference with the MPLS-integrated solution.

Actually, the overall end-to-end QoS network is seen as a full MPLS network (actually the network from first QoS-PRN architecture to last QoS-PRN through the end-to-end path is full MPLS), if seen at the Relay Layer. Technically, the network portions may be seen by the QoS-PRN as "abstract nodes" that are defined ([RFC3209]) as a group of nodes whose internal topology is opaque to the ingress node of the LSP. An abstract node is said to be simple if it contains only one physical node. In the case presented in this work, the "opacity" is complete, concerning not only QoS routing ([RFC3209]) but also QoS technologies that can be different from MPLS.

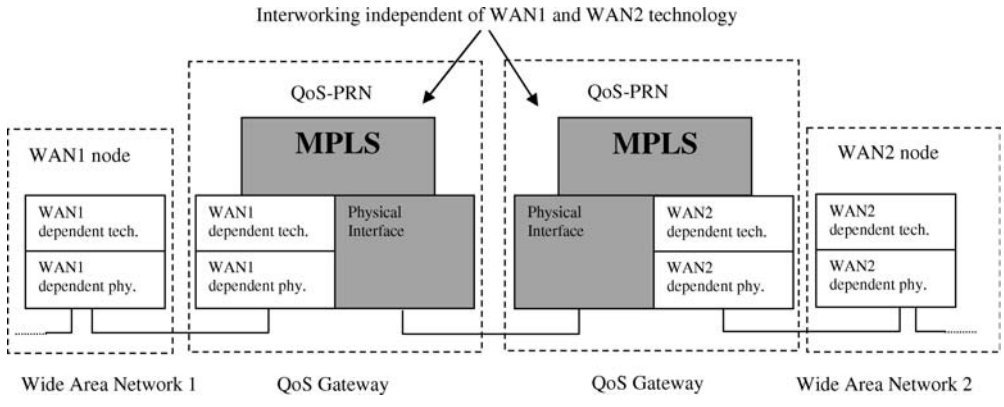


Figure 6.24 MPLS-centric QOS-PRN

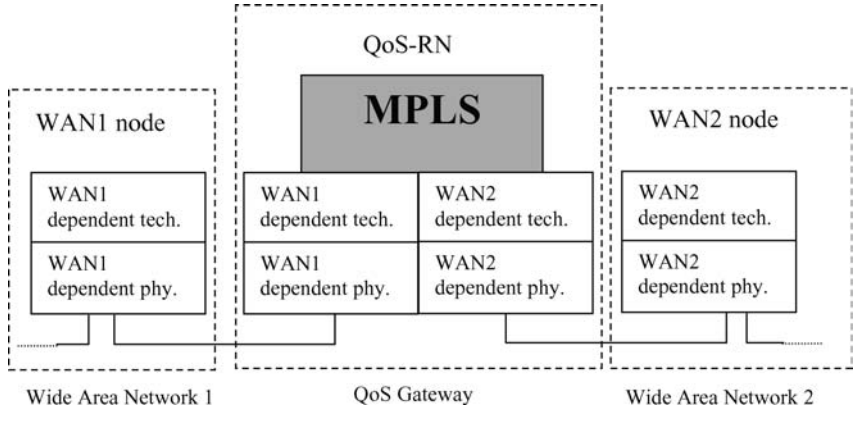


Figure 6.25 MPLS-centric QOS-RN

Figure 6.26 contains a picture showing the concept of abstract node applied to the QoS-PRN at Relay Points. MPLS packets are transported over private domains embedded in the proprietary technology together with the host architecture.

From the operative viewpoint, pure host packets are passed to the MPLS layer that adds the label and forwards it to the next QoS-PRN. MPLS packets are transported over the Relay Points and over the WAN backbones. MPLS labels can easily be carried along WANs which are not necessarily MPLS capable. A traffic flow composed by IP packets plus the MPLS label can be tunnelled along an ATM-based, ISDN-based WAN backbones and along another IP network. The encapsulation of MPLS in IP is also a topic addressed by the IETF (see, e.g. [RFC4023]).

In this context, MPLS represents the reference communication protocol among WANs. The choice, even if new for its application, is justified by recent literature. Reference [RFC4023], for instance, states "… it is possible to replace the top label of the 'MPLS stack' with an IP-based encapsulation, thereby enabling the application to run over networks which do not

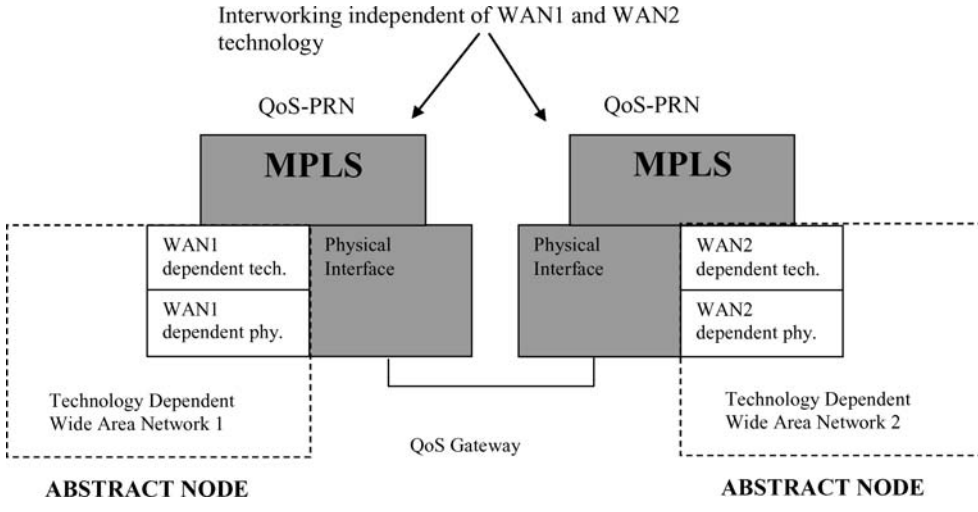


Figure 6.26 Abstract nodes

have MPLS enabled in their core routers". The Full-MPLS-centric approach allows using MPLS in all its powerful features both concerning QoS, because it can transmit and assure QoS requirements for each specific connection, and concerning the role of convergence technology. In more detail, Full-MPLS-centric architectures

- allow "per flow" QoS by using MPLS switching (e.g. Figure 3.14) in each QoS-PRN (RN);
- admits a large number of SLSs;
- can manage, by means of proper definitions of Forward Equivalent Classes available in the network, full MLPP capabilities;
- allow traffic-engineering capabilities among the WANs (see, e.g. [ID-Vasseu03]).

Using MPLS at the Relay Points is a very easy solution to implement for an IP network and brings significant advantages when it is applied to ATM and ISDN networks. It has (for now and future) clear advantages for the QoS management of the overall network.

Some examples about data encapsulation at QoS-PRNs may be added to further clarify the behaviour from the operative viewpoint. The IP host carries a data application and implements the necessary IP stack in the example of Figure 6.27. The first QoS-PRN met along the end-to-end path acts as an LER and identifies the traffic applying the MPLS label. The same operation will be implemented at the last QoS-PRN before the destination. Intermediate QoS-PRNs act as LSRs, where the IP host packets at the Relay Layer are initially encapsulated within the MPLS information and transported over the ATM backbone through the AAL interface. At the exit of the ATM backbone that, for the QoS-PRNs, is only an opaque portion of the MPLS end-to-end path, ATM information is dropped and the MPLS encapsulated IP host packet is passed through the next proprietary Layer 2 protocol. The traffic over the WANs comes from the host protocol stack with the addition of the MPLS shim header (the MPLS label) and is tunnelled along the network WANs, which are not necessarily MPLS capable. Similarly to the IP-centric approach, Figures 6.28 and 6.29

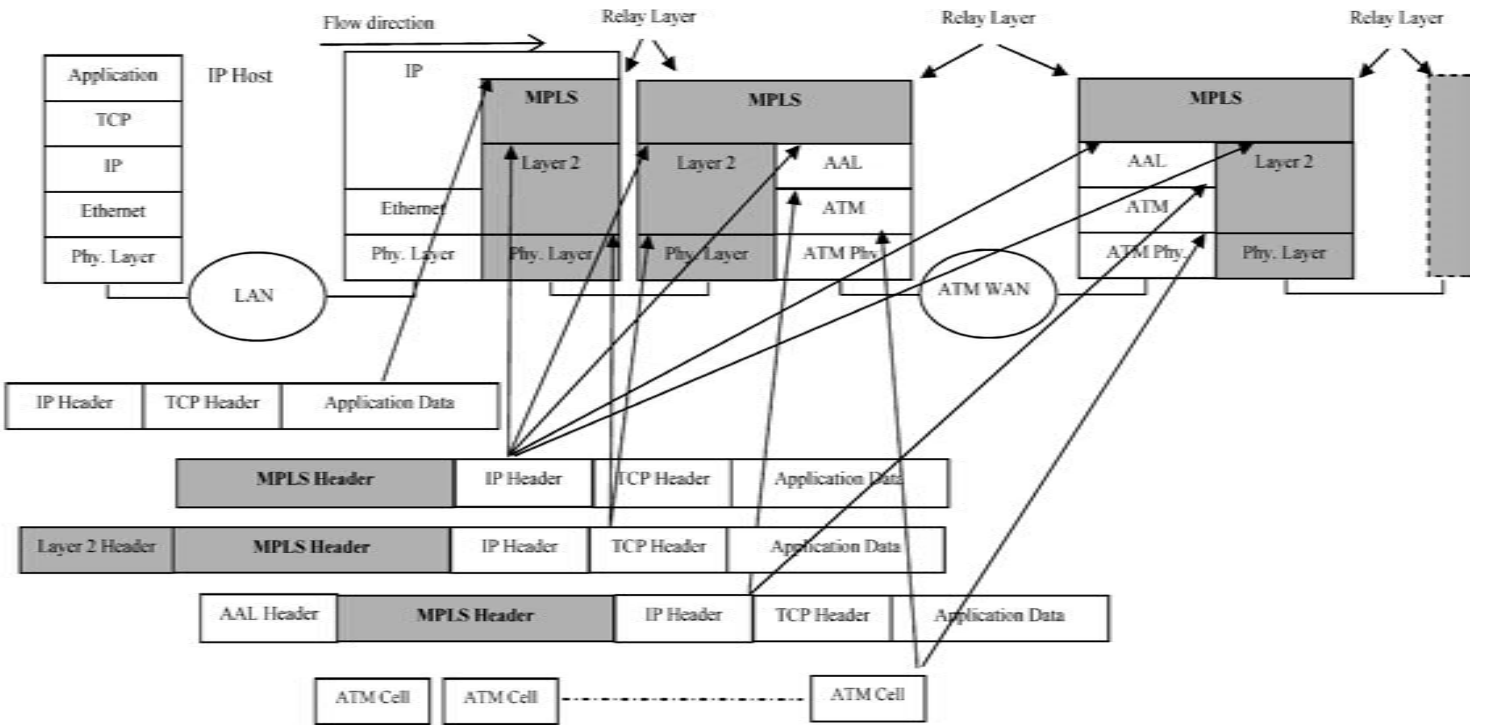


Figure 6.27 MPLS-centric QoS-PRN connecting an IP host to an ATM network portion: Data flow

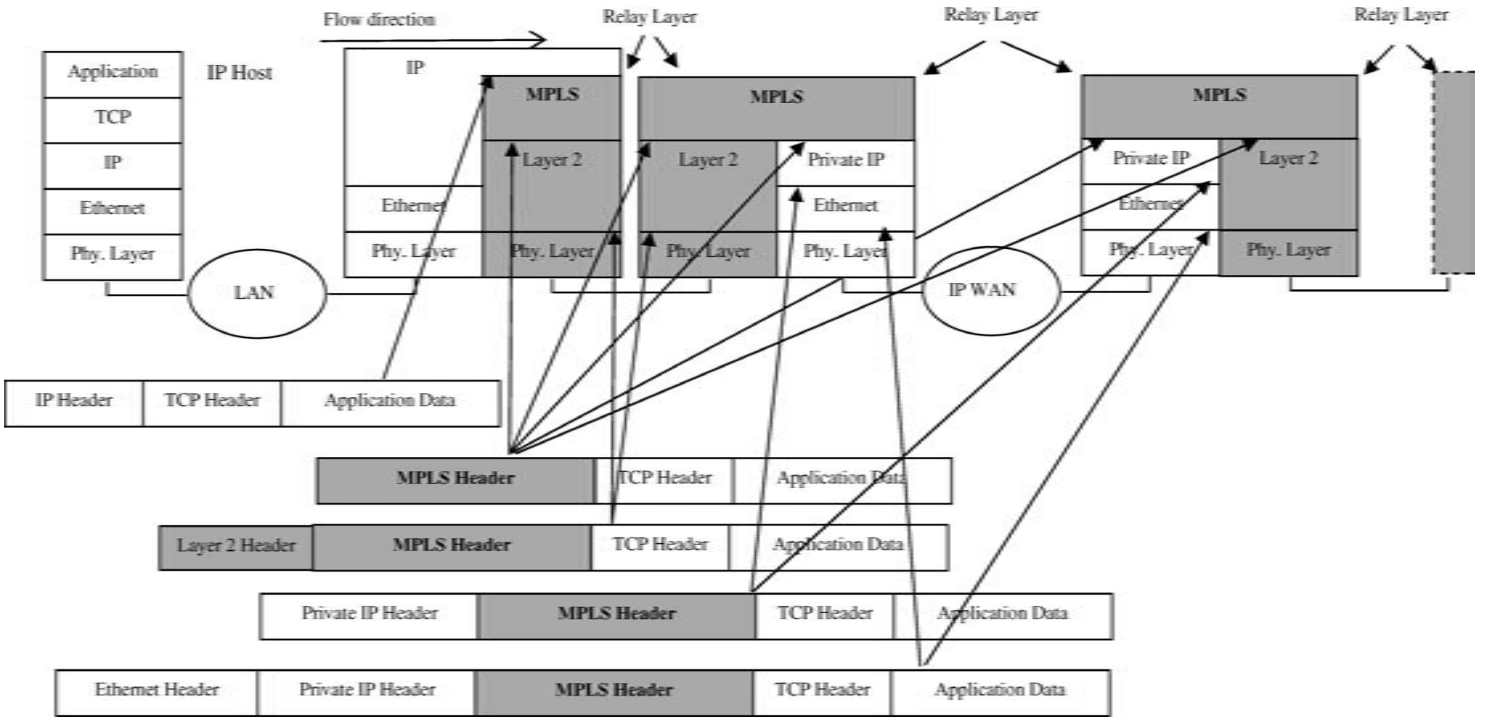


Figure 6.28 MPLS-centric QoS-PRN connecting an IP host to an IP network portion: Data flow

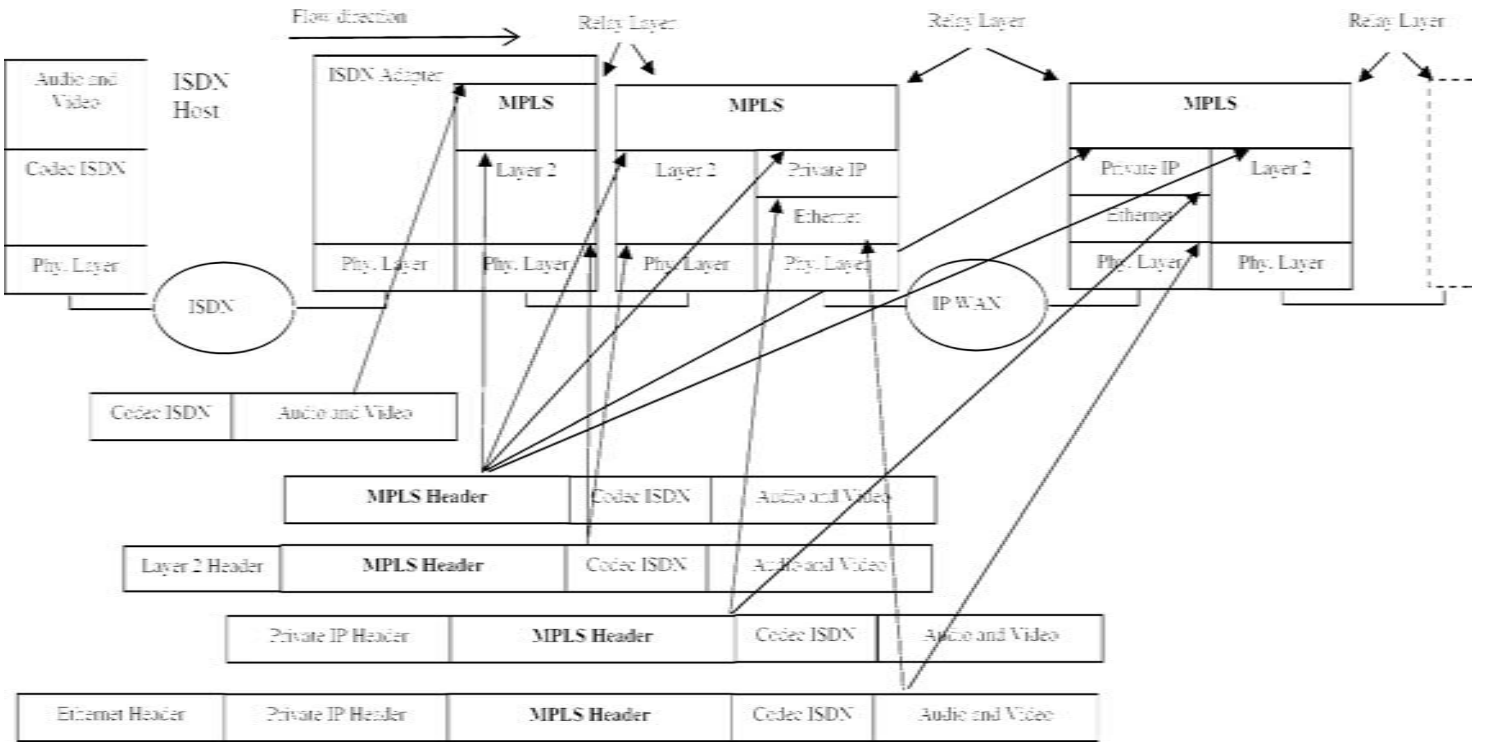


Figure 6.29 MPLS-centric QoS-PRN connecting an ISDN host to an IP network portion: Data flow

show the data flow encapsulation over a private IP WAN for an IP host and an ISDN host respectively.

Concerning VoIP and ISDN interoperability, the observations made in the IP-centric approach are still valid.

MPLS-centric approach has many advantages but implies a big effort for vertical QoS mapping because an SLA distinguished for each connection must be mapped also over private technologies that do not necessarily provide services for each flow as, for example, IP DiffServ. Indications about it will be provided in the chapter dedicated to vertical mapping.

While, as far as IP-centric approaches are concerned, there are many studies and projects related to end-to-end QoS architecture, the Full-MPLS-centric approach is an idea for future and it does not include operative architecture. Nevertheless, the composition of the architecture may be the same, with the central role of QoS-PRNs (RNs). The only problem may be addressing the QoS-PRNs, which are MPLS-switches with increased functionalities. Actually, there is no addressing scheme for MLPS. How can signalling reach the QoS-PRN locations? An addressing scheme is mandatory. There are two choices: either designing a new addressing mechanism based on the previous experience (e.g. ATM, ISDN, plain telephone networks) or using IP again, as done for the MPLS products in the market. The latter is definitely simpler. In this case, while data flows through the end-to-end network encapsulated in MPLS packets, signalling uses a full IP network that overlays the MPLS one. In practice, each MPLS-centric QoS-PRN owns an IP address used only for signalling. In more detail, without entering the signalling protocols details, left for the next chapter, [Fineberg02] suggests that "MPLS may be provisioned statically or use a signalling protocol such as RSVP tunneling extensions (RSVP-TE) or Constrained Routing Label Distribution Protocol (CR-LDP)". A signalling protocol "can perform a variety of functions, such as setting up LSPs traversing specified sequences of the LSRs derived from the CR analysis; create the path state in each LSR by performing label allocation, distribution, and binding; reserve resources in each LSR, including bandwidth, delay, and packet loss bounds; reassign the network resources as necessary; dynamically reroute during network congestion and failures; monitor and maintain explicitly routed LSP state; provide LSP information to the network management system; and other functions" [Fineberg02]. The sentence is referred only to the core domain in [Fineberg02], as widely

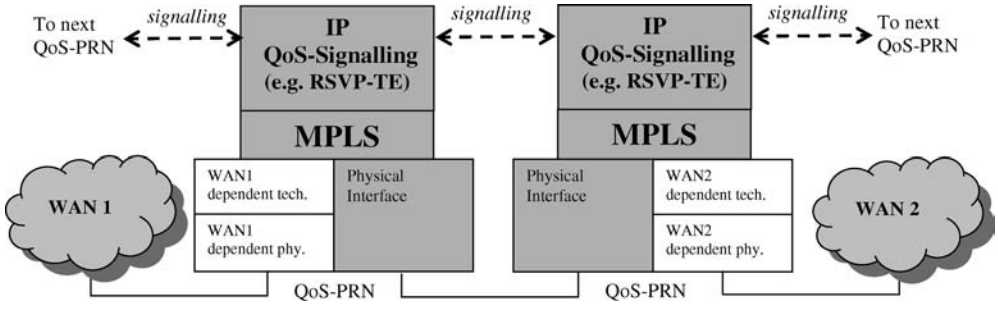


Figure 6.30 MPLS-centric QoS-PRN end-to-end architecture

explained before, but it perfectly matches the need of Full-MPLS-centric end-to-end architectures. The end-to-end protocol architecture for signalling cannot be the same as used for data flow (as done for IP-centric approach) but a new architecture where IP acts over MPLS is strongly necessary. Figure 6.30 contains the mentioned architecture including two generic WANs (WANs 1 and 2), always using the generic expression of Figure 6.7 together with a possible example of QoS signalling protocol used in MPLS and called "RSVP-TE". RSVP-TE is used to set the labels over the path (at each QoS-PRN) and to signal QoS requirements (SLS) over the path. Its features will be explicitly clarified in the next chapter, dedicated to signalling. SLS requirements need to be mapped carefully from MPLS to private network portions and this is the object of vertical QoS mapping. It is assumed that an IP address plane is available in each QoS-PRN for signalling, thus allowing any QoS-PRN to manage a proper routing scheme (e.g. by means of MPLS traffic engineering functionalities [ID-Vasseu03], [OSPF-TE02], [TE-QOS01]) among the WANs and to allow signalling to flow through the end-to-end path. As regards security considerations, the major issue is the need to control and authenticate secure access along the bandwidth pipes carried along the WANs. This requirement is strictly related to the need to assure security capabilities to the tunnels employed in the WAN backbones. Hence, the security considerations of, for instance, [RFC2747] should be taken into account during the deployment of the overall network.

6.6 IPv6-centric QoS Approach

The advantages for QoS management offered by the Full-MPLS-centric approach may be combined with the simplicity of the IP-centric approach. As shown in Chapter 3, there are two versions of the IP, IPv4 and IPv6, whose main difference concerning QoS stands in the features to identify a flow. Actually, if either DiffServ or IntServ approach is used, there is no difference between IPv4 and IPv6 from the QoS viewpoint [IPJ-June05], but it is possible to exploit the IPv6 features to manage QoS through a wider vision over future. In other words, if the 8-bit "Traffic class" field is used to identify flows, IPv6 QoS features are strictly equivalent to IPv4 ones (where the ToS field is used) and DiffServ paradigm may be applied exactly in the same way. The same holds true for IntServ. On the other hand, if the 20-bit "Flow Label" is exploited, there is a strict equivalence with MPLS and with ATM concerning flow identification. It opens the door to new QoS management possibilities. The full use of IPv6 flow identification features implies to overcome the DiffServ paradigm. The idea is to use IP together with all the QoS-control schemes, which characterize ATM and MPLS. In practice, it allows to keep the simplicity of IP concerning the operational protocol features, as well as addressing and interfacing, together with the QoS management power of ATM and MPLS. The concepts have been used in Chapter 3 where the features of IPv6 concerning QoS have been outlined up to the definition of CSF6N (section 3.6), which extends the number of traffic classes within a DiffServ-like scenario, and of F6SN, which introduces a full IPv6-switched network. Chapter 3 introduces CSF6N and F6SN solutions for technological homogeneous networks but the concepts may also be extended to QoS-PRNs. The idea of IPv6-centric architectures is to extend SLS and other features given for IPv6 networks (CSF6N and F6SN) to the overall end-to-end network.

Going back to the architecture, the IPv6-centric is the same as IPv4 one, simply substituting IPv4 with IPv6. The QoS-PRN and QoS-RN architectures are reported in Figures 6.31 and 6.32 for the sake of completeness, specifying also the possible QoS paradigm to use (CSF6N/F6SN).

It means that, as in the IPv4 case, QoS-PRN acts as an IP router and can be addressed by using the IP format. It is a huge advantage concerning signalling. The reference QoS language among PRNs is no longer the DiffServ paradigm (if it is there is no advantage concerning QoS), but the detailed SLS is specified either for CSF6N or for F6SN paradigm in strict dependence of the QoS choice performed for the QoS-PRN (-RN). In the first case, the QoS-PRN (-RN) includes the feature of a CSF6N router managing, in theory, 2^{20} traffic classes. In the second case, QoS-PRN (-RN) is a F6SN node, as shown in Figure 3.22, which contains one Flow Label Translation Table for each possible outgoing link (e.g. outgoing WAN, in the QoS-PRN case), similarly as shown in Table 3.10. It is also true that each entry of the table needs to be mapped over a specific technology within private WAN. If the

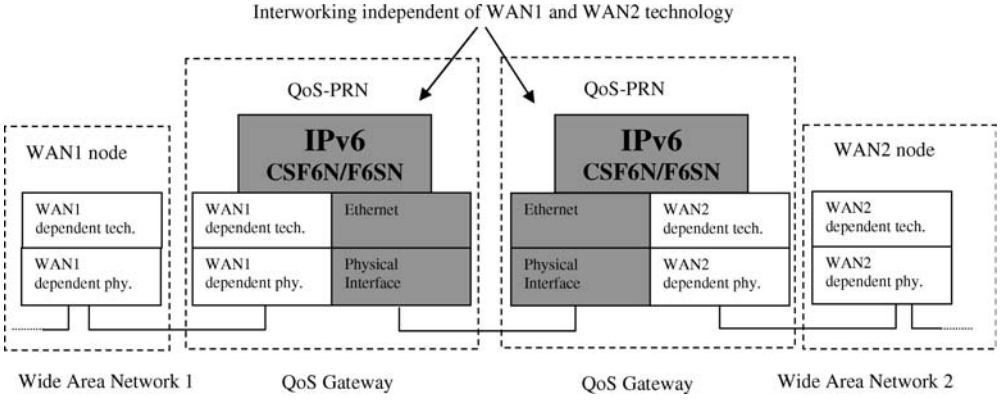


Figure 6.31 IPv6-centric QoS-PRN

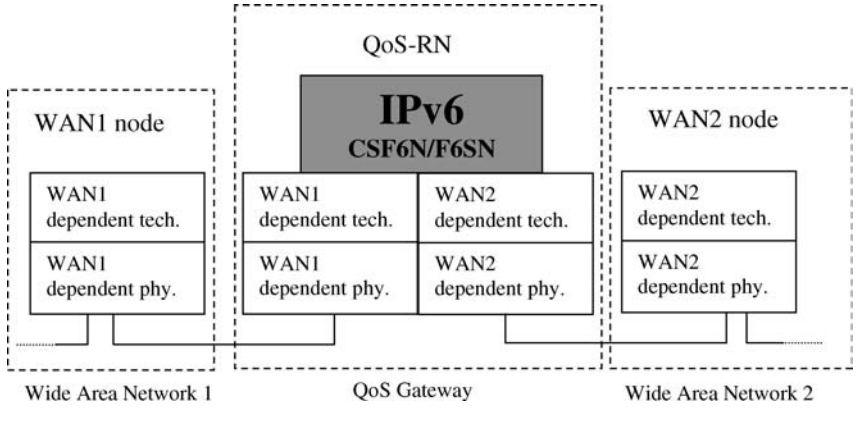


Figure 6.32 IPv6-centric QoS-RN

WAN cannot identify the same number of flows, the WAN itself will take care of the mapping action encapsulating different flows within tunnels and allocating sufficient resources to tunnels so that the operation is transparent for the user, from the QoS viewpoint.

If a full-featured IPv6 is adopted, the overall network is an IPv6 meta-network for QoS management, which connects WANs that implement possibly other technologies. The vertical mapping problem is particularly relevant in this case because the number of traffic classes assured by IPv6-flow labels may be simply matched over MPLS and ATM WANs, but some additional effort is required if IPv4 WANs are traversed. Actually, the effect is multiplexing traffic from a larger to a smaller number of queues. This situation will be deeply investigated in Chapters 8 and 9.

The data flow is the similar as the IPv4 case. Figure 6.33a shows the IPv6-centric QoS-PRN connecting an IPv4 host (explicitly indicated in the figure this time) to an ATM network portion, while Figure 6.33b shows the same case but having an IPv6 host at the source. Similarly as discussed for Figures 6.13a and 6.13b, the same QoS paradigm is used for both IPv6 host and IPv6 QoS-PRN so allowing dropping an IPv6 header. Figure 6.34 shows the data flow over a private IP WAN, which may be thought both as an IPv4 and as an IPv6 network that uses a different QoS paradigm with respect to QoS-PRNs. The ISDN host case has been already discussed for IPv4-centric architectures. The same comments may be applied.

Due to the extended SLS, the control techniques used within the network are more and more important for QoS guarantee. The management effort is more serious than in the DiffServ case. Again, each QoS-PRN (-RN) may be modelled as a battery of virtual buffers implemented in software, which separate the traffic of the different traffic classes supported by the network. Each buffer will be provided with a specific bandwidth. The number of buffer is much larger than the previous case. Management techniques should be inherited from ATM more than by IPv4. An overall architecture, including signalling protocols to manage flow labels, is strongly needed. Strictly speaking of QoS architectures, the approaches previously presented for the IP-centric approach (which, anyway, includes also IPv6-centric approaches) are perfectly suited also in this case. In more detail, the possible abstraction of the QoS end-to-end architecture at management plane proposed in this book and reported in Figure 6.6 obviously includes also the IPv6-centric approach. The peculiarity stands in the fact that the common language, for both addressing/signalling and data flow, is IPv6. The IPv6-centric protocol architecture at the management plane is shown in Figure 6.35 (similarly as in Figure 6.7) by using the generic QoS-signalling identifier. Obviously, the use of a full-QoS support is supposed in both IPv6 approaches. In other words, the CSF6N solution that, theoretically, might be applied also over a No control DiffServ approach is supposed to be implemented together with BBs and all the other required control entities.

6.7 QoS Overall Architecture

Generalizing the architectural schemes of Figures 6.1 and 6.6, it is possible to summarize the features required by QoS architectures [ETSI-TS-102462]. Figure 6.36 contains the end-to-end QoS network model using single RN instead of double PRN for the sake of simplicity. There is a resource controller for each network portion. Controllers are shown as single entities but,

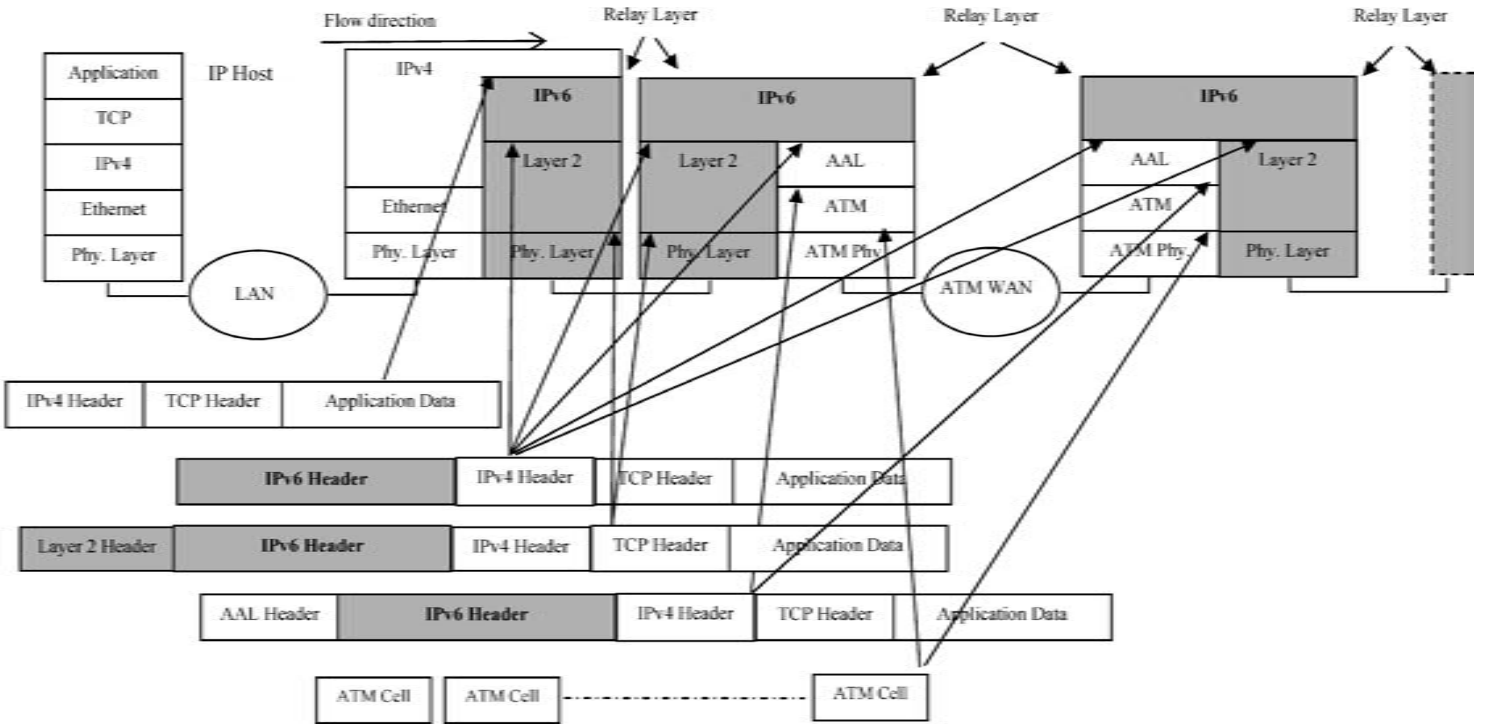


Figure 6.33a IPv6-centric QoS-PRN connecting an IPv4 host to an ATM network portion: Data flow

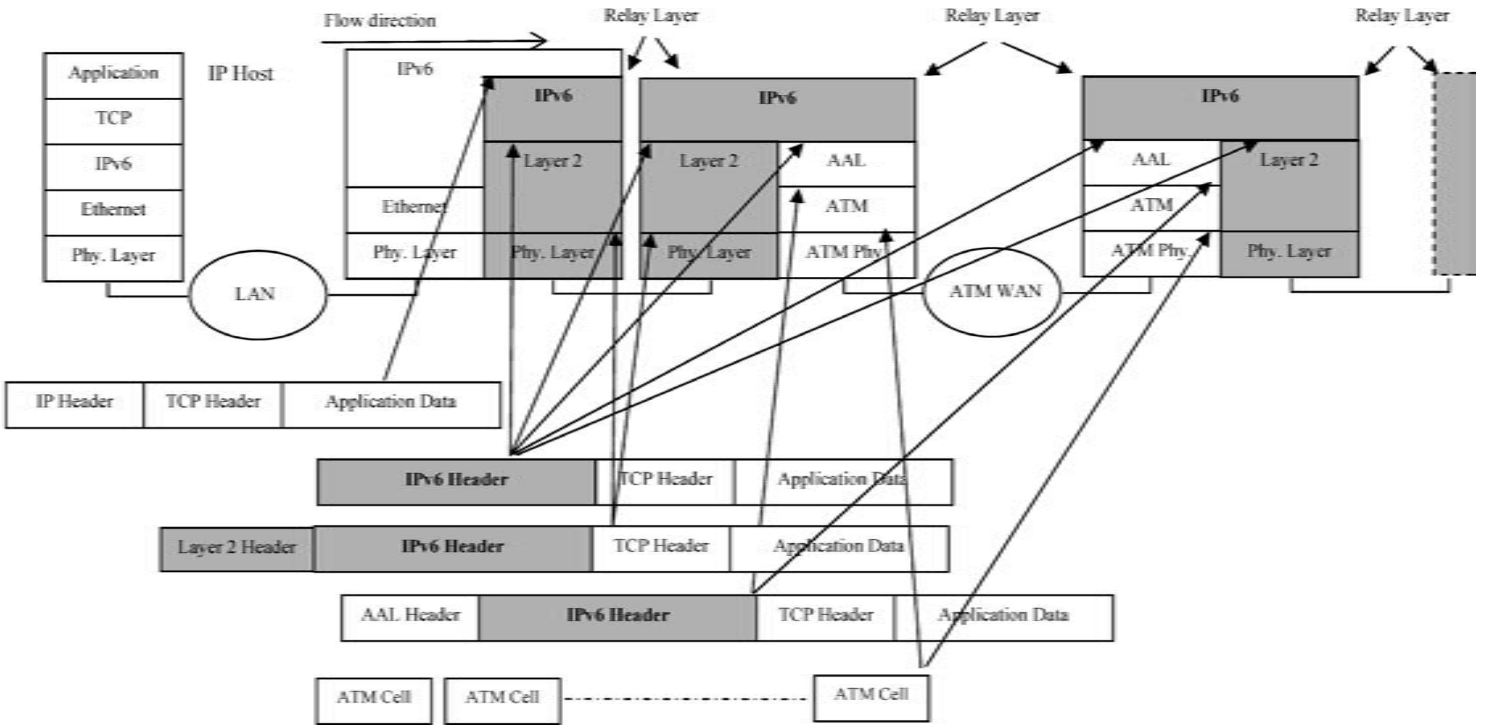


Figure 6.33b IPv6-centric QoS-PRN connecting an IPv6 host to an ATM network portion: Data flow

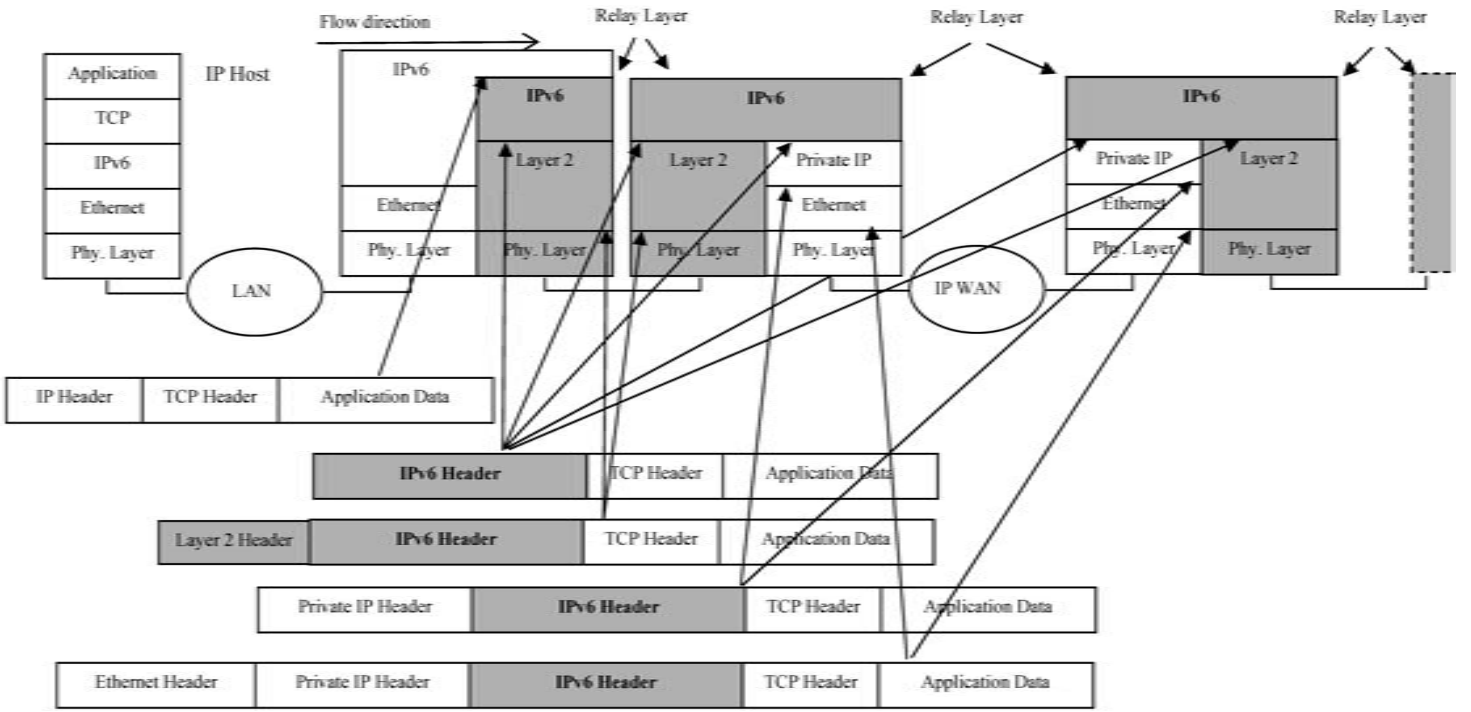


Figure 6.34 IPv6-centric QoS-PRN connecting an IP host to a private-IPv4 network portion: Data flow

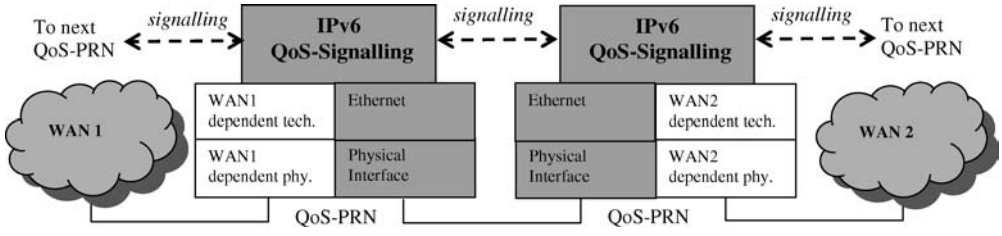


Figure 6.35 IPv6-centric QoS-PRN end-to-end architecture

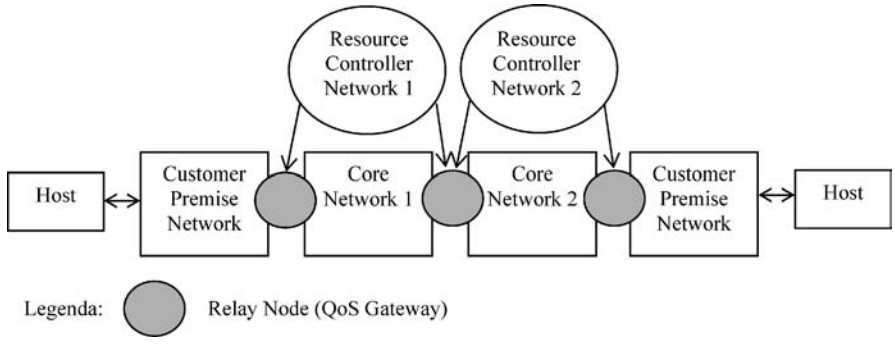


Figure 6.36 End-to-end QoS network

actually, they could also either distribute their functionalities through the network portion into specific blocks or concentrate their operative procedures within the QoS-PRNs/RNs (the QoS gateways). Once fixed the end-to-end QoS reference scheme, each single QoS-PRN/RN may be described as in Figure 6.37, partially derived from [ETSI-TS-102462] where the scheme is shown for a broadband satellite network. Figure 6.37 summarizes, in form of blocks, all the functions required by a QoS-PRN, as already detailed in Chapter 4 and at the beginning of this chapter. The figure is divided into Control and User Plane. In particular, the bold arrows represent the different traffic flows: data, signalling, routing information, and session and service management information. Examples of Session Management Information are represented by SIP [RFC3261, RFC4411 and RFC4412] and, at some extent, H.323 [ITU-T-H.323] protocols. The Session Initiation Protocol (SIP) is an application-layer control (signalling) protocol to create, modify and terminate sessions with one or more participants. The reference applications are Internet telephone calls, multimedia distribution and multimedia conferences. SIP protocol only concerns the application layer. SIP invitations are used to create sessions and carry session descriptions, which contain a set of compatible media types. It uses specific elements called "proxies" to help route requests, authenticate users for services, and implement provider call-routing policies. There can be a link between SIP and Resource Reservation. Reference [RFC3312] defines preconditions that simply require a participant to use existing resource reservation mechanisms before beginning the session. In the case of Figure 6.37, it means to activate resource reservation through

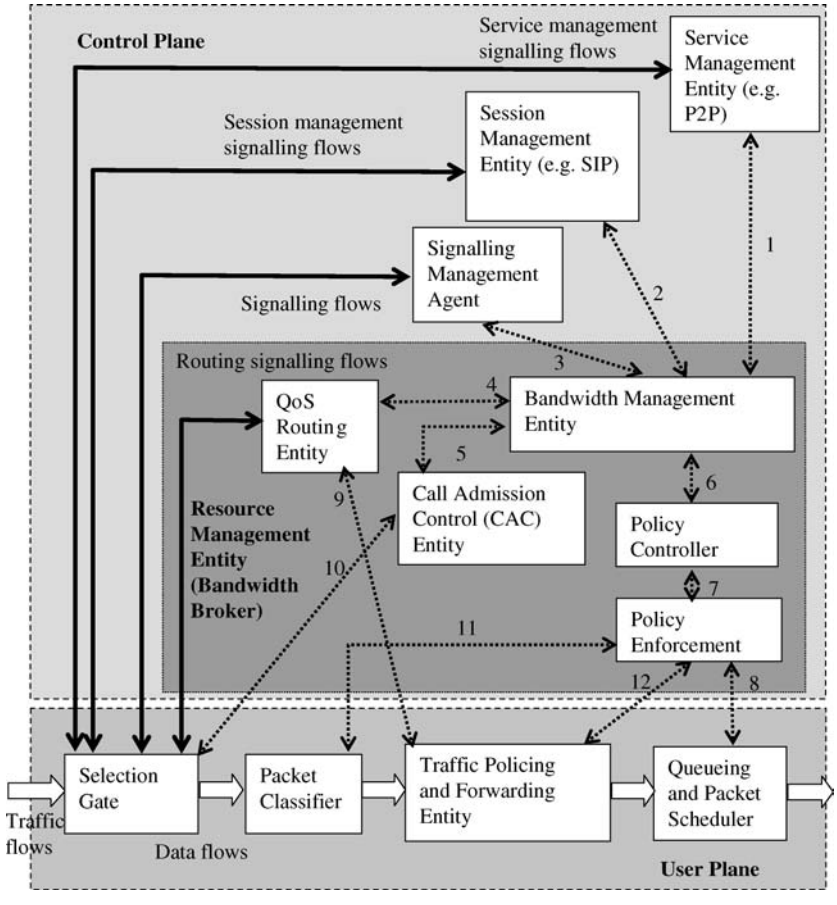


Figure 6.37 QoS Relay Node (QoS gateway) QoS and forwarding function

the Signalling Management Agent (SMA). More recently, [RFC4412] defines two new SIP header fields for communicating two new resource management messages: "Resource-Priority" and "Accept-Resource-Priority". The former can influence the behaviour of SIP user agents (such as telephone gateways and IP telephones) and SIP proxies but it is not a resource reservation signalling protocol. SIP can work without resource reservation. If QoS is necessary then SIP, as indicated in [RFC3312], needs to rely on resource reservation protocols.

Similar observations can be reported for H.323. Reference [ITU-T-H.323] covers the technical requirements for multimedia communications systems in those situations where the underlying transport is a packet-based network, which may not provide a guaranteed QoS. These networks may consist of a single network segment, or they may have complex topologies, which incorporate many network segments interconnected by other communications links. The components of a H.323 system are typically related to the application and include Terminals, Gateways, Gatekeepers, Multipoint Controllers, Multipoint Processors and Multipoint

Control Units. Reference [ITU-T-H.323] defines how the components communicate. H.323 terminals provide audio and optionally video and data communications capability in point-to-point or multipoint conferences. H.323 Gateways accomplish interworking with other terminals. Gatekeepers provide admission control and address translation services. Multipoint Controllers, Multipoint Processors and Multipoint Control Units provide support for multipoint conferences. In other words, H.323 specifies the use of Audio and Video Coders and of proper Call Signalling. Even if not explicitly contained within any recommendation, the relation between H.323 entities and resource reservation might be thought as in SIP case.

The addition of a Service Management Entity is defined by [ETSI-TS-102462] and it is related to other applications (neither SIP nor H.323 based). Also in this case, there is no explicit request (and need) of resource reservation. Actually, the Service Management and the Session Management Entity that appear in Figure 6.37 are independent of Resource Reservation. Obviously there can be a relation, represented by arrows marked by numbers 1 and 2, specifically defined below. The role of each single control action has been defined in Chapter 4. All control functions are conveyed within one single block called "Resource Management Entity". The SMA will transport resource reservation signalling through proper protocols, investigated in the next chapter. It communicates with the peer SMAs located in the neighbour RNs as shown by using the dashed line in Figures 6.35 (IPv6), 6.30 (MPLS), and 6.18 (IP). The mentioned dashed line is "implemented" through the bold double arrow line connecting the SMA with the Selection Gate. Actually the signalling data as well as the data of the other management entities flow through the real network component. Figure 6.37 is an extended and detailed version of Figure 6.6, where each control component is sketched within a function block. The functionalities of all the interfaces between the control modules used in Figure 6.37 are listed in Table 6.1 for the sake of completeness. The function of the Selection Gate may also be included in the Packet Classifier, as often done in the IP paradigm.

Table 6.1 Communication interfaces between control entities

Interface identifying number	Interface description
1	Service request
2	SIP service request
3	Resource reservation request/service level agreement negotiation
4	QoS routing table and related requests
5	CAC queries and answers
6	Bandwidth allocations
7	Policy decisions
8	Queuing and policy implementations
9	Forwarding table
10	Admission policy information
11	Classification policy
12	Traffic conditioning policy

It is important to say that the "User Plane" may represent both 1) and 2) in the following:

1) A single network component within the RN. In this case QoS Routing, CAC and Policy Enforcement Entities control the WAN-dependent technologies generically indicated in Figure 5.14 and applied in the rest of this chapter (e.g. ATM and IP).
2) An overall network portion implemented over homogeneous technology. In this case the control entities within the Resource Management Entity communicate with Control Entities linked to a control "Network Portion Resource Manager" through an Internal Resource Signalling (IRS). The overall architecture is shown in Figure 6.38. It contains the details of what is shown in Figure 6.36. Each control block is referred to a specific network portion. Also in this local case, the figure may be representative of the entire node chain between two RNs but it may also be referred to a single network node where the control functions are decentralized. The real flow of the IRS depends on the used protocol and on the real implementation. For example, the two entities that really communicate may be the bandwidth controllers or, alternatively, if the local network controllers need to be of reduced complexity, the communicating entities may be the two Policy Enforcement blocks only. Anyway, virtually, all the entities shown in Figure 6.38, if present, exchange information between them. It is the meaning of the white horizontal arrows shown in the figure. The black vertical arrow that links the control modules with the user plane substitutes the numbered dashed arrows reported in Figure 6.37. After querying the local controller (or the local controllers) about the network conditions and after reserving the necessary resources between the two RNs, if possible, information about acceptance/rejection of the new call and about other control requests (e.g. bandwidth reservation) may be transmitted by the Resource Management Entity through the SMA (Figure 6.37) to the neighbouring RNs. An interface similar to Table 6.1 may also be defined locally to each network portion. If the local technology should be ATM, obviously all ATM signalling features [ATMSignalling] should

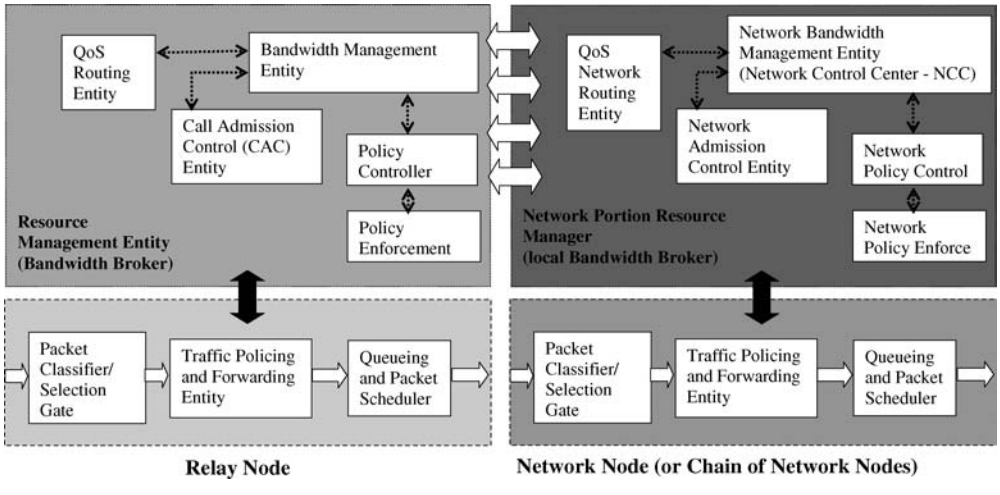


Figure 6.38 Relay node and related network portion controller

be implemented. The role of the local controllers and, in particular, of local bandwidth assignment is particularly clear in case of satellite communication where the transmission medium is shared among the users and each RN is a satellite earth station. Satellite networks deserve a special attention. Chapter 10 will be dedicated to QoS-supporting satellite architectures, including resource reservations, which are detailed in Chapter 11.

6.8 QoS Architectures Comparison

6.8.1 Comparison of the Features

Before detailing the possible signalling protocols that can be applied to the QoS architectures presented in this chapter, the presentation of a first comparison among them may be useful to better understand the different features. As already said in the specific sections, not all QoS architectures have the same level of technological consolidation. For some of them there are already commercial products in the market. Full-MPLS and IPv6 solutions are presented for future evolution but the comparison is performed having in mind all the features that a specific solution can potentially provide, also in the next and far future.

Table 6.2 reports a comparison between IntServ IP, DiffServ IP, MPLS and IPv6 solutions at the QoS-RN (-PRN). DiffServ IP provides four alternatives: No control (where only a basic priority mechanism is applied), Static trunks, DiffServ-PCN, and BB, not available in the market for now but a great potential for QoS management. MPLS provides two possibilities: MPLS-integrated, commercially available, to be used as support to IP QoS-RNs (-PRNs) for hard guarantee QoS, and Full-MPLS, not available in the market, but supportable with relative effort and very interesting for future extensions. IPv6 offers two solutions: CSF6N and F6SN.

Emphasis is put on traffic management, bandwidth optimization, CAC, MLPP, QoS signalling protocols and network planning. When no distinction is made among the alternatives in Table 6.2, it means that the feature is valid for all of them. Even if the comparison may support network operators to choose the most suitable technology for their specific needs, it is more oriented for possible future scenarios. The level of consolidation may be checked by considering standards, scientific literature and device availability in the market. For instance, it is remarkable that MPLS intra-domain QoS management is a consolidated technology, while MPLS inter-domain QoS management is currently under study in standardization bodies and scientific communities.

To summarize the results, it must be noted that the Full-MPLS and IPv6 solution are flexible and powerful for resource management. The presence of QoS signalling protocols, like RSVP-TE, is essential to optimize resource allocation and to support timely fault countermeasures. Indications about them are also reported in Table 6.2. They will be detailed in the next chapter. It is remarkable that even using the MPLS and IPv6 solutions, the presence of loose guarantee QoS within the private network portions can destroy the support of hard guarantee QoS. Mapping the QoS requirements over the different private technology is topical for QoS management. A specific chapter will be dedicated to it. Fault tolerance is not guaranteed if the RN of a given network portion is not able to trigger re-routing decisions within a required time interval, as, for example, if the local technology does not support any internal signalling. The same concept applies if CAC and MLPP are considered.

Table 6.2 QoS architectures comparison.

Technological features	DiffServ IP QoS-PRN [No control, Static trunks, PCN DiffServ, BB] IntServ IP QoS-PRN	MPLS QoS-PRN [Integrated-MPLS, Full-MPLS]	IPv6 QoS-PRN [CSF6N, F6SN] (CSF6N is supposed to use an architecture with bandwidth broker)
	Traffic Management		
Additional headers to IP stack (IP host is supposed, otherwise additional header is always necessary as discussed in the text)	No (with the exceptions evidenced in the text)	Yes, 32 bits of MPLS header	Yes, the IPv6 header, if the host implements IPv4 No, as for IPv4 QoS-PRN, if the host implements IPv6
Label Switching	No Not implemented for IntServ	Yes but only within portions for Integrated MPLS Yes for Full-MPLS	No for CSF6N Yes for F6SN
Power of aggregation and scalability	Not required for No control Yes for all DiffServ solutions No for IntServ	Yes for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Yes for Full-MPLS	Yes
Concurrency in accessing resources	No for No control No for Static trunks Yes for PCN DiffServ Yes for BB Yes for IntServ	Yes for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Yes for Full-MPLS	Yes
SLS/SLA classification	Limited Single flow distinction for IntServ	Limited for MPLS Integrated Huge for Full MPLS (up to single flow distinction, if required)	Huge for CSF6N Huge for Full F6SN (up to single flow distinction, if required)
Dynamic Bandwidth Management	No for No control No for Static trunks No for PCN DiffServ Yes for BB Yes for IntServ	Yes for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Yes for Full-MPLS	Yes
Resource assignment procedure	No assignment for No control Manual for Static trunks Manual/Automatic for PCN DiffServ	Automatic for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN	Automatic

Technological features	DiffServ IP QoS-PRN [No control, Static trunks, PCN DiffServ, BB] IntServ IP QoS-PRN	MPLS QoS-PRN [Integrated-MPLS, Full-MPLS]	IPv6 QoS-PRN [CSF6N, F6SN] (CSF6N is supposed to use an architecture with bandwidth broker)
	Automatic for BB Automatic for IntServ	Automatic for Full-MPLS	
	Bandwidth Optimization		
Probing	Useless for No control Useless for Static trunks Necessary for PCN DiffServ Not used for BB Not used for IntServ	Not used for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Not used for Full-MPLS	Not used
Bandwidth allocation scheme	Not implemented for No control Over-provision for Static trunks Static for PCN DiffServ Dynamic for BB Dynamic for IntServ	Dynamic for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Dynamic for Full-MPLS	Dynamic
Resource allocation control	Not implemented for No control Planning level for Static trunks Planning level for PCN DiffServ Call level for BB Call level for IntServ	Call level for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Call level for Full-MPLS	Call level
Bandwidth wasting when aggregating heterogeneous SLAs	Not considered for No control Large waste for Static trunks Medium waste for PCN DiffServ Medium waste for BB No waste for IntServ	Medium waste for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN No waste for Full-MPLS	No waste
	Call Admission Control (CAC)		
CAC	No for No control No for Static trunks Yes for PCN DiffServ Yes for BB Yes for IntServ	Yes for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Yes for Full-MPLS	Yes

Table 6.2 (*Continued*)

Technological features	DiffServ IP QoS-PRN [No control, Static trunks, PCN DiffServ, BB] IntServ IP QoS-PRN	MPLS QoS-PRN [Integrated-MPLS, Full-MPLS]	IPv6 QoS-PRN [CSF6N, F6SN] (CSF6N is supposed to use an architecture with bandwidth broker)
Precision of bandwidth computation during CAC	Not applicable for No control Not applicable for Static trunks Limited for PCN DiffServ High for BB Very high for IntServ (up to single flow precision)	Very high for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Very high for Full MPLS (up to single flow precision, if required)	Very high for CSF6N Very high for F6SN (up to single flow distinction, if required)
Preemption during CAC	Not applicable for No control Not applicable for Static trunks Implementable for PCN DiffServ Implementable for BB Implementable for IntServ	Implementable for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Implementable for Full-MPLS	Implementable
	Multi Level Precedence and Pre-emption (MLPP)		
MLPP	Not applicable for No control Not applicable for Static trunks Applicable for PCN DiffServ Applicable for BB Simply Applicable for IntServ	Applicable for Integrated-MPLS Simply applicable for Full-MPLS	Applicable for CSF6N Simply applicable for F6SN
Precision of bandwidth computation during MLPP	Not applicable for No control Not applicable for Static trunks Limited for PCN DiffServ High for BB Very high for IntServ (up to single user flow precision)	Very high for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Very high for Full MPLS (up to single user flow precision, if required)	Very high for CSF6N Very high for F6SN (up to single user flow distinction, if required)

(*continued overleaf*)

Technological features	DiffServ IP QoS-PRN [No control, Static trunks, PCN DiffServ, BB] IntServ IP QoS-PRN	MPLS QoS-PRN [Integrated-MPLS, Full-MPLS]	IPv6 QoS-PRN [CSF6N, F6SN] (CSF6N is supposed to use an architecture with bandwidth broker)
Time interval before pre-emption completed	Not applicable for No control Not applicable for Static trunks Not controlled for PCN DiffServ Controlled for BB Much controlled for IntServ	Much controlled for Integrated-MPLS within each portion; concerning the overall network see IPv4 QoS-PRN Much controlled for Full MPLS	Controlled for CSF6N Much controlled for F6SN
	Signalling Protocols		
Inter-domain protocols	No signalling for No control qBGP for Static trunks (standardized) qBGP/modified RSVP for PCN (standardized) NSIS/Dedicated QoS Protocol for BB (not standardized)	See IPv4 QoS-PRN for Integrated-MPLS qBGP/RSVP-TE for Full-MPLS (not standardized)	NSIS/Dedicated QoS Protocol (not standardized)
	Network Planning		
Necessity of a priori over-provision	Yes (if a minimum of quality is required) for No control Yes for Static trunks Partial for PCN DiffServ Partial for BB No for IntServ	Partial for MPLS Integrated No for Full MPLS	Partial for CFF6N No for F6SN
Planning phase development	Not required/trivial for No control Simple for Static trunks Average difficulty for PCN DiffServ Difficult for BB Difficult for IntServ	Difficult	Difficult

Obviously there is no "best" solution, because the quality of a solution strictly depends on the application needs. For instance, if over-provision may be applied because bandwidth is not a scarce resource, also the IPv4-centric solution based on the No control DiffServ paradigm may reveal more than sufficient, even if no control algorithm is applied. This condition may be satisfied in specific wired environments, but it is hardly applicable for terrestrial wireless and satellite links. If there are no signalling schemes to manage resources dynamically, the SLS support is left to the experience of network operators at network planning level.

6.8.2 SLS Separation versus Aggregation

To conclude the comparison among the QoS architectures, it is important to focus on one of the features: the possibility to distinguish many SLSs. The importance of flow identification has been highlighted in Chapter 4 but, here, it is important to specify which can be the impact on bandwidth allocation at QoS-PRN (-RN) level and, for extension, over the entire end-to-end QoS network.

As said in Table 6.2 one of the advantages of the full MPLS and of the IPv6 solutions is the availability of a large set of SLSs, which can be defined. The possibility to differentiate many traffic flows is not only an advantage for QoS management because it increases the flexibility of the QoS offer and the adaptation power to customer needs, but it may be used to optimize bandwidth management. The topic is explicitly mentioned in Table 6.2 under the term "Bandwidth wasting when aggregating heterogeneous SLSs". It is privileged here with respect to other aspects of QoS management, listed and compared in Table 6.2, because many studies confirm the efficiency of aggregating homogeneous traffic, but the performance of non-homogeneous trunks is still an interesting scientific open issue and can be stimulating for the readers who wish to work in this field.

The power of differentiating a great number of SLSs is distinctive for Full MPLS and IPv6 QoS-PRNs (and -RNs). The performance investigation reported here may be seen as a deeper comparison between the mentioned options and the other solutions. It is a right but incomplete viewpoint in the intention of the author. Actually, non-homogeneous traffic aggregation is one of the most meaningful issues for QoS networking with great implications also on pricing schemes. The author's attempt here is to raise the attention about it, well beyond the comparison of two technological alternatives.

Two traffic types are considered for the performance evaluation: VoIP and video. VoIP SLA considers the on–off source model suggested in the ITU P.59 recommendation. As far as the video service is concerned, real traces have been taken from www.tkn.ee.tu-berlin.de/research/trace/trace.html. The QoS constraint, the SLS here, is the Packet Loss Probability (P_{loss}). Up to 8 different SLSs are considered for each traffic category by changing P_{loss} within the range $\left[10^{-2} \text{ to } 10^{-9}\right]$. The tests are performed supposing that two SLSs need to be aggregated. Different aggregations of VoIP and video SLSs are considered. The rate assigned to them is dimensioned in order to support the most stringent SLS, thus potentially introducing bandwidth waste. Differently from the results reported in Chapter 4 (totally simulative), the results shown here are obtained through the equivalent bandwidth approach presented in section 4.1.3, taken from [Guerin91]. The choice also

allows computing very low packet loss values. The assumption here is to have bufferless conditions. An ad hoc simulator in C++ has been used to get the results. The width of the confidence interval over the performance measures is less than 1% for the 95% of the cases.

The bandwidth required to support performance requirements when the traffic of the two classes is aggregated together is compared with the overall bandwidth necessary to guarantee QoS if the two traffics are kept separate. The amount of required additional bandwidth to maintain the same service level is shown in Figures 6.39 and 6.40. The former is computed for an overall number of connections set to 50; the latter for an overall number of connections set to 150. The VoIP SLS is fixed to $P_{\text{loss}} \leq 10^{-2}$. Video requirement is varied from SLS with $P_{\text{loss}} \leq 10^{-2}$ to SLS with $P_{\text{loss}} \leq 10^{-9}$.

Due to the multiplexing gain in presence of bursty sources, below a threshold given by the intersection point of each curve with the X-axis (no additional bandwidth required), aggregating is always convenient (the gain is negative), despite QoS heterogeneity. Above the threshold, a portion of bandwidth is wasted if the traffic classes are not kept separated. Such a threshold is the equilibrium point where traffic aggregation and separation is indifferent for bandwidth allocation.

The qualitative and quantitative information may be used by network managers but something more general can be stated through a closer insight of the results. Figures 6.39

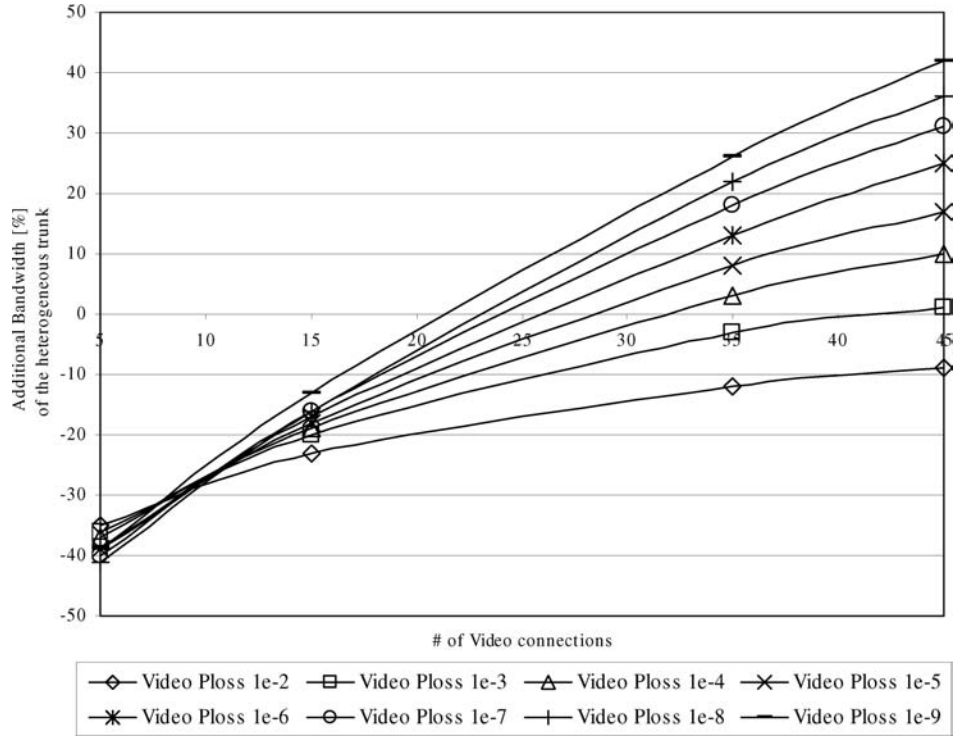


Figure 6.39 Aggregation, VoIP and video traffic and 50 overall connections

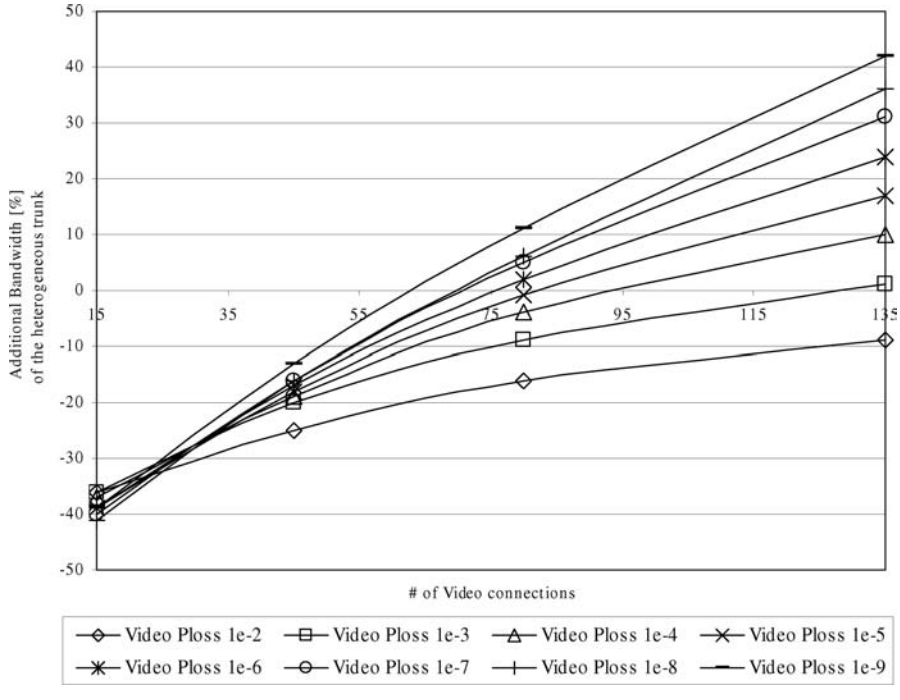


Figure 6.40 Aggregation, VoIP and video traffic and 150 overall connections

and 6.40 show the same trend. It means that the position of equilibrium points is almost invariant if the number of total connections is scaled up, but depends on the percentage of traffic type. For instance, the curve of "Video Ploss 1e-3" meets the *X*-axis at the point "42 video connections" in Figure 6.39 and at the point "130 video connections" in Figure 6.40. Both cases correspond to the 85% of video connections in the aggregated trunk. This information constitutes a powerful and simple tool for planning the aggregation of SLSs. For each Ploss curve, the quantity of additional bandwidth to be provided is a function of the traffic percentage in the aggregation. The analysis is also validated by other simulation's results, not reported here, by taking fixed the video instead of VoIP SLS. A first idea about the indifference of the equilibrium point to the number of flows could have also been got from Figure 4.16, even if the numerical values are not the same being the composition of traffic totally different, as well as the buffer values and the simulative scenario. The application of other traffic categories and other levels of abstractions would impact on the slope of the bandwidth curves reported. The equilibrium point precise behaviour is left to specific research studies. The aim of this section is to attract the attention towards the importance of traffic aggregation for inter-domain and intra-domain QoS management. Aggregation implies bandwidth requirement variations ranging from −40 to 40% in the example of Figures 6.39 and 6.40. It is immediate to think of the economic effect if the bandwidth assignment is linked to a pricing mechanism.

7

Signalling over QoS Architectures

7.1 Introduction

The aim of signalling protocols is to carry information through the networks. Even if the word "signalling" may have a broader meaning including many aspects related to network management and control (e.g. routing algorithms), it is related, in this book, to QoS signalling. In this case, the interest is to carry information about the requested SLS through Relay Layers. In practice, it is necessary to have a common format flowing through Relay Layers (i.e. through the overall meta-network) and carrying information about the flows entering the network. The signalling protocol should work together with CAC and other controls needed at the connection set-up but it does not implement any algorithm. Its role is to provide a common format and a common language, which can be understood by all relay layers, and to collect the acceptance/rejection by single WANs along the end-to-end paths so as to carry the information to the neighbouring WANs during the backward direction. The management of the bandwidth within each single WAN is left to the WAN itself. If a WAN implements a QoS-providing technology, it will make use of a private signalling protocol. The specific choice is outside the scope of this book but important indications may be derived by the discussion reported in this chapter. It is often dependent on the technology provided to interconnect heterogeneous networks: IPv4 (IntServ or DiffServ)-centric, MPLS-centric and IPv6-centric. Signalling protocols are often developed just for single solutions. The important thing here is to have a proper interface between the Relay Layer (and its signalling protocol) and the WAN Control Agents shown in the QoS Architectures in Figures 6.6, 6.37 and 6.38.

The signalling protocol transports QoS SLS for flows up to QoS-PRNs by using one of the protocol architectures presented. It can use both in-band and off-band channels. QoS-PRNs map the SLS over a bandwidth request for private WANs so getting a "bandwidth pipe" of proper dimension to guarantee SLS guarantees up to next QoS-PRNs. Within this operation the "bandwidth pipe" could be not available. The check is performed locally within each single private WAN querying a database constantly updated about the WAN resource state (actually, a Control Agent; a Bandwidth Broker (BB), as in [Jia01], or a Quality Network

QoS over Heterogeneous Networks Mario Marchese

Server, as in [Fineberg02]). If no resource is available, a CAC, as in [Jia01], rejects the connection. BBs cannot communicate with each other but can be locally implemented to the WAN. Also, the communication protocol between QoS-PRN and private BB may be implemented by using a proprietary protocol as well as the format of each single BB, as said above. The advantage is that private WANs need to have, in common, only the definition of the QoS (SLS) and its comprehension.

The definition of the bandwidth pipe that supports the requested QoS depends on the choice performed to implement the Relay Layers. This is the key of QoS and one of the topical points of this book. A full chapter has been dedicated to highlight the performance difference among the possible choices. An important problem concerns QoS routing. If it is managed, as commonly done in MPLS networks, among QoS-PRNs, the tunnels connecting the WANs are crossed along opaque network portions. Hence, it is difficult to assure an end-to-end delay (which strongly depends on the number of nodes of the chosen routing path) if no a priori knowledge is available about the entire network topology. Following the suggestions reported in [RFC2746], such a drawback can be solved by configuring proper tunnels in each backbone WAN, which could carry traffic destined to other WANs. The tunnels are aimed at carrying the traffic dedicated to one (or more) destination hosts placed in a different WAN, with fixed delay and loss. The vertical QoS mapping problem investigated in detail in the following can help find ideas to solve this problem.

Concerning the transport of QoS SLS, based on MESCAL and other analysed architectures, it is possible to establish a generic QoS framework to understand signalling action better. Figure 6.1 already suggests decoupling service and transport domain. Also [ETSI-TS-102462] suggests going forward a clear separation between functions implemented to guarantee services and functions to transport information flows. Open interfaces between Service and Information Transport function groups will be designed in such a way to allow full interworking, also allowing independence between the two function groups. Actually, future heterogeneous network should allow providing new services independently of network and access technology. Figure 7.1, which generalizes MESCAL proposal also including the physical location of the control and data encapsulation functionalities of QoS-PRNs, should help understand the basic concepts partially taken from [ETSI-TS-102462] and [Sarangan2006]. The latter contains a comparative study for dynamic service negotiation in the next-generation Internet. It illustrates the definition of SLA and SLS, proposes a generic framework to describe existing service negotiation protocols and lists the characteristics of a generic service negotiation protocol. It will be taken as reference for signalling description in this book.

The Service Functions are aimed at managing group of functionalities to transmit and negotiate SLS and to take needed actions to assure the service guarantees. Referring to Figure 6.6, it means "Management and Control Plane" composed of Admission Control Agent, SLA Agent, Configuration Agent and Resource Control Agent. Service Functions block may be further decomposed into two entities [Sarangan2006], negotiation manager (NM) and resource manager (RM), but the functionalities should be clear now. In practice, following the approach introduced in this book, the dashed line (the "Horizontal QoS Signalling") assures SLS resource reservation requests transmission between different network portions. It carries out the signalling functions of the horizontal QoS mapping (Chapter 5, section 5.3). The vertical arrow (the "Vertical QoS Signalling") matches the signalling functionalities of the vertical QoS mapping, i.e. within the same network portion (as in Chapter 5, section 5.2). A complete list of the building blocks acting over QoS Architectures sketched in Figure 7.1

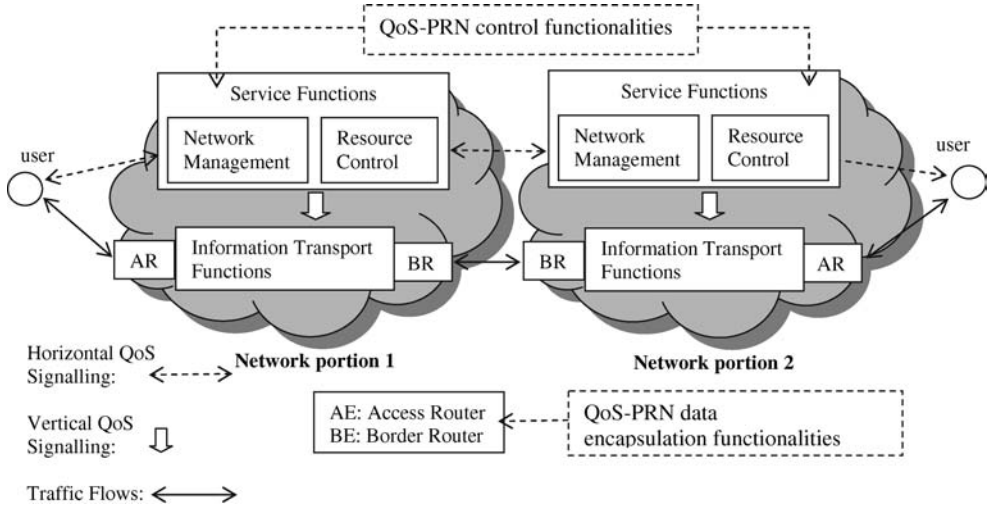


Figure 7.1 QoS Architecture: Service and Information Transport Functional groups

is proposed by [MESCAL] and reported at the beginning of Chapter 6. A more generic classification is reported in [ETSI-TS-102462] and summarized below. Again, the decomposition between User, Control and Management plane is used. Each user/customer asks for service subscription, service invocation and data transmission. The network "answers" through the following:

- User Plane
  - Traffic Classification
  - Packet Marking
  - Traffic Shaping/Policing
  - Flow Control
  - Queuing
  - Scheduling
- Control Plane
  - Service Invocation
  - Resource Reservation
  - Call Admission Control
  - QoS Routing
- Management Plane
  - Service Subscription
  - Service Level Agreements
  - Billing
  - Policy

User and Control Plane functions have already been reported and explained in Chapter 4. SLA has been widely investigated in Chapter 1. The word "Policy" means, in this context, the set of functions to administer, manage and control the access to network resources.

Reference [Sarangan2006] traces some basic features of a service negotiation signalling protocol (in practice the Horizontal QoS Signalling).

- *Negotiation capability* – Signalling should make the customer able to specify and request a service and also to accept and reject possible renegotiations; the provider to notify acceptance and rejection of the request and to manage possible negotiations with the customer. The provider should also be able to modify a service already accepted by the customer, if necessary.
- *QoS architecture compatibility* – Signalling should be designed also considering the composition of the QoS architectures, in particular, if they are standardized.
- *Physical layers compatibility* – Signalling should be able to go through the physical components of the end-to-end network.
- *Reduced action* – Signalling protocols should have limited functionalities to avoid wasting network resources for the only use of it.
- *Transparency to SLS* – Signalling protocols should not be tuned on specific SLS formats but they should also be able to manage possible modifications and extensions. Besides, there is no need of designing all protocols again.

Some existing protocols are already suited for QoS management needs and can be simply extended.

There are many signalling protocols in the literature. The aim of this book is to present some possible example solutions applied to the QoS architectures presented in the previous chapter. Not necessarily, a protocol solution studied and designed for a QoS architecture cannot be applied to another (or to a new) architecture. A possible way to present the issue is to identify a signalling protocol for each technology-centric architecture presented in Chapter 6. An alternative is to present the different signalling protocols leaving the readers to apply them to the different solutions. The way followed in this book is in the middle: different signalling protocols are presented. Comments to evidence possible applications to technology-centric QoS architectures are added directly in each description. A final summary paragraph focusing on the signalling solutions for each QoS architecture is also added at the end of this chapter.

7.2 RSVP QoS Signalling

The first considered QoS signalling solution is RSVP [RFC2205]. It receives a great deal of attention in this book because it is considered the basic protocol from which extensions can be developed. It has been designed to be used by a host to request specific QoS services from a network. Service is requested for a specific flow or group of flows. It is used also by routers to deliver (QoS) requests to all nodes along the path(s) and to create and maintain state resource reservations so to match the host request for specific flows. RSVP allows activating mechanisms to reserve resources. Actually, as any other signalling protocol, it does not reserve any resource but provides the proper container to transport requests. It works in just one direction. If it is necessary to activate reservations in both directions, it is necessary to

activate two RSVP instances. It has been thought as operating over IP (like Internet control protocols) even if the essential concepts might be applied also over other network types. In other words, its main application is for IP-centric QoS architectures and, in particular, within the IntServ paradigm (Figure 6.11 applied together with IntServ), as should be clear in the following, but essential features may also be extended to other frameworks.

7.2.1 RSVP Architecture

Figure 7.2 [RFC2205] reports the QoS architecture on which RSVP is based. It is fully in conformity with the model used up to now and enforces the explained concepts: the link of Figure 7.1 and Figure 7.2 is immediate. Figure 7.2 allows also doing a step forward for a full definition of a QoS Architecture, which will be explicitly reported in the following sub-chapters. Also, a host may be considered as an RSVP entity. In this case, packets are generated directly by an Application entity. Packets enter the packet classifier. Network Management (Figure 7.1) manages QoS requests. They are passed to the RSVP management process, which forwards the requests along the end-to-end path. The mechanisms to assure QoS have been detailed in Chapter 4 and include packet classifier, CAC and packet scheduler. The packet classifier determines the QoS of each single packet. Obviously, for each outgoing link, the scheduler action needs to be associated with technology-dependent link-layer mechanism to achieve the promised QoS. This is part of "Vertical QoS Mapping", introduced in Chapter 5. Details about possible solutions will be given in Chapters 8 and 9. The definition of traffic class is linked to the chosen SLS even if RSVP has been thought of as a solution within the IntServ environment (Chapter 3). It means that RSVP sets a session for a specific data flow, which is defined for a particular destination and transport-layer protocol. In more detail, the following triple is used to set a session (DestAddress, ProtocolId and DstPort) [RFC2205]. The DestAddress is the IP address of the destination (unicast or multicast); the ProtocolId is

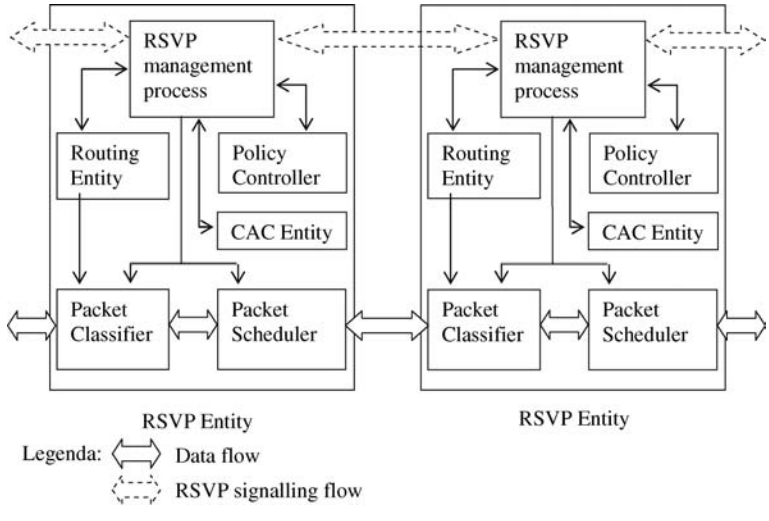


Figure 7.2 RSVP-based QoS architecture

the IP protocol ID (RSVP has been thought for both IPv4 and IPv6). The DstPort (optional) is the UDP/TCP destination port field but, actually, as explicitly said in [RFC2205], it may be considered as a "generalized destination port", i.e. either as a demultiplexing field within an alternative transport protocol, or as an application-dependent field acting at the application layer. It is important to stress that session definition does not require the mandatory presence of DstPort and that it is essential to distinguish the flows addressed to a single IP interface.

The CAC entity checks if the RSVP entity (i.e. the network node) has sufficient available resources to guarantee the SLS. The Policy control entity decides if the user has administrative permission to make the reservation. If both controls provide a successful result, reservations are performed within packet classifier and scheduler so as to assure the requested QoS to the proper data packets (defined by the "filter spec" explained below), otherwise the reservation is rejected and RSVP reports an error message to the receiver generating the request. The precise mechanisms to satisfy QoS requests depend on the link-layer technology, as said above. Reservations (actually bandwidth reservations) performed by RSVP are not permanent for the overall duration of the communications (as done in telephone networks), but they need to be periodically renewed. For this motivation, RSVP introduces the concept of "soft state" allocation. In practice, it means that if the reservation is not renewed through refresh messages that maintain the set states along the path, the state (i.e. the reservation itself) automatically decays and it is deleted.

7.2.2 RSVP Objects

Elements of the RSVP control messages are called "objects", which are identified by means of "classes", further subdivided into class types (C-type) [RFC2205]. For example, as said above, three parameters are used to set a session. In more detail, the SESSION class (whose identifier is number 1) is used, subdivided into two different C-types: "IPv4/UDP SESSION object: Class = 1, C-Type = 1" and "IPv6/UDP SESSION object: Class = 1, C-Type = 2", shown in Figures 7.3 and 7.4, respectively. The Destination Address is the IP unicast or multicast address of the destination and it is a mandatory non-zero field. The Protocol IP is the Protocol Identifier. Also, this field is mandatory and it must be non-zero. The field Flags is set to "00000001" in binary code and it is called "E_Police flag". It is used to determine the effective "edge" of the network, to control traffic policing. Its real use is outside the scope of the book and it is left to specific literature. The DstPort is the UDP/TCP destination port for the session. The field is not mandatory and its value may be set to zero to mean "none". DstPort may also assume a different meaning if set to alternative values (e.g. parameters at the application layer).

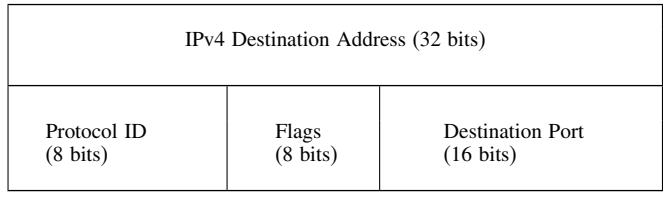


Figure 7.3 RSVP IPv4/UDP SESSION object: Class = 1, C-Type = 1

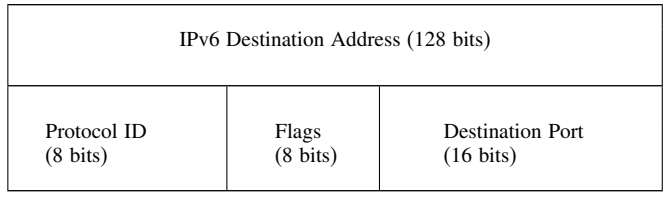


Figure 7.4 IPv6/UDP SESSION object: Class = 1, C-Type = 2

Basic RSVP reservation request consists of a "flowspec" and of a "filter spec", transported through Resv messages. Together, they are called "flow descriptor". The flowspec defines the QoS and represents a sort of SLS. It is connected to resource reservation and acts on the Packet Scheduler and CAC, taking Figure 7.2 as reference architecture. A simple example of flowspec may be represented by the amount of requested bandwidth in terms of multiples of a basic resource quantity (called B, in [RFC2205]). It contains information generated by each receiver describing the desired QoS control service, a description of the traffic flow to which the resource reservation should apply (called "Receiver Tspec"), and the parameters required to invoke the service (called "Receiver Rspec"). Possible compositions of Receiver Tspec and Rspec will be reported in the following. Information carried in flowspecs may change at intermediate nodes in the destination–source path due to reservation mechanisms as well as merging, explained in the remaining sections of this chapter. The filter spec, together with a session specification, defines the flow or group of flows, which will receive the QoS contained in the related flowspec. In practice, it represents the flow identifier and acts on the Packet Classifier in the architecture of Figure 7.2. Flows or group of flows addressed to particular session, which do not match any filter spec, are considered as best-effort traffic.

Filter specs are coded through the class "FILTER_SPEC", identified with number 10. There are three C-Types:

1. IPv4 FILTER_SPEC object: Class = 10, C-Type = 1 (Figure 7.5)
2. IPv6 FILTER_SPEC object: Class = 10, C-Type = 2 (Figure 7.6)
3. IPv6 Flow-label FILTER_SPEC object: Class = 10, C-Type = 3 (Figure 7.7)

The Source Address is the IP source address for the sender host. It is a mandatory non-zero field. The source port is the UDP/TCP source port for a sender. It can also be set to zero. The Flow Label is the 24-bit Flow Label, defined in IPv6, to identify the packets belonging to a specific flow. It has been extensively explained in Chapter 3 and it is the key tool (the switching label) to implement the F6SN, introduced in Chapter 3. This feature of RSVP

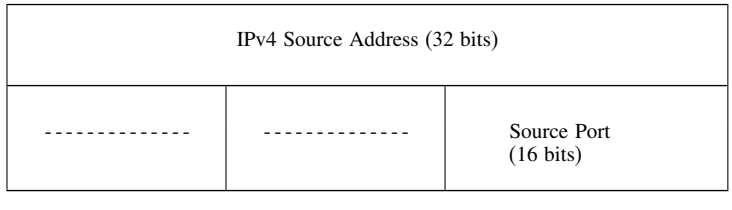


Figure 7.5 IPv4 FILTER_SPEC object: Class = 10, C-Type = 1

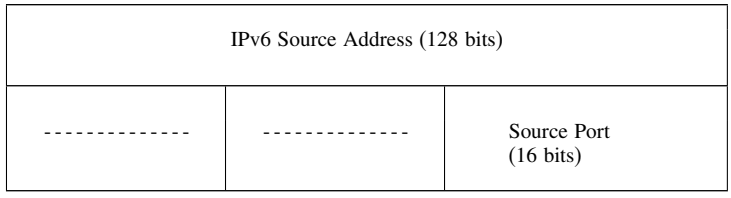


Figure 7.6 IPv6 FILTER_SPEC object: Class = 10, C-Type = 2

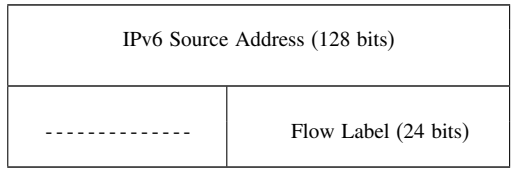


Figure 7.7 IPv6 Flow-label FILTER_SPEC object: Class = 10, C-Type = 3

FILTER_SPEC makes RSVP a feasible signalling protocol to implement the IPv6-centric QoS approach, introduced in this chapter. In practice, IPv6 Flow Label FILTER_SPEC object provides the technological tool to transport QoS requirements and information for a specific flow, identified by IPv6 Flow Label.

RSVP Protocol mechanism is based on two messages: Resv (reservation request) and Path. Each receiver sends (Resv) messages towards the senders. Resv messages follow exactly the reverse of the path used by the data packets sent by the senders and generate reservations within each node along the path from the destination to the source, where Resv messages are finally delivered. After receiving the Resv messages, each single source can configure appropriate control parameters.

The senders (information sources) send the Path messages towards the destination. They follow a path decided by the routing protocols that is not part of RSVP and store a "path state" within each node of the path. The path state must include at least the IP address of the previous hop. The Resv messages will use it to reach the sender in the reverse direction. Previous hop information within Path is transported through PHOP messages. Reversely, next hop information within Resv messages is transported through NHOP objects. They both use RSVP_HOP class, identified through Class = 3. IPv4 RSVP_HOP object (C-Type = 1, shown in Figure 7.8) is distinguished from IPv6 RSVP_HOP (C-Type = 2, shown in Figure 7.9). RSVP_HOP contains the IP address of the last RSVP-knowledgeable interface that forwarded the message and the Logical Interface Handle (LIH), which allows specifying logical outgoing interfaces. They are used, for example, to solve the RSVP

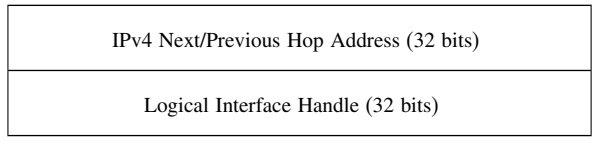


Figure 7.8 IPv4 RSVP_HOP object: Class = 3, C-Type = 1

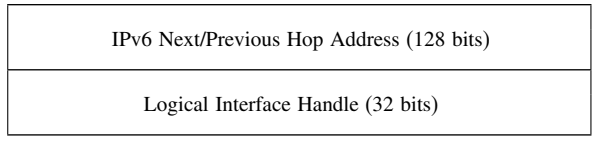


Figure 7.9 IPv6 RSVP_HOP object: Class = 3, C-Type = 2

messages routing problem in multicast tunnels. Multicast tunnels perform IP encapsulation of multicast packets for transmission through routers, without multicast capabilities, and appear as logical outgoing interfaces to be mapped over real physical interfaces. Routing protocols supporting tunnels describe a route by using a list of logical interfaces. RSVP packets find their way within multicast tunnels as follows: when node A forwards a Path message (i.e. from the source to the destination) through a logical outgoing interface, it includes an identifier of the logical interface itself in the PHOP object. It is called "LIH" (Figures 7.8 and 7.9). The next node (called B) in the path memorizes the LIH value in the path state. In the reverse direction, when node B sends the Resv message to A, it adds the LIH value taken from the path state in the NHOP object. When the NHOP object reaches node A, the resource reservation is transferred to the right logical interface through the LIH value. LIH field is created and interpreted by node A. Its value is opaque to node B.

The previous hop is the first information carried by the Path messages. In addition, it carries the Sender Template, the Sender Tspec and the Adspec, which are described in detail in the following.

The Sender Template describes the format of sender data packets. The related object is of great interest: [RFC2205] specifies the SENDER_TEMPLATE Class. It is identified as Class 11; it is used to select the packets of a specific sender from the others belonging to the same session over the same link. The definition, as expected, is the same as FILTER_SPEC. In short,

IPv4 SENDER_TEMPLATE object – Class = 11, C-Type = 1; same definition as IPv4 FILTER_SPEC object – Class = 10, C-Type = 1 (Figure 7.5);
IPv6 SENDER_TEMPLATE object – Class = 11, C-Type = 2; same definition as IPv6 FILTER_SPEC object – Class = 10, C-Type = 2 (Figure 7.6);
IPv6 Flow-label SENDER_TEMPLATE object – Class = 11, C-Type = 3; same definition as IPv6 Flow-label FILTER_SPEC object – Class = 10, C-Type = 3 (Figure 7.7).

Sender Tspec carries information about the sender traffic characteristics. It is used by traffic control to avoid over-reservations and unnecessary CAC failures at destination. Sender Tspec content is not modified by intermediate nodes along the source path.

Adspec carries "One Pass With Advertising" (OPWA) information [OPWA95]. OPWA control packets are sent from the source to the destination, following the data paths. They are aimed at gathering information that may be used to predict the end-to-end QoS, and they are used at the receivers to make reservation decisions. This information might include available services, delay and bandwidth estimates, and operating parameters used by specific QoS control services. When Adspec is received in a node, it is passed to the traffic controller that updates the OPWA information and composes an updated Adspec. If the traffic controller reveals that the node cannot implement a QoS control service contained in the Adspec, a flag

is set in the updated Adspec. The new Adspec is sent forward to the next node in the Path. The process repeats at each node and Adspec is finally delivered to the receiver. Final Adspec at the destination is a summary of QoS information about source–destination path. Adspec size should remain roughly constant as it passes each hop so to get good scaling properties.

In practice, Sender_Tspec and Adspec represent QoS specifications, together with Flowspec, whose simplest representation is composed of the pure bandwidth requirement. The used objects are introduced in [RFC2205] but they are specified by documents prepared by the IntServ working group [RFC2210]. Actually, they are not totally part of the signalling protocol but concern how the signalling protocol is used within a particular QoS paradigm: IntServ, for example, as in this case. QoS definition should be revised in case the paradigm is different but RSVP definitions may still be used.

For the sake of completeness, the following objects are defined to manage QoS:

FLOWSPEC Class = 9, C-Type = 1 is obsolete. The only valid definition is IntServ Flowspec object – Class = 9, C-Type = 2
SENDER_TSPEC Class = 12. IntServ SENDER_TSPEC object – Class = 12, C-Type = 2
ADSPEC Class = 13. IntServ ADSPEC object – Class = 13, C-Type = 2.

As said, all these objects are related to IntServ QoS paradigm and are defined in [RFC2210]. Specifications of Guaranteed Services are reported in [RFC2212], while specifications for Controlled-Load Services (CLS) are contained in [RFC2211]. The general characterization parameters are shown in [RFC2215]. If the SLS and the QoS model should be different, the mentioned definitions should be totally changed. Anyway, it is interesting to check how these objects are composed within the IntServ paradigm to guarantee QoS control services.

Without entering the details of the format, the essential information, for this book, contained in RSVP SENDER_TSPEC is shown in Figure 7.10. It is used for both Guaranteed and Controlled-Load Services, and reports part of the real SENDER_TSPEC class, which includes also message format version, overall length, service header and other parameters fully defined in the mentioned RFCs. Token Bucket Rate and Size define the generated traffic from the sender's viewpoint. Peak Data Rate may be either the sender's peak traffic generation rate (if it can be controlled) or the physical interface line rate (if it is known); otherwise, it may be set to a fixed value called "positive infinity". The Minimum Policed Unit

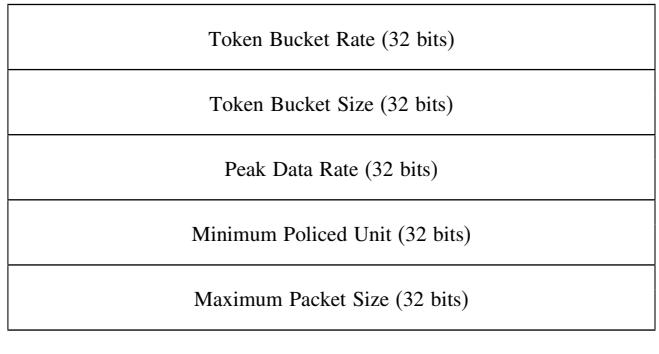


Figure 7.10 SENDER_TSPEC object: Class = 12, C-Type = 2 (selection of real parameters)

Parameter contains the size of the smallest packet generated by the application, including the application data and all protocol headers at or above the IP level (IP, TCP, UDP, RTP, etc.). Link-level headers are not included, because these headers will change as the packet crosses different portions of the network, as clear in Figure 7.1 and in the book portion dedicated to QoS architectures (e.g. Figures 6.11, 6.19 and 6.31). The Minimum Policed Unit Parameter is used by each node to compute the amount of bandwidth, which should be allocated to the flow. Value "0" is not allowed. The Maximum Packet Size Parameter is the size of the largest packet the application can generate and it is used for bandwidth allocation. At this point of the book, it is important to note that SENDER_TSPEC contains a description of the flow to which a bandwidth allocation may be associated. Even if totally polarized towards the IntServ paradigm, the parameters contained are part of the description parameters listed in Chapter 1, as SLS examples. Actually, a SENDER_TSPEC-like packet will always be necessary if a QoS signalling is implemented. Obviously, parameters can change.

Flowspec structure, carried together with FILTER_SPEC by Resv messages, differs if it is used for Controlled-Load Services and for Guaranteed Services. Concerning CL S, the parameter structure is very similar to the SENDER_TSPEC structure. Actually, Figure 7.10 may be used again for FLOWSPEC. Parameter meaning is obviously different because FLOWSPEC flows from the destination to the source and it practically contains reservations but the structure is the same. A Controlled-Load Flowspec parameter setting is fully reported in [RFC2210]. As far as Guaranteed Service is concerned, the Flowspec object is shown in Figure 7.11: the parameters "Token Bucket Rate", "Token Bucket Size", "Peak Data Rate", as well as "Minimum Policed Unit" and "Maximum Packet Size" are called "Tspec" (Receiver Tspec). They reflect the traffic parameters of the desired reservation. The terms "Rate" and "Slack" are aimed at getting bandwidth and delay guarantees and are called "Rspec" (Receiver Rspec). All parameters are specifically defined in [RFC2212] because they are linked to the reservation strategy implemented by IntServ. The Rate must be larger than or equal to the Token Bucket Rate; the Slack needs to be non-negative. Rate is measured in bytes per second and has the same range and representation as the Token Bucket Rate and the Peak Data Rate. Slack is expressed in microseconds. The Rspec rate R can be larger than the Tspec rate because it implies a reduction of the queuing delay.

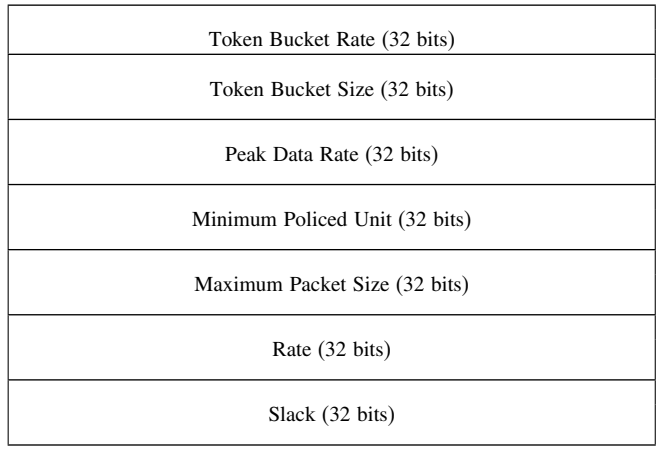


Figure 7.11 FLOWSPEC object: Class = 9, C-Type = 2 (selection of real parameters)

Without going into the reservation techniques details reported in [RFC2212], it is important to establish an order concerning Tspecs and Rspecs. There are two Tspecs: Tspec1 and Tspec2. Tspec1 is better or equal to Tspec2 (i.e. Tspec1 can substitute Tspec2) if the following five conditions are true at the same time:

1. Token Bucket Rate Tspec1 $\geq$ Token Bucket Rate Tspec2
2. Token Bucket Size Tspec1 $\geq$ Token Bucket Size Tspec2
3. Peak Data Rate Tspec1 $\geq$ Peak Data Rate Tspec2
4. Minimum Policed Unit Tspec1 $\leq$ Minimum Policed Unit Tspec2
5. Maximum Packet Size Tspec1 $\geq$ Maximum Packet Size Tspec2

On the other hand, Tspec1 is less or equal to Tspec2 if

1. Token Bucket Rate Tspec1 $\leq$ Token Bucket Rate Tspec2
2. Token Bucket Size Tspec1 $\leq$ Token Bucket Size Tspec2
3. Peak Data Rate Tspec1 $\leq$ Peak Data Rate Tspec2
4. Minimum Policed Unit Tspec1 $\geq$ Minimum Policed Unit Tspec2
5. Maximum Packet Size Tspec1 $\leq$ Maximum Packet Size Tspec2

Some operations need to be defined: two or more Tspecs can be "merged" together. A merged Tspec is computed by selecting the largest Token Bucket Rate, the largest Token Bucket Size, the largest Peak Data Rate, the smallest Minimum Policed Unit and the largest Maximum Packet Size. A number of Tspecs may be summed together so generating a "summed" Tspec by taking the sum of the Token Bucket Rates, the sum of the Token Bucket Sizes, the sum of the Peak Data Rates, the smallest Minimum Policed Unit and the Maximum Packet Size. Within a set of traffic flows, a Tspec sufficient to describe each flow in the set is defined as the "least common Tspec". It may be computed as the largest Token Bucket Rate; the largest Token Bucket Size; the largest Peak Data Rate; the smallest Minimum Policed Unit and the largest Maximum Packet Size.

Similar definitions may be applied to Rspecs. Rspec1 is better or equal to Rspec2 if

- Rate Rspec1 $\geq$ Rate Rspec2
- Slack Rspec1 $\leq$ Slack Rspec2

The other operations are defined as for Tspec. The mentioned definitions will be topical to understand RSVP reservation styles, briefly summarized in the following paragraphs.

The essential fields of ADSPEC format are shown in Figure 7.12. They compose the overall ADSPEC together with the version number of the message format and the message length not including header word. The Default General Parameters fragment is always present, while Guaranteed and Controlled-Load Service fragments are present only in case of Guaranteed and Controlled-Load Service transports, respectively.

The Default General Parameters fragment is reported in Figure 7.13. Actually, some flags as well as some length fields are also contained in the format, which are identified as Other parameters in Figure 7.13. The Default General Parameters fragment contains the parameters defined in [RFC2215]. The following details the parameters.

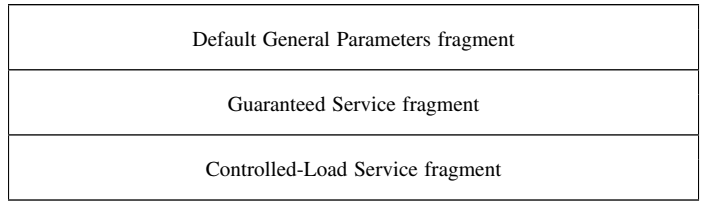


Figure 7.12 ADSPEC object: Class = 13, C-Type = 2 (selection of real parameters)

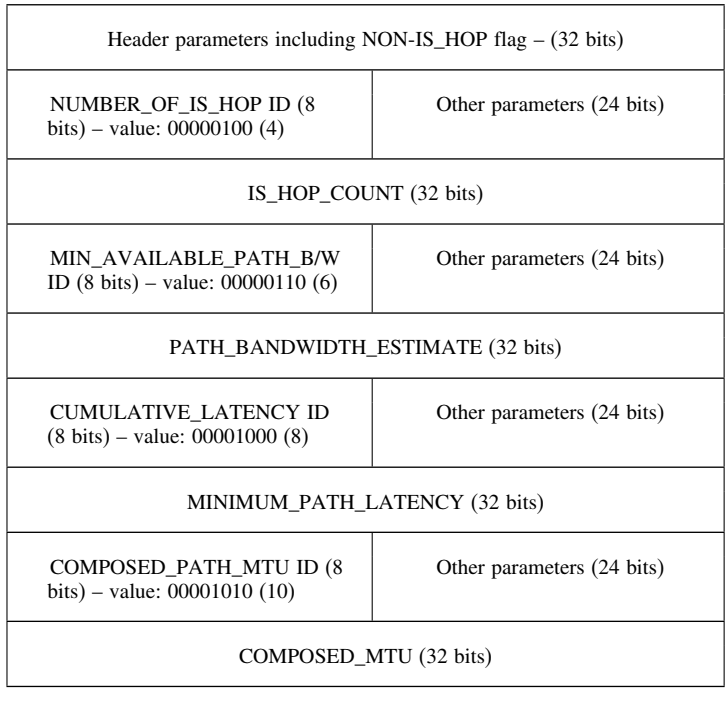


Figure 7.13 Default General Parameters fragment (selection of real parameters)

NON-IS_HOP flag (1 bit): It provides information about the presence of network elements that do not implement IntServ QoS control services along the end-to-end data path. Actually "IS" stands for Integrated Services. It is also called "Global Break Bit". If at least one network element along the path does not support QoS control, it is set to 1, otherwise it is set to 0. If applied to the heterogeneous network chain, whose QoS interworking is guaranteed through IntServ-IP-centric QoS Architecture, shown in section 6.4.2, it means that at least one of the QoS-PRNs (-RNs) within the end-to-end chain does not implement IntServ QoS control. In other words, one or more QoS-PRNs (-RNs) do not implement the QoS architecture shown in Figure 7.2. An example of such an element may be an IP router offering only best-effort packet delivery and not supporting any QoS control. Anyway, the presentation of solutions for QoS over heterogeneous network being the aim of the book, the simple presence of a PRN having a different QoS paradigm is

sufficient to set the NON-IS_HOP flag to 1 within the IntServ-IP-centric QoS Architecture. A network element that does not support IntServ has no information how to set this flag. The real setting of NON-IS_HOP flag may be implemented by QoS-PRNs, the signalling protocol itself and manual configurations of operators monitoring network behaviour. It is linked to the implementation of network solutions and it is outside the scope of this book. NON-IS_HOP flag is also present in service-specific versions, as well as in the Guaranteed and Controlled-Load fragments. In this case, it means that special control services are not implemented within network elements (QoS-PRNs and QoS-RNs, in this case) along the end-to-end chain. For example, it may mean that at least one network element along the path does not support the Guaranteed Service; obviously, it may support CLS. The real structure of the NON-IS_HOP flag is composed of one single bit at fixed position for each specific IS control. Anyway, it might also be regarded as a flag field assuming different values depending on the meaning, if it is thought as a single field. It is important to note that, except for the specific offered services (Guaranteed and Controlled-Load) and their requested QoS model specifically described through the SENDER_TSPEC and FLOWSPEC objects, NON_IS_HOP flag functionalities are necessary in any QoS-supporting signalling. It is true also for other functions described in the following. For this motivation, RSVP is described in detail in this book. It may be seen as a good example also for other needed signalling algorithms. If the NON-IS_HOP flag is set to 1 for a path, the receiver should consider the values of all other parameters contained in the ADSPEC, as possibly inaccurate. The overall service may be affected; actually, it is reasonable. Applied to the IntServ-IP-centric QoS architecture, it would mean that, being one or more private sub-networks not aligned with the others from the QoS viewpoint, the overall end-to-end QoS provision is affected. The conformance of all QoS-PRNs is necessary. More tolerance may perhaps be shown towards specific control services, if the overall end-to-end QoS is not affected.

NUMBER_OF_IS_HOP ID: It identifies the NUMBER_OF_IS_HOP parameter defined in [RFC2215], as well as all other parameters contained in ADSPEC. It is set to number 4 and it is binary coded.

IS_HOP_COUNT: It contains the parameter NUMBER_OF_IS_HOP, which counts the number of network elements that can implement the Integrates Services control architecture along the end-to-end path. It is incremented by 1 at each IS-aware hop.

MIN_AVAILABLE_PATH_B/W ID: It identifies the parameter MINIMAL_AVAILABLE_PATH_BANDWIDTH parameter. It is set to binary coded number 6.

PATH_BANDWIDTH_ESTIMATE: It contains the value of the parameter MINIMAL_AVAILABLE_PATH_BANDWIDTH, which is aimed at collecting information about the bandwidth available along the end-to-end path. It is originated from a similar parameter, called AVAILABLE_PATH_BANDWIDTH, which acts locally in each network element. Again, a network element may be considered the single PRN. The AVAILABLE_PATH_BANDWIDTH is an estimate of the bandwidth available by the network elements for the packets. In the QoS-PRN case, it is the available bandwidth within the private technology-dependent sub-network "governed" by each single QoS-PRN, whose control architecture in this case is also represented in Figure 7.2. This information may also be derived from private signalling protocol acting locally at each single network portion. The AVAILABLE_PATH_BANDWIDTH parameter is identified by the ID 5. If the service is restricted to a limited portion of the overall available bandwidth, the service module needs

to specify this smaller value. The minimum between the value specified by the network element and by the restricted value is identified by the ID 6 in [RFC2215] and called here as MINIMAL_AVAILABLE_PATH_BANDWIDTH. In practice, it specifies the minimum available bandwidth link along the end-to-end path; that is the bandwidth bottleneck of the path.

CUMULATIVE_LATENCY ID: It identifies the parameter CUMULATIVE_ MINIMUM_PATH_LATENCY parameter. It is set to binary coded number 8.

MINIMUM_PATH_LATENCY: It contains the value of the parameter CUMULATIVE_MINIMUM_PATH_LATENCY, which is the latency of the packet forwarding process associated with the end-to-end path. Also in this case, the cumulative parameter is based on a local parameter, which acts within each network element. It is called MINIMUM_PATH_LATENCY and the ID number 7 identifies it. It is defined by the smallest possible packet delay added by the network element. If the IntServ-IP-centric QoS architecture in Figure 6.11 were considered, the MINIMUM_PATH_LATENCY would be the minimum delay that a single sub-network can provide. All the queues in the path are considered empty because of the minimum delay assumption.

If packets traversing a network element (also a single network portion, in this book) may experience different minimal delays by following different paths, the MINIMUM_PATH_LATENCY should consider and report a latency value for each path. It means that, if the next QoS-PRN may be reached through different paths within the network portion, the latency values for each path should be reported. If the considered network portion implements more than one technology, the latency value is technology dependent. Alternatively, also the minimum value among all the possible network portion paths may be reported. In this case, the MINIMUM_PATH_LATENCY value is not dependent on the path followed within each portion. The choice totally concerns the control modules of QoS-PRNs (-RNs). If the control module does not support this feature, it is also possible to estimate the MINIMUM_PATH_LATENCY parameter through off-line computations and real measures. The MINIMUM_PATH_LATENCY values reported for each network element are summed. It gives origin to the CUMULATIVE_MINIMUM_PATH_LATENCY parameter, identified by ID set to 8, which informs about the minimal packet delay along the end-to-end path. The parameter is measured in microseconds.

COMPOSED_PATH_MTU ID: It identifies the COMPOSED_PATH_MTU parameter. It is set to binary coded number 10.

COMPOSED_MTU: It contains the value of the parameter COMPOSED_PATH_MTU, i.e. the Maximum Transmission Unit (MTU) for the packets along the end-to-end path. It is identified by the number 10 and is computed through a minimum operation of the MTU values revealed in each single network element. This local parameter is called PATH_MTU and identified by the ID number 9. The MTU of a network element is the maximum transmission unit the network element can forward without fragmenting. It includes IP and upper layers protocol headers but it does not consider link-layer headers. The definition perfectly adapts if applied to the IntServ-IP-centric QoS architecture. The PATH_MTU is the maximum transmission unit the single traversed sub-network can forward without fragmenting.

Concerning the Controlled-Load Service ADSPEC, there is no need for additional information, except for the Controlled-Load break bit, whose meaning is the same as of the NON-IS_HOP flag.

The Guaranteed Services fragment contains information useful to compute the maximum end-to-end delay bound as required by IntServ GS approach. The meaning of each single

parameter is strictly linked to the model used to get the delay bound and, for this, it is strictly linked to IntServ approach. Reference [RFC2212] contains the model used for GS IntServ. Each network element must assure that the offered service may be approximated by the "fluid model" at fixed service rate (e.g. R), which is essentially the service offered by a dedicated wire providing bandwidth R and connecting the source to the destination. In practice, Generalized Processor Sharing (GPS) scheduling, see Chapter 4 for details, is assumed in each network element (in each network portion, concerning the IntServ-IP-centric QoS architecture considered here). It is necessary to check both the maximum deviation that each single network element can introduce away from the fluid model and the global end-to-end deviation as a summation operation of the local quantities. Locally, two parameters are used: C and D. C is a rate-dependent term and represents the delay that a packet can experience due to the flow rate parameters. An example may be the encapsulation delay due to the change of technology. It is divided by the rate R to have a fair metric and it is measured in units of bytes. D is measured in microseconds. It is rate-independent and it is based on the worst-case non-rate-based transit time variation through the network element. For example, always having the IntServ-IP-centric QoS Architecture in mind and the heterogeneous structure of Figure 5.7, if one portion is implemented over satellite (special QoS gateways for satellites will be presented in Chapter 10), D computation is related to the channel access scheme. If the channel access is ruled through TDMA a specific slot is given to a single flow within each transmission frame, even if a packet is ready to be transmitted, it needs to wait until it gets the right slot. A packet may need to wait the entire frame length in the worst case. D should be set to the frame length. Composing together the different values of C and D by a summation operation including all the network elements of the end-to-end path, the worst end-to-end case concerning the delay that a packet can experience due to the flow rate parameters and the non-rate-based transit time variation are obtained. The quantities C_{tot} and D_{tot} identify the two compositions, respectively. They are used to compute the maximum end-to-end delay bound as indicated in [RFC2212]. Their value is explicitly contained in the Guaranteed Services fragment. Additionally, the partial sums of C and D computed from the most recent network point, where there is an attempt to restore the possibly distorted traffic's shape to conform Tspec (reshaping point), give origin to two more parameters called C_{sum} and D_{sum}, respectively. Due to the aim of this book, there is no need to provide more details in this direction. The overall features are contained in [RFC2212] and references therein.

The Guaranteed Services fragment (in Figure 7.14) contains the following:

- A header including length and Guaranteed Services break bit, whose meaning is the same of the NON-IS_HOP flag but it is related to Guaranteed Services.
- The binary coded identifiers for each parameter: 133 for C_{tot}, 134 for D_{tot}, 135 and for C_{sum} and 136 for D_{sum}.
- Parameters' flag and length, indicated through "Other parameters".
- Service-specific fields, in addition, if necessary.

In addition to the mentioned objects, Class number 15, called Resv_CONFIRM is used to confirm a reservation. It uses two types: the IPv4 RESV_CONFIRM object, C-Type = 1, which is composed of the IPv4 Destination Address (32 bit) and the IPv6 RESV_CONFIRM object, C-Type = 2, which is composed of the IPv6 Destination Address (128 bit).

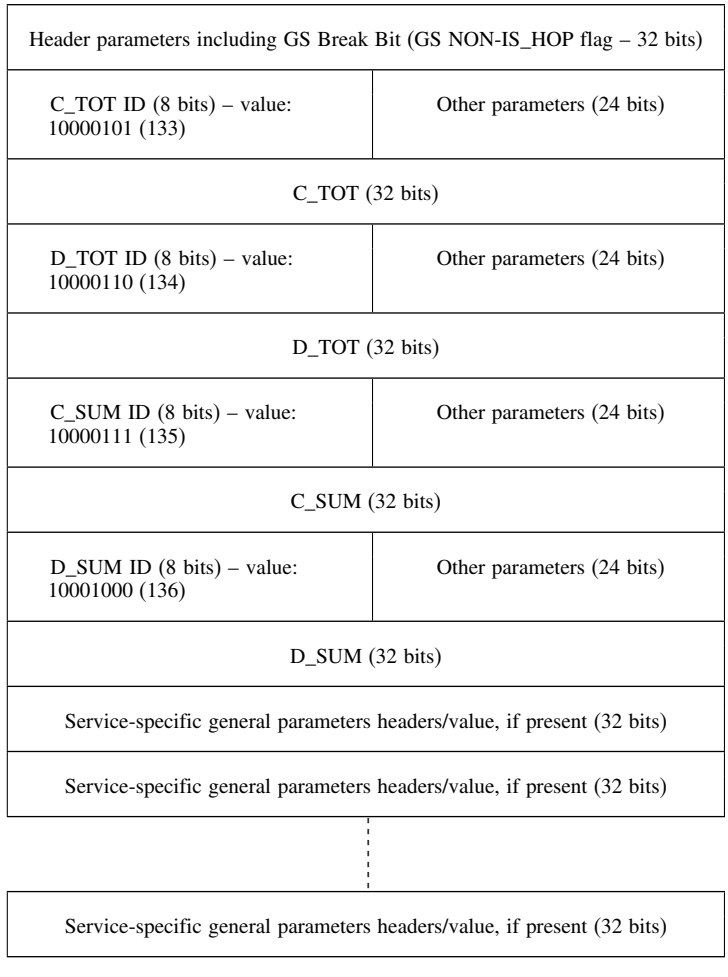


Figure 7.14 Guaranteed Services fragment (selection of real parameters)

7.2.3 RSVP Entities and Resource Reservation Applied to QoS Architecture

It is important to note that each single QoS gateway (QoS-PRNs), by following the terminology used both in Chapter 6 and in Figure 7.1, should implement RSVP and support the guaranteed service model to achieve end-to-end guaranteed service. It means to substitute, as often done in this chapter, the generic network element with the QoS gateway (implemented as Private Relay Node – PRN). The overall scheme is shown in Figure 7.15, where the relation between RSVP network entities (RSVP Entities) and IntServ-IP-centric QoS architecture is clarified. Acting as a QoS gateway, each control entity is in touch with the local control entities of the single network portion both to export the parameters' value, as well as the GS fragment ones, and to decide the acceptance and the rejection of a new user.

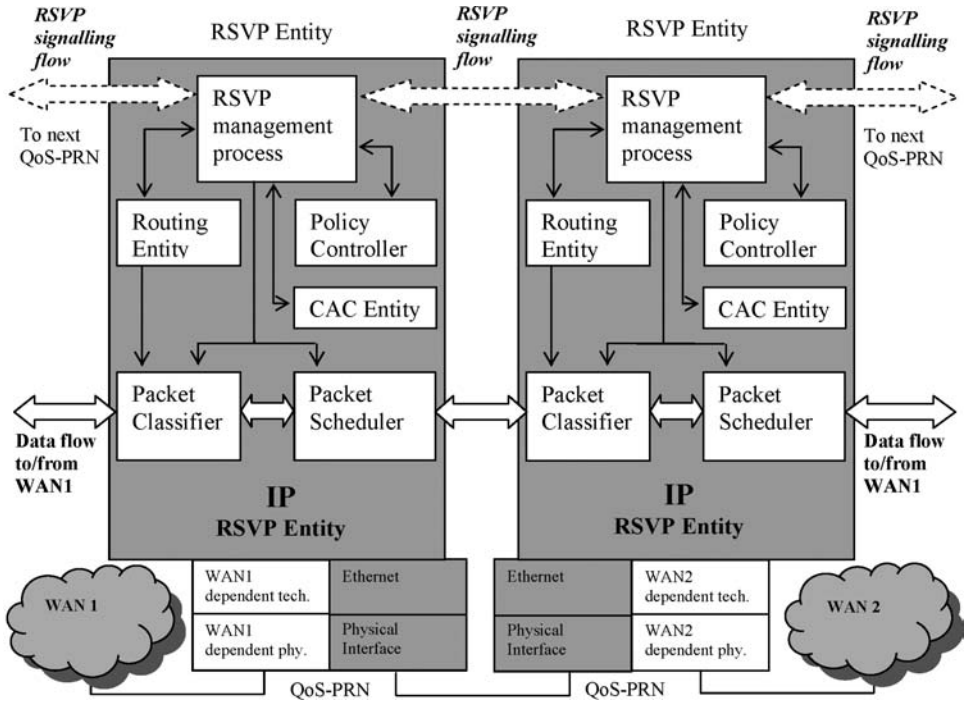


Figure 7.15 IntServ-IP-Centric QoS Architecture Signalling Entities

Obviously, the same functional scheme may be applied to the simplified QoS-RN approach reported in Figure 6.12. In this case, there is one RSVP Entity for each Relay Node.

The RSVP Entities are in charge of RSVP implementation and, in particular, of resource reservation requests transportation. The signalling protocol, on the base of the Tspecs and Rspecs order established before, can implement different resource reservation options, called "styles". One option concerns the reservations for different senders (sources) in the same session. It is possible to have a separate reservation for each single sender. This choice is identified as "Distinct" reservation. Alternatively, it is possible to have a single reservation that includes all the packets of selected senders. This choice is identified as "Shared" reservation. The second reservation option regards the selection of senders. There are two possible choices: either the selected senders are reported within an "Explicit" list or all the senders of the session are implicitly selected through a "Wildcard". The former requires that each FILTER_SPEC object (Figures 7.5, 7.6 and 7.7) match one precise sender. The latter does not require any FILTER_SPEC. Therefore, in short, the sender selection may be Explicit or Wildcard; the reservations may be Distinct or Shared. It gives origin to the reservation styles [RFC2205] reported in Figure 7.16. Shared reservation together with Wildcard sender selection gives origin to Wildcard-Filter (WF) Style. Shared reservation joined with Explicit sender selection provides Shared-Explicit (SE) Style. Distinct reservation together with Explicit sender selection gives origin to Fixed-Filter (FF) Style. No style is defined in the other case because it has no physical meaning.

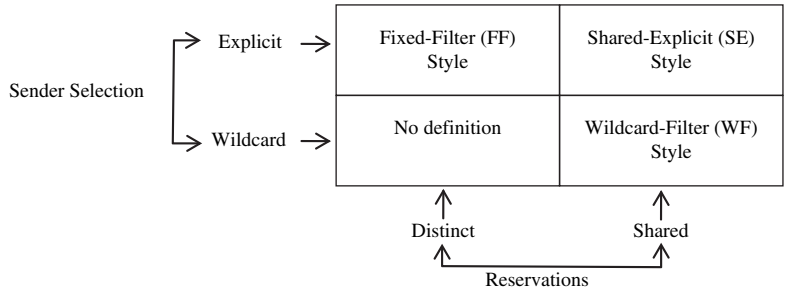


Figure 7.16 RSVP reservation styles

The reservation styles are described in more detail in the following.

7.2.3.1 Wildcard-Filter Style

This style generates a single reservation shared by the flows coming from all sources. In practice, this shared reservation is a common bandwidth pipe whose dimension is the largest bandwidth request of the receivers. It is independent of the number of sources. Following the format in [RFC2205], the WF style reservation request is represented as $WF\,(*\,\{FLOWSPEC\})$. The * is the wildcard sender selection. It means all the senders of the session. *FLOWSPEC* is the Flowspec object previously defined.

7.2.3.2 Shared-Explicit (SE) Style

A single reservation is created (as well as for the WF style), but the senders that share the reservation are explicitly selected. The format is $SE\,((S_1, S_2, \ldots, S_N)\,\{FLOWSPEC\})$. The N information sources identified as $S_1, S_2, \ldots, S_N$ share the single reservation specified through *FLOWSPEC*.

7.2.3.3 Fixed-Filter (FF) Style

This style creates a distinct reservation for each particular sender. The reservation is not shared with the other sources of the same session but it is separate. IntServ allows specifying a single user flow (customer) and the FF style allows assigning a specific bandwidth to this flow. The reservation format is $FF\,(S_1\,\{FLOWSPEC_1\})$, for a single source S_1 and $FLOWSPEC_1$. Multiple FF style reservations are allowed to be transported within the same request. The format is $FF\,(S_1\,\{FLOWSPEC_1\}, S_2\,\{FLOWSPEC_2\}, \ldots, S_N\,\{FLOWSPEC_N\})$.

WF and SE styles are suited for multicast sources that do not generate traffic at the same time. A good example is audio-conference where reserving bandwidth for one single voice flow a time is often sufficient because, typically, the participants speak one at a time. FF style is the only choice where each single customer wants a specific level of quality. Not all the mentioned reservation styles can be merged together. In line with [RFC2205], examples of reservation styles are reported in the following applied to RSVP Entities shown

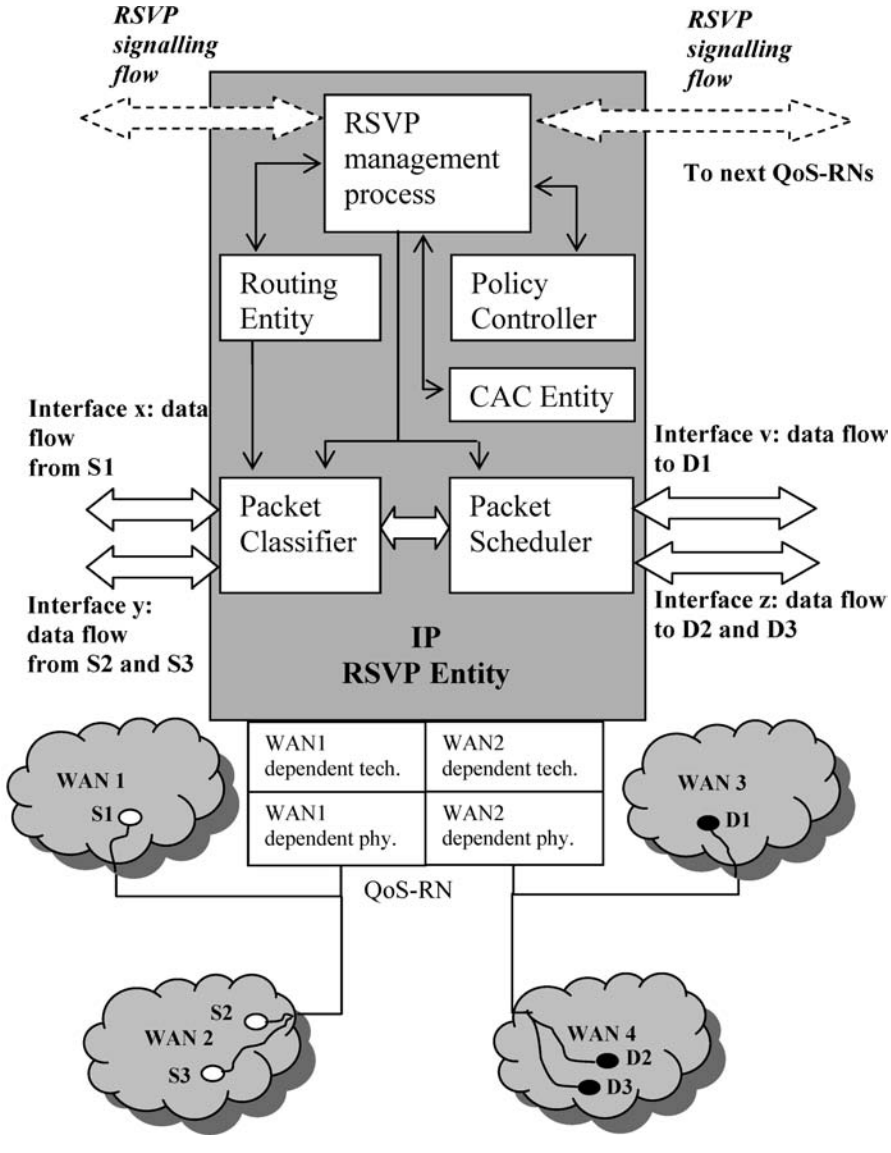


Figure 7.17 RSVP entity configuration

in Figure 7.15. Figure 7.17 shows the RSVP Entity configuration for QoS-RN. RSVP Entity has two input interfaces, identified as x and y, and two output interfaces, called v and z. A QoS RN instead of a PRN is taken as reference to simplify the approach. Input Interface x receives traffic from the traffic source (sender) S_1; input Interface y conveys traffic from the traffic sources identified through S_2 and S_3. Output Interface v forwards traffic to the destination (receiver) called D_1, while output Interface z forwards traffic to destinations D_2 and D_3.

Table 7.1 Interface configurations for WF reservations

Received resource reservations	Implemented resource reservations	Forwarded resource reservations
$WF\left(*\left\{FLOWSPEC_{D_1}\right\}\right)$ at Interface v $WF\left(*\left\{FLOWSPEC_{D_2}\right\}\right)$ at Interface z $WF\left(*\left\{FLOWSPEC_{D_3}\right\}\right)$ at Interface z	$\left(*\left\{FLOWSPEC_{D_1}\right\}\right)$ at Interface v $\left(*\left\{MergedFLOWSPEC_{D_2D_3}\right\}\right)$ at Interface z	$WF\left(*\left\{MergedFLOWSPEC_{D_1D_2D_3}\right\}\right)$ at Interface x $WF\left(*\left\{MergedFLOWSPEC_{D_1D_2D_3}\right\}\right)$ at Interface y

7.2.3.4 Wildcard-Filter Style Example

RSVP Entity receives three WF reservation requests: one from D_1 through Interface v and $FLOWSPEC_{D_1}$; one from D_2 through Interface z and $FLOWSPEC_{D_2}$ and the last one from D_3 through Interface z and $FLOWSPEC_{D_3}$. Flowspecs, as said before, may be merged together by selecting, concerning Tspec, the largest Token Bucket Rate and Size, the largest Peak Data Rate, the smallest Minimum Policed Unit and the largest Maximum Packet Size, and, concerning Rspec, the largest Rate and the smallest Slack. The operation is performed to forward the mentioned reservations. Being WF reservations, $FLOWSPEC_{D_2}$ and $FLOWSPEC_{D_3}$ are merged together to perform reservation over the Interface z. The merging result is called $MergedFLOWSPEC_{D_2D_3}$. $FLOWSPEC_{D_1}$ and $MergedFLOWSPEC_{D_2D_3}$ are merged together, giving origin to $MergedFLOWSPEC_{D_1D_2D_3}$. The composed request is forwarded through Interfaces x and y. More schematically, Table 7.1 contains the resource reservation operations.

7.2.3.5 Shared-Explicit (SE) Style Example

Reservation requests are separate for each single source but the reservations are shared among all the receivers that emit the requests. RSVP Entity receives five reservation requests: two shared reservations from D_1 to S_1 and S_2 at Interface v through $FLOWSPEC_{D_1}^{S_1S_2}$; two shared reservations from D_2 to S_1 and S_3 at Interface z through $FLOWSPEC_{D_2}^{S_1S_3}$; and one reservation from D_3 to S_2 at Interface z through $FLOWSPEC_{D_3}^{S_2}$. Resources specified in $FLOWSPEC_{D_1}^{S_1S_2}$ are reserved at Interface x to S_1 and S_2. $FLOWSPEC_{D_2}^{S_1S_3}$ and $FLOWSPEC_{D_3}^{S_2}$ are merged and the resources specified in $MergedFLOWSPEC_{D_2D_3}^{S_1S_2S_3}$ are reserved at Interface z. $FLOWSPEC_{D_1}^{S_1S_2}$ and $MergedFLOWSPEC_{D_2D_3}^{S_1S_2S_3}$ are merged so getting $MergedFLOWSPEC_{D_1D_2D_3}^{S_1S_2S_3}$, whose request is forwarded both at Interface x and at Interface y towards the sources. SE interface configurations are reported in Table 7.2.

7.2.3.6 Fixed-Filter (FF) Style Example

Reservation requests are separate for each single source. RSVP Entity receives five reservation requests: two from D_1 to S_1 and S_2. They are transported through Interface v by using

Table 7.2 Interface configurations for SE reservations

Received resource reservations	Implemented resource reservations	Forwarded resource reservations
$SE\left((S_1, S_2)\left\{FLOWSPEC_{D_1}^{S_1S_2}\right\}\right)$ at Interface v	$\left((S_1, S_2)\left\{FLOWSPEC_{D_1}^{S_1S_2}\right\}\right)$ at Interface v	$SE\left(S_1\left\{Merged\ FLOWSPEC_{D_1D_2D_3}^{S_1S_2S_3}\right\}\right)$ at Interface x
$SE\left((S_1, S_3)\left\{FLOWSPEC_{D_2}^{S_1S_3}\right\}\right)$ at Interface z	$\left((S_1S_2S_3)\left\{MergedFLOWSPEC_{D_2D_3}^{S_1S_2S_3}\right\}\right)$ at Interface z	$SE\left((S_2, S_3)\left\{Merged\ FLOWSPEC_{D_1D_2D_3}^{S_1S_2S_3}\right\}\right)$ at Interface y
$SE\left(S_2\left\{FLOWSPEC_{D_3}^{S_2}\right\}\right)$ at Interface z		

$FLOWSPEC_{D_1}^{S_1}$ and $FLOWSPEC_{D_1}^{S_2}$, respectively. Two reservations come from D_2 and one reservation from D_3. Both are received at Interface z. The formers are directed to S_1 and S_3 and are formally described through $FLOWSPEC_{D_2}^{S_1}$ and $FLOWSPEC_{D_2}^{S_3}$, respectively. The latter concerns S_1 and is described as $FLOWSPEC_{D_3}^{S_1}$. Merging operation is again fundamental. Reservation requests that share the source are merged together and the most demanding reservation is dominant. The used symbols are the same as used for WF and SE reservations. Resource reservations coming from D_2 and D_3, respectively, and dedicated to S_1 are merged together at Interface z giving origin to $MergedFLOWSPEC_{D_2D_3}^{S_1}$. $FLOWSPEC_{D_1}^{S_1}$ is merged with $MergedFLOWSPEC_{D_2D_3}^{S_1}$ so generating $MergedFLOWSPEC_{D_1D_2D_3}^{S_1}$. The reservation $FF\left(S_1\left\{MergedFLOWSPEC_{D_1D_2D_3}^{S_1}\right\}\right)$ is so forwarded through the Interface x. There is no other merge because only one destination specifies requests for S_2 and S_3. $FLOWSPEC_{D_1}^{S_2}$ and $FLOWSPEC_{D_2}^{S_3}$ are used to forward requests at Interface y. Table 7.3 contains the performed operations.

The reservation style is transported through a suited class, as well as the other RSVP objects: the STYLE class. The number 8 and C-Type 1 identify it. It is shown in Figure 7.18. The Flags field is not assigned. The Option Vector specifies the value for the reservation options and it is composed, as shown in Figure 7.18, without respecting any proportion, by 19 reserved bits, 2 bits called Sharing Control and 3 bits identified as Sender Selection Control. The field Sharing Control is binary coded as shown in Table 7.4; the Sender Selection Control as in Table 7.5. The two tables together give origin to the Reservation Style coding, reported in Table 7.6.

7.2.4 RSVP Functional Specification (RSVP Packet Format)

The RSVP message that transports all the information specified above has a particular format. The RSVP packet format is shown in Figure 7.19. It is structured into two parts: the Common Header, which is 64 bits and is shown in light grey in Figure 7.19, and the Object Formats, which is at least 32 bits. The Common Header is present in any RSVP packet

Table 7.3 Interface configurations for FF reservations

Received resource reservations	Implemented resource reservations	Forwarded resource reservations
$FF\left(S_1\left\{FLOWSPEC_{D_1}^{S_1}\right\}, S_2\left\{FLOWSPEC_{D_1}^{S_2}\right\}\right)$ at Interface v	$\left(S_1\left\{FLOWSPEC_{D_1}^{S_1}\right\}\right)$ at Interface v	$FF\left(S_1\left\{MergedFLOWSPEC_{D_1D_2D_3}^{S_1}\right\}\right)$ at Interface x
$FF\left(S_1\left\{FLOWSPEC_{D_2}^{S_1}\right\}, S_3\left\{FLOWSPEC_{D_2}^{S_3}\right\}\right)$ at Interface z	$\left(S_2\left\{FLOWSPEC_{D_1}^{S_2}\right\}\right)$ at Interface v	$FF\left(S_2\left\{FLOWSPEC_{D_1}^{S_2}\right\}, S_3\left\{FLOWSPEC_{D_2}^{S_3}\right\}\right)$ at Interface y
$FF\left(*\left\{FLOWSPEC_{D_3}^{S_1}\right\}\right)$ at Interface z	$\left(S_1\left\{MergedFLOWSPEC_{D_2D_3}^{S_1}\right\}\right)$ at Interface z	
	$\left(S_3\left\{FLOWSPEC_{D_2}^{S_3}\right\}\right)$ at Interface z	

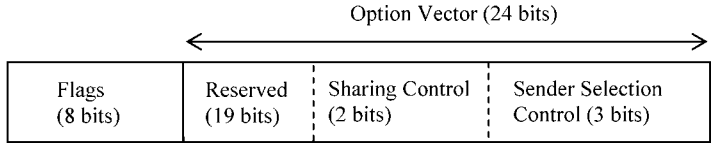


Figure 7.18 STYLE object: Class = 8, C-Type = 21

Table 7.4 Sharing Control field coding scheme

Field Code	Field Meaning
00	Reserved
01	Distinct Reservations
10	Shared Reservations
11	Reserved

Table 7.5 Sender Selection Control field coding scheme

Field Code	Field Meaning
000	Reserved
001	Wildcard
010	Explicit

Table 7.6 Reservation Style coding scheme

Reservation Style Code	Reservation Style Name
10001	WF – Wildcard Filter
01010	FF – Fixed Filter
10010	SE – Shared Explicit

and contains information useful for any possible transported object. The following is the detail:

Version (4 bits): it is the protocol version number
Flags (4 bits): Reserved
Message Type (8 bits): it contains the type of message that is transported; for example, Path message is binary coded as "00000001" (number 1), Resv message as "00000010" (number 2). Path and Resv messages have been described in the text. For the sake of completeness, the codes of each RSVP message are contained in Table 7.7, together with a quick description of their functionalities.

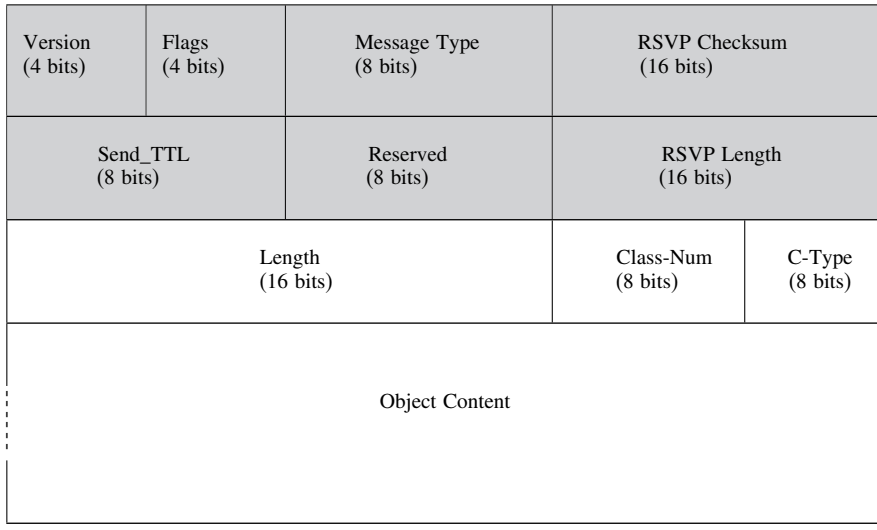


Figure 7.19 RSVP packet format

Table 7.7 Message Type Codes

Message Type (Code)	Message Name	Functionality
00000001	Path	Path control message. It contains SENDER_TEMPLATE, SENDER_TSPEC and, optionally, ADSPEC. It travels from the source (sender) to the destination (receiver)
00000010	Resv	Reservation request control message. It carries reservation requests from the destinations to the sources through the STYLE object and the flow descriptors (FLOWSPEC, FILTER_SPEC)
00000011	PathErr	Path error control message. It is sent to the originating source if an error is revealed during Path message processing
00000100	ResvErr	Reservation error control message. It reports an error in processing Resv messages. It signals either that a reservation request has failed or that a reservation has been cancelled by administrative pre-emption. It travels towards destinations
00000101	PathTear	Path teardown control message. It deletes the path state defined by the proper objects (SESSION, SENDER_TEMPLATE, and Previous Hop – PHOP). It travels towards destinations
00000110	ResvTear	Reservation teardown control message. It deletes the reservation state defined by the proper objects (SESSION, STYLE, FILTER_SPEC and LIH in RSVP-HOP). It travels towards sources
00000111	ResvConf	Reservation confirmation control message. It confirms that a reservation has been successfully installed. It is sent to the destinations

IPv4 Error Node Address (32 bits)		
Flags (8 bits)	Error Code (8 bits)	Error Value (16 bits)

Figure 7.20 IPv4 ERROR_SPEC object: Class = 6, C-Type = 1

IPv6 Error Node Address (128 bits)		
Flags (8 bits)	Error Code (8 bits)	Error Value (16 bits)

Figure 7.21 IPv6 ERROR_SPEC object: Class = 6, C-Type = 2

Without entering the details, error management is performed through a special class, called ERROR_SPEC class, identified by number 6. There are two different C-types: C-Type = 1, for IPv4 ERROR_SPEC, and C-Type = 2, for IPv6 ERROR_SPEC. The format of both is reported in Figures 7.20 and 7.21, respectively. The fields IPv4 (in Figure 7.20) and IPv6 (in Figure 7.21) Error Node Address contain, respectively, the IPv4 and IPv6 address of the node where the error is detected. The field Flags provides details about special conditions and it is used in ResvErr messages. Error Code is the error descriptor and Error Value contains additional information about the error. It is important to note the similarity with the format of IPv4 and IPv6 FILTER_SPEC (Figures 7.5 and 7.6).

7.2.5 Summary of RSVP Protocol Mechanism

Taking Figure 7.17 as reference, Figure 7.22 contains the data and fundamental message direction through the defined interfaces. RSVP works as follows. Each RSVP source (sender) transmits a Path message to the destination(s) along the route fixed by a routing protocol. The path followed is the same as for data. When the Path message traverses an IP RSVP Entity, it fixes path states within each Entity. The Path state is used by the Resv messages in

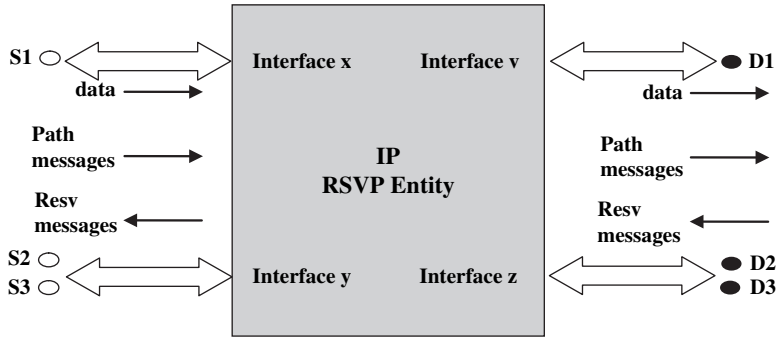


Figure 7.22 Path and Reservation message direction through RSVP Entity interfaces

the opposite direction. Each destination (receiver) transmits Resv messages to the sources. Resv messages follow exactly the reverse of the path followed by data and Path messages. They create and keep alive "reservation states" in each traversed RSVP Entity. Reservation requests are merged together: the "largest" reservation request is forwarded along the path from the destination to the source so no user is penalized. The reservation state is not permanent; it needs to be periodically refreshed through Path and Resv messages. These temporary reservations are called "soft states". If no refresh message has arrived before time-out expiration, the reservation is cancelled. Reservations may also be explicitly dropped by teardown messages (PathTear and ResvTear). Teardown action may be started by an end-user application and by an RSVP Entity either because soft state time-out is expired or because a more important reservation request is arrived at the Entity interface (pre-emption). Errors are managed by ResvErr and PathErr messages.

7.2.6 RSVP Extension for DiffServ QoS Signalling

As clearly indicated in [RFC4094], there is a number of RSVP-based protocol extensions, which provide additional features to the original protocol. A good example is RSVP-TE [RFC3209], object of the next paragraph, which is mainly dedicated to MPLS support. Some extension simply presents additional interfaces such as DiffServ. In detail, [RFC2996] introduces a new Object dedicated to transport Differentiated Services Code Points (DSCP) within RSVP messages so allowing the use of RSVP also in the DiffServ solution, if necessary. The same QoS entities shown in the previous paragraph and graphically reported in Figures 7.15 and 7.17 may be still applied. The reference framework is the integration of IntServ over DiffServ networks [RFC2998].

The format of the new object, called DCLASS, is shown in Figure 7.23.

The Length field contains the overall length of the object in bytes. The class number is 225 in binary. C-Type is 1. The first word is part of the Object Header as in Figure 7.19.

Length (16 bits)	Class Num (8 bits)	C-Type (8 bits)	
Unused (24 bits)		1st DSCP (6 bits)	
Unused (24 bits)		2nd DSCP (6 bits)	
Unused (24 bits)		3rd DSCP (6 bits)	
Unused (24 bits)		4th DSCP (6 bits)	
Unused (24 bits)		k-th DSCP (6 bits)	

Figure 7.23 DCLASS object: Class = 225, C-Type = 1

It is reported here for the sake of completeness. The rest of the object is a list of DSCP values whose number is determined by the Length field. The interpretation and use of the DSCP list is left to the specific resource control implementation. The possibility to transport multiple DSCPs may be interesting to distinguish specific flows within an aggregate or to give a specific meaning that can join bandwidth allocation to the order of DSCP appearance in the object. Also a single DSCP may be transported.

The main aim of DCLASS is to carry information about DSCP between the DiffServ network and the nodes where the packets are marked by using DSCP values [RFC2996]. More generically speaking, it is a tool to convey DSCP information in RSVP messages. In the case treated in this book, the idea is to use it to transport DSCP information among the DiffServ-IP-centric QoS-PRSs. The basic tool is standard RSVP Path message, which is sent towards the receiver by the information source (the first QoS-PRN, in this book). Within the IntServ over DiffServ environment, at some point, the Path message reaches the DiffServ portion. Within the DiffServ-IP-centric QoS architectures, RSVP Path message is generated within the RSVP Entity implemented at the first QoS-PRN along the end-to-end path composed of DiffServ-based PRNs. The Path message traverses the QoS-PRNs chain, and RSVP Entities make a decision regarding the admissibility of the signalled flow. If the network element determines that the request represented by the Path and consequent Resv messages is admissible for the DiffServ-IP-centric QoS architectures, the proper DiffServ service level (behaviour aggregate) for the traffic represented in the RSVP DCLASS-driven request is fixed. Appropriate DSCP packet marking is conveyed and implemented within QoS-PRNs.

Even if the use of RSVP over DiffServ-IP-centric QoS-PRSs may seem strange, it may be an interesting tool for Bandwidth Broker-based DiffServ-IP-centric QoS architectures.

7.3 RSVP-TE

RSVP-TE (Traffic Engineering) has been introduced in [RFC3209]. It contains the necessary RSVP extensions in terms of new objects and their use to create and maintain Label Switched Path (LSP) in MPLS networks. Its features are summarized here taking just [RFC3209] as reference, both because RSVP-TE signalling is perfectly suited for MPLS-centric QoS architectures and because it may also be applied to other environments, well beyond its initial scope. The author of this book thinks that it may be also used as reference for the design of IPv6-centric QoS signalling.

7.3.1 Introduction

Within Full-MPLS-centric architecture introduced in section 6.5.2, the traffic that flows through the QoS-PRNs composes an LSP between the first and the last QoS-PRN. The LSP is defined by the label applied at the first QoS-PRN that acts as ingress node of the LSP within an MPLS network. The same concept, as explained in section 6.5.1, may be applied at the borders of the MPLS portion ruled by two QoS-PRNs, in the Integrated-MPLS architecture. The LSP is created only within the MPLS network portion in this last case. Integrated-MPLS approach, being thought as a way to carry hard QoS guarantees in a DiffServ-based approach, perfectly matches the reference framework where RSVP-TE has been designed, but Full-MPLS allows to enlarge the application field of RSVP-TE and to

show a wider scope, already hidden in the intention of the protocol designers, as clear from the signalling features. Reference to Full-MPLS and Integrated-MPLS will be explicitly done if necessary, otherwise the scenario is represented by MPLS-centric generic solution. In practice, concerning signalling, the difference is only in the extension of the network. Full-MPLS is an overall MPLS network, where the label is assigned at the first QoS-PRN, dropped at the last one and switched at each QoS-PRN. Integrated-MPLS is an IP-based network (also for signalling) where MPLS is used to provide QoS guarantees; the label is assigned and dropped at each QoS-PRN. The LSPs can be treated as tunnels for normal IP traffic. They can be referenced as "LSP tunnels". Each single traffic flow is identified by the MPLS label at the QoS-PRN and each MPLS label may identify a single SLS or a group of SLSs. As already highlighted, the use of MPLS allows using network optimization and control strategies. RSVP-TE permits to establish LSP tunnels by using RSVP. The definition of new objects is essential to get to the aim, but it is important to say that all described objects and messages are optional with respect to RSVP to allow a full integration between RSVP and RSVP-TE. In more detail, the request to bind labels to specific LSP tunnels is initiated at ingress nodes (the first QoS-PRNs in QoS architectures) by using RSVP Path messages, whose features are extended through a LABEL_REQUEST object. Path messages travel from the source to the destination (downstream). On the other hand, just following the RSVP procedure, in response to label requests, the assigned labels need to be propagated upstream (from the destination to the source) through Reservation messages. In consequence, Resv messages are extended by the new LABEL object. RSVP-TE allows also establishing a specific path from the source to the destination. Concerning the topic of this book, it means to fix a route specifying the QoS-PRNs to be traversed so making available a powerful tool to connect bandwidth allocation, QoS signalling and routing, as mentioned in the QoS architectures composed of different control modules and presented in Chapter 6. The tool used to support explicit routing capability is the EXPLICIT_ROUTE object, which extends the features of RSVP Path messages through the encapsulation of a concatenation of hops. The technical details are even more interesting. Strictly following the specification contained in [RFC3209], the explicitly routed path defined by the mentioned concatenation can be administratively specified and/or computed by a special entity based on QoS and policy requirements, also considering the network state. This suitable entity is just the QoS-PRN, in the defined QoS architectures, which takes decisions based on SLS requirements (Chapter 1) and control algorithms (Chapter 4). Referencing Full-MPLS-centric solutions, end-to-end paths are defined as a concatenation of QoS-PRNs, and the intermediate private network portions may be seen as abstract nodes (Figure 6.26). It might be a problem for QoS signalling but, once again, [RFC3209] already embeds the solution. The concept of explicitly routed LSPs is generalized by the notion of abstract nodes. As already defined in Chapter 6, an abstract node is a group of nodes whose internal topology is opaque to the ingress node of the LSP. It is defined as "simple", if it is composed of only one physical node. In consequence, an explicitly routed LSP can be specified as a sequence of IP addresses and/or a sequence of Autonomous Systems (in the wider sense of network portions explained in Chapter 2) by using this abstract concept. It is exactly what is necessary to manage LSPs within MPLS-centric QoS-PRNs, which are identified by IP addresses and connect different Autonomous Systems, more generically identified as heterogeneous network portions. In other words, the path between the first and the last QoS-PRN is an LSP tunnel defined as a sequence of abstract nodes (i.e. as a sequence of network portions). Figure 6.26 perfectly

matches this concept that is essential for the comprehension of MPLS-centric QoS solutions. Moreover, even if the reference traffic in this book is mainly IP based, also alternatives may be considered. For example, the host stack description in Figure 5.15 includes also ISDN, as well as the flow encapsulation examples in Chapter 6. RSVP-TE specification helps solve also this problem. The LABEL_REQUEST object not only transports the label request but also provides information about the network layer protocol to be carried over the LSP.

There are two more objects that extend the features of RSVP: RECORD_ROUTE and SESSION_ATTIBUTE. The former is added both to the Path and to the Resv messages. It memorizes the actual route that the LSP tunnel traverses and it is used for loop detection and, by the senders, to check possible changes to the routing path. The latter, added to Path messages, aids to identify sessions and contains additional control information.

7.3.2 New Objects Definition

In short, the following objects are defined:

1) LABEL_REQUEST, carried in Path messages
2) LABEL, carried in Resv messages
3) EXPLICIT_ROUTE, carried in Path messages
4) RECORD_ROUTE, carried in Path and Resv messages
5) SESSION_ATTRIBUTE, carried in Path messages.

They are part of the RSVP packet format reported in Figure 7.19 and Table 7.7. New C-Types for SESSION, SENDER_TEMPLATE and FILTER_SPEC objects are also defined.

Concerning the LABEL_REQUEST object, Class = 19, there are three possible C-Types: C-Type = 1 is a label request without any indication of the label range; C-Type = 2 is a request with an ATM label range and C-Type = 3 refers to a Frame Relay label range. Except for the indication of the label range, the basic meaning is exactly the same and, in this book, only C-Type = 1 is considered. The LABEL_REQUEST object C-Type = 1 format is reported in Figure 7.24. The Reserved field is set to zero on transmission and ignored on receipt. L3PID defines the above Layer 3, which is using the path. The same codes used within the Type field (2 bytes) of the Ethernet frame to identify the upper layer protocol are applied here. IP, for example, is coded as "0800" (hexadecimal code for binary "0000100000000000"). LABEL_REQUEST is used to establish an LSP tunnel. The LABEL_REQUEST object, sent through the Path message, indicates that a label binding for this path is requested, and provides an indication of the network layer protocol that needs to be carried over this path. So, non-IP network layer protocols may also be conveyed through an LSP. When the Path message that contains the request reaches the receiver, a label is allocated and forwarded to the source through the LABEL object carried in the corresponding Resv message. The label value must be allocated within the indicated range, if any. LABEL_REQUEST and LABEL objects processing is strongly associated. If the receiver accepts, the LABEL_REQUEST

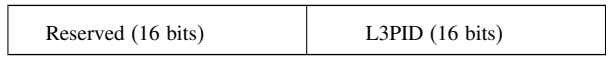


Figure 7.24 LABEL_REQUEST object: Class = 19, C-Type = 1

object must also include a LABEL object in Resv messages related to that Path message. On the contrary, if a LABEL_REQUEST object is not included in the Path message, a node must not contain a LABEL object in the corresponding Resv message. If a node should not be able to support a LABEL_REQUEST (i.e. the LSP tunnel cannot be created), an error message is sent containing the failure of the MPLS label allocation. The label request is forwarded together with the Path message node by node (QoS-PRN by QoS-PRN). The L3PID value is kept step by step. If a particular L3PID is not supported by the receiver, an error message of unsupported L3PID is sent back and the RSVP session is interrupted.

The mentioned LABEL objects are transported through the Reservation messages and immediately follow the FILTER_SPEC. The format is reported in Figure 7.25. The content of a LABEL object is just a single label of 32 bits. Numerically, it is an unsigned integer ranging from 0 to 1,048,575. Class is set to 16 and C-Type to 1. Each label identifies a specific flow within each QoS-PRN coming from a specific physical interface. Once a label is allocated, the node composes a new LABEL object and forwards it to the previous hop (i.e. the previous QoS-PRN in the end-to-end sequence) through the Resv message. The label is carried also during the refresh phase, and a node is expected to send a Resv message before its refresh timers expire if the content of the LABEL object has changed. Under normal working conditions, a node should never receive a LABEL object in a Resv message if it has not included a LABEL_REQUEST object in the corresponding Path message, but, if an RSVP entity does not recognize the LABEL object, it sends an error message coded as "Unknown object class" towards the receiver so causing the reservation failure.

The EXPLICIT_ROUTE object, also referenced through the acronym ERO, is used to specify the routes that the LSP needs to follow explicitly. Class = 20 and C-Type = 1. The format is reported in Figure 7.26. The content of a LABEL object is simply a series of variable length items called "Sub-objects". The RSVP entities that do not support the object will send an "Unknown Object Class" at the reception of the EXPLICIT_ROUTE object.

As detailed in the introduction to RSVP-TE, this object perfectly matches the requirements of MPLS-centric QoS-PRNs because an explicit route is identified by a list of groups of nodes, which must be traversed by traffic flows. Each group of nodes is identified by a specific sub-object. The explicit route is encoded as a series of sub-objects contained in the EXPLICIT_ROUTE object. In consequence, directly from the EXPLICIT_ROUTE object format, an explicit route is a specification of groups of nodes to be traversed. The group of nodes is called "abstract node". Consequently, an explicit route is defined as a list of abstract nodes. A good example of explicit route, also reported in [RFC3209] in a very similar way, is represented by the sole concatenation of heterogeneous network portions, also defined as "Autonomous Systems". They may be explicitly indicated as Sub-objects through

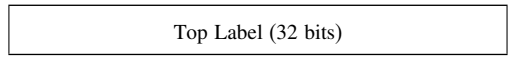


Figure 7.25 LABEL object: Class = 16, C-Type = 1

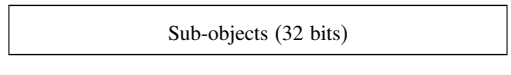


Figure 7.26 EXPLICIT_ROUTE object: Class = 20, C-Type = 1

Autonomous System number sub-objects. Each sub-object is an Autonomous System in the end-to-end communication. "In this case, each Autonomous System is an abstract node and the explicit route is a path that includes each of the specified Autonomous Systems. There may be multiple hops within each Autonomous System, but these are opaque to the source node for the explicit route" [RFC3209]. The sentence matches the scenario of MPLS-centric QoS-PRNs architectures and the relation between QoS-PRNs established through heterogeneous network portions.

Each sub-object has the format reported in Figure 7.27. If the bit *L* is set to "0", the sub-object represents a strict hop in the explicit route, otherwise, if the *L* bit is set to "1", the sub-object is a loose hop in the explicit route. If a sub-object is strict, the route between the sub-object and its preceding sub-object cannot include other network abstract nodes in the middle but, if a sub-object is loose, the path between the sub-object and its preceding sub-object may also include other network abstract nodes in the middle. In short, a loose sub-object communicates that it is necessary to go from sub-object 1 to sub-object 2, but the route in the middle is not fixed; a strict sub-object fixes also the path in the middle. The Type field identifies the type of content of the sub-object. The values defined in [RFC3209] are 1 for IPv4 prefix, 2 for IPv6 prefix, 32 for Autonomous System number. Others may be defined for future meanings. The field Length contains the total length of the sub-object in bytes, including *L*, Type and Length fields. It should be at least 4 bytes. It is a multiple of 4 bytes. Other sub-objects may be defined in case of need. It could further help to manage heterogeneous networks.

The formats of the three mentioned sub-objects are reported in Figures 7.28, 7.29 and 7.30. They contain, respectively, IPv4 prefix, IPv6 prefix and Autonomous System Number sub-objects. The general sub-object header represented by *L*, Type and Length is reported again in the mentioned figures for the sake of completeness.

Concerning Figure 7.28, Type, as already said, is set to 1; Length is always 8 bytes. It means that one IPv4 address is contained within each single sub-object. The IPv4 address, whose overall length is 32 bits, is considered a prefix based on the value reported in the Prefix Length field. Bits not included in the prefix are ignored at the reception of the message and are set to zero when the sub-object is sent forward. The Prefix Length field is in bits. The use of prefixes is a powerful tool because it allows the definition of an IP sub-net as abstract node. Padding bits are set to 0 on transmission and ignored when the message is processed.

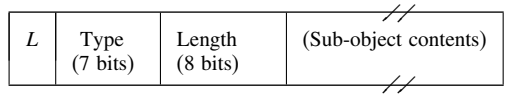


Figure 7.27 EXPLICIT_ROUTE subobject

L	Type (7 bits)	Length (8 bits)	IPv4 address (16 bits)	
IPv4 address (16 bits) [continued]			Prefix Length (8 bits)	Padding (8 bits)

Figure 7.28 EXPLICIT_ROUTE IPv4 prefix sub-object

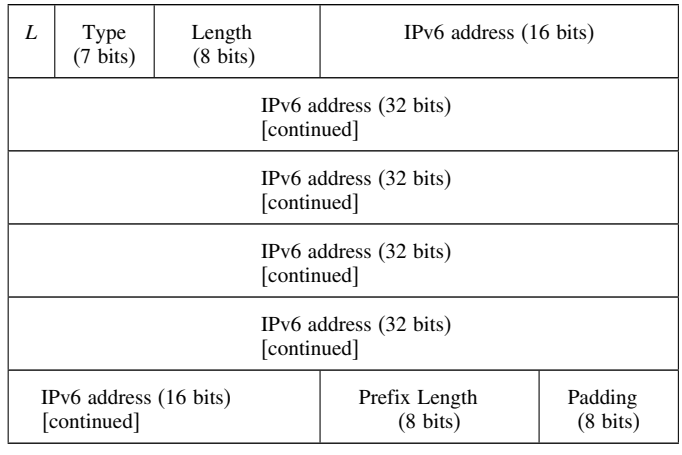


Figure 7.29 EXPLICIT_ROUTE IPv6 prefix sub-object

Figure 7.30 EXPLICIT_ROUTE Autonomous System number subobject

EXPLICIT_ROUTE IPv6 prefix sub-object format is shown in Figure 7.29: Type is set to 2. Being the overall dimension of an IPv6 address 128 bits, Length is 20 bytes (160 bits). The interpretation of the other fields is the same as for IPv4. Again, within the sub-object, one IPv6 address is transported. It may represent an IPv6 group of nodes (i.e. an IPv6 sub-net) by the help of Prefix Length field.

An Autonomous System may be defined as Abstract Node through the EXPLICIT_ROUTE Autonomous System Number sub-object, whose format is reported in Figure 7.30. Type is set to 32, Length to 4 bytes (32 bits). The 2-bytes Autonomous System Number identifies the AS whose group of networks and routers (more generically, network devices) represent the abstract nodes.

When a node receives the EXPLICIT_ROUTE object, it must take a decision about the route to follow to get to the next hop. If the Prefix Length value is 32 for IPv4 sub-object and 128 for IPv6 sub-object, i.e. if the IPv4 and IPv6 addresses represent a specific IP interface, the next hop is fixed by the sequence of sub-objects itself. On the other hand, a sub-object, as clearly stated before, is an abstract object that can represent a group of nodes. It may be an IP sub-network and, through the subject shown in Figure 7.30, an autonomous system. So, the decision about the route implies a selection among alternatives. To better understand the procedure used by a node when it receives the EXPLICIT_ROUTE object and to state a precise link with the QoS architectures presented in Chapter 6, the action steps are listed by keeping a constant look at a QoS-PRN within a Full-MPLS-centric QoS architecture.

1) The first action that the node takes at EXPLICIT_ROUTE object reception is to check the first sub-object. If the node (the QoS-PRN) is not part of the abstract node described by the sub-object, e.g. the IPv4 address of the QoS-PRN is neither the IPv4 address appearing in the sub-object nor it is part of the IP sub-network described by the sub-object, it means that the node has received the EXPLICIT_ROUTE object in error. It should not have been sent to that node. A "Bad EXPLICIT_ROUTE object" error message is sent back.
2) If the first step succeeds, the second action is the analysis of the second sub-object, which should contain an indication of the next hop. Actually, if the second sub-object contains the address of an IP interface (the IP address of the next QoS-PRN), the indication of the next hop interface is explicit. Things are more complex if the second sub-object contains the description of a group of nodes. In detail, if there is no second object, it means the end of the explicit route described by the message. Otherwise, if the node is also part of the abstract node described by the second object, the first sub-object is removed so shifting up the sub-objects in the message and the analysis begins again from point (2). If the node is not part of the abstract node described by the second sub-object, it is necessary to distinguish two cases reported in points (3) and (4).
3) The node (i.e. the QoS-PRN that has received the message) is topologically adjacent with the abstract node contained in the second object. It means that if the sub-object contains an explicit IP interface, there is a direct link between the two nodes (the two QoS-PRNs) and the selection of the next hop is mandatory. More generically, including sub-networks and autonomous systems, the node selects a particular next hop that is part of the abstract node of the second sub-object and removes the first object shifting up the other sub-objects, if any. The algorithm to select the particular next hop is totally dependent on the node implementation. In other words, it is a decision private to the single node.
4) The node is not topologically adjacent with the abstract node contained in the second object. It means that if the sub-object contains an explicit IP interface there is no direct link between the two nodes (the two QoS-PRNs). In this case, the node selects a next hop by choosing an alternative among the nodes belonging to the abstract node described by the first sub-object. In other words, it chooses a node "close to itself", belonging to the same abstract node in the hope that, from there, there is a path to the second sub-object. Two cases may happen. They are described below in points (5) and (6).
5) If the second sub-object is strict (*L* bit is set to 0), there is an error because a direct link is required between abstract nodes (if the abstract nodes describe real interfaces, a direct physical link is required). A "Bad strict node" message is returned.
6) If the second sub-object is loose (*L* bit is set to 1), the abstract node of the second sub-object may be also reached indirectly through intermediate steps. The node selects any next hop along the path to the next abstract node. It will be the care of the next hop to actually reach the abstract node. If there is no path at all, a "Bad loose node" error message is generated.
7) The last action is to replace the first sub-object, with the abstract node representing the next hop selected through the rules given in the previous points, so that the next hop can accept it at reception.

The general structure of the RECORD_ROUTE object is identical to the format of the EXPLICIT_ROUTE object. Figure 7.26 may be applied again, but the format of the

sub-objects changes. The Class identifier is 21 and C-Type is 1. There are three defined sub-objects (IPv4 address, IPv6 address and Label), but, as in the EXPLICIT_ROUTE case, more definitions may be provided, if necessary. The format of IPv4 sub-object is shown in Figure 7.31. Type is set to 1. The field Length contains the total length in bytes including all fields. It is set to 8 here. IPv4 address contains a 32-bit interface address. No sub-network is allowed and, in consequence, Prefix Length value is set to 32. The field Flags may assume two values: 1 to state that the link following this node is protected by a local mechanism and 2 to indicate that a local mechanism is active to maintain the tunnel. Figure 7.32 contains the IPv6 address sub-object. Type is set to 2: Length is 20 bytes. IPv6 address is a real IPv6 interface of 128 bits, so Prefix Length value is 128. The definition of Flags is identical to the IPv4 case. The Label sub-object is shown in Figure 7.33. Type is 3. Length and Flags have the same meaning as in IPv4 sub-object. C-Type is copied from the Label Object as well as from the Content of the Label Object itself.

The RECORD_ROUTE object is used for three main reasons: to find out possible routing loops; to collect information about the followed route and to be directly used as input for the EXPLICIT_ROUTE object with minor changes. RECORD_ROUTE may be embedded

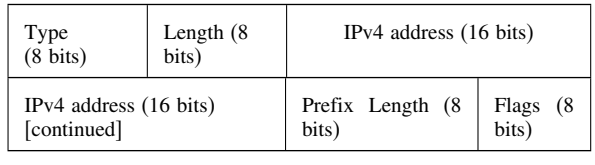


Figure 7.31 RECORD_ROUTE IPv4 sub-object

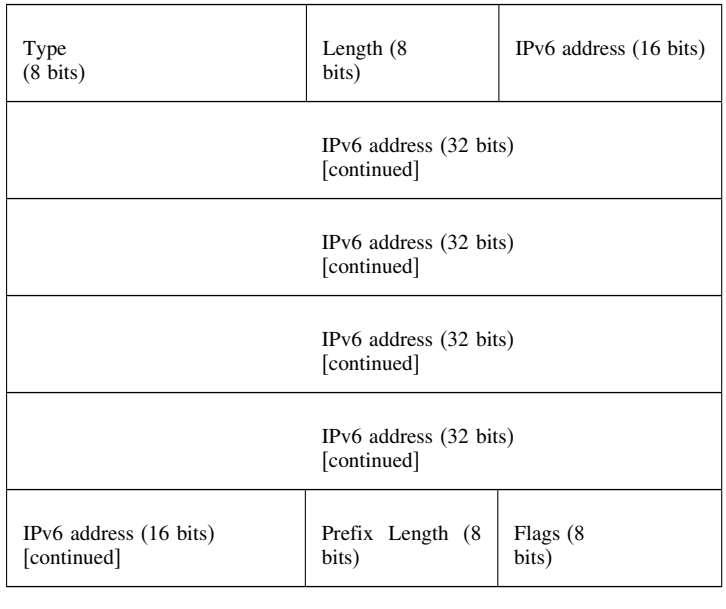


Figure 7.32 RECORD_ROUTE IPv6 sub-object

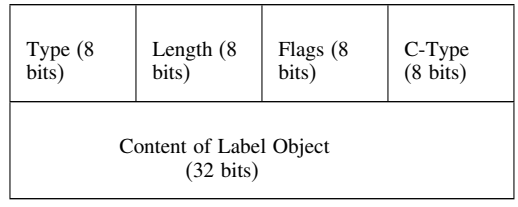


Figure 7.33 RECORD_ROUTE Label sub-object

in both Path and Reservation messages and may complete its action both from the source to the destination and in the opposite direction.

The last new object defined for RSVP-TE is the SESSION_ATTRIBUTE. Class is set to 207. There are two C-Types: LSP_TUNNEL C-Type = 7 and LSP_TUNNEL_RA C-Type = 1, which contains exactly the same content of C-Type 7 together with additional resource affinity information. LSP_TUNNEL_RA details are left to specific literature (e.g. [RFC3209] and references therein) because resource affinity procedure requires a specific treatment that is outside the scope of this book. The format of LSP_TUNNEL C-Type is shown in Figure 7.34.

Set-up Priority ranges from 0 (highest priority) to 7 (lowest priority). It is the priority to get resources and governs the decision of pre-empting other sessions when the session enters the network. Holding Priority has the same range of Set-up Priority. It measures the priority to hold resources after getting them. In practice, it decides if the session may be pre-empted or not by another session entering the network. The Set-up Priority value should not be higher than the Holding Priority value for the specific session that transports the SESSION_ATTIBUTE object. The precise values and corresponding meaning of the flags are left to [RFC3209] for the sake of simplicity. The Name Length measures the display string before padding in bytes, while the Session Name is a null-padded string of characters.

To complete the presentation of RSVP-TE, the definition of the C-Types added to SESSION, SENDER_TEMPLATE and FILTER_SPEC objects is necessary.

The SESSION object adds two C-Types to the original definition in Figures 7.3 and 7.4: LSP_TUNNEL_IPv4, identified through C-Type = 7, and LSP_TUNNEL_IPv6, identified through C-Type = 8. They practically define the LSP within IPv4 and IPv6 scenario so allowing strict QoS requirements through MPLS. The formats that allow reaching the aim are reported in Figures 7.35 and 7.36, respectively for LSP_TUNNEL_IPv4 and LSP_TUNNEL_IPv6. The IPv4 Tunnel End Point Address is the egress node of the LSP tunnel. It is the point where the tunnel finishes. Tunnel IP is a 16-bit identifier of the tunnel, which remains constant over the overall tunnel duration. It is used in the SESSION. Extended ID Tunnel is also an identifier that remains constant over the overall tunnel duration and

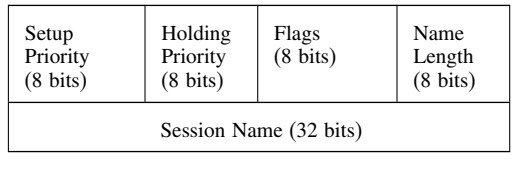


Figure 7.34 SESSION_ATTRIBUTE object: Class = 207, C-Type = 7

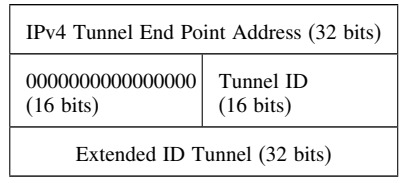


Figure 7.35 LSP_TUNNEL_IPv4 SESSION object: Class = 1, C-Type = 7

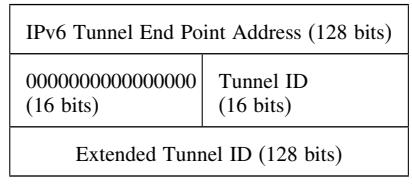


Figure 7.36 LSP_TUNNEL_IPv6 SESSION object: Class = 1, C-Type = 8

it is used in the SESSION. It is typically set to all zeros, but it can be used to identify the ingress node of the tunnel so restricting the SESSION scope to ingress–egress pair. In this case, it contains the IPv4 address of the ingress node. The definitions are the same for LSP_TUNNEL_IPv6 but applied to IPv6 addresses.

FILTER_SPEC (Class = 10) provides two more C-Types in addition to IPv4, IPv6 and IPv6 Flow Label C-Types, shown in Figures 7.5, 7.6 and 7.7, respectively. The new C-Types are defined as LSP_TUNNEL_IPv4 C-Type = 7 and LSP_TUNNEL_IPv6 C-Type = 8. Their formats are shown in Figures 7.37 and 7.38. IPv4 Tunnel Sender Address is the address of the sender node. LSP ID is the label-switched path used in FILTER_SPEC and SENDER_TEMPLATE (defined below).

SENDER_TEMPLATE (Class = 11) provides exactly the same C-Types as FILTER_SPEC: LSP_TUNNEL_IPv4 C-Type = 7 and LSP_TUNNEL_IPv6 C-Type = 8. The format is the same as in Figures 7.37 and 7.38.

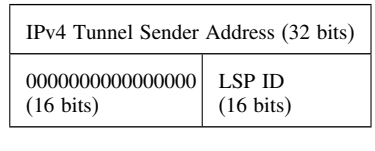


Figure 7.37 LSP_TUNNEL_IPv4 FILTER_SPEC object: Class = 10, C-Type = 7

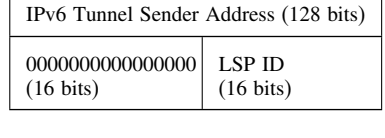


Figure 7.38 LSP_TUNNEL_IPv4 FILTER_SPEC object: Class = 10, C-Type = 8

7.3.3 Control Actions

The described elements of RSVP-TE allow implementing full resource control in MPLS networks. The key idea is to manage specific traffic tunnels as a function of network resources and users' needs. The procedure of building a set of controlled tunnels within an MPLS core by RSVP-TE is summarized.

The operations performed by the ingress LER (the first QoS-PRN along the path) are to store information about traffic and traffic engineering, examine user-defined constraints, calculate the physical path for the LSP, represent path as an explicit route and pass the EXPLICIT_ROUTE object for RSVP-TE signalling. The procedure acts as follows. It starts at the ingress node that performs CAC on the outgoing link with respect to the required bandwidth of the LSP to be established. If the CAC is positive, the ingress node builds the RSVP-TE Path message and forwards it to the next hop. On receipt of the Path message, the mid-point router (intermediate QoS-PRN) performs a CAC check through the control entities defined in the previous chapter. If the CAC is positive, the operation on the Path message is repeated up to the destination.

If the CAC check fails, the mid-point QoS-PRN generates a path error (PathErr) message upstream towards the LER (the first QoS-PRN) to report errors in the Path message processing. If CAC is positive on each intermediate LSR (QoS-PRN), the egress node allocates the incoming label for the tunnel, installs the incoming label, builds the Resv message, inserts the incoming label in the Label object of the Resv message and forwards it upstream. The upstream (mid-point) LSR checks whether the previously allocated bandwidth is still available. This check allows managing concurrency. If the bandwidth check is passed, the mid-point router performs the following tasks: reserves the bandwidth, allocates an incoming label, installs the remotely received outgoing and locally assigned label for the LSP, inserts the locally assigned incoming label in the Label object and forwards the message to the upstream LSR. If the bandwidth check fails, the mid-point router sends a reservation error (ResvErr) message downstream.

Eventually, the Resv message arrives at the ingress LER. On receipt of it, the ingress node checks whether the previously allocated bandwidth is still available. If it is still available, it reserves the bandwidth, installs the remote-received label and informs the control module that the tunnel has been established successfully. Then the ingress node starts forwarding data traffic onto the tunnel.

7.3.4 RSVP-TE and Scalability

RSVP-TE uses connectionless IP. It means that it either refreshes periodically or implements TCP-like mechanisms. The former is, in general, adopted as for RSVP. It means that control messages may be lost and that adjacent nodes may fail without notification. Refreshes help make sure that LSP state is properly synchronized between adjacent nodes. This allows RSVP-TE to pick up changes to the routing scheme automatically. The RSVP-TE network sensitivity from this periodic refresh depends on the sensitivity to failures, which is chosen by configuring the refresh timer. An RSVP-TE Path message will be of the order of 128 bytes, increased of the EXPLICIT_ROUTE object bytes per hop if the explicit route option is used. A Resv message will be of the order of 100 bytes. With 10,000 LSPs (a reasonably high number) and a refresh period of 30 s, this consumes over 600 Kbps of link bandwidth. Whether this is significant depends on the link and on the amount of carried traffic.

To support scalability, RSVP-TE introduces the concept of "summary refresh". RSVP-TE can refresh many LSP states in a single RSVP-TE BUNDLE message: the ability to indicate that nothing has changed on a given Path or Resv rather than having to send the entire normal payload reduces the network refresh flows for RSVP-TE. The RSVP-TE BUNDLE messages have to list the message IDs for each Path or Resv refreshed by the bundle. However, there is a significant difference compared with the number of messages that would travel without this feature. This extension, which requires additional definitions not reported here, effectively transits RSVP-TE from its traditional soft state model to a firm-like state model. In a firm-like state model, refresh messages are still transmitted, but the volume of traffic, the amount of CPU utilization and the response latency are strongly reduced. The Bundle message extension reduces the overall volume of RSVP-TE messages, which must be periodically transmitted and received. It consists of a bundle header followed by a body consisting of a variable number of standard RSVP-TE messages (e.g. Path, PathErr, Resv, ResvTear, ResvErr, etc.). It is used to aggregate multiple RSVP-TE messages within a single protocol data unit (PDU).

7.3.5 Remarks

In conclusion, RSVP_TE provides not only proper formats to assign and manage MPLS labels but also important concepts such as the abstract nodes, which match the needs of heterogeneous QoS interworking. In addition to RSVP, which already included very interesting features such as FILTER_SPEC and SENDER_TEMPLATE IPv6 Flow Label objects, RSVP-TE opens the door to future applications within heterogeneous QoS interworking based on IPv6. Obviously, it needs further study and possibly additional features but it is not a utopian design. The NSIS (Next Steps in Signalling) environment represents a practical future framework where it is possible to design future QoS supporting signalling.

7.4 NSIS QoS Signalling

7.4.1 Requirements and Application Scenarios

NSIS is not just a protocol, but it is a working group that evaluates requirements and protocols for signalling information about a data flow along a network path. The architecture and design goals of NSIS signalling action are listed in [RFC3726]. The reference environment is a heterogeneous network environment. Signalling traverse multiple network portions, exactly as done in this book. This is the motivation because NSIS is so important. The reference network environment and the signalling protocol problem perfectly match the general needs of a QoS-based heterogeneous network. There is no standard for now but only good indications that can be used for future research in the field. In summary, directly from [RFC3726], NSIS should, concerning architecture, signalling messages and control information

– provide information about the network resource state on demand without reserving any resource;
– adapt to any kind of network and be open to future network scenarios;

– be decoupled from transported control information and from particular control schemes for QoS satisfaction;
– be able to traverse opaque objects;
– work over multiple scenarios such as host to host, host to network, network to network, edge to edge, i.e. over the single components of a heterogeneous network;
– work over path-coupled signalling modes (where signalling is routed through nodes belonging to the data path) and path-decoupled signalling modes (where signalling is not routed on the data path);
– hide the internal technology and the topology of a particular network portion;
– traverse specific path segments transparently without any required interpretation of the signalling mechanism;
– allow erasing states along the path when they are no longer necessary and releasing states after failures;
– allow sending notifications among NSIS entities about recoverable errors, unrecoverable errors, service degradation, repair indication and available service upgrade;
– allow sending notifications about set-up state establishment and refusal, possibly providing motivations for refusal;
– allow information exchange within local domain controllers (e.g. the local control modules of the single network portions connected through the heterogeneous network) and allow adding and removing information related to local domains;
– allow modification of the parameters contained in the signalling messages;
– allow addressing protocol states independently of the flow identification;
– allow the modification of already established states as, for example, to upgrade and downgrade resource use, without interrupting the service;
– send just one signalling message for a group of micro-flows (i.e. an aggregate of flows).

Concerning the performance to be reached, it would be recommendable that NSIS

– be scalable;
– allow low latency set-up time to adapt to most challenging applications that require immediate human or machine interaction as well as voice and command and control;
– exploit low bandwidth availability so as to adapt to environments where bandwidth is a scarce and precious resource;
– allow to constraint the load on devices where it is needed (i.e. on QoS-PRNs);
– optimize or, at least, increase network utilization so as to minimize bandwidth waste.

Moreover, NSIS should include the capability to select and change the level of aggregation in dependence of the flow aggregation power and of the customer and provider needs and decisions. This requirement concerns the flexibility of the protocol as well as the requirement about the need to allow unidirectional and bidirectional state set-up. Other requirements contained in [RFC3726] concern the following:

- *Security and authentication*: authentication and authorization of signalling requests, protection of signalling messages against unauthorized modifications, channel security between signalling NSIS entities, prevention of denial-of-service attacks, confidentiality of signalling messages, state ownership;

- *Mobility*: quick and efficient re-establishment of service after handover;
- *Interworking with other protocols*: support and interaction with IP and IPSec tunnels, full interaction with both IPv4 and IPv6, independency of pricing models and charging infrastructures, full interaction with traditional routing protocols;
- *Operation behaviours*: capability to assign a given QoS level to signalling messages, graceful failure and problem handling of NSIS entities with respect to other systems.

The details are left for a deeper investigation.

To better state the application field of NSIS working group, the reading of [RFC3726] is strongly recommended not only to have complete information of QoS signalling, but also to get some technological examples about QoS provision over heterogeneous scenarios including wireless networks, third generation (3G) wireless access, wired portions of wireless networks. The reference network is always the one reported in Figure 2.3, where by chance, the origin of the information is a mobile terminal (possibly a mobile sensor). Terminal mobility is a very important issue concerning QoS provision over heterogeneous networks [RFC3726]. Hand-off is the main issue, in particular, not only for voice calls, but also for data. It may be requested by QoS management for performance reasons, but it may also be forced by mobility management. It may have a heavy impact on QoS because, typically, it may generate packet losses caused by the delay of admission requests and also call blocking due to mobile network load. Moreover, wireless networks deserve special QoS solutions and provide their own technologies for QoS management. They need also to map higher layer QoS requirements over the radio bearer. Signalling protocol must also consider these important aspects and, in particular, allow the dynamic renegotiation of resources when the mobile host enters a new wireless network. Concerning these issues, [RFC3726] reports a specific example, referred to 3G communication, which is particularly meaningful for the topic of this book. It is shown in Figure 7.39. Matching the special case reported with the general scheme shown in Figure 6.37 is important to have a deeper view of the problems and of the application environment of NSIS signalling protocols. The control blocks and functions that appear in Figure 6.37 are mapped over a real wireless access architecture following the view of the author. Alternatives are also possible. The Policy Decision Function block has all per-flow and per-aggregate (per-class) information. It might also act as NSIS entity even if, in Figure 7.39, the CAC functions have been located separately. The comparison with Figure 6.37 is also important because it states possible alternatives to the proposal.

[RFC3726] allows highlighting two topical problems that NSIS signalling must match: the QoS reservation and negotiation from the access to the core network and the QoS reservation and negotiation traversing administrative boundaries (i.e. different Autonomous Systems). Figure 2.3 and, in particular, Figure 5.7 match the two issues graphically.

Concerning the former, the following are the main issues: the reservation object is probably an aggregate of flows over the core; the timescales of states are different over access and core networks; the dimension of the aggregate needs often to be changed and, in consequence, also the traffic specification. If QoS signalling traverses the border between different Autonomous Systems, it is the case of inter-domain signalling. Additionally to the previous case, possible problems are related to the competition among administrative domains, as, for instance, the exchange of information about topology and technology implemented within the network portion. This issue is considered also in the vertical mapping part of this book, where

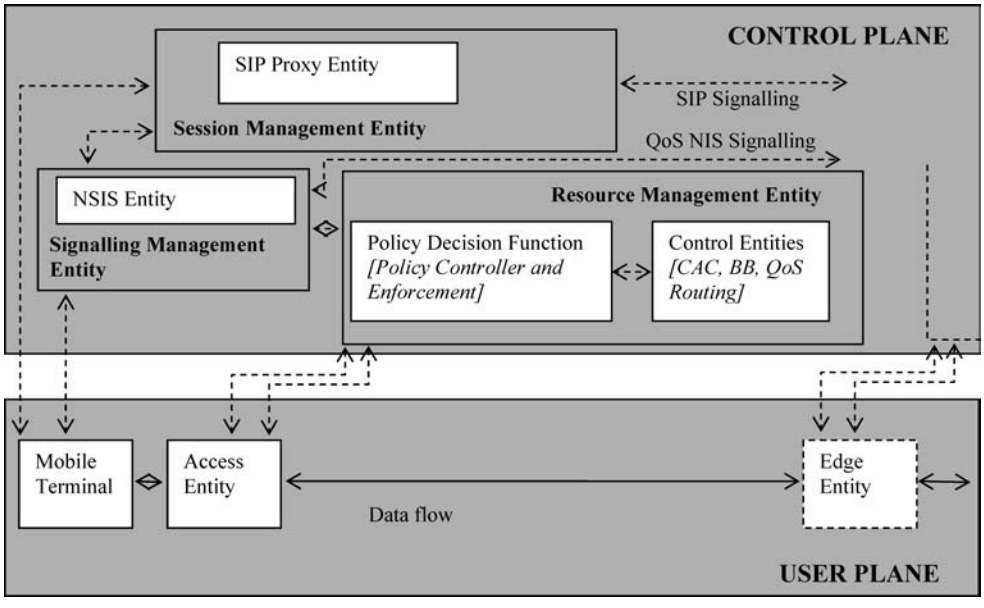


Figure 7.39 Wireless access within heterogeneous environment

possible common interfaces are defined to bypass this problem. Other possible issues are authorization and accountability. They should be solved at SLA level.

The last example is reported (again from [RFC3726]) to better identify the application scenario in strict connection with the reference environment of this book. Figure 7.40 contains the connection of a telephone gateway (Plain telephone network or ISDN) with a backbone QoS network. It allows also introducing the concept of bandwidth pipe, which will be used at the end of the book in Chapters 10 and 11. The heterogeneous network shown in Figure 7.40 must transport phone and data calls. The attention is focused on voice here. The voice quality suffers due to packet loss, latency and jitter, as shown in Chapter 1. The QoS signalling protocol must identify and admit a flow also minimizing these impairments. A telephone gateway may handle thousands of calls simultaneously and, in this case, scalability is a serious issue as well as QoS assurance. Concerning the latter, as stated in Chapter 4, there are many management functions that need to be implemented. Flow identification is of main importance in this case because, as typically done in telephone networks, per call (per user flow) admission control and bandwidth assignation would be needed. This tackles the scalability issue. The solution may just be in the new MPLS-centric and, in particular, IP6-centric QoS architectures proposed in Chapter 6. The aim is just to bring the phone call on the other side of the network providing the same service of the telephone network. It means to offer a specific bandwidth pipe from the source to the destination and, in particular, over the backbone network independently of the used technology. To signal the need of the phone call and to transmit the bandwidth need and control information is the duty of the NSIS signalling protocol, in strict cooperation with SS7 telephone signalling.

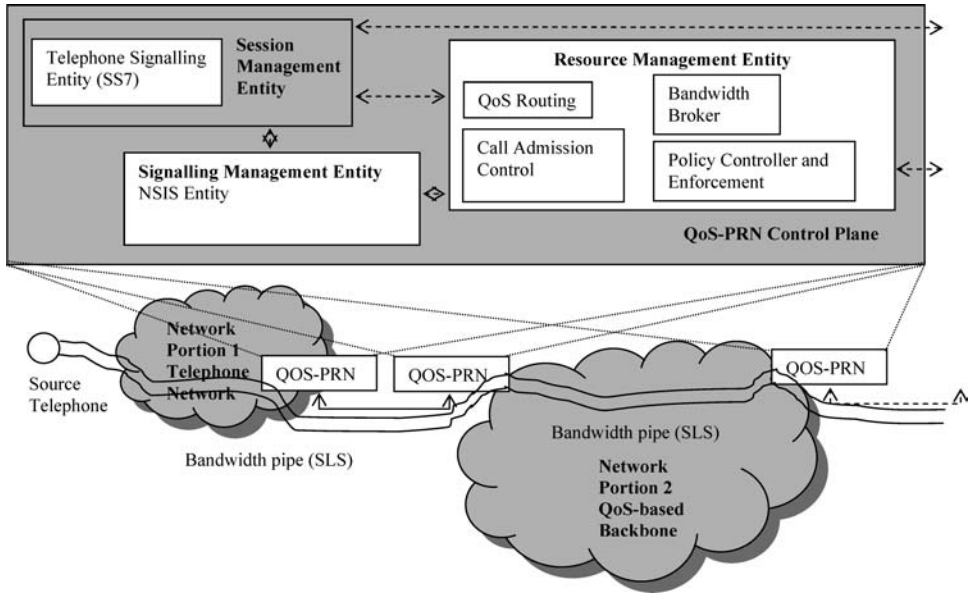


Figure 7.40 Telephone network and QoS-based backbone

7.4.2 NSIS Structure

NSIS aim is similar to RSVP one but it is intended to wider scenarios. Its purpose goes over the QoS signalling used here. Its structure is fully described in [RFC4080] from which most of the material of this paragraph is taken. The main components of NSIS signalling are NSIS Entity, which implements NSIS protocol; NSIS Signalling Layer Protocol (NSLP), which is the component of the NSIS protocol, which supports specific signalling applications; NSIS Transport Layer Protocol (NTLP), which is the component of the NSIS protocol, which supports lower layer application independent of signalling functions. Two more definitions are important. The application layer flow that deserves installation, monitoring, change and removal of states along the network is defined in NSIS session (called simply session in the remaining part of this paragraph as done in [RFC4080] for the sake of generality). In practice, it is the application that requires a specific degree of service. QoS management, SLS transmission and control information exchange (i.e. the aim of NSIS signalling) are defined as NSIS signalling application (and called signalling application in the remaining part of this paragraph, as done in [RFC4080], for the sake of generality). NSIS Entities communicate through peers. It means that they are connected but not necessarily through a direct IP hop. They can communicate through abstract nodes, whose meaning of "group of nodes" has been clarified in the paragraph dedicated to RSVP-TE. The object QoS-PRN and the two examples reported in Figures 7.39 and 7.40 are ideal to focus on the concept of "NSIS domain". It may be intended as a group of nodes, and also a group of networks not necessarily IP based. In means multiple concatenated heterogeneous network portions with trust relations between them. It perfectly matches QoS-PRNs intervention area, as shown in the previous chapters, and the basic assumption of this book, where a network is a sequence of Autonomous Systems.

As clearly stated in [RFC4080], the distinguished features of NSIS protocol are that

- it is related to specific flows and not to general network operations;
- it involves the nodes of the network and it does not traverse them transparently.

In consequence, it means that per-flow information needs to be transported. This information is defined as "control state" in the network. NSIS protocol can install, change, update, read and cancel the control state of a node of the network for a particular data flow, obviously including also a simpler per-class identification granularity.

NSIS protocol suite is structured into two different layers:

1. NSIS Transport Layer Protocol (NTLP), independent of the signalling applications, responsible for peer-to-peer signalling messages delivery. Concerning protocol functional stack, it is located just above the network layer. It is designed as located just over IP in [RFC4080].
2. NSIS Signalling Layer Protocol (NSLP), which refers to signalling application layer protocols, and defines the message formats and other information related to application layer needs. The QoS signalling, which is treated in this chapter, is a signalling layer protocol.

The hierarchy between NSLP and NTLP is shown in Figure 7.41.

Due to the layered structure of the NSIS suite, some intermediate NSIS entities are authorized to bypass the NSLP action. The functional layered separation of NTLP and NSLP makes the signalling protocol more scalable and adaptive to different network features. A possible concatenation of NSIS entities is reported in Figure 7.42. The action at intermediate NSIS nodes is reduced at a minimum processing. The following may be an example: NSLP may impose a per-flow traffic distinction, while NTLP may implement a coarser traffic aggregation acting transparently to the NSLP layer. A formalization of it, even if not explicitly mentioning NSLP and NTLP, is given in the next chapter when the vertical QoS mapping problem is described exactly as a model of queues in cascade. One of the described problems is just to pass from a granularity level for traffic distinction to a coarser one. NTLP entities may need to group flows together and to exchange signalling information for the overall aggregate, instead of for each single flow. As seen in Chapter 3, there are many ways to aggregate traffic. DSCP (DiffServ Code Point) aggregation is of special importance

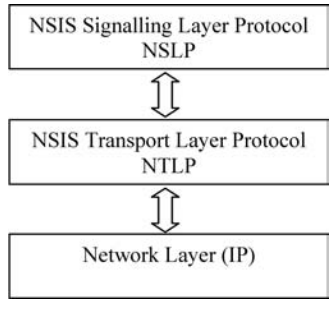


Figure 7.41 NSLP and NTLP protocol stack

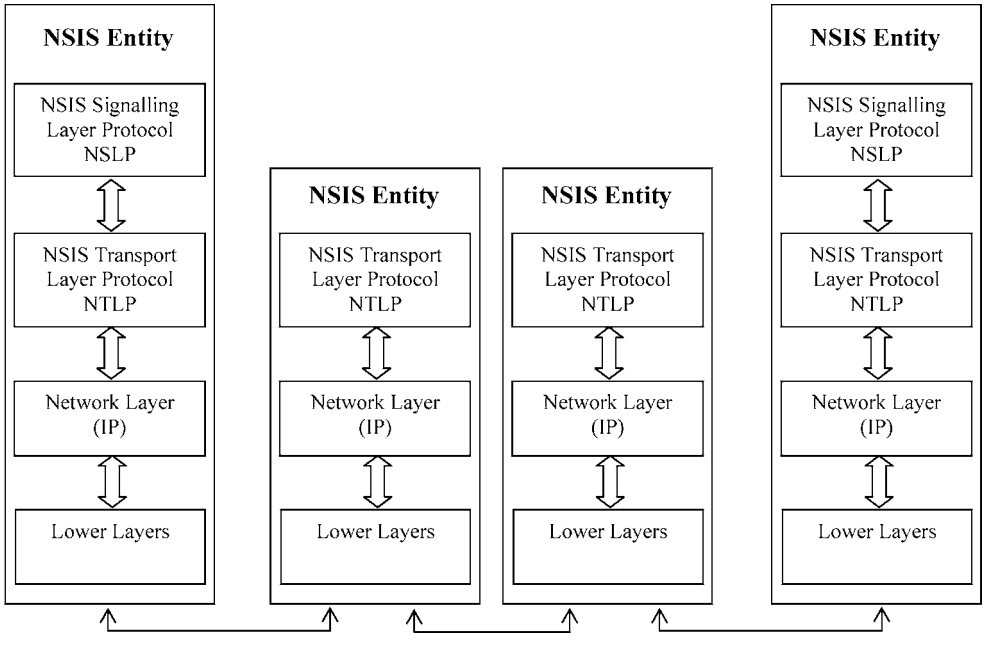


Figure 7.42 NSIS Entities concatenation example

in this case: NSIS protocol stack is also suited for DiffServ transport and may be applied to DiffServ-IP-centric QoS architectures.

More generally speaking, flow identification has the aim to identify traffic flows uniquely and all the packets belonging to a specific flow have the same identifier, which defines the treatment that the packet receives in the network. Within the NSIS suite, the flow identification is not hidden at NSLP layer but it is an explicit part of the NTLP block. It allows managing traffic flows independently of the original application. This feature adds great flexibility to the signalling structure. In detail, [RFC4080] specifies the following flow identifiers, perfectly in line with the philosophy of the book:

1. Source, destination, protocol identifier and higher layer addressing (port address), as done for IntServ
2. Flow Label, as for IPv6
3. DSCP field, as in DiffServ paradigm.

Flow label identification may be applied also for MPLS so completing all the flow identification paradigms identified for the technology-centric QoS architectures of Chapter 6 and enlarging the application scope of NSIS. [RFC4080] includes also a specific identifier for IPsec environment, which is outside the scope of this book.

Additionally, it seems important also to allow the identification of the signalling application independently of the flow identifier. [RFC4080] suggests using a session identifier for that. Even if it is an identifier related to the application, it is assumed to be available also at

NTLP layer. It allows more flexibility and detailed control interventions within NTLP. More details can be found in [RFC4080].

Without entering the details of the NTLP protocol, it is important to say that, due to the aim of the layer, acknowledgement mechanisms are probably needed. In this view, if existing protocols such as TCP are available, they could be included within NTLP functionalities, being NTLP in the same transport position within the functional protocol architecture.

More specifically, QoS signalling, it is necessary to remind, is only an aspect of NSIS structure, even if it is of main importance for the topic treated in this book: NSIS entities may be functionally structured into three components. They are not necessarily separate hardware components and the different functions may be conveyed within just one object. On the other hand, conceptual separation is helpful to better understand

- QoS NSIS Initiator (QNI), which makes the resource request;
- QoS NSIS Responder (QNR), which acts as endpoint of the signalling action;
- QoS NSIS Forwarder (QNF), which is the intermediate entity that forwards NSIS signalling along the NSIS path.

All the entities defined above and synthetically indicated as NSIS Entity in Figures 7.39 and 7.40 interact with the Resource Management Entity, which is in charge of resource management and control schemes. The formal communication interface between the two entities is implemented through the SLS or, if other considerations such as pricing are also involved, through the SLA. Also physical co-location of Resource Management and Signalling Entity is possible. QNI and QNR must contain both NTLP and NSLP. QNF must contain NTLP and may contain NSLP. Concerning NSLP within QoS NSIS Entity, [RFC4080] proposes a possible set of basic messages. It is reported in Table 7.8. The message direction identifies only the course taken by the messages. It means that the real path could also be shorter as, for example, QNI to QNF.

NSIS is a very powerful framework and may be applied to any QoS-oriented environment.

Table 7.8 Possible set of basic messages for QoS NSIS Entities

Message name	Message direction	Message function
Request	QNI to QNF to QNR	To create a new reservation for a specific flow or group of flows
Modify	QNI to QNF to QNR	To modify a reservation already established
Release	QNI to QNF to QNR and QNR to QNF to QNI	To release a reservation already established
Accept	QNR to QNF to QNI	To confirm the acceptance of a reservation request and/or of a modification
Reject	QNR to QNF to QNI	To reject the acceptance of a reservation request
Notify	QNI to QNF to QNR and QNR to QNF to QNI	To report events happened in the network
Refresh	QNI to QNF to QNR	To implement state management

7.5 Q–BGP (QoS-enhanced–Border Gateway Protocol)

7.5.1 Introduction to BGP

Border Gateway Protocol is not a QoS signalling protocol. It is a routing protocol, but its presence in this section is due to an evolution of BGP, the Q-BGP (QoS-enhanced–BGP) [Boucadair05], which gives some information about the reachability of the destinations with a given QoS. Even Q-BGP is a routing protocol and not a QoS signalling one. It is briefly summarized here because, within a loose QoS provision, as said in Chapter 6, it can be of great help.

Routing protocols may be classified as in Table 7.9 [Forouzan].

Intradomain routing concerns the path of packets within a specific network portion. Interdomain routing regards the path composed of network portions and it is the decision taken within QoS-PRNs. Distance vector considers the least cost route between two nodes as the route with minimum distance and keeps a vector of minimum distance in every node to take decisions. RIP (Routing Information Protocol) is a simple protocol based on the distance vector algorithm, where the distance is simply defined as the number of links that are used to reach the destination. Link state routing is based on the Dijkstra's algorithm. Each node knows the overall network portion topology including the list of nodes and links, the connection type, the cost and the link condition (up or down). Each node builds its own routing table. Open Shortest Path First (OSPF) protocol is part of the link state family.

Path vector routing is an inter-domain protocol. There is one (or more) node(s) in each network portion (Autonomous System), which acts as representative of the AS. It is called "speaker" node. It creates a routing table and advertises it to the other speaker nodes of the neighbour ASs. The routing table is composed of paths. In other words, the speaker node advertises the ASs that must be traversed to reach a specific destination in its AS and in other ASs. Figure 7.43 contains a simplified version of the AS network used in Chapter 2 to define an Autonomous System. Only one speaker node for AS and a more limited number of nodes within each AS are supposed here. The speaker nodes (i.e. the QoS-PRNs) are identified through the letter R and the AS number. For example, the speaker node of AS1 is called R1. Each device (e.g. router) within each AS is identified through the letter R, the AS number and a sequential identifier. For instance, internal devices to AS1 are called R11, R12 and R13.

Initially, each speaker node has information only about the reachability of nodes inside its own AS. The initial tables for R1, R2, R3 and R4 are reported, respectively, in Tables 7.10, 7.11, 7.12 and 7.13.

Each speaker node shares its table with the neighbour nodes and, after a stabilization period, each speaker has a table which contains complete information on how to reach destinations in other ASs. The related complete tables for all the speaker nodes in Figure 7.43 are reported in Tables 7.14–7.17. The selection of the best route depends on the performed

Table 7.9 Classification of routing protocols

Intradomain routing protocols	Interdomain routing protocols
Distance vector (RIP) Link state (OSPF)	Path vector (BGP)

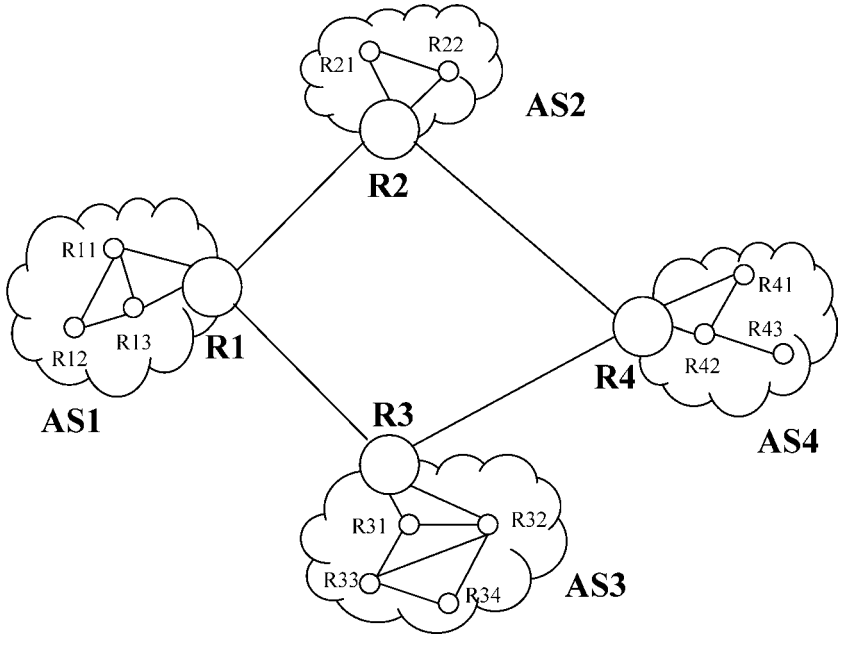


Figure 7.43 Heterogeneous network as a composition of Autonomous Systems

Table 7.10 Initial routing table for speaker node R1

Destination	Path to reach the destination
R1	AS1
R11	AS1
R12	AS1
R13	AS1

Table 7.11 Initial routing table for speaker node R2

Destination	Path to reach the destination
R2	AS2
R21	AS2
R22	AS2

choice. The smaller number of traversed autonomous systems may be a criterion and also QoS, security and reliability may be applied.

BGP, whose complete presentation is contained in [RFC4271], is an inter-domain protocol that uses path vector routing. It applies the routing tables presented above but, instead of using, as destinations, a list of Autonomous Systems, exploits a list of attributes, each of

Table 7.12 Initial routing table for speaker node R3

Destination	Path to reach the destination
R3	AS3
R31	AS3
R32	AS3
R33	AS3
R34	AS3

Table 7.13 Initial routing table for speaker node R4

Destination	Path to reach the destination
R4	AS4
R41	AS4
R42	AS4
R43	AS4

Table 7.14 Stabilized routing table for speaker node R1

Destination	Path to reach the destination
R1	AS1
R11	AS1
R12	AS1
R13	AS1
R2	AS1–AS2
R21	AS1–AS2
R22	AS1–AS2
R3	AS1–AS3
R31	AS1–AS3
R32	AS1–AS3
R33	AS1–AS3
R34	AS1–AS3
R4	AS1–AS2–AS4
	AS1–AS3–AS4
R41	AS1–AS2–AS4
	AS1–AS3–AS4
R42	AS1–AS2–AS4
	AS1–AS3–AS4
R43	AS1–AS2–AS4
	AS1–AS3–AS4

Table 7.15 Stabilized routing table for speaker node R2

Destination	Path to reach the destination
R1	AS2–AS1
R11	AS2–AS1
R12	AS2–AS1
R13	AS2–AS1
R2	AS2
R21	AS2
R22	AS2
R3	AS2–AS1–AS3
	AS2–AS4–AS3
R31	AS2–AS1–AS3
	AS2–AS4–AS3
R32	AS2–AS1–AS3
	AS2–AS4–AS3
R33	AS2–AS1–AS3
	AS2–AS4–AS3
R34	AS2–AS1–AS3
	AS2–AS4–AS3
R4	AS2–AS4
R41	AS2–AS4
R42	AS2–AS4
R43	AS2–AS4

Table 7.16 Stabilized routing table for speaker node R3

Destination	Path to reach the destination
R1	AS3–AS1
R11	AS3–AS1
R12	AS3–AS1
R13	AS3–AS1
R2	AS3–AS1–AS2
	AS3–AS4–AS2
R21	AS3–AS1–AS2
	AS3–AS4–AS2
R22	AS3–AS1–AS2
	AS3–AS4–AS2
R3	AS3
R31	AS3
R32	AS3
R33	AS3
R34	AS3
R4	AS3–AS4
R41	AS3–AS4
R42	AS3–AS4
R43	AS3–AS4

Table 7.17 Stabilized routing table for speaker node R4

Destination	Path to reach the destination
R1	AS4–AS2–AS1
	AS4–AS3–AS1
R11	AS4–AS2–AS1
	AS4–AS3–AS1
R12	AS4–AS2–AS1
	AS4–AS3–AS1
R13	AS4–AS2–AS1
	AS4–AS3–AS1
R2	AS4–AS2
R21	AS4–AS2
R22	AS4–AS2
R3	AS4–AS3
R31	AS4–AS3
R32	AS4–AS3
R33	AS4–AS3
R34	AS4–AS3
R4	AS4
R41	AS4
R42	AS4
R43	AS4

them giving information about the path. The list of attributes helps choose the best route based on the policy chosen by the QoS-PRNs. BGP messages are sent over TCP connections and messages are processed when they are entirely received. The maximum message size is 4096 bytes. The smallest message that may be sent is the pure BGP header whose length is 19 bytes.

7.5.2 BGP Message Formats

The Message Header format is contained in Figure 7.44. The field Marker has only a compatibility aim and it is set to all 1. The field Length contains the total length of the message, including the header. It is computed in bytes. Type shows the type code of the message. [RFC4271] defines four message types: Open, Update, Keepalive and Notification. The relative Type codes are: 1 – "00000001" for Open, 2 – "00000010" for Update, 3 – "00000011" for Keepalive and 4 – "00000100" for Notification.

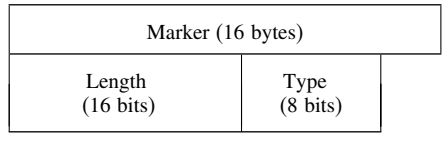


Figure 7.44 BGP message header

Version (8 bits)	My Autonomous System (16 bits)	Hold time (16 bits)
Hold time [continued]	BGP identifier (32 bits)	
BGP identifier [continued]	Optional Parameters Length (8 bits)	Optional Parameters (variable length)

Figure 7.45 Open message format

All BGP messages contain the BGP message header at the beginning of the message. The message will follow in strict sequence after the last bit of the Type field. To create a relationship with the neighbour, a network device running BGP opens a TCP connection and sends an Open message. If the neighbour accepts, it sends a Keepalive message.

Figure 7.45 shows the Open message format. The field Version indicates the protocol version number of the message. The current BGP version number is 4. The field My Autonomous System contains the Autonomous System number of the sender of the Open message. Hold Time (16 bits) indicates the number of seconds, which the sender proposes for the value of the Hold Timer. The value represents the maximum number of seconds, which may elapse between the reception of successive "Keepalive" and "Update" messages from the sender. More details may be found in [RFC4271]. The BGP Identifier (32 bits) is the BGP Identifier of the sender. The value is set by a given BGP speaker to the IP address assigned to that BGP speaker. The field Optional Parameters Length contains the length in bytes of the following Optional Parameters field, which contains a list of optional parameters whose format is reported in [RFC4271].

The Keepalive message is composed only by the BGP header message. These messages are constantly exchanged among network devices running BGP.

A Notification message is sent when an error condition is detected. The BGP connection is closed immediately after it is sent. It contains the following fields: Error code (8 bits), to identify the error type; Error sub-code (8 bits), to provide more precise information about the nature of the reported error; and Data, whose length is variable, to have a diagnosis of the notification reason. Details about error codes and sub-codes may be found in [RFC4271]. Their analysis is out of the scope of this book.

The most important message, at least concerning the topic of this book, is the Update message. It is used by a network device running BGP to announce a route to new destinations and to withdraw destinations that have been advertised previously. In short, it is the message that contains the tables with the reachability of destinations shown in the previous paragraph together with their related attributes. The format of the Update message is shown in Figure 7.46. Not all fields are contained in each Update message.

The field Withdrawn Routes Length contains the length of the following Withdrawn Routes in bytes. If it is set to "0", it means that no routes have been withdrawn and that there is no Withdrawn Routes field in the Update message. Withdrawn Routes contains a list of IP address prefixes of the withdrawn routes. Each IP address prefix is encoded in the form "Length (8 bits), Prefix (variable length)". The Total Path Attribute Length reports the length of the Path Attributes field in bytes. A "0" value means that neither the Network Layer

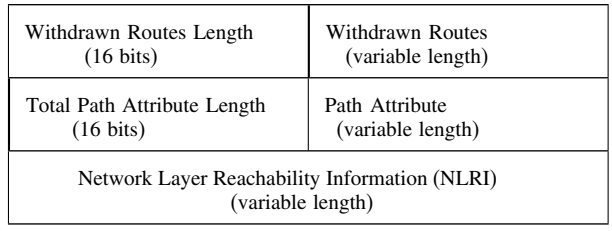


Figure 7.46 Update message format

Reachability Information (NLRI) field nor the Path Attribute field is present in the "Update" message. The field Path Attributes is essential for Update messages. Path Attributes are classified into four categories: Well-known mandatory, Well-known discretionary, Optional transitive and Optional non-transitive. All BGP implementations must recognize well-known attributes. Among them, the mandatory attributes must be included in every Update message which contains NLRI. Discretionary attributes may or may not be included in Update messages. In addition to well-known attributes, each path may have one or more optional attributes. Well-known mandatory attributes include the following:

ORIGIN, which is generated by the speaker that originates the associated routing information.
AS_PATH, which identifies the autonomous systems traversed by the routing information carried in the Update message.
NEXT_HOP, which defines the IP address of the router, which should be used as the next hop to the destinations listed in the Update message. The details of BGP are out of scope of this book and can be found in [RFC4271], but some more information is reported below.

Concerning the format, each single attribute is described through a triple "Attribute Type, Attribute Length, Attribute Value", which has variable length. The format is reported in Figure 7.47.

Attribute Type is 16 bits. It is composed of 8 bits called Attribute Flags (Flags, in Figure 7.47) and 8 bits that define the Attribute Type Code (Type Code, in Figure 7.47). The first bit (bit 0) of the Attribute Flags is the higher-order bit called "Optional" bit and defines if the attribute is optional (Optional bit set to 1). If the Optional bit is set to 0, it means that the attribute is defined as well known. The second higher-order bit of the Attribute Flags is called "Transitive" bit and defines if an optional attribute is transitive (Transitive bit set to 1) or non-transitive (Transitive bit set to 0). The Transitive bit is set to 1 for well-known attributes. The third bit is called "Partial" bit and defines if the information contained in the optional transitive attribute is partial (Partial bit set to 1) or complete (Partial bit set to 0). Well-known optional non-transitive attributes have the Partial bit set to 0.

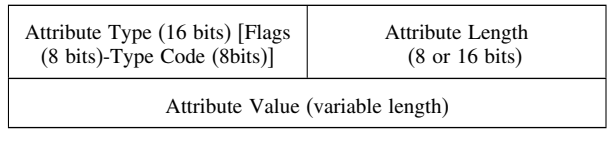


Figure 7.47 Path Attribute field format

The fourth bit is the Extended Length bit. It states whether the Attribute Length field is 1 (bit to 0) or 2 bytes (bit to 1). The other low order 4 bits are not used. The Attribute Type Code identifies precisely the attribute. The Attribute Length field contains the length of the following Attribute Value. The Attribute Value specifies more information about the attribute. For example, the following are the well-known mandatory attributes:

ORIGIN: It is identified by Type Code = 1 – "00000001". Attribute Value may assume three values (binary coded): 0, 1 and 2. They mean, respectively, NLRI is interior to the originating AS; NLRI is exterior to the originating AS and is learnt by a particular protocol, the EGP – Exterior Gateway Protocol defined in [RFC904]; NLRI is exterior to the originating AS and is learnt through different protocols.

AS_PATH: It is identified by Type Code = 2 – "00000010". It is composed of a sequence of AS path segments and each of them is described as a triple "path segment type (8 bits), path segment length (8 bits), path segment value (16 bits)". The path segment type may assume the following binary coded values: 1 for AS_SET, which represents an unordered set of traversed Ass; and 2 for AS_SEQUENCE, which represents an ordered set of traversed Ass. The path segment length contains the number of ASs in the path segment value field. The path segment value field contains the AS numbers, which identify the traversed ASs, whose type has been defined in the path segment type and whose number in the path segment value.

NEXT_HOP: It is identified by Type Code = 3 – "00000011". Its function has been defined above.

There are other attributes whose definition (in [RFC4271]) is outside the scope of this book.

The last field contained in the format of Path Attribute is called NLRI, whose length may be computed through the following formula:

$$\text{NLRI Length} = \text{UPDATE message Length-23-Total Path Attributes Length} - \text{Withdrawn Routes Length}$$

Update message Length is the value encoded in the fixed-size BGP header. Total Path Attributes Length and Withdrawn Routes Length are the values encoded in the variable part of the UPDATE message. The number "23" is a combined length of the fixed-size BGP header, the Total Path Attributes Length field and the Withdrawn Routes Length field.

NLRI field contains the list of one or more destinations, essentially, as listed in the routing tables of section 7.5.1. The used format to describe each destination within the NLRI field is reported in Figure 7.48.

The Prefix Length contains the length of the Prefix field in bits. If the Prefix Length is all zeros, it means all IP addresses. In this case also, the Prefix is set to zero. The Prefix field is

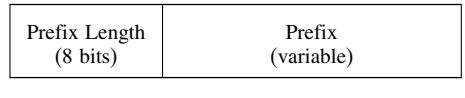


Figure 7.48 Format of each destination within the NLRI field

an IP address prefix, which represents an IP sub-net. An UPDATE message advertises only one set of path attributes that can be shared by multiple destinations, contained in the NLRI field. All the attributes reported in one specific UPDATE message apply to all destinations contained in the NLRI field of that UPDATE message.

7.5.3 Additional Information Carried by Q-BGP

The general idea of this BGP extension [Boucadair05] is to provide, together with the Path Attributes, also the possible QoS that can be offered if a specific destination is reached through that path.

Two service capability groups are defined. The first group requires propagating only an identifier that has been agreed between adjacent peers. This identifier could be the DSCP value used to signal a QoS class to deliver information between ASs. The second group requires the propagation of a set of QoS performance features associated with an identifier.

The carried QoS performance characteristics could be taken into account by the Q-BGP routing selection process to select an optimal path in terms of available QoS. This would enable to tune the route selection process in order to select routes according to more sophisticated routing policies (e.g. route with highest available rate and lower delay).

A new parameter is included in the optional parameters of the OPEN message of Q-BGP sessions to indicate that the specific BGP device supports the attributes related to Q-BGP. In order to indicate that a given inter-domain QoS delivery solution belongs to a given group (either Group 1 or Group 2), a new parameter called "QoS Service Capability" is introduced. The composition is reported in Figure 7.49. The first octet set to all 1's indicates that the offered inter-domain delivery announcements are based on Group 1 classification. The second octet set to all 1's indicates a Group 2 classification.

The set of attributes included in Path Attribute adds two new types (in draft version): a QoS_NLRI attribute for Group 1 and a QoS_NLRI attribute for Group 2. The former is composed of a simple field of 8 bits called QoS Class Identifier, shown in Figure 7.50. It contains the DSCP code. The latter is more complex and is reported in Figure 7.51. The QoS Information Length field carries the number of the QoS Information Code, which will be sent by the Q-BGP speaker in a single Q-BGP UPDATE message. The QoS Information Code field identifies the type of QoS information in terms of 0 for reserved, 1 for packet rate, 2 for one-way delay metric and 3 for inter-packet delay variation. The QoS information subcode field carries the sub-type of the QoS information. The sub-types are 0 for none,

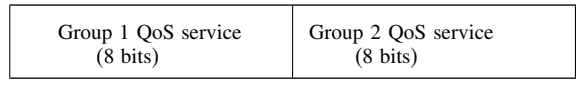


Figure 7.49 QoS Service Capability attributes

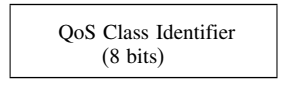


Figure 7.50 QoS_NLRI attribute for Group-1

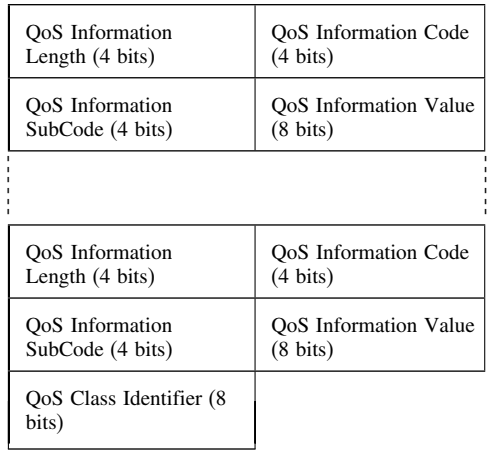


Figure 7.51 QoS_NLRI attribute for Group-2

1 for reserved rate, 2 for available rate, 3 for loss rate, 4 for minimum one-way delay, 5 for maximum one-way delay and 6 for average one-way delay. The QoS Information Value field indicates the value of the QoS information. The corresponding measure units depend on the instantiation of the QoS information code. The QoS class identifier field indicates, again, the identifier as DSCP.

An important issue concerns the triggering period of Q-BGP messages. In practice, two solutions are possible. The first one is based on the administratively enforcement of some static trunks to differentiate the QoS among the classes. Over-provision may also be applied. In this case, the QoS information does not change too frequently. In the second solution, the QoS implementation can be much more dynamic, based on signalling protocols such as RSVP and possibly using active measurements and/or probing mechanisms. The frequency of changes could be much higher in the dynamic case.

7.6 Final Remarks Concerning Signalling

Possible applications of the presented signalling schemes over the QoS architectures of Chapter 6 have been evidenced in the text. Table 6.2 already contained a first indication of it. Anyway, some more comments may be added now, after the presentation of signalling scenarios and algorithms. Obviously, there are many other signalling schemes that have not been considered here. Reference [Vali2004], for example, contains a list of inter-domain signalling protocols, which evidences also the reservation granularity characteristics (per-flow, per-aggregate) and the architecture where to apply them. Nevertheless, the aim of this book is not to enter the details of the different signalling protocols in the market, but only to evidence the need of signalling to provide QoS and to indicate a possible path for its design. Many design details have been reported for RSVP and RSVP-TE because, in the view of the author of this book, they represent a good example of how a signalling protocol should be designed, in strict dependence of the application environment. Much attention has been also dedicated to NSIS workgroup that represents the future of signalling architectures

and protocols. Q-BGP has been presented because it represents a first step towards full QoS management. It can be applied to each QoS architecture presented, if a loose QoS guarantee is the aim.

In summary, for each specific technology-centric QoS architecture in Chapter 6, a signalling solution may be individuated, but this must be seen as a challenge. The problem is still open, not only from the implementation viewpoint but also for the research. The following may be considered a starting point to find new efficient solutions.

- IP-centric QoS architecture
  - IntServ-IP-centric QoS architecture: RSVP solution is perfectly suited for this QoS architecture.
  - DiffServ-IP-centric QoS architecture
  - No control: No signalling at all; a minimum QoS may be got through bandwidth over-provision.
  - Static trunks: Only loose QoS may be provided. No signalling is required and QoS may be obtained through over-provision. Nevertheless, a first step towards the respect of QoS requirements can be done by using Q-BGP, which allows individuating the quality through which destinations may be reached.
  - DiffServ-Pre Congestion Notification (DiffServ-PCN): This solution has been designed having loose QoS in mind. Also in this case, a combination of Q-BGP and over-provision may help provide some guarantees. Additionally, a light version of RSVP may also be used as a step forward towards QoS provision.
  - Bandwidth Broker (BB): Even if without distinguishing each single user flow, the idea is to provide hard QoS guarantees through a pure DiffServ architecture. RSVP Extension for DiffServ is the first solution that can be applied. NSIS solution represents the future for this approach. A full dedicated QoS signalling should be designed by strictly following NSIS recommendations and retaining most of RSVP protocol for features and packet formats.

- MPLS-centric QoS approach
  - MPLS-integrated QoS approach: It has been designed to provide hard QoS guarantees over a DiffServ network portion. RSVP-TE is the signalling protocol designed for this aim. Concerning end-to-end signalling, the DiffServ-IP-centric QoS solutions are the reference.
  - Full-MPLS-centric QoS approach: RSVP-TE is applicable also in this case.

- IPV6-centric QoS approach: This architecture represents the real challenge for future. NSIS is the reference environment without forgetting the steps followed by RSVP and the formats of its packets. A dedicated QoS signalling solution should be designed, in this context, concerning both CSF6N and F6SN. In more detail, F6SN-IPv6-centric QoS approach should consider an RSVP-based new NSIS solution. CSF6N-IPv6-centric QoS approach derives from the DiffServ paradigm and, for this, should also base its new NSIS protocol on the experience of RSVP Extension for DiffServ.

8

Vertical QoS Mapping

8.1 Reference Architecture

As said in Chapter 5, all functional layers need to be involved in QoS management. In particular, one of the problems is to implement bandwidth reservation decisions taken at Layer 3 (e.g. at IP layer) and to map it over the lower layers. The reference environment is the ETSI BSM Protocol Architecture shown in Figure 5.4. The mentioned architecture has been studied for the satellite communication but it has a wider application. This is the motivation to choose it as reference for the investigation of "vertical QoS mapping" in this book. ETSI BSM has provided a big effort concerning QoS architectures and the results of it can be applied also to other environments. Much material contained in this chapter is inspired to ETSI BSM material contained in [ETSI-TS-102462], [ETSI-TS-102463] and [ETSI-TS-102464]. Reference [ETSI-TS-102463] is an open specification to enable QoS services over IP-based multimedia systems based on the IntServ paradigm, extensively explained in the previous chapters. Its focus is on the mapping of IP QoS functionalities over the satellite-specific QoS features. The Satellite Independent Service Access Point (SI-SAP) interface is used. In the same general QoS context, Reference [ETSI-TS-102464] refers to DiffServ and specifies QoS mapping between DiffServ Code Points (DSCPs) and Satellite Dependent services.

The following chapter tries generalizing the "vertical QoS mapping" issue but without forgetting the huge work developed within the ETSI BSM technical committee to which the author belongs.

As said in Chapter 5, the layers whose features depend on the technology are identified as Technology Dependent (TD; Satellite Dependent (SD) in the BSM architecture), while layers whose characteristics are independent of the transport technology are classified as Technology Independent (TI; Satellite Independent (SI) in the BSM architecture). The interface between them is defined as Technology Independent Service Access Point (TI-SAP), which is the ETSI BSM SI-SAP.

QoS over Heterogeneous Networks Mario Marchese

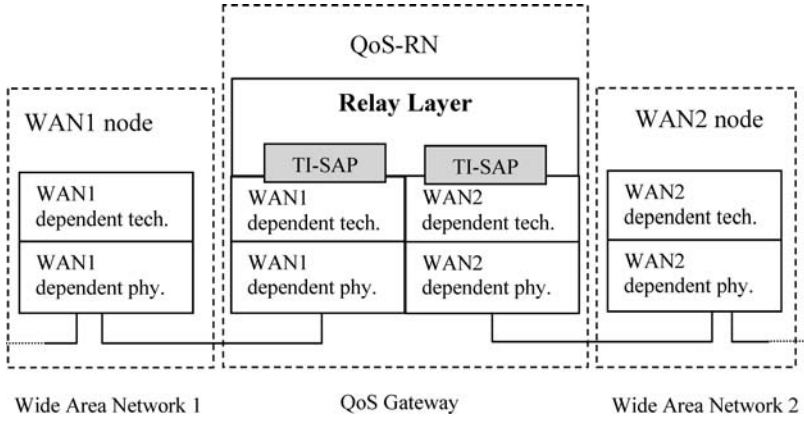


Figure 8.1 TI-SAP location within generic Relay Node (RN)

Figure 8.1 identifies exactly the TI-SAP action point within the generic Relay Node defined in Figure 5.14. Figure 8.2 completes the identification also for the Private Relay Node of Figure 5.13, for which the difference is only in the property of the QoS gateway. Only the interface towards Wide Area Networks is of interest. The internal connection between QoS-PRNs may be considered only a physical connection (supposed to be fully over-dimensioned). If this assumption seems to be strong, there is no problem to define also a TI-SAP over the intermediate connection. It is shown as a dashed line in Figure 8.2. Anyway, the distinction between Relay Node and Private Relay Node is not essential concerning the "vertical QoS mapping". Only Relay Nodes will be referenced in the remaining part of the chapter. The scheme in Figure 8.1 may be obviously applied in cases where the used technology is explicated, as well as in Figures 6.12, 6.20, 6.25 and 6.32. The specific case of a satellite network, where the idea of splitting TD and TI layers was born, is shown in Figure 8.3 by using the proper SI-SAP terminology. Relay Nodes, in this case, correspond to the Satellite Terminals (STs) that give access to the satellite portion. IPv4/IPv6 has been

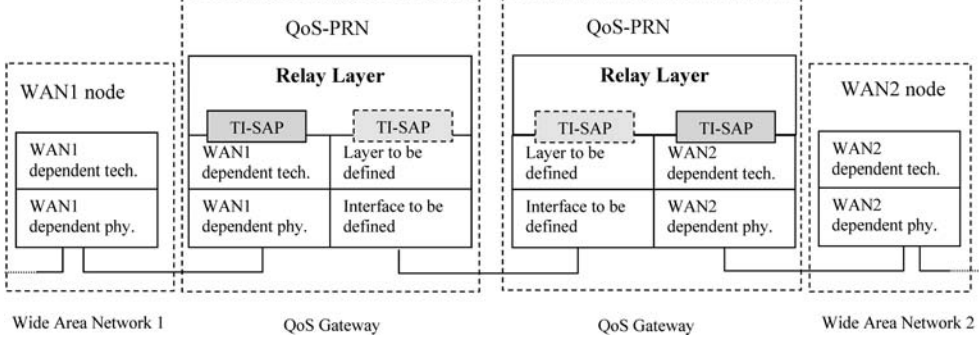


Figure 8.2 TI-SAP location within generic Private Relay Node (PRN)

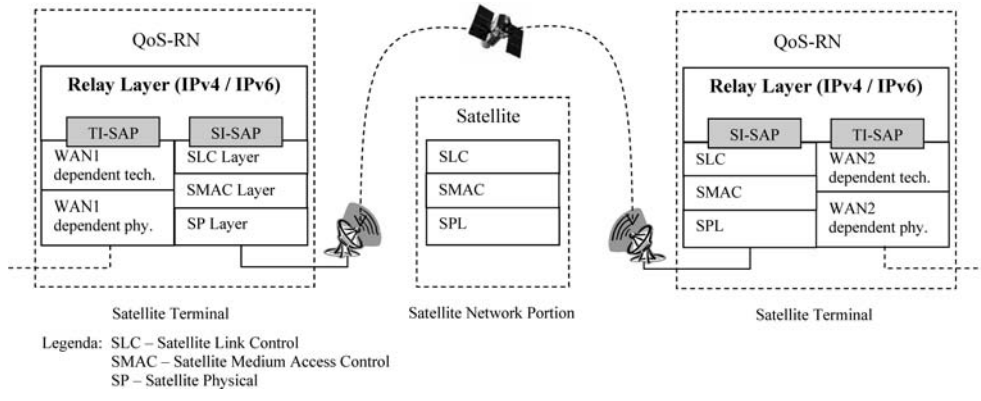


Figure 8.3 SI-SAP location within a satellite network portion

used as Relay Layer because SI-SAP (even if it has a general meaning) has been formally defined only for IP-based network, as shown in Figure 5.4.

Even if Relay Nodes are the reference, it is essential to mention that vertical QoS mapping is not limited to it, but it has a general application when there are two or more layers in cascade and the lower layer must offer a QoS guarantee to the upper layer. Another application, also mentioned in this book, is vertical QoS mapping within each network node, that is within private network portions.

It is important to note that ETSI BSM (in Figure 5.4) has not been totally applied. The Satellite Independent and Dependent Adaptation Functions have been ignored. Actually, they strongly depend on the interface implementation issues. Some details about them will be provided in the following without any ambition of completeness.

Formally, vertical QoS mapping problem may be described as a chain of buffers in cascade. Figure 5.5 shows the used model in the IP DiffServ over ATM environment. Figure 8.4 tries to increase the level of abstraction and to describe the problem with no reference to

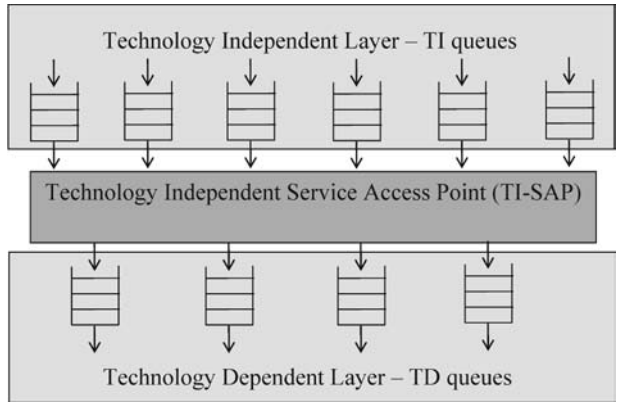


Figure 8.4 Vertical QoS Mapping model

a specific technology. Both TD and TI layers are described as batteries of queues acting, respectively, at Layer 2 and Layer 3. The choice has a direct correspondence with reality where hardware and software buffers are used.

8.2 Control Modules

The architecture shown in Figure 8.4 must be connected to the related control plane and with the functionalities shown in Figures 6.37 and 6.38. Following the lines drawn in [ETSI-TS-102463] and [ETSI-TS-102464], the control plane is composed of the following control blocks:

- The Resource Management Entity (also called TI layer Resource Management Entity, when it is necessary to avoid confusion with the control block acting at TD layer and to stress the action at TI layer) if vertical QoS mapping is applied within the Relay Node (as shown in Figure 8.1); the Network Portion Resource Manager, if vertical QoS mapping is applied within each Network Portion or within each single Network Node. The first case is referenced in the chapter for the sake of simplicity. There is no difference in the other case, from the control viewpoint.
- The TD Resource Management Entity, which physically allocates the necessary resources at Layer 2 and often works in parallel with a local bandwidth allocator, which may also correspond to the Network Portion Resource Manager, if it has information about the physical state of the links. The local bandwidth allocation (identified also as Network Bandwidth Management Entity in Figure 6.38) is called "Network Control Centre" (NCC) for the sake of generality. The acronym NCC is much used in the case of radio and satellite networks where the bandwidth to be allocated is shared among different stations. The bandwidth to be allocated to each single station to match vertical QoS requirement may overcome the maximum limit capacity of the shared network. Chapter 11 is dedicated to this important issue, which is only mentioned in this chapter.
- The QoS Mapping Management Entity (QoS Mapping Manager) that receives the resource allocation requests from the Resource Management Entity. A request may concern resource reservation, release and modification. The communication between QoS Mapping Management and Resource Management Entity should be established through a proper interface. A proposal specific for satellite interfaces SI–SAPs comes from [ETSI-TS-102463]. It refers to the composition of a set of primitives to establish the mentioned communication. It is referenced in the remaining part of the chapter because it is very interesting and, if properly modified, it could be applied to other environments. After receiving the resource allocation request, the QoS Mapping Manager maps it on the lower layer. In other words, it translates the request into reservation, release and modification actions to be applied at TD layer.

Besides formal definitions, QoS mapping action is very interesting from the scientific viewpoint. It includes the use of control algorithms, and measures and tackles different embedded problems, explicitly described in section 8.4.

Figure 8.5 contains the mentioned control modules and connects the vertical QoS model of Figure 8.4 with the control architecture shown in Figures 6.37 and 6.38. It includes the distinction between user and control plane, as well as the presence of the mentioned Network Control Centre.

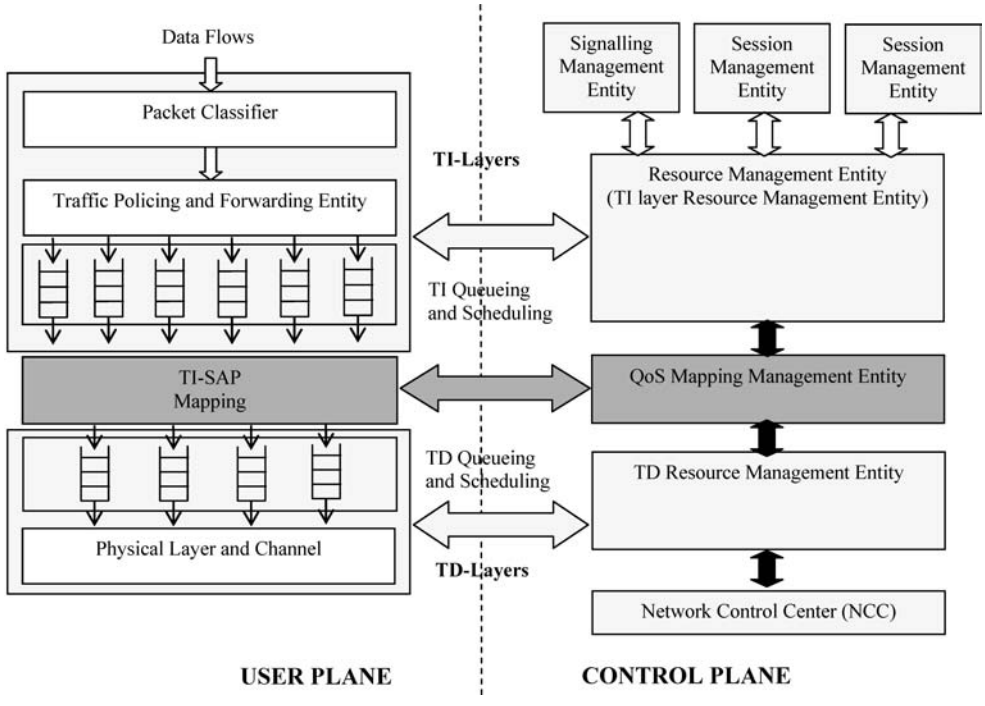


Figure 8.5 Vertical QoS mapping: Information forwarding and control module interaction

8.3 Technology Independent Layers' Implementation

If Figure 8.5 shows user and control plane with no reference to technology, it is also important to show the different architectures when specific QoS-supporting solutions are used. It has no impact on TD layers because the two blocks have been decoupled by the TI-SAP but, anyway, there is impact on QoS management, as shown in Chapters 3 and 4, starting from the power of flow identification offered by the different solutions.

Concerning TI layers, Chapter 6 evaluated five different possibilities. The ATM solution, even if theoretically possible and powerful from the QoS viewpoint, has not been considered as a TI solution, because its use is complex from the signalling viewpoint. Actually, ATM requires a proper signalling stack that makes interworking with other networks complex. Besides, due to its diffusion, IP is considered here as the only technology feasible for host terminals together with ISDN. It suggests the use of ATM at lower layer, as transport technology. As a consequence, ATM will be considered as one of the main solutions for the SD layers.

The solutions considered as possible Relay Layers within the Relay Nodes in Chapter 6 are IP IntServ, IP DiffServ, MPLS, F6SN and CSF6N. The advantages and drawbacks of them have been outlined in Chapter 3, and for QoS Architectures in Chapter 6. The signalling proposals for them have been considered in Chapter 7, within the framework of QoS architectures. They can be applied now to differentiate the QoS provision power offered by the TI layers.

As said in Chapter 3, IP offers two ways to mark traffic:

1. The vector composed of "IP source address", "IP destination address", "Protocol", all contained within the IPv4 header; "TCP/UDP source port" and "TCP/UDP destination port", contained in the TCP/UDP header. The vector characterizes each single customer, but there is no single label available to identify a group of flows with the same performance requirements.
2. ToS field (8 bits in the IPv4 header), whose first 6 bits define the DSCP (DiffServ Code Point) field. The packets identified by the same DSCP are treated in the same way by network components acting at IP layer.

The first way to mark traffic is used by the Integrated Services, which defines (see Chapter 3 for general description and Chapter 7 for signalling) two service classes: guaranteed service (GS) and controlled-load service (CLS). GS is a service that assures that IP packets arrive within a specific maximum delivery time and are not discarded due to queue overflow, if the traffic flow respects the declared traffic parameters. CLS offers a service that guarantees high probability that transmitted packets are successfully delivered to the destination and end-to-end transit delay measured for high percentage of packets does not exceed the minimum end-to-end transit delay, computed by considering transmission delays and fixed processing delay within network elements.

Vertical QoS mapping model applied by using IntServ at TI layers is shown in Figure 8.6. As extensively said in Chapter 7, the Signalling Management Entity is the RSVP Entity. Resource requests are transported through the RSVP objects presented in section 7.2.2. The other control entities are the same as in Figure 8.5. Figure 8.6 shows three groups of queues dedicated to GS, CLS and BE services. Each single queue transports a specific traffic flow identified by the vector "IP source address", "IP destination address", "Protocol", "TCP/UDP

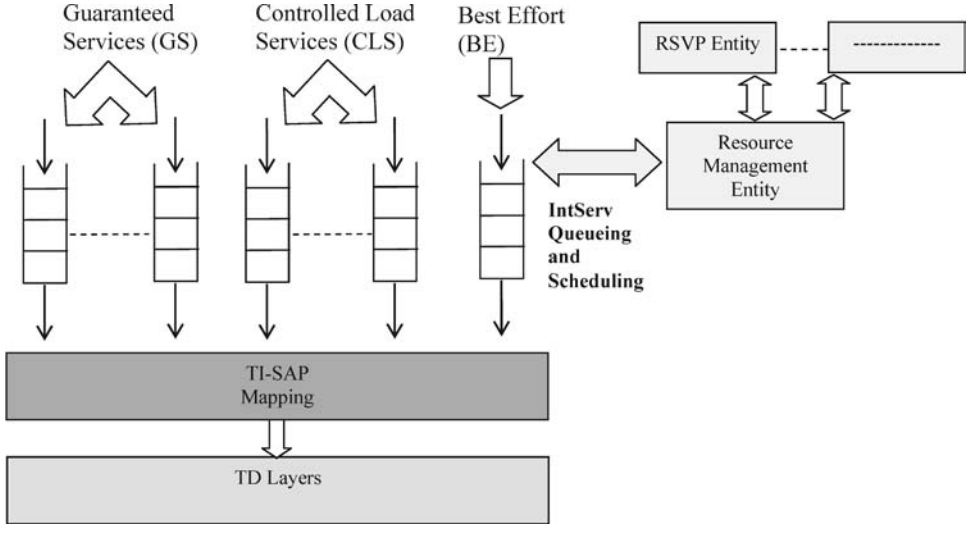


Figure 8.6 Vertical QoS mapping: IntServ at TI layers

source port" and "TCP/UDP destination port". The service of each single IntServ queue needs to be mapped on the buffers implemented at TD layers through the interface TI–SAP.

The Differentiated Services (Diffserv) use the DSCP (DiffServ Code Point) value contained in the first 6 bits either of IPv4 header field ToS or of the IPv6 header Traffic Class field. The 6-bit field DSCP specifies the forwarding behaviour (technically called "Forwarding Equivalence Class") that the packet will receive. The forwarding behaviour is called PHB (Per Hop Behaviour), within each network node. Offered QoS is strictly linked to the DSCP assignation, specified in section 3.4.4. The DSCP identifies a specific traffic class and implies that all the packets identified with the same DSCP receive the same treatment. The class selector PHB offers three forwarding priorities: Expedited Forwarding (EF) is characterized by a minimum configurable service rate and is oriented to low delay and low loss services; Assured Forwarding (AF) is specified for four independent classes (AF1, AF2, AF3 and AF4) and, within each AF class, offers three "drop precedence" categories; Best Effort (BE) does not provide any performance guarantee and does not define any QoS level. Besides general DSCP assignation rules, Tables 3.6, 3.7 and 3.8 report complete proposals about the PHB offered by the DiffServ paradigm. Concerning signalling in QoS architectures, Chapter 7 reports the possible solutions for the DiffServ environment. Following this line, Figure 8.7 shows the vertical QoS model when the DiffServ paradigm is applied at TI layers. Traffic is classified into three groups of flows: EF, AF and BE. Each single queue is identified by the DSCP value and the signalling action is managed by the "DSCP-based Signalling" entity, which implements one of the solutions explained in Chapter 7.

MPLS identifies the traffic flows through the Label Value of 20 bits, contained in the MPLS header. Each label is associated with control modules that allow guaranteeing the level of quality specified in the SLS for each flow. The vertical QoS models applied for MPLS TI layers are shown in Figure 8.8. Each SI queue receives the traffic of the flows entering the Relay Node (or any other Network Node where vertical QoS mapping is required). The flows and the related queues are classified by using the Label Values, numbered from 1 to *n*.

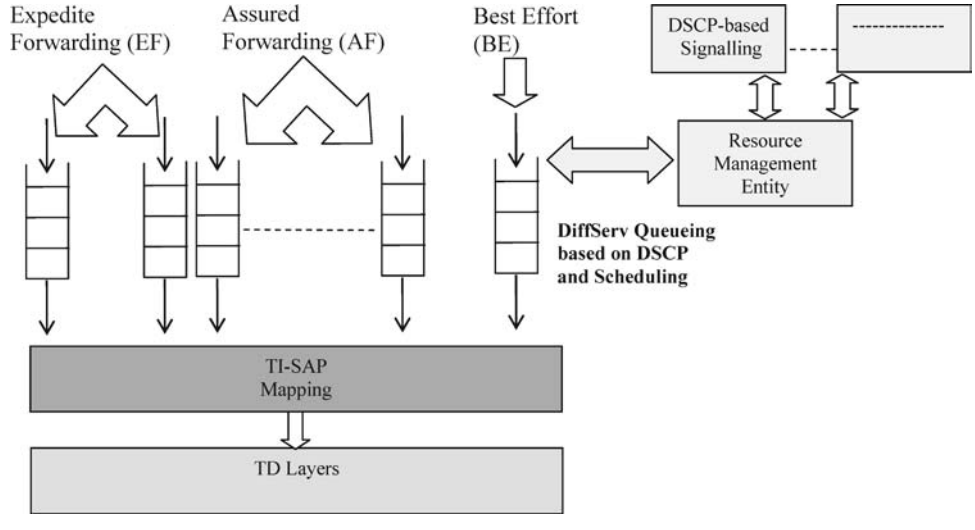


Figure 8.7 Vertical QoS mapping: DiffServ at TI layers

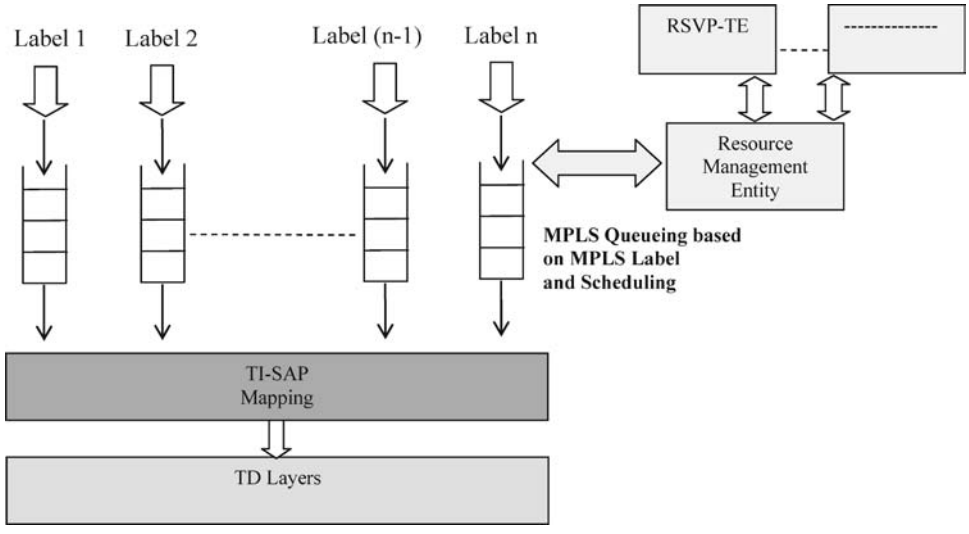


Figure 8.8 Vertical QoS mapping: MPLS at TI layers

The Resource Management Entity implements RSVP-TE, as clarified in Chapter 7. As in the other cases, the services provided by the TI queues will be mapped on the TD layer through the interface TI–SAP.

IPv6 can use two fields to identify the traffic flows. They are contained in the IPv6 header: the Flow Label field (20 bits) and the Traffic Class field (8 bits). The latter is functionally equivalent to IPv4 ToS field and contains the DSCP field in the first 6 bits leaving the rest

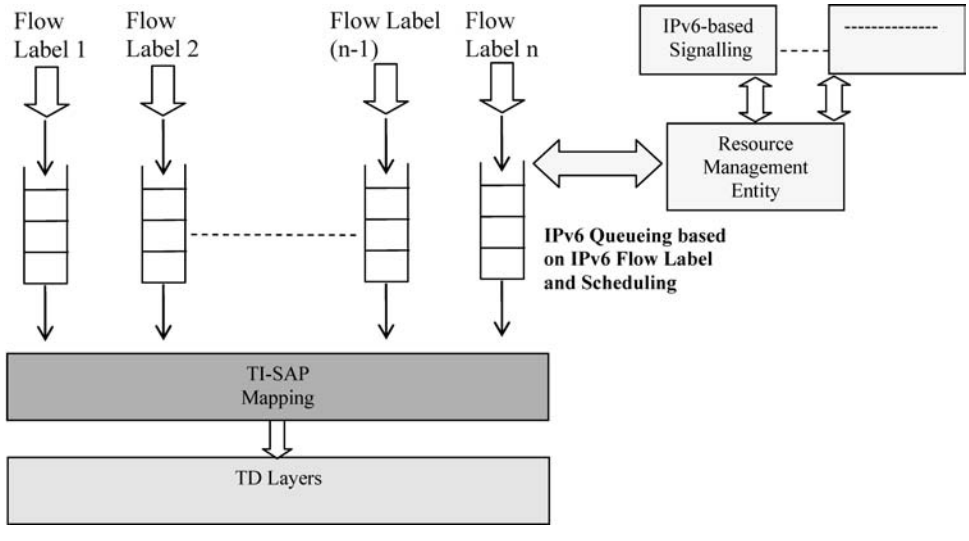


Figure 8.9 Vertical QoS mapping: IPv6 at TI layers (CSF6N and F6SN architectures)

for future extensions, as in IPv4. The vertical QoS mapping model, in this case, is again DiffServ. The former opens the door to two novel QoS architectures: CSF6N (in section 3.6) and F6SN (in section 3.7), whose architectures have been specified in section 6.6. Even if CSF6N and F6SN use the flow identifier Flow Label with a different meaning, the vertical QoS model (as well as the QoS architecture) is the same for both. It is reported in Figure 8.9. Each single queue is identified by the Flow Label value. The signalling action is managed by the "IPv6-based Signalling" entity, which implements one of the solutions explained in Chapter 7. TI-SAP will transfer the TI services on the physical technology at TD layers.

8.4 Technology Dependent Layers' Implementation

The encapsulation used at TD layers may vary depending on the environment. Some examples have been reported in section 5.2.1 and, again, in Figure 8.10. The cloud containing the different technologies means that each of the listed solutions may be used to encapsulate the packets coming from the upper layers. The list is not exhaustive. Even if TI-SAP is aimed at decoupling TI and TD implementations, the real QoS mapping provided over Layer 2 technology heavily depends on the features of Layer 2 encapsulation and on the characteristics of the used hardware. TD layers, for their definition, depend on the implementation details such as buffer dimensions, number of queues, medium access control implementation and bandwidth allocation schemes, if present. The role of TI-SAP is just to hide these details to the upper layers but, in the framework of this book, is very important, also in view of real future implementations, to specify the problems of vertical QoS mapping at the TD layers. These issues are ignored for choice by technical reports for the motivations reported above. This book dedicates the entire next chapter to describe possible control schemes that try solving the vertical QoS mapping at the TD layers. It will suggest practical implementations about the action of Layer 2.

The control blocks acting at TD layer are shown in Figure 8.11. Information is multiplexed over the common channel by serving the TD buffers. The physical channel is shared among all TD buffers, but it can also be shared among different remote Relay Nodes, as in the case of wireless and satellite channels. Figure 8.3 is exemplary of the satellite/wireless case. The difference between the two situations is very important. In both the cases, the physical bandwidth is assigned by a specific control block (called Network Control Centre). If there is a dedicated channel for the single Multiplexing serving one Relay Node (or any other Network Node, where necessary), the Network Control Centre corresponds, at least in theory, with the Bandwidth Management Entity (Bandwidth

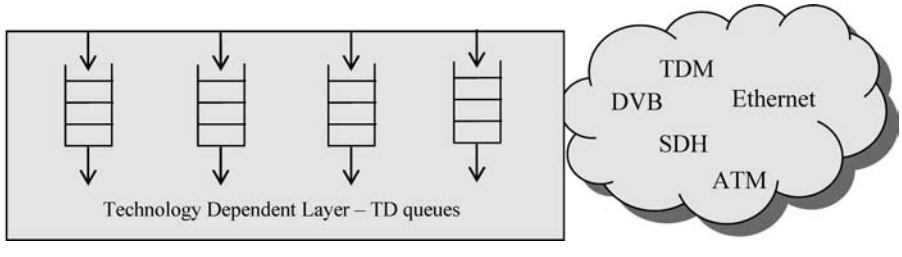


Figure 8.10 SD layer possible implementations

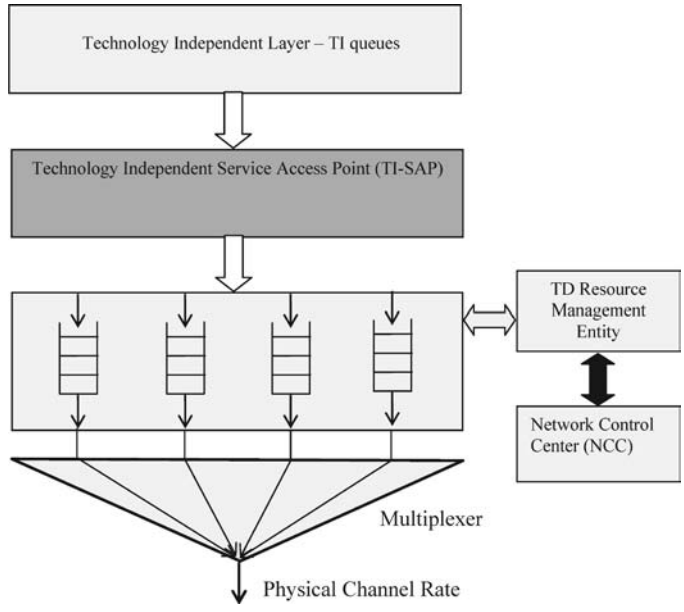


Figure 8.11 SD layer control modules

Broker), if the control is implemented within a Relay Node, and with Network Bandwidth Management Entity, if the control is implemented within a Network Node (and/or within a private network whose bandwidth allocation is decided by the mentioned management entity). Obviously, the problems introduced by vertical QoS mapping listed in detail in the remaining part of the chapter are still to be solved but, theoretically, there is no need to interact with another control block: the BB (or the other mentioned blocks) assigns the bandwidth at SI buffers. This service capacity needs to be transferred to the SD layers in transparent way but the channel capacity, which is typically fixed for a cable and variable over time for wireless and satellite links, is entirely dedicated to the SD queues. Figure 6.38 shows the referenced control Entities. If the channel is shared among different Relay Nodes as in Figure 8.3, the bandwidth allocated at SI layers does not necessarily correspond to the bandwidth physically allocated on the channel, because the available channel rate varies over time, so making the mapping problem more complex. It means that, in addition to the mapping needs, it is important to consider that the channel is shared with other users remotely located, that the bandwidth is not infinite, that the sum of all bandwidth requests may overcome the entire bandwidth capacity and that the channel rate may be affected by external phenomena, such as fading. This last problem, which characterizes wireless and satellite links, exists also if the channel is not shared. For these motivations, the presence of an additional control block (the Network Control Centre), acting at TD layer and using information about the channel state relieved at frame and physical layer, must be forecast. The specific action of the Network Control Centre depends on the physical link features and, in general, implies the introduction of complex mathematical models. The topic is of main interest for research. An example of it will be given in Chapter 11, specifically dedicated to satellite architectures and bandwidth allocation mechanisms.

8.5 TI-SAP Implementation

It is very important to have a model to describe the TI-SAP. Actually, the literature contains, for now, a formal description of it only for satellite communications. The documents [ETSI-TS-102462], [ETSI-TS-102463] and [ETSI-TS-102464] contain a formal description of the interface SI–SAP. The same concepts will be used here and referred to a generic TI–SAP interface. Obviously, it is important to remember that the extension to generic TI–SAP is done by the author just for this book and that the basic concepts used appear in the literature as referred only to the satellite interface.

The idea is to model TI-SAP through a block of buffers, similarly as done for SI and SD layers. Each single queue is individuated by a specific identifier that is called QID (Queue Identifier) in [ETSI-TS-102462] and related documents. Each QID represents an abstract queue available at the interface TI–SAP and identifies a specific level of QoS to transfer packets from TI to TD layer. The TD layers are responsible to assign the necessary capacity to each single abstract queue so that the QoS levels that the TI layers should receive from the TD layers can be satisfied. Actually, using the concept of abstract queues allows decoupling the mapping problem into two different separate problems: TI queues mapped over QIDs and QIDs mapped over TD queues. The QoS Mapping Management Entity shown in Figure 8.5 has the role of managing QIDs and related mapping problems. It is called QID Resource Manager in [ETSI-TS-102462] just for this motivation. QoS Mapping Management Entity is referenced here also as QID Management when it is necessary to stress the role of QID manager. Referring to satellite environment and SI-SAP, in particular, this issue is matched by the control blocks together with SIAF (related to SI layer to QIDs mapping) and SDAF functions (related to QIDs to SD layer mapping) depicted in Figure 5.4. Anyway, the assumption about the existence of a battery of buffers at the TI-SAP, actually modelling the TI-SAP, is the most important concept. It means that the Relay Layer (or any other Network Node where vertical QoS mapping is implemented) provides different levels of QoS within a transport bearer through a number of QIDs that govern the service offered by the TI-SAP at upper layers. TI layers can access and modify the QIDs so as to request and possibly receive a different QoS service. QID management may be static or dynamic. Figure 8.12 shows the general concept of abstract queues and the related control module. The number of QIDs should be large enough to support the desired QoS model.

As said above, the problem may be decomposed into two separate mapping problems: first, mapping between TI queues and QIDs and secondly mapping between QIDs and TD queues. Decoupling the problem is an implementation choice and it is not mandatory. TI queues may be mapped directly on TD queues. Anyway, the double step allows more elastic implementation solutions.

The TI-to-QID mapping may be modelled as in Figure 8.13. Three different situations may happen:

1) One TI queue is associated with one QID (one-to-one correspondence)
2) More than one TI queue is associated with one QID (multiplexing)
3) One TI queue is associated with more than one QIDs (demultiplexing).

The first two options are more common. The third option is a special case that is applied when the packets belonging to the same TI traffic class and, therefore, requiring the same level of QoS are routed towards different destinations. This solution may be used in very

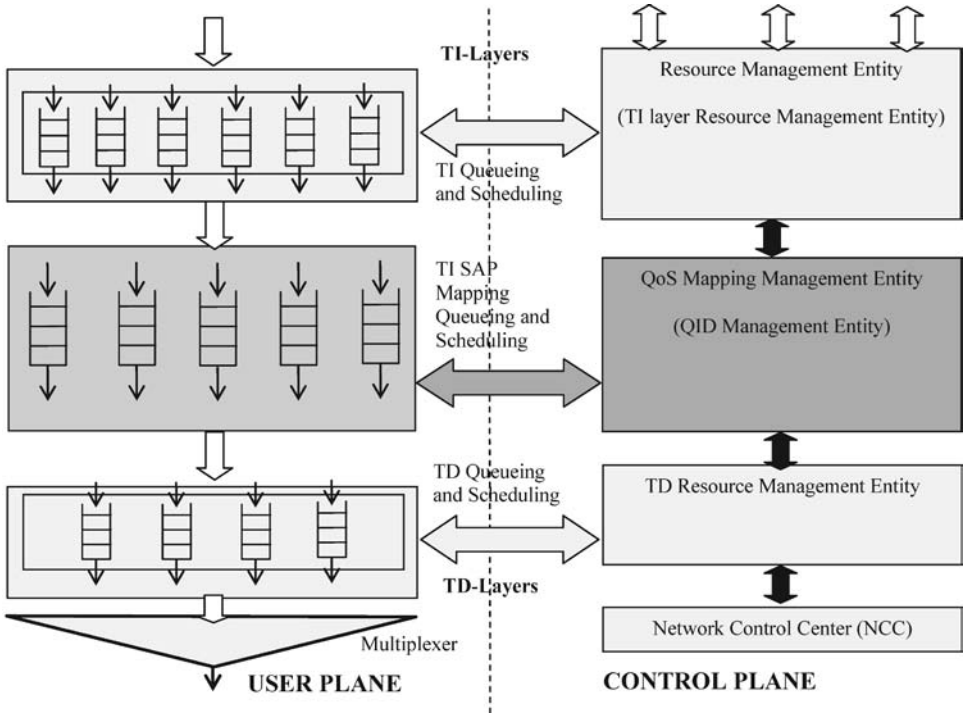


Figure 8.12 TI-SAP model

large mesh network where the large number of next hop nodes requires their aggregation into fewer IP queues. Obviously, a mesh network may also use separate IP queues for all next hop nodes, as done in typical router implementations, and also supposed in this book. In this case, options 1 and 2 are applied again. A TI-to-QIDs mapping table is required to

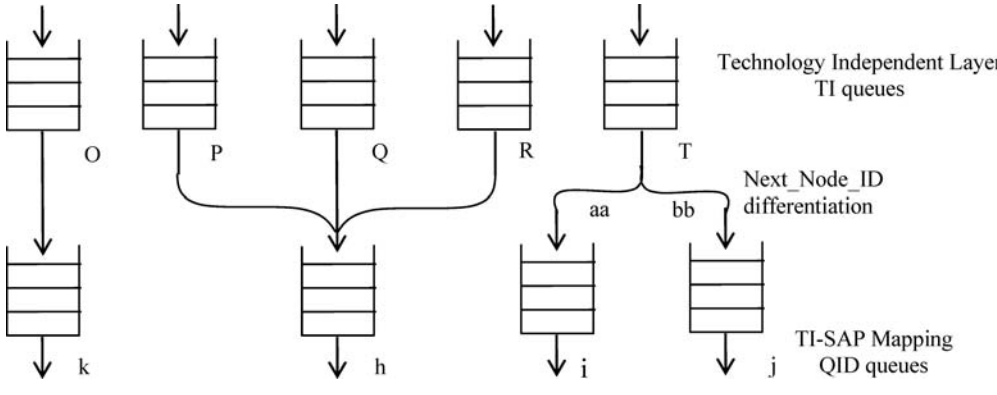


Figure 8.13 TI-to-QID mapping model

Table 8.1 TI-to-QID mapping table

TI queue identifier	Next_Node_ID	QID
O	all	k
P	all	h
Q	all	h
R	all	h
T	aa	i
T	bb	j

implement the described operations. Table 8.1 shows it for the queue cascade in Figure 8.13. The implementation of option 3 implies the knowledge of a next node identifier. It should be the Layer 2 address but it might also be a private identifier used within the TI-SAP Resource Management Entity. It is called Next_Node_ID in Table 8.1 and its meaning is clear from the application framework of the third option. If option 3 is not forecast, this table entry is useless.

Mapping of QIDs over TD queues may be modelled similarly as the previous case except for option 3, which has no physical meaning in QID-to-TD case. Two different situations may happen:

1) One SI queue is associated with one QID (one-to-one correspondence)
2) More than one SI queue is associated with one QID (multiplexing)

The two options are graphically shown in Figure 8.14, while the related mapping table is contained in Table 8.2. As clear in Figures 8.13 and 8.14, two batteries of buffers acting in cascade represent the model of the problem. Each single layer that contains the buffer must implement a protocol and implies a specific encapsulation. A change of encapsulation is necessary when information is transferred between adjacent layers. Information coming from two or more buffers is often conveyed to a single buffer at lower layer so implementing a multiplexing action. Moreover, the physical channel bandwidth that acts as server rate for the queues at the SD layer may vary over time. These three issues will be formalized in the next paragraph so as to allow the introduction of a general format for the description of vertical QoS mapping.

Table 8.2 QID-to-TD mapping table

QID	TD queue identifier
k	v
h	w
i	w
j	w
l	z
m	z

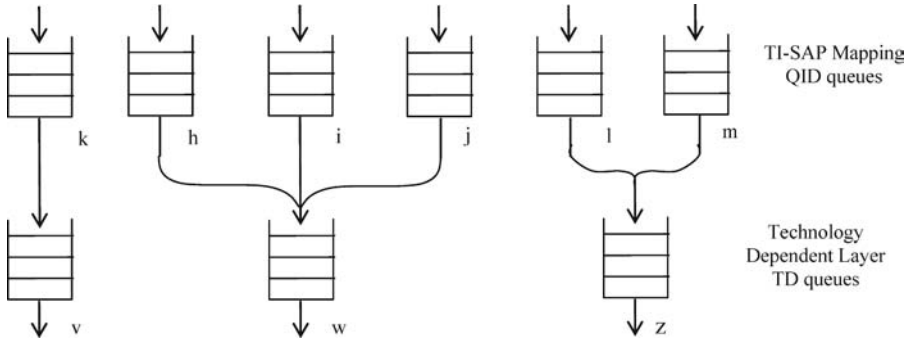


Figure 8.14 QID-to-TD mapping model

Concerning the control plane, it is important to establish a communication mechanism between the entities dedicated to control the mapping action and shown on the right in Figures 8.5 and 8.12. In particular, it is interesting to check how the TI–SAP interface can be really implemented. Actually, the queue models allow understanding the problems, but they do not provide any information about real implementations. Also in this case, the work developed by the ETSI SES BSM technical committee is essential. The reference document, in this case, is represented by the technical specification [ETSI-TS-102463], even if written only for the satellite environment.

The mechanism proposed in [ETSI-TS-102463] uses the model recommended by OSI for the communication between adjacent layers. It defines a set of primitives through which the set of queues may be opened, modified and closed. The set of primitives represents the view that the TI layers (the SI, in case of satellite environment, and the IP, in the detail of document [ETSI-TS-102463]) have of the TD layers through the abstract queues of the TI-SAP. In other words, the set of primitives formally defines the communication between the TI and TD layers. The action on the TD layers to provide the necessary bandwidth to satisfy the desired performance requirements concerns the manufactures and cannot be standardized. Nevertheless, the problem is scientifically so interesting that its formalization and its possible solution will be reported, respectively, in the next paragraph and in the next chapter. The latter mainly concerns research issues and it is probably of interest only for scientists working in the field.

The set of primitives proposed by [ETSI-TS-102463] and applied for a general TI-SAP here is reported in Table 8.3. All actions are ruled by the four basic primitive types of the OSI-layered architecture explained at the beginning of Chapter 3: request, confirm, indication and response. The following parameters may be used to carry information through the primitives:

1. TI queue identifier (shown in Table 8.1). It identifies the queue at TI layer (e.g. the IP queue).
2. QID (shown in Tables 8.1 and 8.2). It identifies the abstract queue.
3. Next_Node_ID (shown in Table 8.1 and Figure 8.13). It specifies the next node to be reached.

Table 8.3 TI-SAP set of primitives

Primitives	Service offered	Primitive Parameters
TI-queue-open.request	It requests to open an abstract queue QID. The request comes from TI layer Resource Management	TI queue identifier, QIDSPEC, Next_Node_ID
TI-queue-open.confirm	It confirms the opening of an abstract queue QID. It is the reply of QID Management Entity solicited by TI-queue-open.request	TI queue identifier, QIDSPEC, Next_Node_ID, QID, Success_flag
TI-queue-modify.request	It requests to modify an abstract queue; the request is performed by TI layer Resource Management	TI queue identifier, Next_Node_ID, QID, QIDSPEC (optional)
TI-queue-modify.confirm	It confirms the modification (or non-modification) of an abstract queue in answer to TI-queue-modify.request	TI queue identifier, Next_Node_ID, QID, QIDSPEC, Success_flag
TI-queue-modify.indication	It communicates a modification request of an abstract queue performed by the QID Management Entity and sent to TI layer Resource Management	QID, QIDSPEC, TI queue identifier (optional), Next_Node_ID (optional)
TI-queue-modify.response	It acknowledges the reception of TI-queue-modify.indication. It is emitted by TI layer Resource Management	QID
TI-queue-close.request	It requests to close an abstract queue. The request is generated by TI layer Resource Management Entity	TI queue identifier, Next_Node_ID, QID, QIDSPEC (optional)
TI-queue-close.confirm	It acknowledges the request to close an abstract queue. It is emitted by QID Management Entity	TI queue identifier, Next_Node_ID, QID
TI-queue-close.indication	It requests to close an abstract queue when decided by QID Management Entity	QID, TI queue identifier (optional), Next_Node_ID (optional)
TI-queue-close.response	It acknowledges the TI-queue-close.indication. It is emitted by TI layer Resource Management Entity	TI queue identifier, Next_Node_ID, QID

(*continued overleaf*)

Table 8.3 (*Continued*)

Primitives	Service offered	Primitive Parameters
TI-queue-state.request	It requires the state of a specific abstract queue identified by QID	QID, TI queue identifier (optional), Next_Node_ID (optional)
TI-queue-state.confirm	It communicates the state of the abstract queue as requested by TI-queue-state.request	QID, QIDSTATE, TI queue identifier (optional), Next_Node_ID (optional)
TI-queue-state.indication	It requires the state of a specific TI layer queue	QID, TI queue identifier, Next_Node_ID (optional)
TI-queue-state.response	It communicates the state of the TI layer queue as requested by TI-queue-state.indication	QID, TI queue identifier, TI queue state descriptors Next_Node_ID (optional)

4. QIDSPEC. It is a key point for QoS management. It specifies the QoS level requested and represents the Service Level Specification (SLS). For example, in case in QoS management provided through IntServ, it is the RSVP FLOWSPEC Object, reported in Figure 7.11. QIDSPEC values will be translated into parameters used to associate QID to TD queues and to allocate resources at TD layer.
5. Success_flag. It is set to "1" if the related operation (e.g. to create a new abstract queue) was successful; it is set to "0" otherwise.

In detail, referring to Figure 8.12, the Resource Management Entity (the IP queue manager, if the TI layers are composed of the IP stack) sends a request to the QID Management Entity to create an abstract queue QID through the primitive "TI-queue-open.request". The request creates a new abstract queue identified by QID and associates it with the IP queues and with the SD queues by following one of the options reported in Figures 8.13 and 8.14. The request transports the following parameters: the TI queue identifier (to identify the request), QIDSPEC (to specify QoS request and allow resource check) and Next_Node_ID (to create a new entry in the TI-to-QID table). The QID Management Entity answers to the request through the "TI-queue-open.confirm" primitive. It identifies the request by using the same TI queue identifier, QIDSPEC and Next_Node_ID parameters used by the request and communicates the assigned QID (to be stored in the TI-to-QID table), if the request has been accepted. The acceptance of the request is signalled through the Success_flag set to "1". An open abstract queue may be modified both by a TI-layer Resource Management request and by an unsolicited indication of the QID Management Entity. The former is implemented through the "TI-queue-modify.request" primitive, which identifies the abstract queue to modify. The identification is performed through the parameters TI queue identifier, Next_Node_ID, QID and, optionally, QIDSPEC, which may be not available at TI layer. After receiving the TI-queue-modify.request, the QID Management Entity confirms or does not confirm the modification through the primitive "TI-queue-modify.confirm", which uses the same parameters of the request primitive: TI queue identifier and Next_Node_ID (to update the IP-to-QID table), QID (to identify which QID as been modified or not)

and QIDSPEC, now mandatory, to update the information about QIDSPECs and related QoS management at QID Management level. Additionally, as done for QID creation, the accomplishment of QID modification is confirmed through the Success_flag that is set to "1" for a positive modification and to "0", if the modification request is not accomplished. Concerning the unsolicited indication of the QID Management Entity, it is communicated through the "TI-queue-modify.indication" primitive, called by the QID Management Entity towards the TI layer Resource Management to inform about the change. Its mandatory parameters are QID and QIDSPEC, to identify the modified queue and the related quality, respectively. Optionally, it can transport also TI queue identifier and Next_Node_ID, if QID Management is aware about their values. The acknowledge to this primitive is provided by TI layer Resource Management through the "TI-queue-modify.response" primitive, whose only parameter is obviously the same QID specified in the indication. An abstract queue may be also closed. The action may be started by the TI layer Resource Management Entity through the primitive "TI-queue-close.request". When TI flows stop, the associated TI queues are released and the related abstract queues are no longer necessary. The parameters are TI queue identifier, QID and Next_Node_ID. QIDSPEC is optional because it could also be unknown at TI layer. QID Management acknowledges the request, which must be accomplished, through the primitive "TI-queue-close.confirm". It contains the same values of the following parameters, used in the corresponding request: TI queue identifier, QID and Next_Node_ID, to identify the request generator and to identify which QID has been closed. The action to close QIDs may be also started by an unsolicited decision of the QID Management Entity. If an abstract queue QID is closed by the QID Management, the TI layer is informed of it through the "TI-queue-close.indication" primitive, which carries the parameter QID to identify the closed abstract queue and, optionally, if QID manager is aware of their values, the parameters TI queue identifier and Next_Node_ID. QID Management acknowledges the indication through the primitive "TI-queue-close.response". It specifies TI queue identifier of the TI queue associated with the closed QID, Next_Node_ID and the same QID used by the indication. Concerning the requests to close a queue, three cases may happen looking at Figures 8.13 and 8.14.

1. The closed abstract queue identified by QID is associated to one TI layer queue. In this case, if the release request comes from the TI layer (TI-queue-close.request), it means that the TI queue is no longer needed and, as a consequence, the related QID is useless. Its release is confirmed by TI-queue-close.confirm. If the release request comes from the QID manager (TI-queue-close.indication), it means that the abstract queue is no longer available. The associated TI queue is released. The action is confirmed through TI-queue-close.response.
2. The closed abstract queue identified by QID is associated to more than one TI layer queue. In case the release request is activated by the TI layer through TI-queue-close.request, it means that all TI queues are released. Also, the associated QID needs to be released. The action is confirmed through TI-queue-close.confirm. If the closing action is started by the QID manager, TI-queue-close.indication is sent to the TI layer. It means that the abstract queue towards which the TI queues address their traffic does not exist any longer. All the associated queues at the TI layer need to be released, but the QID manager can ignore the number of associated TI layer queues. So, it sends just one TI-queue-close.indication. The TI layer replies with as many TI-queue-close.response as the number of TI layer queues released by closing action.

3. The closed abstract queue identified by QID is associated with a TI layer queue that is mapped also over other QIDs because the traffic is addressed to different next hops (Next_Node_ID). If the action is taken by the TI layer, it means that the TI traffic conveyed to that particular Next_Node_ID has stopped and that the QID is no longer useful and can be released. A TI-queue-close.request is sent. The QID manager replies through TI-queue-close.confirm. When the closing action is generated by the QID manager, it means that one particular QID has been released. A TI-queue-close.indication is sent to the TI layer, but the TI layer, after updating the IP-to-QID table, does not release the associated TI queue because it serves at least another QID. The TI layer acknowledges the modification through TI-queue-close.response.

Two types of requests may be generated to get information about the state of the queues. The first one is generated by the TI layer Management Entity to know the state of the associated QID. It is very useful to tune control actions as well as CAC at TI layer. The request may be referenced as TI-queue-state.request. The QID identifier must be specified. TI queue identifier and Next_Node_ID may be optional in case the QID Manager would like to know the origin of the request. The same parameters are used to communicate the state to the TI layer Management Entity through the primitive TI-queue-state.confirm. Additional parameters are necessary to specify the state of the queues. Obviously, it depends on the control algorithm. They are generally indicated as QIDSTATE here. Examples may be the percentage occupancy of the queue, the quantity of lost information, the service rate of the buffer, the average delay time and the average jitter. The next chapter, which contains a theoretical scheme to vertically map the QoS in a cascade of buffers, gives a clear example about possible required parameters. On the other hand, also the QID Manager may require information about the state of the TI queues. In this case, it is possible to use the TI-queue-state.indication. Requested information is transmitted through TI-queue-state.response. The parameters to be indicated are TI queue identifier, Next_Node_ID and QID, together with the TI queue state descriptors. TI-queue-state.response may also be unsolicited. It means that, within the framework of a control mechanism, it may not require TI-queue-state.indication to be generated.

8.6 Vertical QoS Mapping Problems

After identifying the technological tools to establish a communication between TI and TD layer, it is time now to mention explicitly the problems of QoS mapping between two different layers in cascade, trying to abstract the concept. The abstract queue, for example, is a very powerful tool, in practice, because it allows decoupling the QoS mapping problem. Anyway, QID queue may be substituted by real TD queues without affecting the nature of the problem so introducing a direct QoS mapping between TI and TD layers. In other words, both the direct TI over TD mapping and the two decoupled problems may be modelled as two layers in cascade, shown in Figure 8.13 and 8.14. The latter is taken as a reference in the remaining part of the book. The two consecutive layers are identified as TI and TD, ignoring the abstract queue layer to simplify the theoretical approach. As already said, it does not affect the generality of the approach. Service requirements about the two layers may be exchanged by using a set of primitives as defined in the previous paragraph.

There are three problems arising from the action of two layers in cascade. The first two may be generically applied. The last one is related to time varying channels. In the author's experience, it is related to satellite communications, but it is not limited to them.

8.6.1 Change of Information Unit

It is the consequence of TI traffic transport over a TD portion that implements a specific technology, as indicated in Figure 5.3. As explicitly reported in section 3.1, at each layer, the information coming from the upper layer and called Service Data Unit (SDU) is encapsulated within a new frame composed also of header and/or trailer (called Protocol Control Information – PCI). The newly created entity is called Protocol Data Unit (PDU). The process is clearly depicted in Figure 3.3. It means that the TI layer packets access the TD queue after being encapsulated in a new frame. It is intuitive that the service rate at the SD layer must consider the additional bits of the header/trailer to keep a fixed level of service. The imposed additional information is called overhead. Figure 8.15 shows the problem of the change of information unit when there are two buffers in cascade. It is important to note that TI and TD are directly used in Figure 8.15, but the mapping problems shown in Figures 8.13 (TI over QID) and 8.14 (QID over TD) may be modelled identically.

The problem may be solved off-line, if information about the dimension of the packet, imposed overhead and current dimension of the TI queue length is available. All information may be unavailable at TD layer, also in presence of a dynamic information exchange between the layers based on a set of primitives. For example, if IP is implemented at TI layer and ATM is chosen for TD layer; if the Service Level Specification imposes a maximum loss of 10^{-2} at IP layer, the minimum bandwidth R_{TI} to guarantee the SLS of TI layer may be provided by simulations, by experience or by equivalent capacity techniques summarized in Chapter 4, given the dimension of the IP buffer. Each single IP packet is encapsulated by using the ATM Adaptation Layer 5 and the successive segmentation in cells of 48 bytes. A header of 5 bytes is added to each cell giving origin to an ATM cell of 53 bytes. The

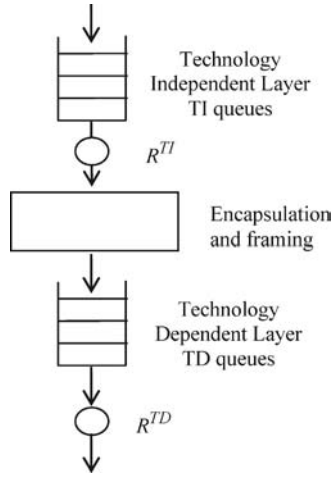


Figure 8.15 Change of information unit: Two buffers in cascade

question is, which is the bandwidth R^{TD} to be allocated at TD layer if a fixed performance, requested by TI layer through the open primitive and a proper set of parameters, similar to QIDSPEC, defined to open an abstract queue, needs to be guaranteed? In numbers, the increase in the bandwidth allocation necessary at the TD layer can be foreseen by means of the overhead effect, called *CellTax*, introduced by the AAL5 with LLC-SNAP encapsulation and by the ATM header. Since, during the generation of the ATM frame, two octets (for the LLC-SNAP overhead) need to be added to each IP packet, the average number of ATM cells for each IP packet is $\overline{\#\text{ATMCells}} = (\overline{\text{DimIPPacket}} + 2)/48$, where $\overline{\text{DimIPPacket}}$ denotes the average IP packet's size in bytes and 48 is the payload of an ATM cell in bytes. Hence, it is possible to compute the overall overhead due to the encapsulation format of the TD frame and the average percentage bandwidth increase necessary in the TD core of the network. The overhead is identified as $\text{CellTax} = (\overline{\#\text{ATMCells}} \times 53 - \overline{\text{DimIPPacket}})/\overline{\text{DimIPPacket}}$. The number 53 is the overall size of an ATM cell in bytes. A possible forecast for the TD bandwidth allocation is ruled by the following heuristic allocation law $R^{TD} = (1 + \text{CellTax}) \cdot R^{TI}$. A similar method can be employed when the transport technologies of interest are different, for example in case of a QoS mapping involving either IPv6 over IPv4 or IPv4 over MPLS at the SI-SAP interface. The knowledge of the precise encapsulation formats is required as well as the average packet dimension. The use of the average packet length is not conservative. The employment of the maximum packet length *MaxIPPacket* would be totally conservative but would imply a great bandwidth waste. Independently of these considerations, another aspect must be considered. This approach implies the knowledge of the IP buffer length because TD buffer length needs to take the IP buffer length and to enlarge it at least of the CellTax. Unfortunately, the knowledge of the IP buffer length is not trivial. The two layers TI and TD, even if communicating, are totally separated and the TD buffer may be obtained within a hardware board that is designed and implemented without any knowledge of future upper layers. It is sufficient to guarantee the implementation of the primitives, but any other aspect may be ignored. Actually, it is the rule of functional layers implementation to improve the flexibility of network elements. It suggests the use of a dynamic algorithm. The concept is enforced concerning the second aspect of vertical QoS mapping.

8.6.2 Heterogeneous Traffic Aggregation

The model is shown in Figure 8.16. The rates R_h^{TI}, R_i^{TI} and R_j^{TI} are properly dimensioned together with the TI queues h, i, j to offer a fixed service at TI layer. For example, if each single TI queue, which may be implemented in IP, requires a specific SLS in terms of packet loss threshold, the service rate and the length of each queue are dimensioned to satisfy the request. Each TI queue may be assigned to a service class, requiring, for instance, $P_{loss-TI}^h \leq 10^{-2}$, $P_{loss-TI}^i \leq 10^{-3}$ and $P_{loss-TI}^j \leq 10^{-5}$. The threshold values on the packet loss give origin to R_h^{TI}, R_i^{TI} and R_j^{TI} values that can vary over time following the entering traffic behaviour. The problem of the vertical mapping is to determine the current service rate R^{TD} to be assigned at layer TD so that the change of layer is transparent to the users. Operatively, the TI layer may ask the TD layer to guarantee a packet loss below a given threshold (e.g. $P_{loss-TD} \leq 10^{-6}$) or, alternatively, to follow run-time the behaviour of the TI queue that has the most severe requirement (e.g. $P_{loss-TD} \leq P_{loss-TI}^j$). The next chapter will report an algorithm that operates in this direction.

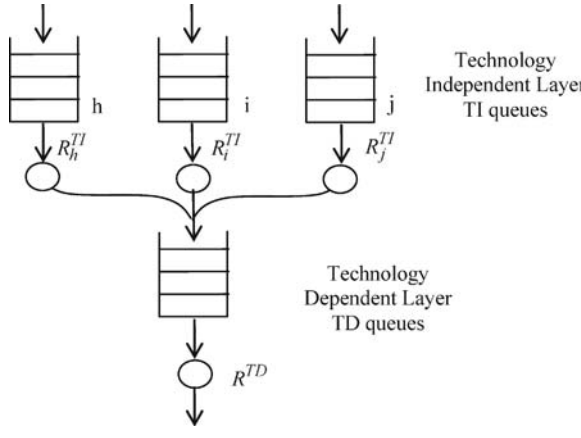


Figure 8.16 Heterogeneous traffic aggregation

The scheme reported in Figure 8.16 is close to reality. For example, referring to the satellite environment and to the related SI–SAP interface, which uses IP as upper layer, as outlined in [ETSI-TS102], "it is accepted in the BSM industry that at the IP level (above the SI–SAP interface) between 4 and 16 queues are manageable for different IP classes. Below the SI–SAP these classes can further be mapped into the satellite dependent priorities within the BSM which can be from 2 to 4 generally". The due association of IP QoS classes to SD transfer capabilities is also limited by hardware implementation constraints.

It is important to note that the problem is applicable both within the abstract queue model in Figures 8.13 and 8.14 and within the TD layer. As said, even if bandwidth allocation at TD layer may be hardly standardized to avoid the violation of implementers' freedom, it is a very interesting technological and scientific problem that this book would like to treat, at least giving some examples and theoretical indications.

8.6.3 Fading Effect

Finally, yet importantly, many transmission environments, such as satellite and wireless links, need to tackle time varying channel conditions due to fading. The model is reported in Figure 8.17.

From the mathematical viewpoint, fading effect may be modelled as a reduction of the bandwidth actually "seen" by the SD buffer, at least in satellite communications [Celandroni03]. The reduction is represented by a stochastic process $\phi(t)$. At time t, the "real" service rate $R_{real}^{TD}(t)$ (available for data transfer) is $R_{real}^{TD}(t) = R^{TD}(t) \cdot \phi(t)$, $\phi(t) \in [0, 1]$, where time dependency is explicitly indicated to enforce the concept of varying channel conditions. Actually, bandwidth allocation, if it is dynamic, is always varying over time.

8.6.4 Joint Problems

The three problems presented above may be seen jointly. The model is shown in Figure 8.18. It is clear that, in this context, the extreme variability of the bandwidth needs and the

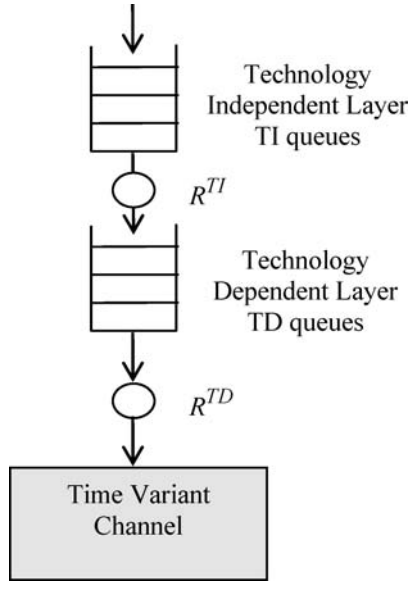


Figure 8.17 Time variant channel

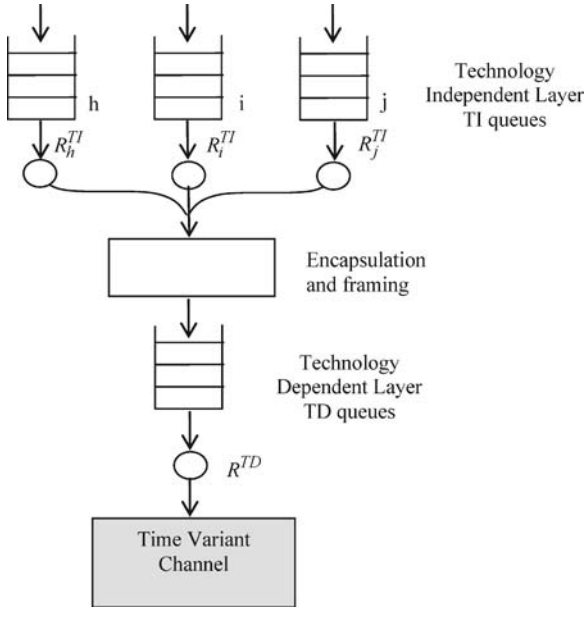


Figure 8.18 Joint model of vertical QoS mapping

complexity of the problem recommend an automatic management of the bandwidth through a proper control scheme.

It is obvious that also other solutions based on off-line measures and on a reasonable over-provision can give good results. The problem is stimulating from the scientific viewpoint. The next chapter is dedicated to present a possible idea to solve the mentioned problems limiting the information exchange between adjacent layers and without using off-line information. The chapter is dedicated to researchers interested to investigate this issue scientifically and to people of industrial research and development interested in applied mathematics and scientific details. The chapter can be ignored at a first reading.

9

Algorithm for Vertical QoS Mapping

9.1 Introduction

As said at the end of the previous chapter, this description contains many mathematical details and it is dedicated to people who have an interest in applied scientific research.

The ETSI-BSM (Broadband Satellite Multimedia) architecture (reported in Figure 5.4) is a good example and it is the reference for the operative example reported in this section. It is referred to satellite communication. As already said, the considered protocol stack separates the layers between the (SD) and the (SI). The former strictly depends on physical implementations often covered by industrial copyright. The latter is composed of IP and upper layers. The interface between SI and SD is defined as Satellite Independent–Service Access Point (SI-SAP). No use of abstract queue is used here to define the SI-SAP. QoS requirements must flow through SI-SAP and be implemented at SD layers. The use of primitives is implicit here. The issues mentioned in the previous chapter are topical when traffic is forwarded from SI to SD layers: change of encapsulation format; possible need of aggregating traffic with heterogeneous performance requirements and satellite channel fading. On the one hand, there is a strong need that SD layers provide a service (described in terms of performance metrics, formally included in a Service Level Specification – SLS) to the SI layers, but, on the other hand, it should be done with the minimum SD-SI bi-directional information exchange, which, ideally, should be limited to the performance requirement. The problem is related to automatic bandwidth adaptation. SD layer needs to compute the bandwidth to be assigned at SD buffer (i.e. the service rate) in real time so that the performance requirements fixed by SI layers can be satisfied.

The chapter proposes a novel control scheme, partially published in the literature [Marchese06] for the optimization of the bandwidth provision at SD layer. Multiple traffic classes implemented at SI layer and aggregated together at SD layer (thus leading to the

QoS over Heterogeneous Networks Mario Marchese

generation of statistically heterogeneous traffic trunks) are considered along with the joint performance metric composed of packet-loss probability and average delay. The aim is to "map" the QoS defined at SI layer onto SD technology.

9.2 Network Optimization: State of the Art

Optimization techniques for telecommunication networks usually follow the minimization of a proper functional cost capturing the system performance to get the best resource allocation so that $^{Opt}\theta = \arg\min_{\theta} \underset{\omega}{E}\{L[\theta, \omega]\}$.

In general, the system performance can be expressed according to specific metrics $L[\theta, \omega]$, such as blocking probability of connection requests, packet-loss probability, packet mean delay or delay jitter and service provider's revenue or network welfare. The vector θ contains decision variables related to the available resources (e.g. service rate, buffer size as expressed in the previous chapter referred to layer SD). The expectation $\underset{\omega}{E}\{L[\theta, \omega]\}$ is provided over all the feasible sample paths of the system (i.e. the set of all possible realizations of the stochastic processes involved in the system). These problems are often solved by means of centralized approaches, in which the control systems are strictly based on closed-form expressions of the performance measure. Such optimization approaches act according to parameter-adaptive certainty equivalent controls [Bertsekas01], where mapping between the current statistical behaviour of the system and the functional cost parameters must be periodically performed on-line, in order to maintain a good performance of the algorithms. Unfortunately, the conditions for the applicability of closed-form functional costs are hardly matched in real contexts. Control theory helps provide active measurement techniques suited for on-line network management, so avoiding analytical models of network phenomena: a wide range of control techniques have emerged to optimize the network performance; advanced techniques (as fuzzy tuning, neural networks, identification methodologies and learning theory) have also been studied and applied to obtain optimized performance. In this perspective, the principle of stochastic approximation [Kushner] is followed in this example by using a specific Perturbation Analysis technique [Wardi02, Cassandras03].

9.3 The SI-SAP QoS Mapping Problem

9.3.1 System Constraints and Assumptions

IP Packet Loss Probability (PLP) and IP Packet Average Delay (AD) are the chosen SLS performance metrics in this chapter. The approach can be generalized including other QoS constraints, such as delay jitter. The control mechanism does not need information about traffic statistics, hardware details (e.g. buffer length) and allocated resources, but only reference values and on-line measures, as suggested by the requirements indicated in the previous chapter.

9.3.2 Stochastic Fluid Model and Optimization Problem

The mathematical framework is based on Stochastic Fluid Models (SFM) [Wardi02, Cassandras03] of the SI-SAP traffic buffers. N SI queues and, without loss of generality, one single

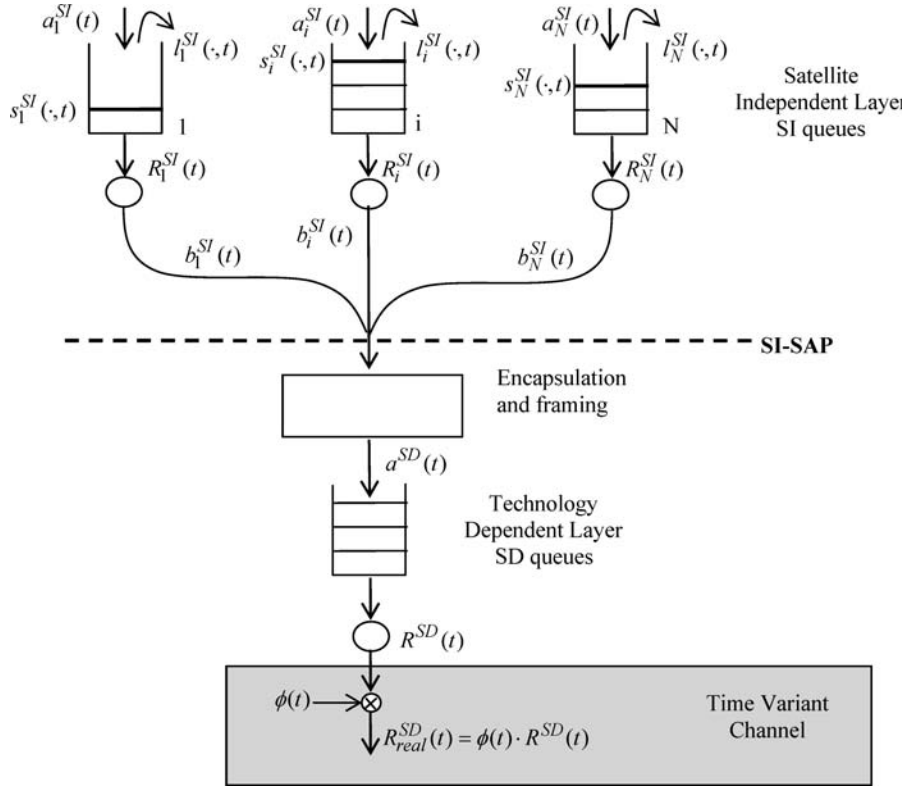


Figure 9.1 Joint model of vertical QoS mapping: SI-SAP interface and traffic formalism

SD queue are considered for the analytical formulation. Each SI queue contains a specific traffic class. Figure 8.18 is taken as reference, even if applied to the satellite (SI and SD) instead of the generic environment (TI and TD). Due to the need of reporting the analytical formulation of traffic flows and of a strict formalism, the reference scheme is reported again in Figure 9.1 together with all necessary formalism.

$a_i^{\mathrm{SI}}(t)$ and $b_i^{\mathrm{SI}}(t)$ are, respectively, the inflow and outflow rate processes of the traffic buffer i at SI layer at time t. SI buffers identifiers range in the interval $[1, N]$. The service rate of buffer i is $R_i^{\mathrm{TI}}(t)$, as indicated also in Chapter 8. The quantity $s_i^{\mathrm{SI}}(\cdot, t)$ is the workload (i.e. the fluid volume) in buffer i; in practice, it is the buffer i state. The function $l_i^{\mathrm{SI}}(\cdot, t)$ measures the overflow rate (i.e. the fluid interpretation of the packet loss) of buffer i. According to the chosen SFM, the loss volumes and the cumulative workloads of the buffers are the quantities representing the performance metrics of interest. The quantity $b_i^{\mathrm{SI}}(t)$ is obtained as in formula (1).

$$b_i^{\mathrm{SI}}(t) = \begin{cases} a_i^{\mathrm{SI}}(t), & \text{if } \mathrm{s}_i(\cdot, t) = 0 \\ R_i^{\mathrm{SI}}(t), & \text{if } \mathrm{s}_i(\cdot, t) \neq 0 \end{cases} \tag{1}$$

Let $a^{SD}(t)$ be the inflow rate process of the buffer at the SD layer at time t. It derives from the outflow rate processes of the SI buffers (or directly from the $a_i^{SI}(t)$ processes, if no buffering is applied at the SI layer) and from the change of the encapsulation format at the SI-SAP. ${}^iLoss_{Vol}^{SD}[a^{SD}(t), R^{SD}(t)\cdot\phi(t)]$ and ${}^i\text{Workload}_{Vol}^{SD}(a^{SD}(t), R^{SD}(t)\cdot\phi(t))$ define the loss volume and the cumulative workload of the i-th traffic class within the SD buffer. They are functions of the following elements: the SD inflow process $a^{SD}(t)$ (deriving from the aggregation of the SI inflow processes $a_i^{SI}(t)$, $i = 1, \ldots, N$, encapsulated in a specific transport technology), the fading process $\phi(t)$ and the SD bandwidth allocation $R^{SD}(t)$. ${}^i\text{Loss}_{Vol}^{SD}(a^{SD}(t), R^{SD}(t)\cdot\phi(t))$ and ${}^i\text{Workload}_{Vol}^{SD}[a^{SD}(t), R^{SD}(t)\cdot\phi(t)]$ have a topical role because they allow capturing the performance level of each traffic class i's IP Packet Loss Probability (PLP) and IP Packet Average Delay (AD), respectively. It is worth noting that no analytical expression for them is available, since there are no instruments for the mathematical description of the statistical behaviour of the packets belonging to a specific traffic class (i.e. $a_i^{SI}(t), i = 1, \ldots, N$) within an aggregated trunk (i.e. $a^{SD}(t)$). The key idea is to "equalize" the QoS measured at the SD layer in dependence of the QoS imposed by the SI layer. To capture this concept, it is useful to think at a "penalty cost function", whose values can be interpreted as an indication about the current inability of the SD layer to guarantee the required QoS. In practice, ${}^i\text{Loss}_{Vol}^{SD}(a^{SD}(t), R^{SD}(t)\cdot\phi(t))$ and ${}^i\text{Workload}_{Vol}^{SD}(a^{SD}(t), R^{SD}(t)\cdot\phi(t))$ chase the dynamic variations of the quantities (representative of the SI layer requests) identified in the following as ${}^i\text{Loss}_{Vol-Thr}^{SD}(t)$ and ${}^i\text{Workload}_{Vol-Thr}^{SD}(t)$, for packet loss and average delay, respectively. Concerning this issue, the chapter proposes two alternatives, reported below:

1) ${}^i\text{Loss}_{Vol}^{SD}[a^{SD}(t), R^{SD}(t)\cdot\phi(t)]$ and ${}^i\text{Workload}_{Vol}^{SD}[a^{SD}(t), R^{SD}(t)\cdot\phi(t)]$ chase threshold values (fixed or time varying) that flow from SI to SD layer through the proper primitives and are representative of the SLS (composed of PLP_i^* and AD_i^*, the packet loss probability and average delay performance requirement thresholds for the i-th traffic class). This method may be denoted as "feedback from SLS reference level". In practice, concerning ${}^i\text{Loss}_{Vol-Thr}^{SD}(t)$, it is the product of the required PLP_i^* and the sub-portion of the SD inflow relative to class i over a given time horizon $[k, k+1]$. It means that

$$ {}^i\text{Loss}_{Vol-Thr}^{SD}(t) = \text{PLP}_i^* \cdot \int_k^{k+1} b_i^{SI}(t)\text{d}t \tag{2} $$

where $b_i^{SI}(t)$ is the outflow rate process of the i-th SI buffer, and

$$ {}^i\text{Workload}_{Vol-Thr}^{SD}(t) = R^{SD}(t)\cdot\text{AD}_i^* - \text{DimPacket}_i(t) \tag{3} $$

where $\text{DimPacket}_i(t)$ is the size of the packet belonging to the i-th class and entering the SD buffer at time t. To simplify the control implementation, $\text{DimPacket}_i(t)$ may be substituted by $\overline{\text{DimPacket}_i(t)}$, the average packet size injected in the system by i-th service class.

$$ {}^i\text{Workload}_{Vol-Thr}^{SD}(t) = R^{SD}(t)\cdot\text{AD}_i^* - \overline{\text{DimPacket}_i(t)} \tag{4} $$

Packet service time is not considered in the delay (i.e. the delay constraint is referred to the time spent in the buffers). ${}^i\text{Workload}_{Vol-Thr}^{SD}(t)$ is a time-varying threshold of the maximum fluid volume allowed in the buffer to guarantee a delay per packet AD_i^*. In other words, ${}^i\text{Loss}_{Vol-Thr}^{SD}(t)$ and ${}^i\text{Workload}_{Vol-Thr}^{SD}(t)$ model the performance requirements that the SI layer

expects from the SD layer. Chasing the threshold means to assign the bandwidth at the SD layer so as to match the required performance.

2) Alternatively, ${}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}[a^{\mathrm{SD}}(t), R^{\mathrm{SD}}(t)\cdot\phi(t)]$ and ${}^{i}\mathrm{Workload}^{\mathrm{SD}}_{\mathrm{Vol}}[a^{\mathrm{SD}}(t), R^{\mathrm{SD}}(t)\cdot\phi(t)]$ can chase the performance measured at SI layer. It means that the SD layer must be informed about it by proper primitives. This method may be denoted as "feedback from measured SI losses". Some additional definitions are necessary for the theoretical model. ${}^{i}\mathrm{Loss}^{\mathrm{SI}}_{\mathrm{Vol}}[a^{\mathrm{SI}}_i(t), R^{\mathrm{SI}}_i(t)]$ and ${}^{i}\mathrm{Workload}^{\mathrm{SI}}_{\mathrm{Vol}}[a^{\mathrm{SI}}_i(t), R^{\mathrm{SI}}_i(t)]$ are the loss volume and the buffer cumulative workload of the i-th IP buffer according to the bandwidth allocation $R^{\mathrm{SI}}_i(t)$, respectively. Using the overflow rate $l^{\mathrm{SI}}_i(\cdot, t)$ and the state $s^{\mathrm{SI}}_i(\cdot, t)$ of buffer i, the loss volume and the buffer workload may be computed over a given time horizon $[k, k+1]$ as

$$ {}^{i}\mathrm{Loss}^{\mathrm{SI}}_{\mathrm{Vol}}(\cdot) = \int_{k}^{k+1} l^{\mathrm{SI}}_i(\cdot, t)\mathrm{d}t \tag{5} $$

$$ {}^{i}\mathrm{Workload}^{\mathrm{SI}}_{\mathrm{Vol}}(\cdot) = \int_{k}^{k+1} s^{\mathrm{SI}}_i(\cdot, t)\mathrm{d}t \tag{6} $$

In this case, the aim is the equalization of the PLP and AD of each traffic class between the SI and the SD layers. In other words, the SD layer follows the performance of the SI layer. Mathematically,

$$ {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol-Thr}}(t) = {}^{i}\mathrm{Loss}^{\mathrm{SI}}_{\mathrm{Vol}}[a^{\mathrm{SI}}_i(t), R^{\mathrm{SI}}_i(t)] \tag{7} $$

$$ {}^{i}\mathrm{Workload}^{\mathrm{SD}}_{\mathrm{Vol-Thr}}(t) = {}^{i}\mathrm{Workload}^{\mathrm{SI}}_{\mathrm{Vol}}[a^{\mathrm{SI}}_i(t), R^{\mathrm{SI}}_i(t)] \tag{8} $$

It is an interesting alternative to point (1): chasing the performance of the layer above, even if it makes worst the overall performance because the two layers are in cascade, may help save bandwidth when the performance of the SI layer is not satisfying. Actually, if the SI layer cannot guarantee a specific level of QoS, probably it is useless to provide effort at SD to assure the PLP and AD thresholds. Moreover, tracking the behaviour of the layer above (or a fraction of it) allows operating without the knowledge of a specific performance requirement at SD layer.

The optimization problem, called QoS Mapping Optimization (QoSMO) Problem, is aimed at finding the optimal bandwidth allocations $R^{\mathrm{SD}}_{\mathrm{Opt-Loss}}(t)$ and $R^{\mathrm{SD}}_{\mathrm{Opt-Workload}}(t)$, so that the cost functions $J_{\mathrm{Loss}}[\cdot, R^{\mathrm{SD}}(t)]$ and $J_{\mathrm{Workload}}[\cdot, R^{\mathrm{SD}}(t)]$ are minimized:

$$ R^{\mathrm{SD}}_{\mathrm{Opt-Loss}}(t) = \arg\min_{\mathrm{R^{SD}}(t)} \underset{\omega\in\Pi}{E} \left\{ J_{\mathrm{Loss}}[\cdot, R^{\mathrm{SD}}](t) \right\} \tag{9} $$

$$ R^{\mathrm{SD}}_{\mathrm{Opt-Workload}}(t) = \arg\min_{\mathrm{R^{SD}}(t)} \underset{\omega\in\Pi}{E} \left\{ J_{\mathrm{Workload}}[\cdot, R^{\mathrm{SD}}(t)] \right\} \tag{10} $$

$J_{\mathrm{Loss}}[\cdot, R^{\mathrm{SD}}(t)]$ is just an average measure over all the traffic classes of the penalty function computed at time t. $J_{\mathrm{Workload}}[\cdot, R^{\mathrm{SD}}(t)]$ is the corresponding measure for the workload.

$$ J_{\mathrm{Loss}}[\cdot, R^{\mathrm{SD}}(t)] = \sum_{i=1}^{N} \left\{ {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol-Thr}}(t) - {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}[a^{\mathrm{SD}}(t), R^{\mathrm{SD}}(t)\cdot\phi(t)] \right\}^2 \tag{11} $$

$$ J_{\mathrm{Workload}}(\cdot, R^{\mathrm{SD}}(t)) = \sum_{i=1}^{N} \left\{ {}^{i}\mathrm{Workload}^{\mathrm{SD}}_{\mathrm{Vol-Thr}}(t) - {}^{i}\mathrm{Workload}^{\mathrm{SD}}_{\mathrm{Vol}}[a^{\mathrm{SD}}(t), R^{\mathrm{SD}}(t)\cdot\phi(t)] \right\}^2 \tag{12} $$

It is important to note that the thresholds to be chased used in the two formulas reported above may be chosen either from (2), (4) or from (7), (8) based on the algorithm's aim. The quantity ω is a realization of the stochastic processes involved in the problem $\left[\phi(t), a_i^{\mathrm{SI}}(t), a^{\mathrm{SD}}(t)\right]$. The operator $\underset{\omega\in\Pi}{E}\{\cdot\}$ is the mean over the set Π of all possible sample paths. All the variables involved are evaluated at time t. It means that an adaptive control algorithm able to consider variable system conditions as well as number of sources in each traffic class, performance requests and fading levels is presented. The algorithm is aimed at equalizing the Loss and Workload measures taken at SD layer with the related thresholds by automatically adapting the bandwidth provided at the SD buffer.

An important observation is that one QoS constraint (e.g. PLP) reveals to be more stringent in terms of bandwidth requirements than the other one, and its satisfaction automatically assures the other constraint. It is true, also, if there are more than two constraints. For this reason, it is possible to assume that $R^{\mathrm{SD}}_{\mathrm{Opt-Loss}}(t) \neq R^{\mathrm{SD}}_{\mathrm{Opt-Workload}}(t)$, even if it is not possible to have a priori knowledge about the most stringent constraint at a given time instant t. So, it is possible to define the solution of the optimization problem. It is the quantity that can assure the QoS constraint for both loss and delay. In short, it is the object of the minimization and is identified as in (13).

$$R^{\mathrm{SD}}_{\mathrm{Opt}}(t) = \mathrm{Max}\left\{R^{\mathrm{SD}}_{\mathrm{Opt-Loss}}(t), R^{\mathrm{SD}}_{\mathrm{Opt-Workload}}(t)\right\} \tag{13}$$

In practice, the aim of the procedure is to get a control algorithm that computes the bandwidth requirements related to loss and delay, and assigns the bandwidth at SD layer according to the most stringent requirement. The solution of the optimization problem is obtained by capturing the current bandwidth need of the SD buffer through a sensitivity estimation procedure. The optimized approach is described below.

Being the involved stochastic processes unknown, the QoSMO problem is solved by taking loss volume and buffer cumulative workload measures over a given observation horizon $[k, k+1]$. These measures must be representative of the average values of the stochastic processes describing loss volume and buffer cumulative workload. The processes describing loss volume and buffer cumulative workload. The processes are supposed ergodic, at least in the convergence time of the algorithm. The length of the observation horizon is very important because, on one hand, it must be long enough to assure that the measures are representative of the average values of the stochastic processes, but, on the other hand, it must be short to assure quick convergence, as better explained in the following.

9.3.3 Reference Chaser Bandwidth Controller (RCBC)

The control scheme needs to minimize the cost functions $J_{\mathrm{Loss}}[\cdot, R^{\mathrm{SD}}(t)]$ and $J_{\mathrm{Workload}}[\cdot, R^{\mathrm{SD}}(t)]$, defined above and representing the distance between current performance and threshold. The derivative of both quantities is necessary to fulfil the operation, because it will be used in the gradient descent that is applied in the minimization.

Considering the PLP constraint first, the cost function $J_{\mathrm{Loss}}[\cdot, R^{\mathrm{SD}}(t)]$ derivative can be obtained from the computation reported in (14), directly applying the derivative operator to the definition in formula (11).

$$\frac{\partial J_{\mathrm{Loss}}[\cdot, R^{\mathrm{SD}}(t)]}{\partial R^{\mathrm{SD}}(t)} = 2\cdot\phi(t)\cdot\sum_{i=1}^{N}\frac{\partial {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}[R^{\mathrm{SD}}_{\mathrm{real}}(t)]}{\partial R^{\mathrm{SD}}_{\mathrm{real}}(t)}\left\{{}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}\left[R^{\mathrm{SD}}_{\mathrm{real}}(t)\right] - {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol-Thr}}(t)\right\} \tag{14}$$

where $R^{\mathrm{TD}}_{\mathrm{real}}(t) = \phi(t) \cdot R^{\mathrm{TD}}(t)$ is the real bandwidth used at SD layer, as explained above. The dependence of the threshold on the bandwidth assigned at SI layer is clear if the threshold choice depends both on (2) and (7). The two quantities ${}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}\left[R^{\mathrm{SD}}_{\mathrm{real}}(t)\right]$ and ${}^{i}\mathrm{Loss}_{\mathrm{Vol-Thr}}(t)$ may be measured. $\frac{\partial {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}\left[R^{\mathrm{SD}}_{\mathrm{real}}(t)\right]}{\partial R^{\mathrm{SD}}_{\mathrm{real}}(t)}$ is the unknown quantity. Before proposing a possible estimation algorithm, it is important to focus on the real physical meaning of $\frac{\partial {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}\left[R^{\mathrm{SD}}_{\mathrm{real}}(t)\right]}{\partial R^{\mathrm{SD}}_{\mathrm{real}}(t)}$. It represents the sensitivity of the packet loss to infinitesimal variations of the bandwidth serving the buffer at SD layer. So the help to compute it comes just from the Infinitesimal Perturbation Analysis (IPA), developed in the field of Sensitivity Estimation techniques for Discrete Event Systems. The theoretical description of IPA is contained in references [Wardi02] and [Cassandras03]. The $\frac{\partial {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}\left[R^{\mathrm{SD}}_{\mathrm{real}}(t)\right]}{\partial R^{\mathrm{SD}}_{\mathrm{real}}(t)}$ values may be obtained in real time on the basis of traffic samples acquired during the system evolution. In other words, they can be obtained by the simple observation of the system behaviour over a given time interval $[k, k+1]$ that separates two consecutive bandwidth allocations at SD layer. Also, intuitively, the sensitivity of the system loss depends on the speed with which the system passes from an empty to a full state. The periods of time in which the buffer is not empty are defined as busy periods. The periods when it is full are defined are full periods. The evaluation period $[k, k+1]$ is divided into N_k busy periods identified by the variable bp. The length of the busy period bp within $[k, k+1]$ is called B^{bp}_k. The derivative approximation is computed as

$$\frac{\partial {}^{i}\mathrm{Loss}^{\mathrm{SD}}_{\mathrm{Vol}}\left[R^{\mathrm{SD}}_{\mathrm{real}}(t)\right]}{\partial R^{\mathrm{SD}}_{\mathrm{real}}(t)} \cong -\sum_{\mathrm{bp}=1}^{N_k} \left\{ {}^{i}\mathrm{at}^{\mathrm{bp}}_{k}\left[R^{\mathrm{SD}}_{\mathrm{real}}(k)\right] - {}^{i}\mathrm{ll}^{\mathrm{bp}}_{k}\left[R^{\mathrm{SD}}_{\mathrm{real}}(k)\right] \right\} \tag{15}$$

where $\left\{ {}^{i}\mathrm{at}^{\mathrm{bp}}_{k}\left[R^{\mathrm{SD}}_{\mathrm{real}}(k)\right] - {}^{i}\mathrm{ll}^{\mathrm{bp}}_{k}\left[R^{\mathrm{SD}}_{\mathrm{real}}(k)\right] \right\}$ is the contribution to the SD loss volume of the i-th traffic class for the busy period B^{bp}_k within the decision interval $[k, k+1]$; ${}^{i}\mathrm{at}^{\mathrm{bp}}_{k}$ is the arrival time of the first packet of service class i within the busy period B^{bp}_k; ${}^{i}\mathrm{ll}^{\mathrm{bp}}_{k}$ is the time when the last loss of class i occurs during B^{bp}_k. If there is just one traffic class, (15) is an equality as proved in [Wardi02] and used in [Marchese06]. It is only an approximation in case of traffic-class aggregation. The equality has not been proofed for the multiple-class case. In this case, the class index i becomes useless, $\mathrm{at}^{\mathrm{bp}}_{k}$ is the time when the bp-th busy period begins (i.e. the instant when the first packet enters the buffer) and $\mathrm{ll}^{\mathrm{bp}}_{k}$ is the time when the last full period of the bp-th busy period finishes (i.e. being in a fluid model, it can be identified with the instant of the last loss within the bp-th busy period). It is rather intuitive in this single-class case, but the same method may be applied to the multiple-class case. The quantities ${}^{i}\mathrm{at}^{\mathrm{bp}}_{k}$ and ${}^{i}\mathrm{ll}^{\mathrm{bp}}_{k}$ are aimed at estimating the sensitivity of class i loss volume to bandwidth allocation. Figure 9.2 shows the practical application of the concepts explained above over the single busy period. The single-class case is shown.

Concerning the delay AD, the derivative of the cost function $J_{\mathrm{Workload}}[\cdot, R^{\mathrm{SD}}(t)]$ can be obtained through (16) as done in (14) for the loss.

$$\frac{\partial J_{\mathrm{Workload}}[\cdot, R^{\mathrm{SD}}(t)]}{\partial R^{\mathrm{SD}}(t)} = 2 \cdot \phi(t) \cdot \sum_{i=1}^{N} \frac{\partial\, {}^{i}Workload^{\mathrm{SD}}_{\mathrm{Vol}}[R^{\mathrm{SD}}_{\mathrm{real}}(t)]}{\partial R^{\mathrm{SD}}_{\mathrm{real}}(t)}$$

$$\left\{ {}^{i}Workload^{\mathrm{SD}}_{\mathrm{Vol}}\left[R^{\mathrm{SD}}_{\mathrm{real}}(t)\right] - {}^{i}Workload_{\mathrm{Vol-Thr}}(t) \right\} \tag{16}$$

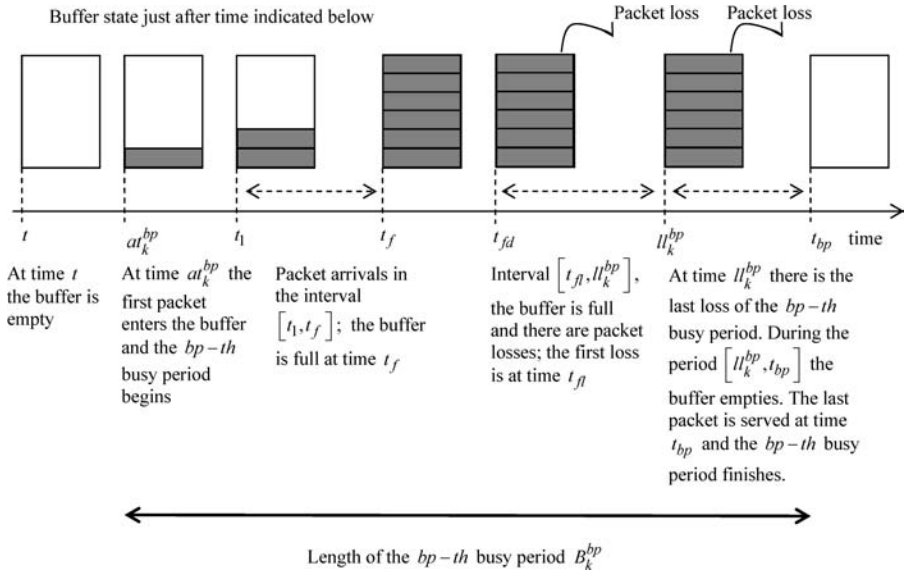


Figure 9.2 Buffer behaviour over the single busy period

Again, all quantities may be measured except for $\frac{\partial^i Workload_{\text{Vol}}^{\text{SD}}[R_{\text{real}}^{\text{SD}}(t)]}{\partial R_{\text{real}}^{\text{SD}}(t)}$ that can be obtained by using the IPA estimator for the buffer cumulative workload. The service rate $R_{\text{real}}^{\text{SD}}$ is the parameter.

$$\frac{\partial^i Workload_{\text{Vol}}^{\text{SD}}[R_{\text{real}}^{\text{SD}}(t)]}{\partial R_{\text{real}}^{\text{SD}}(t)} \cong -\frac{1}{2}\sum_{\text{bp}=1}^{N_k}\sum_{\text{fp}=1}^{M_k^{\text{bp}}+1}\left\{{}^i at_{\text{bp,fp}}\left[R_{\text{real}}^{\text{SD}}(k)\right] - {}^i ll_{\text{bp,fp}-1}\left[R_{\text{real}}^{\text{SD}}(k)\right]\right\}^2 \qquad (17)$$

For each busy period B_k^{bp}, $bp = 1, \ldots, N_k$ within the decision interval $[k, k+1]$, there are M_k^{bp} periods where the buffer is full (as, e.g., in Figure 9.2, the period $\left[t_{\text{fl}}, ll_k^{\text{bp}}\right]$, which is also the last and unique full period within the busy period B_k^{bp}). They are called full period and identified as $F_k^{\text{bp,fp}}$, $fp = 1, \ldots, M_k^{\text{bp}}$. The time instants ${}^i at_{\text{bp,fp}}[R_{\text{real}}^{\text{SD}}(k)]$ and ${}^i ll_{\text{bp,fp}}[R_{\text{real}}^{\text{SD}}(k)]$ represent the arrival time of the first packet and the last packet loss for service class i within the full period $F_k^{\text{bp,fp}}$, $fp = 1, \ldots, M_k^{\text{bp}}$. Obviously, the quantity ${}^i ll_{\text{bp,fp}-1}(R_{\text{real}}^{\text{SD}}(k))$ appearing in (17) is the last loss for service class i within the full period $F_k^{\text{bp,fp}-1}$. Two quantities are still undefined in (17): ${}^i ll_{\text{bp},0}\left[R_{\text{real}}^{\text{SD}}(k)\right]$ and ${}^i at_{\text{bp},M_k^{\text{fp}}+1}\left[R_{\text{real}}^{\text{SD}}(k)\right]$. The former has been already defined above. It is the time of the first packet arrival of service class i within B_k^{bp}. In short, ${}^i ll_{\text{bp},0}\left[R_{\text{real}}^{\text{SD}}(k)\right] = {}^i at_k^{\text{bp}}$. The latter is the last packet arrival of service class i within B_k^{bp}. Again, the equality holds [Wardi02], if just one traffic class enters the buffer: $at_{\text{bp,fp}}\left[R_{\text{real}}^{\text{SD}}(k)\right]$ is the time when the fp-th full period within the bp-th busy period begins (i.e. the arrival time of the packet filling the buffer) and $ll_{\text{bp,fp}-1}\left[R_{\text{real}}^{\text{SD}}(k)\right]$ is the last instant of the $(fp-1)$-th full period within the busy period bp-th (which may be considered the time of the last packet loss, being a fluid model). Figure 9.3 reports the graphical interpretation of the reported definitions.

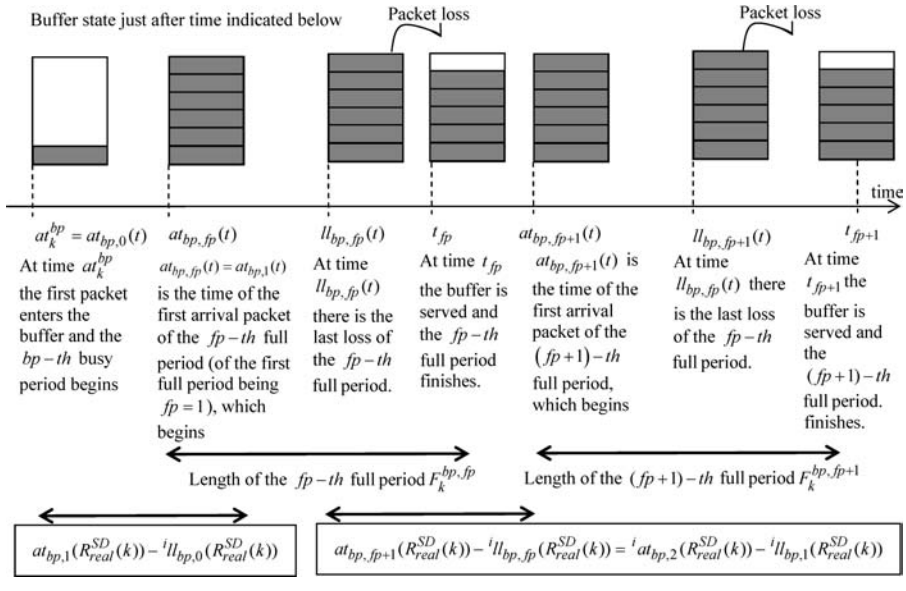


Figure 9.3 Buffer behaviour over full periods

Both for loss $\left\{ \frac{\partial\ {}^{i}Loss_{\mathrm{Vol}}^{\mathrm{SD}}\left[R_{\mathrm{real}}^{\mathrm{SD}}(t)\right]}{\partial R_{\mathrm{real}}^{\mathrm{SD}}(t)} \right\}$ and delay $\left\{ \frac{\partial\ {}^{i}Workload_{\mathrm{Vol}}^{\mathrm{SD}}\left[R_{\mathrm{real}}^{\mathrm{SD}}(t)\right]}{\partial R_{\mathrm{real}}^{\mathrm{SD}}(t)} \right\}$, the key idea is to measure the contribution of the IP packets belonging to the *i-th* traffic class to the lengths of the busy and full periods of the SD buffer. These quantities allow "measuring" the QoS received by the *i-th* traffic class within the aggregated trunk. Equations (14) and (16) imply that the gradient estimation is performed with respect to the "real" service rate $R_{\mathrm{real}}^{\mathrm{SD}}$, so capturing the effects of the realization of the fading stochastic process in each decision period.

The proposed optimization algorithm performs a sequence of bandwidth reallocations $R^{\mathrm{SD}}(k), k = 1, 2, \ldots$; it is based on the gradient method, whose descent steps are ruled by the formulae (18)–(21). The notations *Loss* and *Workload* are substituted with the compact indication $QoS-req = Loss, Workload$. It means that $QoS-req$ can assume both value *Loss* and *Workload*, and allows avoiding formula duplications. The notation $QoS-req-Max$ indicates the most restrictive requirement at the decisional instant.

$$R_{\mathrm{QoS-req}}^{\mathrm{SD}}(k+1) = R_{\mathrm{QoS-req}}^{\mathrm{SD}}(k) - step_{\mathrm{QoS-req}}^{k} \cdot \left. \frac{\partial J_{\mathrm{QoS-req}}[\cdot, R^{\mathrm{SD}}(t)]}{\partial R^{\mathrm{SD}}(t)} \right|_{R^{\mathrm{SD}}(t)=R^{\mathrm{SD}}(k)} \tag{18}$$

$$R^{\mathrm{SD}}(k+1) = R_{\mathrm{QoS-req-Max}}^{\mathrm{SD}}(k+1) \tag{19}$$

$$QoS\text{–}req\text{–}Max = \arg \underset{\mathrm{QoS-req}}{Max} \left\{ \left. \frac{\partial \bar{J}_{\mathrm{QoS-req}}\left[\cdot, R^{\mathrm{SD}}(t)\right]}{\partial R^{\mathrm{SD}}(t)} \right|_{R^{\mathrm{SD}}(t)=R^{\mathrm{SD}}(k)}, QoS\text{–}req = Loss, Workload \right\} \tag{20}$$

$\frac{\partial \bar{J}_{\text{QoS}-\text{req}}[\cdot, R^{\text{SD}}(t)]}{\partial R^{\text{SD}}(t)}$ is the normalized cost derivative, defined in (21).

$$\frac{\partial \bar{J}_{\text{QoS}-\text{req}}[\cdot, R^{\text{SD}}(t)]}{\partial R^{\text{SD}}(t)} = \frac{\frac{\partial J_{\text{QoS}-\text{req}}[\cdot, R^{\text{SD}}(t)]}{\partial R^{\text{SD}}(t)}}{\frac{\partial J^{\text{Max}}_{\text{QoS}-\text{req}}[\cdot, R^{\text{SD}}(t)]}{\partial R^{\text{SD}}(t)}}, \quad QoS\text{–}req = Loss, Workload \tag{21}$$

The quantity referenced as $\frac{\partial J^{\text{Max}}_{\text{QoS}-\text{req}}[\cdot, R^{\text{SD}}(t)]}{\partial R^{\text{SD}}(t)}$ is the maximum value achievable for $\frac{\partial J_{\text{QoS}-\text{req}}[\cdot, R^{\text{SD}}(t)]}{\partial R^{\text{SD}}(t)}$ over a given decision period. The two gradient stepsizes are $\text{step}^k_{\text{Loss}}$ and $\text{step}^k_{\text{Workload}}$, summarized in $\text{step}^k_{\text{QoS}-\text{req}}$ to reduce notation and computed for the reallocation instant k. In practice, the need to find out the most stringent QoS requirement is expressed in (20). The direct comparison of the normalized sensitivity estimators $\frac{\partial \bar{J}_{\text{QoS}-\text{req}}[\cdot, R^{\text{SD}}(t)]}{\partial R^{\text{SD}}(t)}$ reveals to be a precise tool to detect the most sensitive QoS threshold.

Since both ${}^i\text{Loss}^{\text{SD}}_{\text{Vol}}(\cdot)$ and ${}^i\text{Workload}^{\text{SD}}_{\text{Vol}}(\cdot)$ are the loss and workload volumes of traffic queues for class i, they can be reasonably assumed to be continuous, differentiable, with a negative derivative in the service rate. As a consequence, the penalty cost functions $J_{\text{Loss}}[\cdot, R^{\text{SD}}(t)]$ and $J_{\text{Workload}}[\cdot, R^{\text{SD}}(t)]$ are also continuous, differentiable with unique minima $R^{\text{SD}}_{\text{Opt}-\text{Loss}}(t)$ and $R^{\text{SD}}_{\text{Opt}-\text{Workload}}(t)$. So, in ergodic conditions, $R^{\text{SD}}_{\text{QoS}-\text{req}}(k) \overset{k\to+\infty}{\longrightarrow} R^{\text{SD}}_{\text{Opt}-\text{QoS}-\text{req}}$, QoS–req = Loss, Workload. It may be argued that the presence of (20) introduces discontinuities in the rate allocations, thus leading to a sequence of rate reallocations, $R^{\text{SD}}(k), k = 1, 2, \ldots,$ not converging to the optimal solution of the QoSMO problem. However, during the convergence process in ergodic conditions, a performance metric sensitivity reveals to be higher than the other one. This is the rationale under (20). In practice, SD reallocations are driven to the steady state by the performance sensitivity currently more sensible to the rate reallocations. A new optimal operating point of the system is obtained assuring the SLS preservation.

The main challenge is to automatically adapt $R^{\text{SD}}(t)$ to counteract variable system conditions, either when the required performance thresholds are redefined or when the traffic statistics and the fading change. Due to the regularity of the penalty cost functions, RCBC can efficiently track the optimal solution. In this perspective, the major concern is related to the ergodicity of the involved stochastic processes. It is necessary to assume that RCBC convergence towards a new operation point is faster than the changes of the stochastic environment.

Some conditions are necessary to guarantee the convergence of the algorithm but they are left to specific research papers.

9.3.4 Alternative Approach: Equivalent Bandwidth Heuristic

Additionally to the practical and simple scheme that evaluates the overhead imposed by the change of encapsulation described in detail in section 8.6.1 for the ATM technology, the equivalent bandwidth technique, explained in section 4.1.3, may be used for bandwidth allocation in vertical QoS mapping problems. As said in section 4.1.3, the equivalent bandwidth techniques are based on the statistical characterization of the traffic by means of descriptors such as peak rate, mean rate and maximum burst size. The inflow process $a^{\text{SD}}(t)$ is the result of the aggregation of the outflow processes of the SI buffers. The mentioned traffic

descriptors may be hardly applied, but the technique described in section 4.3.1 may be still applied. It is important to know the mean value $m_a{}^{SD}$ and the standard deviation $\sigma_{a^{SD}}$ of the $a^{SD}(t)$ process. It is called "Equivalent Bandwidth – EqB" approach. As well as RCBC, EqB exploits on-line measures without assuming any a priori knowledge of traffic statistics and buffer size. With $k = 1, 2, \ldots$ the time instants of the SD rate reallocations, $m_{a^{SD}}(k)$ and $\sigma_{a^{SD}}(k)$ being the mean and the standard deviation, respectively, of the SD inflow process measured over the time interval $[k, k+1]$, the bandwidth provision $\left[R^{SD}(k+1)\right]$ at the SD layer, assigned over the time interval $[k+1, k+2]$, is computed as in (22).

$$R^{SD}(k+1) = m_{a^{SD}}(k) + z(\varepsilon) \cdot \sigma_{a^{SD}}(k) \tag{22}$$

where $z(\varepsilon) = \sqrt{-2\ln(\varepsilon) - \ln(2\pi)}$ and ε is the PLP upper bound, as defined in section 4.3.1. The operative proposal is identified as EqB in the following. An important note concerns the heterogeneity of the required QoS levels within SD layer aggregate. Equivalent bandwidth is the minimum rate allocation necessary to guarantee a specific QoS to a single statistically homogeneous flow. The definition is generalized to be applied in this example, since the minimum R^{SD} satisfying all the QoS levels requested by the SI classes is the target. EqB approach may be adapted to this need, as done here, by choosing the parameter ε as the most stringent PLP* required at the SI layer and by properly dimensioning the buffer size to control the most demanding AD* constraint.

9.4 Performance Analysis

This paragraph is aimed at showing the performance of the presented algorithm and stimulating further research in this important field. A C++ simulator has been developed for the SI-SAP architecture of Figure 9.1, taken as reference for the tests. The first part, widely taken from [Marchese06], is dedicated to the evaluation of the scheme to solve only the problem related to the change of encapsulation format as evidenced in section 8.6.1. The second part concerns the traffic aggregation issue, as described in section 8.6.2. The third part is related to the fading problem, as suggested in section 8.6.3.

9.4.1 Encapsulation

The scenario presented in Figure 8.15 is considered for the test. The IP over ATM scenario (e.g. EF traffic over ATM) is taken into account and AAL5 is based on LLC-SNAP encapsulation. RCBC faces only the encapsulation problem for now, as described above.

The convergence behaviour of the proposed control algorithm together with its capability of tracking a time variant bandwidth requirement in different data traffic conditions is shown. The bandwidth assignment of RCBC is compared with the allocation computed by the CellTaxAllocation presented in section 8.6.1, also highlighting the equivalent bandwidth shift due to the change of the encapsulation format for different SD buffer sizes. VoIP traffic is used in this last case. The aim is to evidence RCBC behaviour chasing both the losses measured at SI layer as in formula (7) and the loss volume computed from a fixed SLS threshold as in (2).

9.4.1.1 Convergence Behaviour and Tracking Capability of the Control Algorithm

In this scenario, the IP traffic flow entering the SI–SAP interface is composed of ten aggregated sources, each of them characterized by the Trimodal distribution, which concerns the packet size. According to it, the packet size can assume three different values: q with probability p_q, r with probability p_r and s with probability $p_s = 1 - p_q - p_r$. The Trimodal distribution is widely used for current Internet traffic. The following values taken from [Mellia02] are assumed: $q = 48$ bytes and $p_q = 0.559$, $r = 576$ bytes and $p_r = 0.2$, $s = 1500$ bytes. The corresponding notation is Trimodal (48, 576, 1500, 0.559, 0.2). It is used for each active IP source in the following.

The duration of the simulation is 120.0 s. In order to verify the adaptive capability of the RCBC, the IP sources' rate is increased after 1 min of simulation. It passes from 1.0 Mbps to 2.0 Mbps. The Service Level Specification (SLS) is represented at SI layer by the Packet Loss Probability (PLP). It is set to $3 \cdot 10^{-2}$. The bandwidth necessary at SI layer R^{SI} to satisfy this requirement is 9.55 Mbps during the first minute and 19.0 Mbps for the rest of the test. It is computed by simulation. Both the SI buffer and SD buffer sizes are set to 150,000 bytes. The losses measured at SI layer as in formula (7) are tracked by the SI layer.

Figure 9.4 shows a sample path of the PLP measured at the buffers versus time. RCBC is applied by setting the time between two SD reallocations to 0.2 s and the gradient stepsize to 10^{-3}. The figure allows giving a first idea of the quick reaction of the scheme to traffic variations. No bandwidth is provided to SD buffer at the traffic load change instant in the case presented. It means that the tracking of the bandwidth needed is started from the null value. The convergence time is much reduced if the bandwidth allocated to SD buffer "starts" from heuristic measures, as should be clearer from the results reported in the following. The proposed control algorithm is able to equalize the PLP of the two buffers after a transient period of about 10.0 s.

Obviously, the optimal value of R^{SD} is time-dependent according to the time-varying IP sources' behaviour. The behaviour of RCBC is clear from Figure 9.5 that shows the PLP

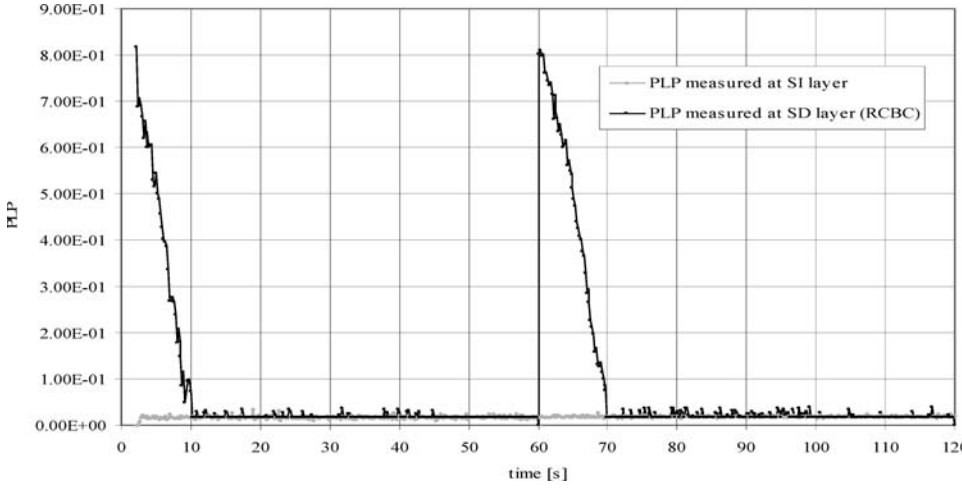


Figure 9.4 PLP at the SI and SD buffers of the SI–SAP interface

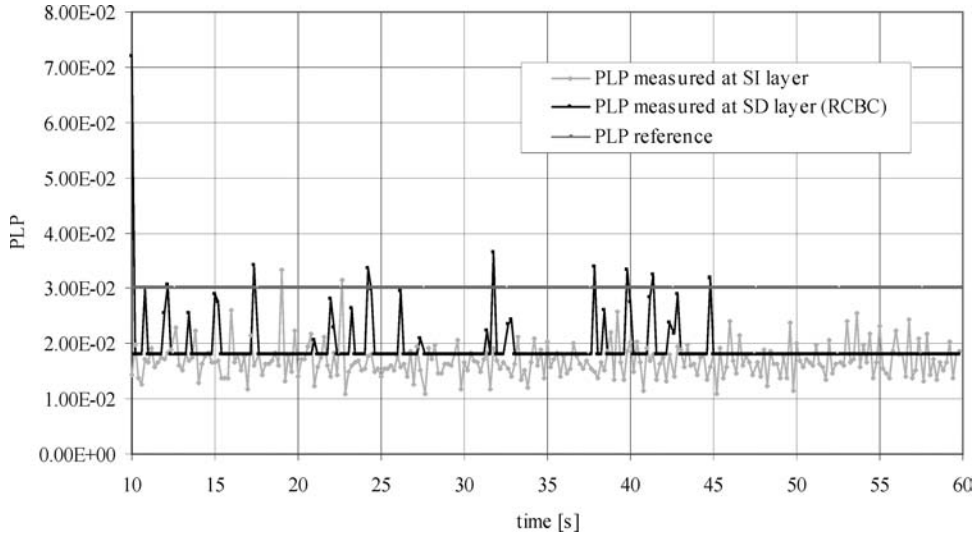


Figure 9.5 RCBC behaviour over time

in the first minute of simulation after reaching the optimal bandwidth value steady state. The bandwidth necessary for SD layer is obtained and a proper "QoS mapping" is applied to guarantee the required SLS, despite the changes in the transport technology and in the inflow rate of the IP sources. RCBC tracks a time variable threshold (SI layer PLP) and can respect the required SLS almost constantly, except for sporadic instants due to quick traffic (and threshold) variations.

9.4.1.2 Comparison with CellTax Heuristic

RCBC is compared with the CellTax allocation heuristic, described in detail in section 8.6.1. CellTax has a perfect knowledge about the bandwidth assignment at the SI layer and about the SI packet size distribution. Traffic conditions are varied. Inter-arrival time between IP packets is modelled by a Pareto distribution. The used mean inter-arrival times are 10 ms and 100 ms and the number of connections in the IP flow is 1, 20 and 100. As in the previous section, the packet size distribution is Trimodal (48, 576, 1500, 0.559, 0.2). The SI buffer size is set to 150,000 bytes (2830 ATM cells). The SLS at SI layer is set to $PLP \leq 10^{-2}$ and R^{SI} is allocated consequently. Table 9.1 contains the comparison between R^{SI} and R^{SD}, both computed by RCBC and CellTax. They are indicated through ${}^{RCBC}R^{SD}$ and ${}^{CellTax}R^{SD}$, respectively. The values are reported for different: mean inter-arrival time of the IP packets (InterTime IP Packets), number of connections (sources) in the flow (#Conn) and buffer length at SD layer, set to 2830 ATM cells (as the SI buffer size) and to 200 ATM cells (well below the SI length). Table 9.1 compares the two allocation methods by showing the bandwidth required at SD layer. Table 9.2 shows the additional bandwidth in percentage required by RCBC and CellTax, taking R^{SI} as a reference. Table 9.3 directly compares RCBC and CellTax by showing the difference of the bandwidth allocation between CellTax and RCBC and the percentage difference taking RCBC as reference. In practice, Table 9.3

Table 9.1 RCBC and CellTax bandwidth allocation

SD buffer (ATM cells)	IP packet inter-arrival time	Number of calls	R^{SI} (Mbps)	${}^{CellTax}R^{SD}$ (Mbps)	${}^{RCBC}R^{SD}$ (Mbps)
2830	0.01	1	0.28	0.337	0.352
2830	0.01	20	5.7	6.858	7.024
2830	0.01	100	28.5	34.288	34.944
2830	0.1	1	0.039	0.047	0.048
2830	0.1	20	0.78	0.938	0.964
2830	0.1	100	4.1	4.933	4.957
200	0.01	1	0.29	0.349	0.413
200	0.01	20	5.75	6.918	8.075
200	0.01	100	28.75	34.589	41.657
200	0.1	1	0.038	0.046	0.06
200	0.1	20	0.78	0.938	1.13
200	0.1	100	3.9	4.692	5.556

Table 9.2 RCBC and CellTax additional bandwidth in percentage

SD buffer (ATM cells)	IP packet inter-arrival time	Number of calls	$\frac{{}^{CellTax}R^{SD} - R^{SI}}{R^{SI}} \cdot 100$ (%)	$\frac{{}^{RCBC}R^{SD} - R^{SI}}{R^{SI}} \cdot 100$ (%)
2830	0.01	1	20.31	25.71
2830	0.01	20	20.31	23.23
2830	0.01	100	20.31	22.61
2830	0.1	1	20.31	23.08
2830	0.1	20	20.31	23.59
2830	0.1	100	20.31	20.90
200	0.01	1	20.31	42.41
200	0.01	20	20.31	40.43
200	0.01	100	20.31	44.89
200	0.1	1	20.31	57.89
200	0.1	20	20.31	44.87
200	0.1	100	20.31	42.46

shows the bandwidth missing to CellTax allocation to satisfy performance requirements and the percentage bandwidth underestimation performed by CellTax.

The width of the confidence interval over the simulated loss measures is less than 1% for 95% of the cases.

As shown previously, ${}^{RCBC}R^{SD}$ obtained after the convergence of the RCBC should converge with the optimal value to dimension the bandwidth necessary at SD layer and it can be taken as the target value for the following comparison.

From these results, it is clear that CellTax produces satisfying results only if the buffers have the same size. If the SD buffer is underdimensioned with respect to SI buffer, CellTax allocation underestimates the bandwidth up to 23%. On the other hand, RCBC reveals to be a promising tool for planning the SI–SAP interface. It is worth noting that RCBC, without any

Table 9.3 RCBC and CellTax direct comparison

SD Buffer (ATM cell)	IP packet inter-arrival time	Number of calls	${}^{\text{CellTax}}R^{\text{SD}} - {}^{\text{RCBC}}R^{\text{SD}}$ (Mbps)	$\frac{{}^{\text{CellTax}}R^{\text{SD}} - {}^{\text{RCBC}}R^{\text{SD}}}{{}^{\text{RCBC}}R^{\text{SD}}} \cdot 100$ (%)
2830	0.01	1	−0.0151	−4.3003
2830	0.01	20	−0.1664	−2.3695
2830	0.01	100	−0.6562	−1.8778
2830	0.1	1	−0.0011	−2.2496
2830	0.1	20	−0.0256	−2.6552
2830	0.1	100	−0.0244	−0.4915
200	0.01	1	−0.0641	−15.5221
200	0.01	20	−1.1573	−14.3317
200	0.01	100	−7.0684	−16.9681
200	0.1	1	−0.0143	−23.8048
200	0.1	20	−0.1916	−16.9554
200	0.1	100	−0.8640	−15.5504

a priori knowledge about statistical behaviour of the traffic sources, encapsulation format and buffer dimension, guarantees the correct QoS mapping, despite the changes in the traffic flow. Only information about the amount of losses is necessary to apply the algorithm. It means that, in theory, the notion of buffer might also be dropped without affecting the validity of the control mechanism. It is a powerful result for future applications. Reference [Marchese06] shows the precise additional bandwidth computed by RCBC by varying the SD buffer size. Its measure is outside the scope of this book, but the information about the algorithm efficiency independently of the SD buffer length is essential to match the requirements fixed in Chapter 8.

9.4.1.3 Additional Information About Convergence

The time that RCBC requires to reach the optimal SD allocation can be reduced both by changing the initial bandwidth value and by means of a proper tuning of the time interval between two SD bandwidth reallocations and by means of the gradient stepsize. If the values of the gradient stepsize are too large, instability in the SD allocation arises; if, on the contrary, they are too small, a long time is necessary to reach the optimal steady state. A proper trade-off between the two parameters must be investigated. For instance, satisfying time interval between two SD bandwidth reallocations and gradient stepsize values are 0.1 s and $3 \cdot 10^{-3}$, respectively. The transient periods are reduced from 10 to about 4 s by using these values for the simulation tests in Figures 9.4 and 9.5.

9.4.1.4 RCBC Behaviour: Feedback from Measured SI Losses and from SLS Reference Level

The aim is to show the RCBC behaviour with a fixed SLS threshold as indicated in formula (2). It is a common situation in real BSM networks where SI layer passes the performance threshold to SD layer through the primitives, as said in Chapter 8. The case of a Voice over IP (VoIP) traffic, coming from a single EF queue at SI layer, and conveyed to a single ATM

buffer at SD layer is considered. Each VoIP source is modelled similarly as in [Byungsuk02] as an exponentially modulated on–off process, with mean on and off times equal to 1.008 s and 1.587 s, respectively, as indicated in [ITU-T-P.59], and, during on-periods it generates 16.0 Kbps over RTP/UDP/IP stack. The mean rate is 6.22 Kbps; VoIP packet size is 80 bytes. The required SLS performance threshold of VoIP flows is set to 2% of PLP.

SI bandwidth provision R^{SI} has been already dimensioned to guarantee the required PLP of SI layer. As said above, two implementation schemes of RCBC are considered: (1) PLP at the SD buffer chases the packet losses measured at the SI buffer, as in the results presented up to now; (2) SD PLP chases the performance level defined in the SLS (2%). They have denoted as "feedback from measured SI losses" and "feedback from SLS reference level", respectively, in section 9.3.2.

The performance is evaluated by increasing the number of VoIP sources in the flow from 70 to 110 over time. The increasing step is of ten sources to stress the working conditions and the reaction of the algorithm. Heuristic based on CellTax introduced in section 8.6.1 is used to initialize the RCBC gradient descent in order to speed up the convergence and to assure fast reactions to time-varying system conditions. The used initialization is $R^{SD}(0) = R^{SD}_{init} = (1 + CellTax) \cdot R^{SI}$. RCBC's bandwidth initialization is performed again when the number of VoIP source changes every 3000 s. A new SD bandwidth allocation is performed through the gradient descent every 30 s. The gradient stepsize is fixed to 7.0 in the "feedback from measured SI losses" case and to 11.0 in the "feedback from SLA reference level" case. They are the best values found by simulation inspection to maximize the convergence speed and to avoid strong SD rate oscillations.

The performance of RCBC is reported in Figure 9.6 for "feedback from measured SI losses" and in Figure 9.7 for "feedback from SLS reference level". Results are very satisfying in both cases. The convergence speed is outstanding.

Table 9.4 reports the average PLPs over the entire simulation horizon. It is easily observable that RCBC that uses measured SI losses yields a PLP very close to the IP one. RCBC chasing SLS threshold satisfies the required performance and allows reaching an average PLP well below the value obtained through CellTaxAllocation, which even if it assures an average value below the threshold, it has a very "jumpy" behaviour not completely satisfying over many periods, as shown in Figure 9.8.

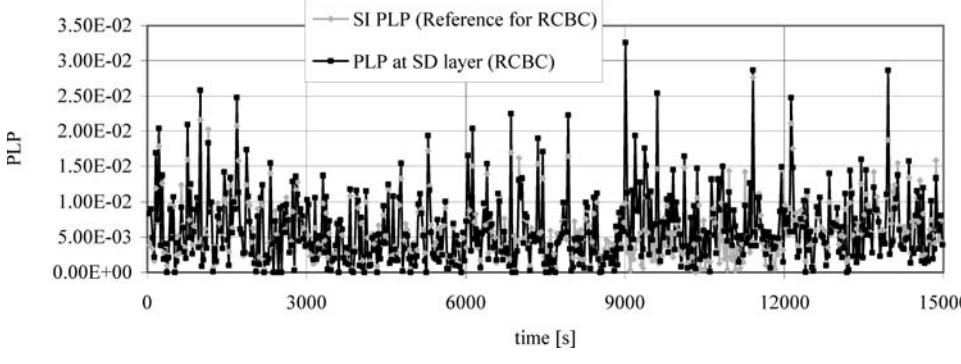


Figure 9.6 RCBC performance: PLP SI and SD layers, feedback from measured SI losses

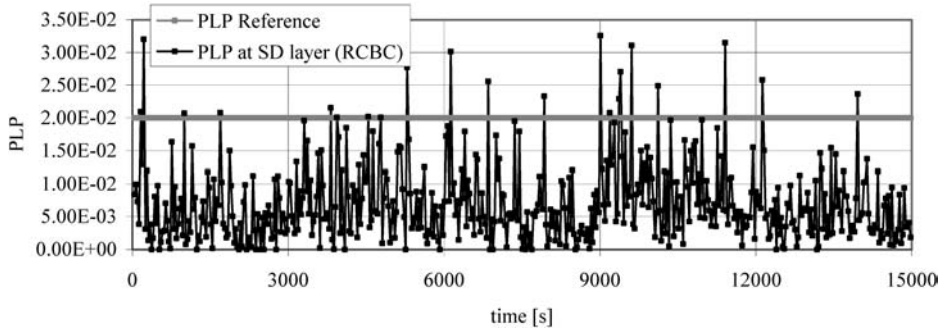


Figure 9.7 RCBC performance: PLP SI and SD layers, feedback from SLS reference level

Table 9.4 Average PLP over the entire simulation

PLP at SI layer	PLP at SD layer RCBC: feedback from measured SI losses	PLP at SD layer RCBC: feedback from SLS reference level	PLP at SD layer CellTax
$5.78 \cdot 10^{-3}$	$6.16 \cdot 10^{-3}$	$7.03 \cdot 10^{-3}$	$1.55 \cdot 10^{-2}$

Both RCBC and CellTax strategies use the same size of the SI and SD buffers, which is fixed to 20 VoIP packets corresponding to 31 ATM cells. Buffer sizes are short to assure low propagation delays at each SI-SAP, thus limiting the delay, which is not part of the SLS for now.

Despite the change in the transport technology, RCBC assures QoS mapping by tracking the best operative point of the system. The bandwidth provisioning at both SI and SD layers is compared in Figure 9.9 to highlight the bandwidth shift computed by RCBC when the encapsulation format changes.

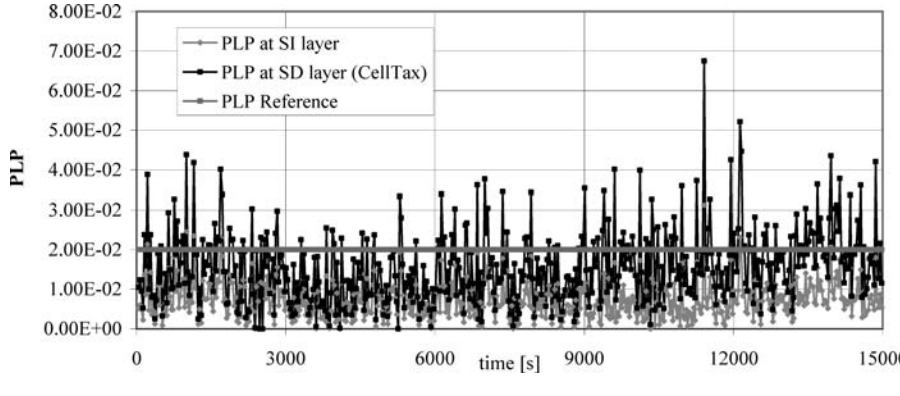


Figure 9.8 CellTax performance: PLP SI and SD layers

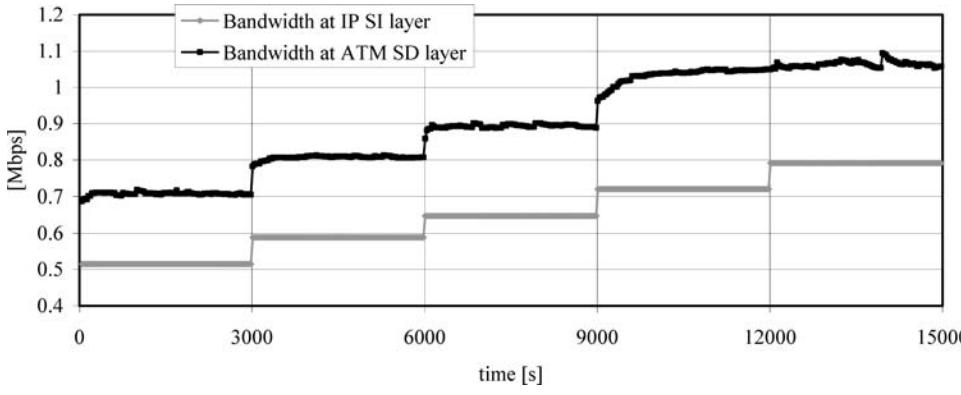


Figure 9.9 SI and SD layer bandwidth shift

9.4.2 Traffic Aggregation

In this test, there are two SI traffic buffers: one for VoIP and one for video traffic. The aim is to check the RCBC behaviour when multiple flows are aggregated together at the SI–SAP interface, as described in section 8.6.2. The first buffer conveys the traffic of 100 VoIP sources. VoIP sources are exponentially modulated on–off processes, as described above. The SI service rate for VoIP must assure the VoIP SLS requirements: $PLP^{*}_{\mathrm{VoIP}} = 10^{-2}$ and $AD^{*}_{\mathrm{VoIP}} = 20\,\mathrm{ms}$. It is computed through simulations and set to $R^{\mathrm{SI}}_{\mathrm{VoIP}} = 232\,\mathrm{kbps}$. The SI VoIP buffer size is 30 IP packets of 1500 bytes each. The second buffer is dedicated to a video service. "Jurassic Park I" video trace, taken from the website referenced in [Video-Traces], is used. Data are H.263 encoded and have an average bit rate of 260 kbps as well as a peak bit rate of 1.3 Mbps. The SI rate allocation for video, measured through simulations, is $R^{\mathrm{SI}}_{\mathrm{Video}} = 350\,\mathrm{kbps}$. It assures $PLP^{*}_{\mathrm{VoIP}} = 10^{-3}$ and $AD^{*}_{\mathrm{VoIP}} = 50\,\mathrm{ms}$, which are the SLS requirements for video. The SI video buffer size is set to 10,500 bytes. Both the outputs of the SI buffers are conveyed towards a single queue at the SD layer. DVB encapsulation (header 4 bytes, payload 184 bytes) of the IP packets through the LLC/SNAP (overhead 8 bytes) ([ETSI-TS-102462]) is implemented in this case. SD buffer size is set to 300 DVB cells. RCBC tracks the SLS performance thresholds, independently of the measured SI performance, by following the feedback from SLS reference level approach. RCBC is compared with the EqB approach, as described in section 9.3.4. The loss probability bound ε for EqB is set to 10^{-3}, being the most stringent PLP constraint imposed at the SI level. The time interval between two consecutive SD reallocations is denoted by T_{RCBC} and T_{EqB}, respectively for RCBC and EqB. T_{RCBC} is fixed to 7 min, while T_{EqB} is set to the following values: $T_{\mathrm{EqB}} = \left\{\frac{1}{3}\cdot T_{\mathrm{RCBC}}, \frac{1}{2}\cdot T_{\mathrm{RCBC}}, T_{\mathrm{RCBC}}, 2\cdot T_{\mathrm{RCBC}}, 3\cdot T_{\mathrm{RCBC}}\right\}$ in different tests. In numbers, it means $T_{\mathrm{EqB}} = \{2.33\,\mathrm{min}, 3.5\,\mathrm{min}, 7\,\mathrm{min}, 14\,\mathrm{min}, 28\,\mathrm{min}\}$. EqB allocation is parameterized by the dimension of the time horizons in which m_{aSD} and σ_{aSD}, the mean and the standard deviation, respectively, of the SD inflow process are computed. PLP is the dominant performance constraint for RCBC.

The results are shown in Figure 9.10. SD bandwidth allocations versus time are shown. RCBC captures the bandwidth need R^{SD} of the SD layer in a single reallocation step and no change is produced for the rest of the simulation. On the other hand, EqB produces strong

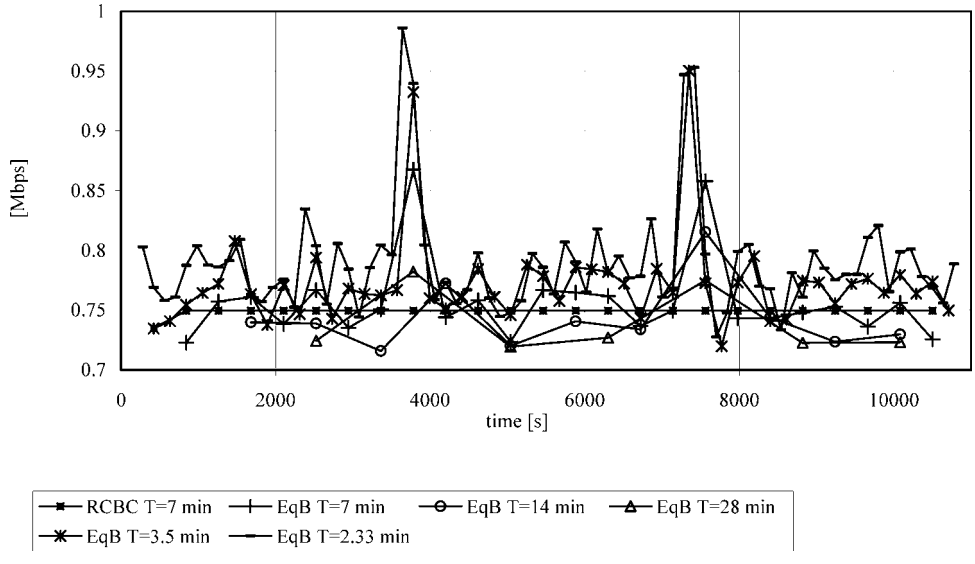


Figure 9.10 SD bandwidth allocations versus time

Table 9.5 Average PLP over the entire simulation

PLP variance at SD layer RCBC	PLP variance at SD layer $T_{EqB} = 2.33$	PLP variance at SD layer $T_{EqB} = 3.5$	PLP variance at SD layer $T_{EqB} = 7$	PLP variance at SD layer $T_{EqB} = 14$	PLP variance at SD layer $T_{EqB} = 28$
$8.40 \cdot 10^{-6}$	$2.07 \cdot 10^{-3}$	$1.52 \cdot 10^{-3}$	$1.23 \cdot 10^{-3}$	$8.01 \cdot 10^{-4}$	$7.34 \cdot 10^{-4}$

oscillations in the SD rate assignment. Oscillation size decreases when T_{EqB} increases, even if the precision of the measure is lower when T_{EqB} is large. It should be clearer from the remaining part of the paragraph. The variance values of PLP are reported in Table 9.5. RCBC performance is outstanding.

The precision of RCBC allocations is clear from Figures 9.11 and 9.12, which show, respectively, the PLP of the video SD buffer and the corresponding bandwidth allocations. The video PLP performance target is 10^{-3} and it is also shown in Figure 9.11. The results are averaged over the entire simulation horizon and shown for RCBC and EqB. RCBC performance is very satisfying because PLP is below the performance threshold, being the RCBC video PLP $7.56 \cdot 10^{-4}$. EqB guarantees "below threshold" result only for frequent reallocations ($T_{\text{EqB}} = 2.33$ min), but this configuration implies large oscillations over time, as it is clear from Figure 9.10 and Table 9.5. Similar comments might be done for VoIP PLP. The average value for RCBC PLP is $1.01 \cdot 10^{-4}$, well below the performance requirement which is 10^{-2}. As shown in Figure 9.12, RCBC not only allows saving bandwidth compared to the "SD layer EqB T = 2.33 min" strategy, which also guarantees the performance requirement, but offers bandwidth allocations comparable to the EqB strategies whose PLP is far from the SI threshold. To match performance requirements, EqB must reduce T_{EqB} to a minimum

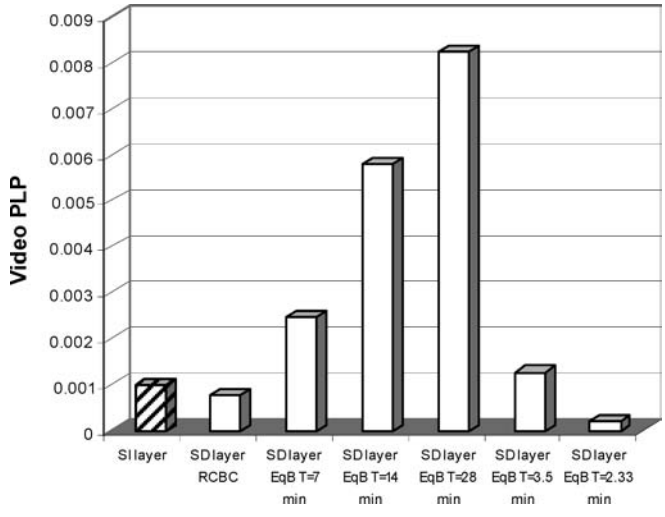


Figure 9.11 Video Packet Loss Probabilities

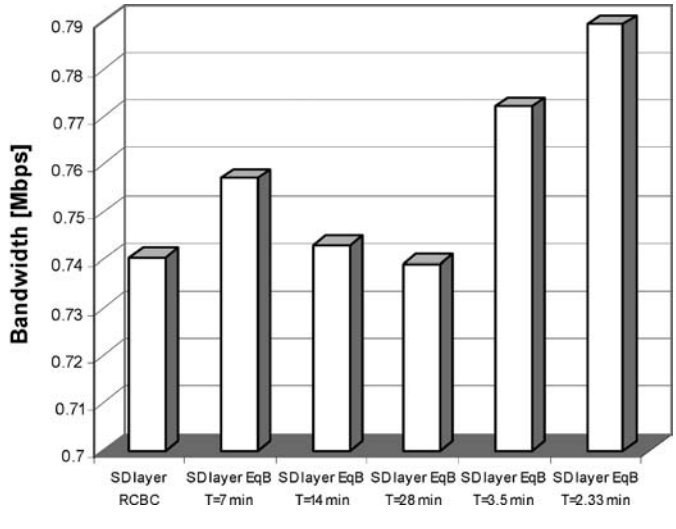


Figure 9.12 SD average bandwidth allocations

and allocate much more bandwidth than RCBC. It outlines the difficulty to find the optimal measurement window for EqB statistics.

9.4.3 Fading Counteraction

The last paragraph of this section is dedicated to analyse the answer of the algorithm to the fading problem, described in section 8.6.3. Even if limited to satellite and wireless

links, this issue is very important because it allows highlighting the dynamic reaction of the algorithm and to check the behaviour of the control scheme over specific environments where bandwidth allocation is topical.

An aggregate trunk of 50 VoIP on-off sources composes the traffic at the SI–SAP interface. ATM is used again as SD transport technology. Only one IP queue and one ATM queue (e.g. EF traffic over ATM) are isolated for this test. The choice is due to the will of identifying, precisely, the contribution of each single problem, such as fading in this case. The time horizon of the simulation scenario is 133.0 min. The SLS is $PLP^*_{\mathrm{VoIP}} = 2 \cdot 10^{-2}$ ms and $AD^*_{\mathrm{VoIP}} = 20$ ms. The buffer size is set to 1600 bytes (20 VoIP packets) at SI layer and to 70 ATM cells at SD layer. T_{RCBC} is set to 1 min. Time-varying channel degradation affects the SD buffer service rate. Fading is modelled by the bandwidth reduction process $\phi(t)$ described in section 8.6.3. It can assume the following values $\{0.0, 0.15625, 0.3125, 0.625, 0.8333, 1.0\}$ at any time instant. Figure 9.13 shows the $\phi(t)$ behaviour versus time. The process generates peaks of channel degradation, especially in the time interval [4800, 6000]. PLP and AD measured versus time are depicted in Figures 9.14 and 9.15, respectively, by using RCBC at SD layer. Only four peaks of performance degradation (PLP and AD above the corresponding QoS requirements) appear and only correspondence of reduction factor changes. It is important to remind that, in this case, when the fading shape varies, RCBC scheme is started from the beginning (i.e. giving the same bandwidth previously allocated in clear sky conditions) to stress the algorithm effort. In these hard conditions, the quick reaction and adaptation to fading variations are evident. RCBC is now applied in its "feedback from measured SI losses" version.

The bandwidth allocations of the SD and SI layers versus time are compared in Figure 9.16. RCBC allocations include the additional bandwidth assigned to match fading counteraction.

From the results presented, it is clear that RCBC effectively produces a quick adaptive response to the channel variations and is able to keep the desired QoS. SD PLP and AD, averaged over the entire simulation horizon, are $1.62 \cdot 10^{-2}$ ms and 17 ms, respectively, which are below the performance requirements. It is important to note that this scenario allows highlighting the RCBC's real time "active learning" capability. Actually, optimal rate allocations may be "memorized", in real time, in correspondence of experienced fading values and given number of connections in the flows. Hence, SD allocation can be optimally tuned for the rest of the time without further exploration steps (and consequent performance degradation) until traffic and/or fading configuration change takes place.

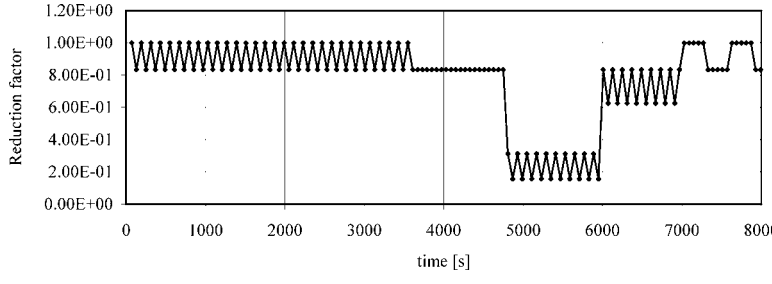


Figure 9.13 Bandwidth reduction due to fading process

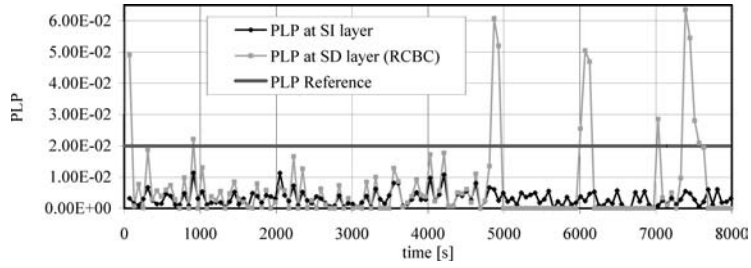


Figure 9.14 Packet Loss Probability at SI and SD layer by using RCBC

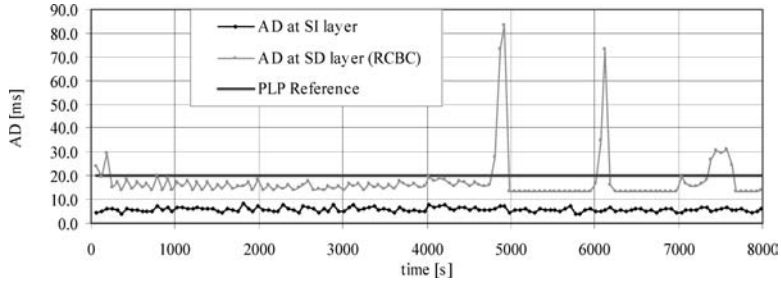


Figure 9.15 Delays at SI and SD layer by using RCBC

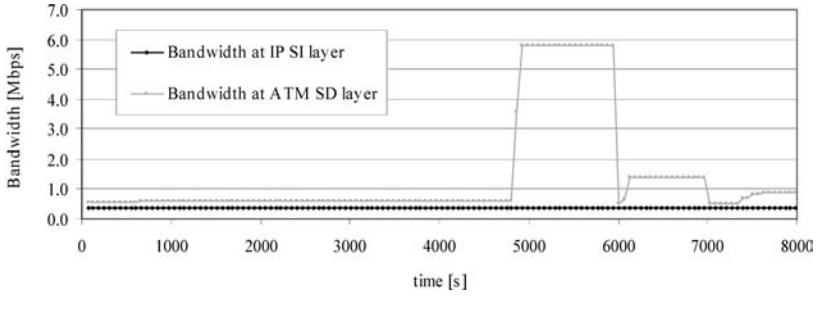


Figure 9.16 Bandwidth allocations at SI and SD layer by using RCBC

This section concludes the performance evaluation of RCBC. It concludes also the chapter itself and the vertical QoS mapping issue.

10

QoS Gateways for Satellite and Radio Communication

10.1 Role of QoS Gateway

The detailed explanation of the horizontal and vertical QoS mapping problems allows clarifying the role of the QoS gateways defined in the previous chapters. The overall action heavily depends on the chosen QoS architectures. It ranges from no quality when no control is implemented to loose QoS assurance when DiffServ-IP-centric approach is applied with Static Trunks and DiffServ-Pre Congestion Notification options, and to hard-QoS guarantees. Among the QoS architectures that operate with hard-QoS guarantees, IntServ-IP-centric, Full-MPLS-centric and IPv6-centric guarantee per flow assurance, while DiffServ-IP-centric QoS architecture with Bandwidth Broker application offer per class (per aggregate) guarantee. Anyway, in general, concerning the hard-QoS architectures, the final action of the QoS-PRNs (-RNs) may be modelled as a bandwidth pipe acting as an end-to-end or, at least, from the first to the last QoS-PRN (-RN), which assigns the necessary bandwidth to guarantee the specified SLS to the source traffic.

The bandwidth pipe derives from the co-operation of all layers. Figure 5.6 is exemplary because it shows as bandwidth needs to be enlarged passing from a layer to another for the vertical mapping adaptation, which should be transparent to the user flow. The overall effect of the bandwidth pipe is shown in the example in Figure 7.40. In more detail, Figure 10.1 shows the bandwidth pipe establishment over the QoS-RN chain. QoS-RNs are used instead of QoS-PRNs to simplify the figure but the meaning is the same. The bandwidth pipe is evidenced by a thick bold black line. It can be dedicated to a single flow or to a traffic class and, obviously, it may be enlarged and restricted both to accomplish vertical QoS needs and to accommodate, in case of traffic classes, new traffic coming from intermediate nodes but, from the point of view of the users, there is always a dedicated portion of bandwidth to assure the SLS.

QoS over Heterogeneous Networks Mario Marchese

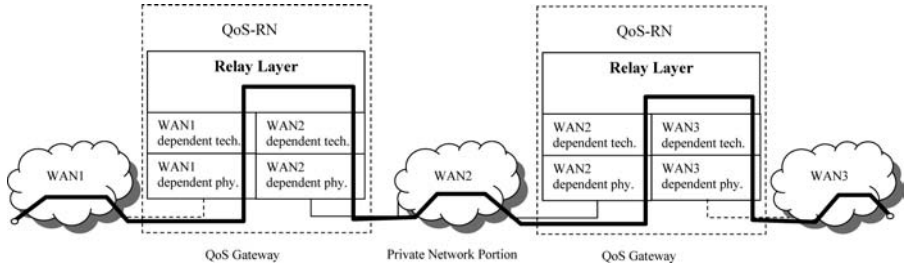


Figure 10.1 Establishment of the bandwidth pipe over the QoS-RN chain

As said, the creation of the bandwidth pipe derives from the action of different control blocks. In particular, bandwidth allocation, developed by the block identified as Network Control Centre (NCC)/Network Bandwidth Management Entity in Chapters 6 and 8, is particularly meaningful if applied to satellite communications, as in Figure 8.3. The problem is just how to allocate bandwidth to the satellite portion of the network. The NCC acts as bandwidth allocator to dimension the bandwidth pipe over the satellite portion, as shown in Figure 10.2. The solution of the problem is linked to the algorithm explained in Chapter 9, but it is generalized to include limitations to the overall available bandwidth provided by the satellite channel, which are ignored in Chapter 9 because they are not part of the solved theoretical problem. The algorithmic solution may be of main interest for researchers in the field and, as done in Chapter 9 for the abstraction of vertical mapping, it is reported in a separate chapter (Chapter 11), which can be ignored if the reader is not interested in this type of research. Figure 10.2 will be also referenced in Chapter 11 to explain the algorithm.

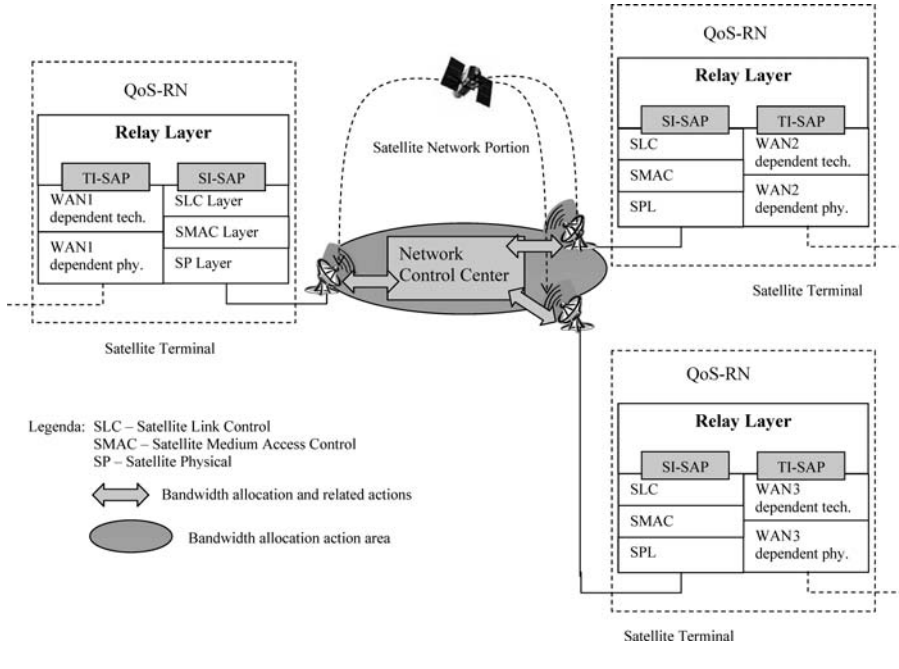


Figure 10.2 Satellite portion bandwidth allocation

10.2 Protocol Optimization Through Layers (POTL)

As promised in Chapter 5 (section 5.3), the action of NCC and of the control blocks to manage vertical QoS mapping allows a wider interpretation of the QoS gateway, which extends the features of the QoS-RNs (and -PRNs). The aim is the improvement and, if possible, the optimization of the overall network performance through a dedicated protocol stack that may be implemented for each specific network portion. It is called "Performance Optimization Through Layers" (POTL), and it is strictly related to the co-operation between layers to offer an overall QoS service. It may also be referenced as "cross-layer interaction". In other words, each single network portion (e.g. WAN2, in Figure 10.1) may deserve a special stack, totally dedicated to it where WAN-dependent layers are designed just to match the peculiar problems of the network portion.

An additional action within the relay layer in the QoS gateway is the operation also at transport and application layer. It does not really mean that the relay layer must implement the full features of transport and application layers, but that it should be able to intercept transport/application layer packets at layer 3 and to mask their behaviour with the source host. The mechanism is often referred as "Performance Enhancing Proxy" (PEP) and it is developed, in particular, at the transport layer to adapt Transport Control Protocol (TCP). The idea is meaningful if applied to special environments such as radios and satellites where the TCP does not use the available bandwidth efficiently. The motivations for this are reported in detail in Chapter 12 together with some results that can provide a precise idea about the TCP behaviour over satellite links. The main problems concern channels with large delay-bandwidth products; channels with errors at transmission level not due to congestion and channels characterized by combined effects of the two characteristics. TCP delays the transmission rate of each source because it either receives acknowledgements in delay or does not receive any acknowledgement and considers the related packet loss due to congestion instead of due to transmission error. If the dedicated protocol stack acting over the network portion can isolate it from the rest of the network splitting the transport protocol action into differentiated parts, the overall communication may be surely improved.

For example, referencing Figure 10.2, the transport layer connection may be divided into three parts: from the source host to the first QoS-RN acting as Satellite Terminal; over the satellite portion between two QoS-RNs and from the destination to the second QoS-RN acting as Satellite Terminal. Figure 10.3 shows a scheme of a possible QoS architecture

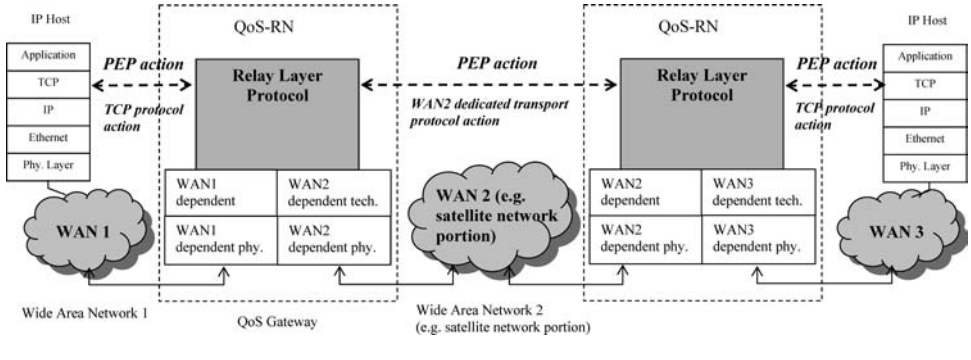


Figure 10.3 QoS-RNs PEP action at the transport layer

where QoS-RNs also act at the transport layer as PEP. TCP acts on the first and the last part. A dedicated transport protocol is implemented over the satellite link. The same comments are still valid for the application layer, if a TCP-like acknowledgement mechanism is applied within the application layer relaxing the role of the transport layer.

10.3 Protocol Stack Optimization Action

The action of improving (or, possibly, optimizing) the protocol behaviour over the overall network depends on the characteristics of the QoS architecture, as said at the beginning of this chapter. If each single flow has a specific bandwidth reserved for it from the source to the destination (or from the first QoS-RN to the last one, supposing over-provisioned local access networks), actually, it would be reasonable to act on the source and to optimize the protocol behaviour within the host. For example, TCP (as better explained in Chapter 12) increases the source rate step-by-step in strict dependence on the number of received acknowledgements. It is efficient if the bandwidth available is not known, as typical over the Internet. In practice, it is a way to test bandwidth availability. If the bandwidth is assured from the source to the destination, the source rate can be adapted on it directly at the information source. It may be implemented at both the application and transport layers. The action is also possible when the traffic is aggregated in classes, but it is more complex because the precise bandwidth allocated to each flow within the aggregate is not trivial to estimate by the source.

Adapting the source emission rate to the available bandwidth seems a simple operation that makes QoS packet-switching networks very close to circuit-switching networks but it is not so. It contrasts with the reality where many commercial products that work at the application layer are based on TCP and their modification would require a strong economic effort. So, the adaptation of the flow rate over a specific network portion is fully justified for any considered hard-QoS architectures, in particular, when the traffic is divided into classes. The action is shown in Figure 10.4 by using a model of a DiffServ-IP-Centric QoS architecture. While referring to Figure 10.3, Figure 10.4 contains the QoS-RN part insisting on WAN2. Each single connection acting either at transport or at application layer (depending on the PEP action) is intercepted by the relay layer. The relay layer, on the base of the control blocks introduced in Chapter 6, has information about the number of flows accepted at a given time instant as well as about the rate assigned to each buffer acting at TI (Technology Independent) layer (SI – Satellite Independent, in case of satellite communication). In consequence, it knows the bandwidth (at least in average) assigned to each single flow. It can be defined as a sort of Complete Knowledge (CK) situation. So, making full use of the available information and of the protocol features and parameters (see Chapter 12, for TCP case), the rate of the flows may be adapted to the available bandwidth so improving the communication performance. The mechanism implementation implies not only the interception and splitting of the transport/application layer flow but the use of additional buffers, where information coming from the source is stored and waits to be transmitted by the dedicated protocol stack.

Chapter 12 contains the details concerning TCP and possible actions on the parameters so as to improve the performance up to the CK situation, where the transmission over the dedicated portion is totally adapted to the available bandwidth. Even if dedicated only to TCP, reading Chapter 12 should help understand the real problems, which are linked to the peculiar features of the traversed networks and to the characteristics of the protocols, which

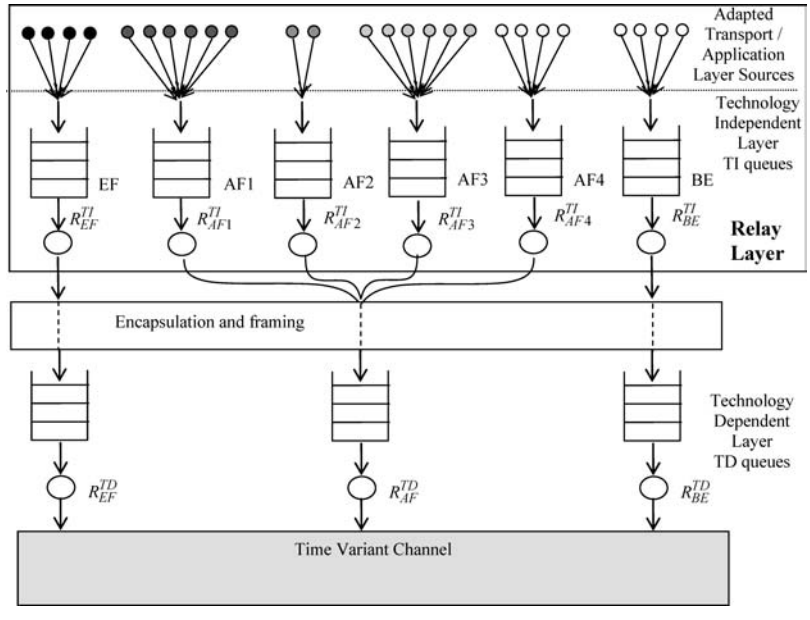


Figure 10.4 Rate adaptation at DiffServ-IP-centric QoS-RN

have been designed either for generic networks or for specific ad hoc situations. In both cases, their behaviour is not adapted to the peculiarities of some networks such as satellites and radios.

The idea to have QoS gateways whose internal stacks are totally adapted to the needs of traversed networks may be the way to get transmission optimization over packet-switched networks. Its implementation implies the solution of many research problems of interest both for academia and for industry. It also requires the intervention and the contribution of standardization bodies. In consequence, it seems also a possible topic for a project requiring the synergy of research, implementation and standardization.

11

Bandwidth Allocation for Satellite Environment

11.1 Introduction

This chapter reports practical examples of bandwidth allocation algorithms over satellite communication, as promised in Chapter 10. The bandwidth allocation is performed by the Network Control Centre (NCC), shown in the network of Figure 10.2, which is generalized and taken as reference for this chapter.

The aim is both to give an idea of the possible NCC action and to introduce an interesting topic for future research.

11.2 System Scenario and Control Architecture

11.2.1 Network Topology

The network portion considered is composed of Z Satellite Terminals (STs)/QoS-RNs ($Z = 3$, in Figure 10.2), modelled as nodes gathering traffic from the sources and connected through a satellite connection. It is shown in Figure 11.1. The control architecture is centralized: the NCC manages the resources and provides the STs with portions of the overall bandwidth C_{tot} (e.g. TDMA slots), which is the service capacity of the buffers contained in each QoS-RN. The NCC may be physically located within each single QoS-RN, which acts as master station in a given period.

The practical aim of the allocation is the provision of bandwidth to each QoS-RN by splitting the overall available capacity and the guarantee of specific QoS requirements (the SLS offered by SD to SI layer) if the overall available bandwidth allows it.

QoS over Heterogeneous Networks Mario Marchese

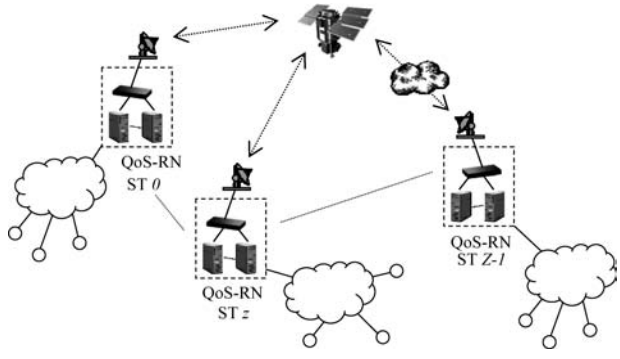


Figure 11.1 Network topology and Satellite Terminal (ST) model

11.2.2 Simple Channel Model

In satellite systems the degradation of the channel quality represents the cause of detriment of communications services.

In practice, satellite channels are affected by the same problems of typical terrestrial wireless channel but additionally they are also significantly corrupted, if the transmission frequencies are high, by meteorological precipitations. In other words, satellite channels are typically affected by rain fading, which is predominant at higher frequencies especially above 10 GHz. It has a negative impact on QoS. The meteorological state over an ST determines the real availability of the channel bandwidth for transmissions. Fading effects may be compensated by a range of Forward Error Correction (FEC) coding schemes, useful at providing efficient broadband services working under different attenuation conditions. Depending on the fading conditions, the amount of bits dedicated to FEC may be increased (or decreased) so improving (or not) the protection power against channel errors. In consequence of the bits dedicated to protection, the size of the transported information changes. In dependence on meteorological conditions, STs may provide low-rate services involving powerful FEC coding schemes, when fading is severe, up to high-rate services when channel conditions are good.

So, the problem related to the usage of powerful coding schemes is the bandwidth reduction that determines the effective bandwidth availability. Due to the redundancy bits introduced at the physical layer, the capacity really "seen" by higher layers of the network is reduced so creating possible bottlenecks. This condition may affect significantly the overall performance. The bandwidth allocation is used simultaneously to compensate the reduction and to obtain physical channel reliability.

The bandwidth reduction due to FEC may be modelled as a multiplicative factor of the overall bandwidth assigned to STs coherently with [Celandroni03]. The model has been presented in section 8.6.3 and it is reported again here for the sake of completeness by slightly changing the notation so as to adapt it to bandwidth allocation. The stress is now on the assignation of bandwidth and not on buffer service rate, even if, from the operative viewpoint, there is no difference at all: the assigned bandwidth is just the service rate "given" to the corresponding ST. Mathematically, it means that the real bandwidth C_z^{real} available

for the z-th ST is its nominal bandwidth C_z reduced by a factor ϕ_z, which is, in general, a variable parameter contained in the real numbers interval [0, 1].

$$C_z^{\text{real}} = \phi_z \cdot C_z; \phi_z \in [0, 1], \phi_z \in \mathbb{R} \tag{1}$$

A specific value ϕ_z corresponds to a fixed attenuation level "seen" by the z-th ST. The channel model, used in the bandwidth allocation methods proposed in the following, maps the link layer corruption problem onto a congestion problem. In this view, the capacity reduction has been considered through the factor ϕ_z and the consequent assumption is that all the packet losses happening during communication may be supposed due to congestion events. On the other hand, in the conditions described, with the use of FEC, the loss due to link layer corruption may be supposed tending to zero and the simple channel model proposed in (1) seems to be a reasonable approximation of the satellite radio channel behaviour. An example of the mapping between signal to noise ratio and ϕ_z is contained in Table 11.1 [Celandroni03, Marchese05].

In more detail, Table 11.1 reports the correspondence between the Carrier Power to One-Side Noise Spectral Density Ratio (C/N_0) and ϕ_z factor. The overall bit rate available for a carrier is fixed to 8.192 Mbps. The values reported in the table allow limiting the bit error probability below 10^{-7}. It is not "0", as supposed in theory, but it is a low value. The parameters ϕ_z shown in the table will be used in the performance comparison among bandwidth allocation methodologies proposed in the following.

In this chapter, each fading level is supposed to happen with an associated probability and associated with a particular FEC. The interpretation of the reduction factor ϕ_z allows making important assumptions:

1. In satellite environments, link corruption due to noise is regular and packet loss is mainly due to it.
2. Nevertheless, if FEC schemes are used, the link corruption may be considered as a congestion event.

In practice, increasing FEC bits, the errors due to faded link may be neglected but, in the same time, the available bandwidth for information is reduced so creating possible bottlenecks and consequent loss.

Table 11.1 Signal to noise ratio and related ϕ_z level

C/N_0 [dB]	ϕ_z
>77.13	1
74.63–77.13	0.8333
72.63–74.63	0.625
69.63–72.63	0.3125
66.63–69.63	0.15625
<66.63	–

The considered simple model actually neglects the instantaneous dynamic of the fading and, in this view, it should be substituted with a more precise channel model. Nevertheless, within the framework described, and coherently with the literature in the field [Bolla, Marchese05, Celandroni03], the assumption is not too severe and seems to be a reasonable approximation of the satellite channel behaviour, at least concerning bandwidth control algorithms.

11.3 General Bandwidth Allocation Architecture

The architecture taken as reference in this chapter takes its origin from a bandwidth allocation scheme, called "Constrained Average Probability-Adaptive Bandwidth Allocation over Satellite Channel" (CAP-ABASC). It is reported in Figure 11.2 and is taken from the literature [Bolla, Marchese05].

Two types of traffic are considered: a guaranteed rate traffic, operating at a fixed speed (measured in Kbps) called "CBR" in the following, and a non-guaranteed best-effort Internet traffic (a superposition of TCP and UDP sources) identified through the index *be*. It corresponds to have two SD buffers in Figure 10.4: one dedicated to CBR traffic (e.g. EF traffic) and one dedicated to best-effort traffic. Best-effort queue conveys Internet traffic, even if only TCP sources are considered in the result section of this chapter.

The allocation structure may be extended to more than two buffers so matching the QoS-RN Architectures introduced in the previous chapters.

The general control mechanism is composed of the high control layer called "NCC", which distributes the bandwidth capacity among the STs, and of the low control layer, called "Local Controller" (LC), which splits the capacity C_z allocated to each ST into two contributions: C_z^{CBR}, the bandwidth needed for the guaranteed rate CBR traffic (in practice R_{EF}^{TD}, in Figure 10.4), and C_z^{be}, the capacity for the non-guaranteed TCP-UDP traffic (best-effort flows), which is called R_{BE}^{TD} in Figure 10.4.

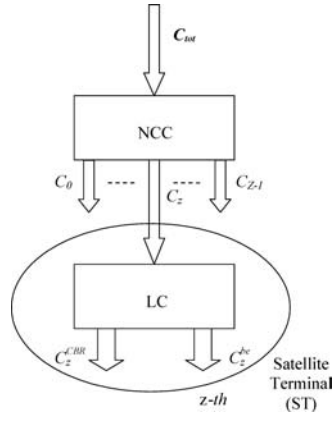


Figure 11.2 Resource allocator control scheme

11.3.1 Local Controller

Each z-th ST solves a local optimization problem to share the assigned capacity between CBR and best-effort traffic. It finds a threshold $\hat{K}_z^{\max}$, which is the maximum number of acceptable CBR calls, which rules the CAC. The aim would be to allow guaranteeing a specific quality in terms of call-blocking probability, which should be upper bounded by a fixed threshold δ_z (always related to a z-th ST), if possible.

The optimization framework, taken from [Bolla, Marchese05] and generalized, is quickly reported in the following. The minimum capacity $C_z^{\min}$ (Kbps), which should be assigned to the CBR buffer of the z-th ST to get the call-blocking probability below the threshold δ_z, is defined in (2):

$$C_z^{\min} = \arg\min_{Y_z} \left\{ Y_z \in \mathbb{R} : P_z^{\text{block}} \left(\left\lfloor \frac{Y_z}{R_z^{\text{CBR}}} \right\rfloor \right) \leq \delta_z \right\}, 0 < Y_z \leq C_{\text{tot}} \tag{2}$$

C_{tot} is the overall channel bandwidth. R_z^{CBR} is the fixed bit rate at which CBR sources at the z-th ST emit data. It is considered the same for all calls. The CAC for each ST is modelled through an M/M/m/m queuing system whose blocking probability is given by the Erlang B formula [Kleinrock]. In practice, the call-blocking probability of the z-th ST is given by (3). It is a function of the maximum number $x_z^{\max}$ of available servers of the M/M/m/m queue:

$$P_z^{\text{block}}\left(x_z^{\max}\right) = \frac{\dfrac{1}{x_z^{\max}!}\left(\dfrac{\lambda_z}{\mu_z}\right)^{x_z^{\max}}}{\displaystyle\sum_{j=0}^{x_z^{\max}} \frac{1}{j!}\left(\frac{\lambda_z}{\mu_z}\right)^{j}} \tag{3}$$

λ_z and $1/\mu_z$ are, respectively, the call arrival rate and the average duration of each call, both of them exponentially distributed.

The maximum number $K_z^{\max}$ of calls that can be served by $C_z^{\min}$ is obtained through:

$$K_z^{\max} = \left\lfloor \frac{C_z^{\min}}{R_z^{\text{CBR}}} \right\rfloor \tag{4}$$

If C_z is the bandwidth allocated to ST z, the maximum bandwidth that CBR traffic can get is ruled by

$$\widehat{C}_z^{\text{CBR}} = \begin{cases} C_z^{\min} & if\ C_z > C_z^{\min} \\ C_z & if\ C_z \leq C_z^{\min} \end{cases} \tag{5}$$

So the maximum number of CBR calls, which can be accepted by ST z is

$$\widehat{K}_z^{\max}(C_z) = \left\lfloor \frac{\widehat{C}_z^{\text{CBR}}}{R_z^{\text{CBR}}} \right\rfloor \tag{6}$$

It will govern the CAC scheme dedicated to CBR traffic.

CBR traffic does not necessarily take the maximum bandwidth $\widehat{C}_z^{\text{CBR}}$ in each time instant: let $k_z(t) \leq \widehat{K}_z^{\max}$ be the number of CBR calls in progress at time t, at ST z. $k_z(t)$ is also the number of busy servers at time t. The current bandwidth taken by CBR traffic at time t is $C_z^{\text{CBR}}(t) = R_z^{\text{CBR}} \cdot k_z(t) \leq \widehat{C}_z^{\text{CBR}} \leq C_z^{\min}$. The residual capacity ($C_z^{be}(t) = C_z - C_z^{\text{CBR}}(t)$) is available for the Internet traffic.

11.3.2 NCC Allocation

The rationale under the bandwidth allocation algorithm is to assure the QoS guaranteed traffic and, in the same time, to help the non-guaranteed traffic to offer the best possible service.

Starting from the theory previously illustrated, considering that each ST is supposed to have here a single buffer (but in general, as shown in [Marchese05], there can be more than one), gathering best-effort traffic from the sources, the practical aim of the allocator, is the provision of bandwidth to each ST by splitting the overall capacity available among the STs. The chosen criterion is to consider the STs as competitive entities of the problem.

In a competitive environment, Multi-Objective Programming (MOP) is suited to be used in the formulation of the allocation problem. From the analytical viewpoint, the bandwidth allocation problem may be formalized as:

$$\mathbf{C}^{\text{opt}} = \{C_0, \ldots, C_z, \ldots, C_{Z-1}\} = \arg\min_{\mathbf{X}} \{\mathbf{F}(\mathbf{X})\}; \mathbf{F}(\mathbf{X}) : \mathbf{D} \subset \mathbb{R}^Z \rightarrow \mathbb{R}^Z, \mathbf{X} \geq 0 \tag{7}$$

where $\mathbf{X} \in \mathbf{D}, \mathbf{X} = \{X_0, \ldots, X_z, \ldots, X_{Z-1}\}$ is the general definition of the vector of capacities that can be assigned to STs. The element X_z, $\forall z \in [0, Z-1]$, $z \in \mathbb{N}$ is referred to the z-th ST; $\mathbf{C}^{\text{opt}} \in \mathbf{D}$ is the vector of optimal allocation, aim of the solution; $\mathbf{D}$ represents the domain of the vector of functions. The solution must respect the constraint:

$$\sum_{z=0}^{Z-1} X_z = C_{\text{tot}} \tag{8}$$

where, as previously considered, C_{tot} is the overall available capacity.

$\mathbf{F}(\mathbf{X})$, dependent on the bandwidth vector, $\mathbf{X}$, is called *performance vector*. It is the quantity to be optimized:

$$\mathbf{F}(\mathbf{X}) = \{f_0(X_0), \ldots, f_z(X_z), \ldots, f_{Z-1}(X_{Z-1})\}; \forall z \in [0, Z-1] \tag{9}$$

The single z-th *performance function* $f_z(X_z)$ (or *objective*) is a component of the vector and may be defined, for example, as the average packet-loss probability related to the best-effort traffic $P_z^{\text{loss}}(\cdot)$ with the addition of a penalty function related to CBR traffic. $f_z(X_z)$ functions, in general, must be supposed to be decreasing with the bandwidth used by a single ST (X_z) and with the fading level ϕ_z. For each ST z, the *performance function* is considered averaged on the fading level ϕ_z, which is considered a discrete stochastic variable ranging among L possible values ϕ_z^l happening with probability $p_{\phi_z^l}$.

Analytically,

$$\begin{aligned} f_z(X_z) &= \underset{\phi_z}{E}\left[P_z^{\text{loss}}\left(\phi_z X_z, \widehat{K}_z^{\max}(X_z)\right) + W_z^{CAP}(X_z)\right] = \\ &= \sum_{l=0}^{L-1}\left[P_z^{\text{loss}}\left(\phi_z^l X_z, \widehat{K}_z^{\max}(X_z)\right) + W_z^{CAP}(X_z)\right] p_{\phi_z^l}; \forall l \in [0, L-1], L \in \mathbb{N} \end{aligned} \tag{10}$$

$P_z^{\text{loss}}(\cdot)$ is a function of the bandwidth X_z and of the maximum number of accepted CBR calls. It represents the average packet-loss probability of non-guaranteed traffic at ST z. The

closed-form expression of $P_z^{\text{loss}}(\cdot)$ has a great impact on the bandwidth allocation concerning non-guaranteed traffic and, as a consequence, on the performance that the control system can offer to best-effort flows.

The penalty function $W_z^{CAP}(X_z)$, which allows considering the constraint related to the CBR calls, is based on the call-block probability:

$$W_z^{CAP}(X_z) = \begin{cases} 0 & \text{if } \underset{\phi_z}{E}\left[P_z^{\text{block}}\left(\widehat{K}_z^{\max}(X_z)\right)\right] \le \delta_z \\ \text{H} & \text{if } \underset{\phi_z}{E}\left[P_z^{\text{block}}\left(\widehat{K}_z^{\max}(X_z)\right)\right] > \delta_z \end{cases} \tag{11}$$

Where H is a very large constant, and

$$\underset{\phi_z}{E}\left[P_z^{\text{block}}\left(\widehat{K}_z^{\max}(X_z)\right)\right] = \sum_{l=1}^{L-1}\left[P_z^{\text{block}}\left(\widehat{K}_z^{\max}(X_z)\right)\right] p_{\phi_z^l} \tag{12}$$

The solution of the minimization problem formalized in (7) depends on the specific methodology used to solve the vector optimization problem. Some possible approaches are proposed in section 11.5 after a short introduction reported in section 11.4.

In the described optimization framework, each *performance function* $f_z(\cdot)$ represents a single cost competing with the others to get bandwidth.

11.4 Pareto Optimality of the Bandwidth Allocation

The optimal solution for MOP problems is called "Pareto Optimal Point" (POP) [Miettinen] coherently with the classical MOP theory, which is strictly related to the concept of efficient allocation discussed in [Miettinen] and [Yaïche] and here formalized. The bandwidth allocation $\mathbf{C}^{\text{opt}} \in \mathbf{D}$ is a POP if a generic allocation $\mathbf{X} \in \mathbf{D}$ does not exist so that

$$\mathbf{F}(\mathbf{X}) \le_P \mathbf{F}\left(\mathbf{C}^{\text{opt}}\right), \forall \mathbf{X} \ne \mathbf{C}^{\text{opt}} \tag{13}$$

Concerning the operator "$\le_P$", given two generic performance vectors $\mathbf{F}^1, \mathbf{F}^2 \in \mathbb{R}^Z$, $\mathbf{F}^1$ *dominates* $\mathbf{F}^2 (\mathbf{F}^1 \le_P \mathbf{F}^2)$ when

$$\begin{aligned} &f_x^1 \le f_x^2 \ \forall x \in \{0, 1, \ldots, Z-1\} \text{ and} \\ &f_y^1 < f_y^2 \text{ for at least one element } y \in \{0, 1, \ldots, Z-1\} \end{aligned} \tag{14}$$

where f_x^1, f_y^1, f_x^2 and f_y^2 are the elements of the vector $\mathbf{F}^1$ and $\mathbf{F}^2$, respectively.

In the considered problem, a POP is a bandwidth allocation where any change to get a lower value of one of the performance functions implies the increment of at least one of the other functions. The constraint in (8) defines the set of POP solutions because, over that constraint, each variation of the allocation, aimed at enhancing the performance of a specific ST, implies the performance deterioration of at least another ST due to the decreasing nature of the objective functions.

It is worth noting that, in the proposed general methodology, no on-line decision method is applied. The system evolution is ruled by stochastic variables. In practice, $\mathbf{F}(\mathbf{X})$ in the optimization problem (7) is considered to be the average value of the performance vector over all the possible realizations of the stochastic processes of the network. The performance

functions are representative of the steady-state behaviour of the system and the allocation is provided with a single infinite-horizon decision.

11.5 Resolution Approaches

In general, the solution of the NCC allocation formalized in problem (7) can be generated with different methodologies. The strategies reported in this chapter provide just one solution to problem (7), out of the overall set of solutions defined by the constraint (8). Even if all the solutions defined by (8) are Pareto optimal, one of them may be preferred depending on a fixed criterion. MOP problems are thought in a fully competitive environment but, for example, if the aim (the criterion) is to get the minimum average packet-loss probability over all STs, it is necessary to use a method to generate the solution that, within the space defined by the POP set, allows choosing the allocation that satisfies the aim. The solutions listed below have different decisional criteria:

- Utopia Minimum Distance (UMD) – is a method recently proposed in [BisioMarchese06] and it is a strategy directly derived from the MOP general theory.
- Fixed Allocation (FIX) – where the allocator assigns the same capacity to each ST independently of the meteorological and traffic conditions.
- Heuristic (HEU) – where the bandwidth allocation is directly proportional to the traffic offered (N_z) and inversely proportional to the fading level (ϕ_z).
- Value Function (VALUE) – where the minimized cost is the sum of the packet loss averaged over the fading level, which is the philosophy of the techniques presented in [Bolla]. Also this method may be considered directly derived from the MOP theory [Miettinen].
- Nash Bargaining Solution (NBS) – where the cost is the sum of the logarithms to base e of the packet loss of each ST averaged over the fading levels (in practice, it minimizes the product of the packet losses averaged over the fading).
- QoS-constrained solutions, which use the same philosophy of the UMD method but explicitly consider QoS constraints within the allocation algorithm.

VALUE and NBS methods imply a partial cooperation among the STs to allocate bandwidth, agreed a priori and formalized through proper cost functions. Such cooperation is not included within UMD, which models bandwidth allocation as a totally competitive problem. The UMD cost function (in the following) is not representative of any prior agreement among the STs but is aimed at approaching the ideal situation for each of them, where every ST has the full bandwidth availability.

In the following, for the sake of synthesis, the TCP packet-loss probability will be considered as $P_z^{loss}(\cdot)$. The explicit expression used is reported in section 11.6 at the end of this chapter. It is important to remark that it is just an example and any other function that does not violate the theoretical assumptions reported in section 11.3.2 concerning $f(x)$ may be used.

11.5.1 Utopia Minimum Distance Method Algorithm

It is aimed at approaching the ideal performance, which theoretically happens when each single ST has the availability of the entire channel bandwidth, by minimizing the Euclidean distance between the performance vector and the ideal solution of the problem. It is a MOP

resolution method also identified as GOAL approach [Miettinen] and bases its decision on the ideal solution defined as *utopia point*. In more detail, the *ideal performance vector* in this case is

$$\mathbf{F}^{\mathrm{id}}\left(\mathbf{C}^{\mathrm{id}}\right)=\left\{f_0^{\mathrm{id}}\left(C_0^{\mathrm{id}}\right),\ldots,f_z^{\mathrm{id}}\left(C_z^{\mathrm{id}}\right),\ldots,f_Z^{\mathrm{id}}\left(C_{Z-1}^{\mathrm{id}}\right)\right\} \tag{15}$$

where

$$f_z^{\mathrm{id}}\left(C_z^{\mathrm{id}}\right)=\min_{X_z}\underset{\phi_z}{E}\left[P_z^{\mathrm{loss}}\left(\phi_z X_z,\widehat{K}_z^{\max}\left(X_z\right)\right)+W_z^{CAP}\left(X_z\right)\right],\ X_z\in[0,C_{\mathrm{tot}}] \tag{16}$$

From equation (16), called *single objective problem*, it is obvious (being each objective function supposed to be decreasing with the bandwidth) that the optimal solution is given by $C_z^{\mathrm{id}}=X_z=C_{\mathrm{tot}},\forall z\in[0,Z-1]$. So, $\mathbf{C}^{\mathrm{id}}=\{C_{\mathrm{tot}},C_{\mathrm{tot}},\ldots,C_{\mathrm{tot}}\}$.

Starting from the definition of the *ideal performance vector*, the problem stated in equation (7) can be solved by the following allocation under the constraint (8):

$$\mathbf{C}_{UMD}^{\mathrm{opt}}=\arg\min_{\mathbf{C}}\left(\left\|\mathbf{F}\left(\mathbf{C}\right)-\mathbf{F}^{\mathrm{id}}\left(\mathbf{C}^{\mathrm{id}}\right)\right\|_2\right)^2 \tag{17}$$

where $\|\cdot\|_2$ is the Euclidean norm.

Utopia Minimum Distance is thought for a fully competitive environment where each ST tries approaching its ideal performance. It is not the only possible choice. Some form of cooperation among the STs may be also included, as said in the next paragraphs. The comparison among the approaches is reported in a short performance evaluation section shown at the end of the chapter.

Figure 11.3 geometrically describes the behaviour of the UMD strategy for 2 earth STs (ST 0 and ST 1). For the sake of simplicity, in the graphical description of the UMD behaviour, each performance function is supposed to be composed only by the $P_z^{\mathrm{loss}}(\cdot)$ ignoring the constant penalty function. It means that either the CBR constraint is supposed always satisfied

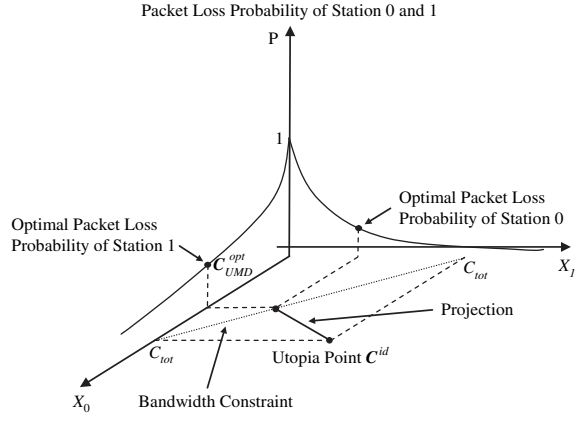


Figure 11.3 UMD behaviour

or, alternatively, C_{tot} is considered the portion dedicated to the Internet (best-effort) traffic. In practice, C_{tot} would be the residual bandwidth.

On the basis of the fading conditions, UMD search the projection of the utopia point, which minimizes simultaneously the packet-loss probability of STs 0 and 1, reported in the axis P. Plane (X_0, P) contains the packet-loss probability of ST 0 versus X_0, while plane (X_1, P) contains the quantity for ST 1 versus X_1. X_0 and X_1 are the axes reporting the capacities that may be allocated to the ST 0 and ST 1, respectively. ST 1 is supposed to be faded in Figure 11.3 and its related packet-loss probability (shown in plane (X_1, P)) is higher than the packet loss of ST 0. In practice, the action of the proposed algorithm provides a bandwidth allocation, which is the solution of the problem (17), representative of the utopia point projection over the bandwidth constraint (8). The solution obviously privileges the faded ST.

11.5.2 Fixed Allocation

A very simple method is the fixed allocation. In this simple case, the bandwidth allocator assigns the same quantity of capacity to each ST independently of the meteorological and traffic conditions. Being Z the overall number of STs,

$$C_z = \frac{C_{\text{tot}}}{Z}; \forall z \in [0, Z-1], Z \in \mathbb{N} \tag{18}$$

It is obvious to see that the constraint reported in equation (8) is respected and the solution is within the POP set.

11.5.3 Heuristic Allocation

Being the fading condition (expressed in terms of ϕ_z) seen by ST z and the traffic load offered to it (expressed in number of active connections, identified by $N_z = k_z(t) + k_z^{be}(t)$, where $k_z^{be}(t)$ is the number of best-effort sources active at time t) the crucial elements of the proposed bandwidth allocation strategies, a simple heuristic allocation scheme can be defined in terms of them. In more detail, concerning HEU, the bandwidth provided to each ST is a weighted portion of the overall available bandwidth.

From the analytical viewpoint, the capacity assigned to the z-th ST is

$$C_z = wbp_z \cdot C_{\text{tot}}; \forall z \in [0, Z-1], Z \in \mathbb{N}; wbp_z \in [0, 1], wbp_z \in \mathbb{R} \tag{19}$$

Where

$$wbp_z = \frac{N_z}{\phi_z} \cdot \left(\sum_{j=0}^{Z-1} \frac{N_j}{\phi_j} \right)^{-1} \quad \text{with} \quad \sum_{z=0}^{Z-1} wbp_z = 1 \tag{20}$$

The bandwidth assigned to an ST increases coherently with the offered traffic and with the rain fading level. Also in this case, the method is proposed as term for comparison with the other techniques.

11.5.4 Value Function

The technique takes its origin from a bandwidth allocation scheme originally dedicated to geostationary satellite channels presented in [Bolla]. The cost function used there is the decisional criterion of this methodology in the MOP framework.

The VALUE method generates a solution compatible with the problem because it is an MOP method (belonging to the *preference function* methods family as defined in [Miettinen]). In particular,

$$\begin{aligned} J_{\text{VALUE}}(\mathbf{X}) &= \sum_{z=0}^{Z-1} \underset{\phi_z}{E}\left[P_z^{\text{loss}}\left(\phi_z X_z, \widehat{K}_z^{\max}(X_z)\right) + W_z^{CAP}(X_z)\right] \\ &= \sum_{z=0}^{Z-1}\sum_{l=0}^{L-1}\left[P_z^{\text{loss}}\left(\phi_z^l X_z, \widehat{K}_z^{\max}(X_z)\right) + W_z^{CAP}(X_z)\right] p_{\phi_z^l}; \\ &\quad \forall z \in [0, Z-1], Z \in \mathbb{N}; \forall l \in [0, L-1], L \in \mathbb{N} \end{aligned} \tag{21}$$

The VALUE strategy distributes the bandwidth by minimizing the sum of the single *performance functions*. The vector $\mathbf{C}_{\text{VALUE}}^{\text{opt}}$ of the capacities assigned by VALUE is computed as

$$\mathbf{C}_{\text{VALUE}}^{\text{opt}} = \arg\min_{\mathbf{X}} J_{\text{VALUE}}(\mathbf{X}) \tag{22}$$

under the constraint (8).

11.5.5 Nash Bargain Solution

A recently proposed bandwidth allocation method concerns the game theory framework. The considered proposal, deeply treated in [Yaïche], is based on the Nash bargaining framework. It uses the cooperative game theory [Muthoo, Nash] and allows reaching network efficiency and fairness.

The idea is to use the NBS in the context of telecommunication networks and, in particular, in the satellite environment. This approach allows obtaining a Pareto Optimal solution, as well as the techniques previously described in the MOP framework, and provides a bandwidth distribution representative of the proportional fair allocation (see [Yaïche] for details). It uses utility functions, defined below.

From the analytical viewpoint, the *utility functions* $(V_z(\cdot))$, one for each ST, is the reciprocal value of the TCP packet-loss probability averaged over the fading levels, i.e. the reciprocal value of the performance function defined in (10).

$$V_z(X_z) = \frac{1}{\underset{\phi_z}{E}\left[P_z^{\text{loss}}\left(\phi_z X_z, \widehat{K}_z^{\max}(X_z)\right) + W_z^{CAP}(X_z)\right]} = \frac{1}{f_z(X_z)}; \forall z \in [0, Z-1], Z \in \mathbb{N} \tag{23}$$

The utility of each ST grows when the average packet-loss probability decreases. In practice, $V_z(\cdot)$ is a crescent function of the capacity assigned to the z-th ST. The solution is got by defining

$$J_{\text{NBS}}(\mathbf{X}) = \prod_{z=0}^{Z-1} V_z(X_z); \forall z \in [0, Z-1], Z \in \mathbb{N} \tag{24}$$

and by solving

$$\mathbf{C}_{\mathrm{NBS}}^{\mathrm{opt}} = \arg\max_{\mathbf{X}} J_{\mathrm{NBS}}(\mathbf{X}) \tag{25}$$

Each single *utility function* being convex, the optimization problem defined in equation (25) may be equivalently expressed by using

$$\hat{J}_{\mathrm{NBS}}(\mathbf{X}) = \sum_{z=0}^{Z-1} \ln\left[V_z(X_z)\right]; \forall z \in [0, Z-1], Z \in \mathbb{N} \tag{26}$$

So, the solution of the problem (25) may also be found by solving

$$\mathbf{C}_{\mathrm{NBS}}^{\mathrm{opt}} = \arg\max_{\mathbf{X}} \hat{J}_{\mathrm{NBS}}(\mathbf{X}) \tag{27}$$

In conformity with the previously described methods, to get a minimization problem also for NBS, it is necessary to change the $\hat{J}_{\mathrm{NBS}}(\mathbf{X})$ sign. Applying the logarithm properties,

$$\tilde{J}_{\mathrm{NBS}}(\mathbf{X}) = -\hat{J}_{\mathrm{NBS}}(\mathbf{X}) = \sum_{z=0}^{Z-1} \ln\left[\underset{\phi_z}{E}\left[P_{\mathrm{loss}}^{z}(\phi_z X_z)\right]\right]; \forall z \in [0, Z-1], Z \in \mathbb{N} \tag{28}$$

The NBS strategy distributes the bandwidth by minimizing the sum of the logarithms (base e) of each single *performance function*. The vector $\mathbf{C}_{\mathrm{NBS}}^{\mathrm{opt}}$ of the capacities assigned by the NBS method is computed as in (29), under the constraint (8).

$$\mathbf{C}_{\mathrm{NBS}}^{\mathrm{opt}} = \arg\min_{\mathbf{X}} \tilde{J}_{\mathrm{NBS}}(\mathbf{X}) \tag{29}$$

11.5.6 QoS-constrained Solutions

In general, additional constraints may be added to match specific QoS performance requirements. In this paragraph, modified MOP methods taken from [Bisio06] are presented as possible allocation strategies aimed at binding the performance functions. A possible approach may be provided by modifying the UMD method previously introduced. QoS constraints have been fixed for each single ST (i.e. for each *performance function*). Analytically, it may be described as:

$$f_z(C_z) \leq \gamma_z, \forall z \in [0, Z-1], Z \in \mathbb{N} \tag{30}$$

where $\gamma_z \in \mathbb{R}$ is the QoS requirement constraint for the z-th ST. In terms of packet-loss probability, typical γ_z values may be set to 10^{-2}, for voice-streaming applications, and to 10^{-3}, for more demanding applications. C_z^{thr} needs to be allocated to generic ST z to satisfy constraint (30):

$$C_z^{\mathrm{thr}} = \{C_z : f_z(C_z) = \gamma_z, \forall z \in [0, Z-1], Z \in \mathbb{N}\} \tag{31}$$

The value C_z^{thr} may also be received directly from each SD layer as output of the vertical QoS mapping solution introduced in Chapter 9 as RCBC scheme.

The practical aim is to guarantee that the bandwidth C_z allocated to z-th ST is

$$C_z \geq C_z^{\text{thr}}, \forall z \in [0, Z-1], Z \in \mathbb{N} \tag{32}$$

The main problem is to match the limitation of the overall available capacity (8). The sum of the required bandwidths C_z^{thr} ($\sum_{z=0}^{Z-1} C_z^{\text{thr}}$) may be larger than the overall capacity provided by the satellite channel. It implies two possible compromises in the framework of the UMD method [Bisio06]:

1) Bandwidth allocation penalizes some STs, whose traffic flows are considered less important, and guarantees the required bandwidth only for a subset of them, when possible. This approach is called "Constrained Utopia Minimum Distance" (CUMD);
2) The algorithm provides bandwidth so as to approach the requested QoS as close as possible for all STs. This approach is called "QoS Point Minimum Distance" (QPMD). This choice may imply that all Z constraints are not satisfied, even if there is enough bandwidth to satisfy a portion of them.

11.5.6.1 Constrained Utopia Minimum Distance

The constraint set reduces the overall possible solutions by creating a subset of possible POPs. In more detail, the Euclidean norm $\left(\left\|\mathbf{F}(\mathbf{C}) - \mathbf{F}^{\text{id}}\left(\mathbf{C}^{\text{id}}\right)\right\|_2\right)^2$ is the decisional criterion also of the CUMD method but the minimization is carried out both under the constraint (8) and under the set of Z constraints defined in (32).

There is no assurance that QoS requirements for each ST are guaranteed, due to the limited amount of available capacity and to the fading conditions "seen" by the earth STs.

Considering only two STs for the sake of simplicity, given the plane (X_0, X_1) of Figure 11.3, Figures 11.4a and 11.4b try synthesizing what can happen. $C_{\text{CUMD}}^{\text{opt}}$ is shown as C_{CUMD}.

Figure 11.4a contains the example when both requirements can be satisfied: constraint (32) defines a continuous set of points. Figure 11.4b shows the situation when constraint (32)

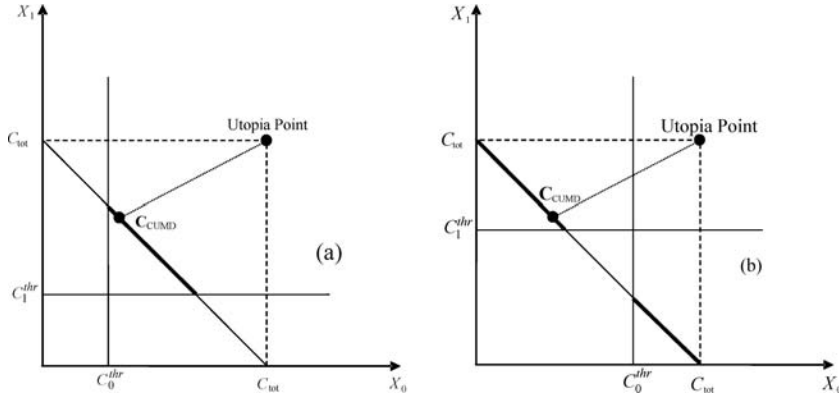


Figure 11.4 CUMD behaviour – constraints not satisfied in the same time

defines a discontinuous set of points where both requirements cannot be satisfied in the same time.

It means that just one ST can reach its QoS level. The traffic of the other ST is considered less important. The choice of the privileged ST depends on the distance with the utopia point. The ST assuring the minimum distance is chosen: ST 1 in the example. It holds true also for more than two STs. If two or more STs assure the minimum distance, the choice is random (it is true also for the situations described just below). Similar behaviour is obtained if at least one of the requirements implies $C_z^{\text{thr}} \geq C_{\text{tot}}$: the bandwidth is assigned to the ST that requires feasible bandwidth ($C_z^{\text{thr}} < C_{\text{tot}}$), so respecting the QoS constraint. The other ST gets the residual channel capacity. If there is more than one ST where $C_z^{\text{thr}} < C_{\text{tot}}$, again the minimum distance choice is taken. The method allows guaranteeing at least one of the performance constraints if the bandwidth needed by one of the STs is lower than the overall capacity available. If $C_z^{\text{thr}} \geq C_{\text{tot}}, \forall z$, then one of the ST gets the entire available channel capacity (through minimum distance choice) and the other(s) is (are) in complete outage condition.

11.5.6.2 QoS Point Minimum Distance

QoS Point Minimum Distance does not minimize the Euclidean distance from the *utopia point* (the situation with no competition) but minimize the distance from the representative point of the desired QoS performance. In practice, it corresponds to use a new definition of the *ideal performance vector*. The new reference point (where QoS is guaranteed) is called *QoS Point* and the related vector is called *QoS performance vector*.

$$\mathbf{F}^{\text{QoS}}\left(\mathbf{C}^{\text{QoS}}\right) = \left\{ f_0^{\text{QoS}}\left(C_0^{\text{QoS}}\right), \ldots, f_Z^{\text{QoS}}\left(C_{Z-1}^{\text{QoS}}\right) \right\} \tag{33}$$

where

$$f_z^{\text{QoS}}\left(C_z^{\text{thr}}\right) = \gamma_z, C_z^{\text{thr}} \in \mathbb{R} \tag{34}$$

which is directly derived from (31). From (33) and (34), obviously $\mathbf{C}^{\text{QoS}} = \left\{ C_0^{\text{thr}}, C_1^{\text{thr}}, \ldots, C_Z^{\text{thr}} \right\}$. The general allocation problem in equation (7), approached by using the UMD method, in presence of QoS requirements, can now be solved through:

$$\mathbf{C}_{\text{QPMD}}^{\text{opt}} = \arg\min_{\mathbf{C}} \left(\left\| \mathbf{F}\left(\mathbf{C}\right) - \mathbf{F}^{\text{QoS}}\left(\mathbf{C}^{\text{QoS}}\right) \right\|_2 \right)^2 \tag{35}$$

The minimization is obviously carried out under the constraint (8).

Also in this case, the behaviour may be described by using the reference situation where two earth STs are considered. Two cases may happen:

1) The *QoS Point* satisfies the constraint (8) and it is within the set of feasible allocations (Figure 11.5a); QoS requirements are matched and the overall performance is surely better than expected because more bandwidth than required is assigned to the STs.
2) The *QoS Point* is outside the feasible allocation region defined by the constraint (8). QPMD provides allocations through equation (35). QoS satisfaction can be approached (Figure 11.5b).

$C_{\text{QPMD}}^{\text{opt}}$ is shown as C_{QPMD} and QoS Point as QoSPoint in Figure 11.5.

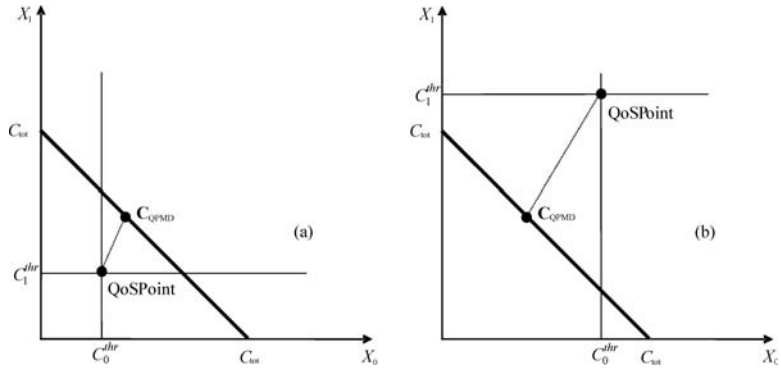


Figure 11.5 QPMD behaviour

11.6 Numerical Examples

In this paragraph, the schemes are compared in terms of allocated bandwidth and packet-loss probability, both analytically evaluated. The comparison concerns UMD, FIX, HEU, VALUE and NBS.

For the sake of simplicity, the presence of the CBR calls has been neglected. Only the TCP traffic is considered and, as a consequence, each performance function is linked to the $P_z^{\text{loss}}(\cdot)$. Coherently with equation (10), each "performance function" $f_z(X_z)$ is defined as the average of the TCP packet-loss probability $P_z^{\text{loss}}(\cdot)$ over the fading level ϕ_z, considered a discrete stochastic variable ranging among L possible values ϕ_z^l happening with probability $p_{\phi_z^l}$:

$$f_z(X_z) = \sum_{l=0}^{L-1} \left[P_{\text{loss}}^z\left(\phi_z^l X_z\right)\right] \cdot p_{\phi_z^l};\ l = 0, 1, \ldots, L-1 \tag{36}$$

In practice, the weight $W_z^{\text{CAP}}(\cdot)$ and the dependency on the number CBR calls $\widehat{K}_z^{\max}(\cdot)$ are removed from (10).

The expression for $P_z^{\text{loss}}(\cdot)$ is taken from [Bisio] and reported in (37).

$$P_z^{\text{loss}}\left(\phi_z^l X_z\right) = \frac{32\left(k_z^{be}\right)^2}{3b_n^z \cdot (m_z+1)^2 \cdot (\phi_z^l X_z \cdot RTT + Q_z)^2} \tag{37}$$

where index z-th identifies the ST, k_z^{be} is the number of TCP sources at the allocation instant, b_n^z is the number of TCP packets ($b_n^z = 1$, in the results proposed in the following) covered by one acknowledgement, m_z is the multiplicative decrease parameter ($m_z = 1/2$, in this chapter), $\phi_z X_z$ is the available bandwidth for the z-th ST as specified in (1) and Q_z is the best-effort traffic buffer size. RTT is the TCP Round Trip Time. Channel errors are corrected by FEC codes (considered through ϕ_z) so getting negligible Bit Error Ratio (BER) values. It implies that the bandwidth available for data is strongly reduced, but makes feasible considering almost all losses (actually all, as assumed in [Bisio]) due to a bandwidth bottleneck (to congestion) and not to channel errors.

Two STs have been considered: "0", always in clear sky, and "1", which varies its fading level ϕ_1 according to real measures also used in [Celandroni03, Marchese05]. The number

of active TCP sources is set to $k_z^{be} = 10$, $z = \{0, 1\}$. The fading level is a deterministic quantity, supposed always fixed in the tests. The overall bandwidth available C_{tot} is set to 4 Mbps and the best-effort buffer size is 10 packets of 1500 bytes for each ST. TCP round trip time is considered fixed and equal to 100 ms for all STs. This value is computed by supposing buffer occupancy of 8 packets and an allocated rate of about 2 Mbps. Anyway RTT values do not affect the qualitative behaviour of the schemes, even if, obviously, the specific values change.

11.6.1 Bandwidth and Packet-Loss Probability

Figure 11.6 shows the Euclidean distance from the utopia point for all the considered schemes by varying the fading level. UMD minimizes it and, in a totally competitive environment, where all STs aim at getting as much bandwidth as possible without any agreed cooperation with the other entities, it provides the optimal allocation. The difference with the other schemes is more evident for serious fading conditions ($\phi_1 \leq 0.625$, in Table 11.1) because, approaching a clear sky condition, all performance functions tend to have the same value.

The second set of results proposed concerns the case where the number of STs ($Z = \{2, \ldots, 8\}$) and the number of TCP sources ($k_z^{be} = \{5, 10\}$) varies. Also in this case, one ST out of Z is seriously faded ($\phi_1 = 0.156$). The others operate in clear sky. Figures 11.7 and 11.8 contain the packet-loss probability for the faded ST (suffix "-F") and for the other STs (suffix "-C"). In this last case, the single values are averaged over the number of STs in clear sky. UMD, VALUE and NBS are considered. UMD does not punish the faded ST and allows, for $k_z^{be} = 5$ (Figure 11.7), to get packet-loss values feasible with most current services (e.g. voice and video streaming) for users connected both through the faded ST and through the other STs, so allowing, if necessary, interaction among them. This is not reachable with VALUE and NBS. The trend is the same for $k_z^{be} = 10$ (Figure 11.8) but the overload does not allow getting practical loss values in any case.

11.6.2 Performance Evaluation in Presence of QoS Constraints

The aim of this section is to show the performance of the allocation techniques UMD, CUMD and QPMD in terms of allocated bandwidth and packet-loss probability. The network

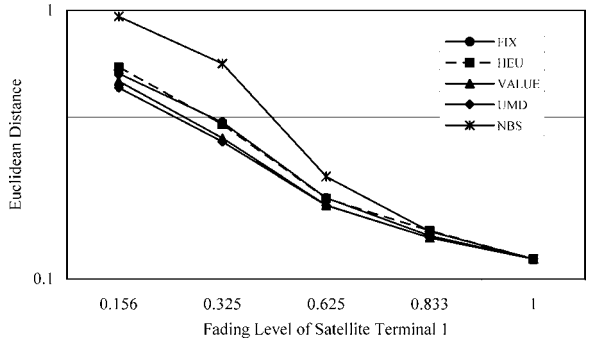


Figure 11.6 Euclidean distance from the utopia point

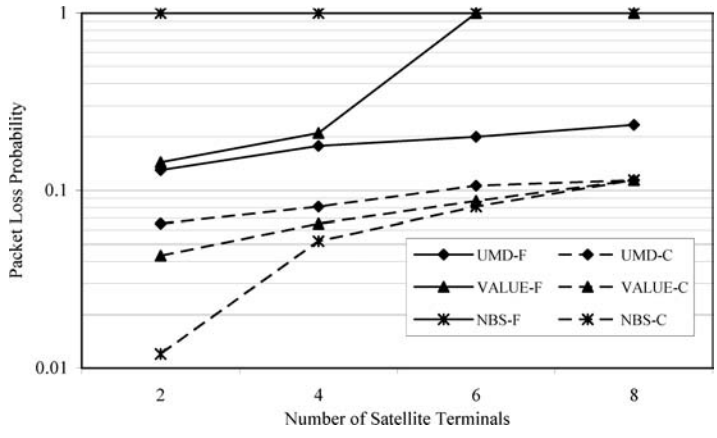


Figure 11.7 Packet loss probability versus number of satellite terminal – $k_z^{be} = 5$

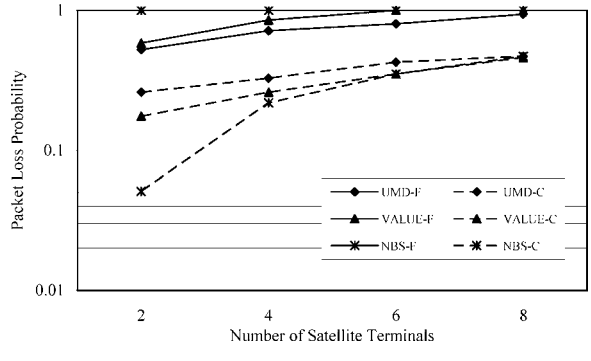


Figure 11.8 Packet loss probability versus number of satellite terminal – $k_z^{be} = 10$

scenario considered is unchanged with respect to the previous case with two STs, but the overall bandwidth available C_{tot} is set to 10.24 Mbps (10,240 Kbps). The RTT is supposed fixed and equal to 520 [ms] for all STs. It is considered comprehensive of the propagation delay of the channel and of the waiting time spent in the buffers of the STs.

In summary,

- UMD represents a completely competitive problem with no constraints. Its aim is to approach the ideal point where each ST (both, in this case) has the complete availability of $C_{\text{tot}} = 10.24\,[\text{Mbps}]$.
- CUMD solves the same problem but adds a set of constraints (32) so getting minimum bandwidth requirements for each ST.
- Following the same philosophy, QPMD tries respecting (or, it not possible, approaching) the constraint set (32) by originating a new ideal point represented by bandwidth assignations that allow respecting constraints.

The performance analysis is carried out by setting a fixed performance constraint (32) and showing bandwidth allocations and packet-loss probabilities for the two involved STs.

In detail, the tests are obtained by using $f_0(C_0) \leq 0.01$, $f_1(C_1) \leq 0.01$. Table 11.2 contains the minimum capacity (C_0^{thr} and C_1^{thr}) requirements to match performance constraints for STs 0 and 1. Again it is important to remind that C_0^{thr} and C_1^{thr} could be received directly by RCBC scheme in Chapter 9 without any constraint for $f_0(\cdot)$ and $f_1(\cdot)$. Table 11.3 shows the bandwidth assigned (Kbps) to STs 0 and 1 versus the fading level ϕ_1 of ST 1. Table 11.4 contains the values of the packet-loss probability for STs 0 and 1 versus the fading level of ST 1.

Concerning CUMD algorithm, values corresponding to $\phi_1 = 0.156$ are the most interesting to check the behaviour of the algorithm. It is the situation where constraints $f_0(C_0) \leq 0.01$, $f_1(C_1) \leq 0.01$ cannot be satisfied in the same time because the overall bandwidth is not sufficient ($C_1^{\text{thr}} \geq 10.24\,[\text{Mbps}]$, see the first row of Table 11.2). CUMD chooses to satisfy $f_0(C_0) \leq 0.01$ because $C_0^{\text{thr}} = 2.67\,[\text{Mbps}] \leq 10.24\,[\text{Mbps}]$ and to consider traffic from ST

Table 11.2 Minimum bandwidth requirement

ϕ_1	C_0^{thr} (Kbps)	C_1^{thr} (Kbps)
0.156	2,670	17,116
0.312	2,670	8,558
0.625	2,670	4,272
0.833	2,670	3,205
1	2,670	2,670

Table 11.3 Allocated bandwidth (Kbps)

ϕ_1	ST 0			ST 1		
	UMD	QPMD	CUMD	UMD	QPMD	CUMD
0.156	1920	1792	2670	8320	8448	7569
0.312	2944	2304	2670	7296	7936	7569
0.625	4224	2688	4224	6016	7552	6016
0.833	4736	2688	4736	5504	7552	5504
1	5120	5120	5120	5120	5120	5120

Table 11.4 Packet loss probability

ϕ_1	ST 0			ST 1		
	UMD	QPMD	CUMD	UMD	QPMD	CUMD
0.156	0.018	0.020	0.010	0.036	0.035	0.042
0.312	0.008	0.013	0.010	0.013	0.011	0.012
0.625	0.004	0.009	0.004	0.005	0.003	0.005
0.833	0.003	0.009	0.003	0.003	0.001	0.003
1	0.002	0.002	0.002	0.002	0.002	0.002

1 less important than traffic from ST 0. The minimum bandwidth (2.67 [Mbps]) is assigned to ST 0 and ST 1 gets the residual capacity (7.57 [Mbps]), as clear in Table 11.3.

The impact on the packet loss may be seen in Table. 11.4. ST 0 value is below the threshold. ST 1 is obviously penalized: the packet-loss probability value is far from the threshold. Also values corresponding to $\phi_1 = 0.312$ show an interesting situation for CUMD because, again, constraints $f_0(C_0) \leq 0.01, f_1(C_1) \leq 0.01$ cannot be satisfied at the same time, but both could be satisfied separately. CUMD chooses to privilege ST 0 because the choice assures minimum distance with the utopia point. Concerning the behaviour of CUMD for $\phi_1 \geq 0.625$, being bandwidth requirements to get performance constraints within the set of feasible allocations, as reported in Table 11.2, it totally overlaps UMD. Actually, the two schemes have the allocation mechanism based on the utopia point and, except for the constraint sets, which, even if acting, do not influence allocations if $\phi_1 \geq 0.625$, are the same algorithm.

Utopia Minimum Distance, being not constrained, has the only aim of minimizing the distance with the utopia point (i.e. to approach the behaviour where STs have the complete availability of channel bandwidth). It does not consider any global benefit for the network but acts in a completely competitive environment, where each ST pursues its own benefit.

QoS Point Minimum Distance changes the nature of allocation because it defines a new reference point (QoS Point) which, implicitly, contains the performance constraints. The QoS point values are the bandwidth allocations contained in Table 11.2. The shape of bandwidth allocations is similar to UMD but its aim is to keep the quantity $\left(\left\|\mathbf{F}(\mathbf{C}) - \mathbf{F}^{\mathrm{QoS}}\left(\mathbf{C}^{\mathrm{QoS}}\right)\right\|_2\right)^2$ minimum. Being the performance requirements the same for both STs, QPMD privileges the faded ST, compared to UMD. The consequent packet-loss probability values are shown in Table 11.4. As required by its aim, QPMD is the best scheme to have a common (among STs) way to approach performance constraints. The three approaches give the same allocations when they act in clear sky ($\phi_1 = 1$).

12

Transport Layer over Satellite

12.1 Introduction

In the previous chapters, the attention has been paid mainly to the design of protocol architectures suited to ensure QoS requirements by implementing proper QoS mapping algorithms and signalling protocols aimed at transporting the indicators of performance expected by the end-users. In this light, solutions working basically at the network layer and below have been analysed by pointing out also the necessity of implementing a Relay Layer, responsible for translating protocol formats and QoS requirements of data flows traversing different ASs, based upon different protocol technologies. This approach seems effective thanks to the frameworks formulated to address the problem of QoS control and to the definition of possible signalling architectures. Nevertheless, the advantages offered by such an approach can be amplified as transport layer design issues are taken under consideration, as explained in Chapter 10. Addressing QoS issues does not exclude the transport and application layer design because the peculiarities of satellite and wireless links (as investigated here) strongly impact on the performance of protocols working at the higher layers. In other words, this means that, potentially, the solutions proposed so far may be improved to guarantee optimal performance results. This observation takes even more importance in the case of file transfers, HTTP Web browsing, ssh application and, in general, in all the services relying on TCP protocol. This is due to the high sensitivity of TCP to all the characteristics, such as long delays and frame errors, which are not present in terrestrial networks. From the observation just drawn on the TCP limitations, it is immediate to observe that even if lower layer techniques offer guaranteed bandwidth to TCP flows, TCP itself may not be able to make effective use of these network resources because of its intrinsic characteristics. As a consequence, the ideal solution would be to implement transport layer protocols able to adapt to the environment characteristics in order to make the QoS solutions, proposed in this book, more effective, as stated in Chapter 10. Two basic approaches can be pursued. The first one considers the possibility of directly adopting powerful transport protocols on the end-users' host, in order to maintain the end-to-end semantic. This solution,

QoS over Heterogeneous Networks Mario Marchese

however, poses the problem of modifying the transport layer protocol on the information source and destination sides and, consequently, would imply the modification of interfaces and applications running over it. The second solution consists in the implementation of a relay layer in the QoS-PRN (-RN) gateway stack able to manage efficiently TCP connections as well as to deal QoS issues. To do that, the adoption of effective mechanisms aimed at translating the transport protocol formats from one domain to another is necessary and, consequently, the implementation of effective data communication mechanisms may be designed. This option is suited for Performance Enhancing Proxy (PEP) architectures, and for the design of QoS gateways that implement QoS-aware mechanisms at any level of the protocol stack, from the transport down to the physical layer. As said in Chapter 10, similar comments may be reported concerning application layer. Anyway, this chapter focuses on transport layer and, in detail, on TCP, so as to allow a full comprehension of the topic through practical examples.

In consequence, an introduction to communication issues arising at the transport layers is highlighted and basic solutions to overcome the hazards derived from the investigated environment are reported in sections from 12.4 to 12.7. For the sake of comprehension, an introduction on how TCP actually works is also given in sections 12.2 and 12.3.

12.2 The TCP Protocol

The Transmission Control Protocol (TCP) is a connection-oriented, end-to-end reliable transport protocol working between hosts in packet-switched networks, and between interconnected systems of such networks. The motivations, the philosophy and the functional specification of the protocol are contained in [RFC793].

Basically, the TCP assumes it can obtain a simple, potentially unreliable service from the lower level protocols and, in principle, the TCP should be able to operate above a wide spectrum of communication systems ranging from hard-wired connections to wireless and satellite networks. Nevertheless, its use within a satellite environment, even if it does not affect the correct working of the protocol, heavily affects the performance, as it will be shown in section 12.3. It is set just above the Internet Protocol (IP) [Comer], which offers a service to the TCP to send and receive information segments of variable length called "datagram" in the IP terminology. Being a network layer, IP manages also issues such as addressing and routing of the information. It may fragment TCP segments, which have to traverse portions of networks with different characteristics at the data link layer, as widely explained in the book.

As noted above, the primary purpose of the TCP is to provide reliable service between pairs of host computers. To match the requirement on top of a less reliable Internet communication, it uses facilities in the following areas: Basic Data Transfer, Reliability, Flow Control, Multiplexing, Connections, Precedence and Security. The detailed description of each operation, which can be found in [RFC793], is beyond the scope of this chapter. Nevertheless, it is important to focus on two areas whose implementation has strong impact on the metrics used to measure the performance when TCP is applied over satellite links: reliability and flow control. TCP rules the amount of data sent by the sender by applying an Automatic Repeat Request (ARQ) system. It assigns a sequence number to each segment and uses an acknowledgement mechanism that flows from the destination to the source to be sure that the information is arrived at the destination. Conceptually, each segment is

assigned a sequence number. The sequence number of the segment is transmitted with the segment itself. When the TCP transmits a segment, it puts a copy on a retransmission queue and starts a timer called Retransmission Timeout (RTO). When the acknowledgement for that segment is received, the segment is deleted from the queue. If the acknowledgement is not received before the timer runs out, the segment is retransmitted. The flow control mechanism employed is explained in detail in section 12.3.

12.3 The TCP Congestion Control

Most of this section is taken from [RFC2581], which is dedicated to describe the algorithms ruling the TCP behaviour in presence of congestion. In detail, it specifies four algorithms: slow start, congestion avoidance, fast retransmit and fast recovery. The definitions contained in Table 12.1 have been used.

Table 12.1 Definition of TCP parameters

Parameter	Definition
SEGMENT	Any TCP/IP data or acknowledgement packet (or both)
SENDER MAXIMUM SEGMENT SIZE (SMSS)	Size of the largest segment that the sender can transmit. It depends on the type of network used and on other factors
RECEIVER MAXIMUM SEGMENT SIZE (RMSS)	Size of the largest segment the receiver can accept
FULL-SIZED SEGMENT	A segment that contains the maximum number of permitted data bytes (i.e. SMSS bytes of data)
RECEIVER WINDOW (rwnd)	The most recently advertised receiver window. It is a receiver-side limit on the amount of outstanding data. It corresponds, at least in the implementations considered, to half of the receiver buffer length, at the beginning of the transmission
CONGESTION WINDOW (cwnd)	A TCP state variable that limits the amount of data a TCP can send. It is a limit on the amount of data the sender can transmit into the network before receiving an acknowledgement (ACK). Some implementations maintain cwnd in units of bytes, while others in units of full-sized segments
INITIAL WINDOW (IW)	Size of the sender's congestion window after the three-way handshake is completed
LOSS WINDOW (LW)	Size of the congestion window after a TCP sender detects loss using its retransmission timer
RESTART WINDOW (RW)	Size of the congestion window after a TCP restarts transmission after an idle period
FLIGHT SIZE	The amount of data that has been sent but not yet acknowledged. Actually, it identifies the segments still "in flight", still inside the network

A segment is considered lost either after the special timer (RTO) expires, as mentioned in the previous section and in [RFC793], or after 3 duplicated acknowledgements (4 ACKs indicating the same sequence number) are received as explained in the following.

The slow start and congestion avoidance algorithms are used by a TCP sender to control the amount of data to be injected into the network. The minimum of cwnd and the minimum between the source buffer and rwnd governs data transmission (the variable TW identifies the real transmission window). Another state variable, the slow start threshold (ssthresh), is used to determine whether either the slow start or the congestion avoidance algorithm is used to control data transmission, as discussed below.

Some more indications about acknowledgements generation are as follows. A TCP receiver should use the delayed ACK algorithm. It means that an ACK is not generated for each full-sized received segment but, in most implementations, for every second full-sized segment and, in any case, must be generated within 500 ms since the arrival of the first unacknowledged packet. Out-of-order data segments should be acknowledged immediately, in order to accelerate loss recovery.

12.3.1 Slow Start

The aim of slow start algorithm is to probe the network to check the available capacity so as to avoid congestion. The initial value of cwnd, IW, must be less than or equal to $2 \cdot \text{SMSS}$ bytes. A non-standard, experimental TCP extension allows using a larger window, whose value is limited by (1)

$$\text{IW} = \min\left(4 \times \text{SMSS}, \max\left(2 \times \text{SMSS}, 4380\,\text{bytes}\right)\right) \tag{1}$$

The initial value of ssthresh may be arbitrarily high and it is reduced in response to congestion. The slow start algorithm is used when cwnd $<$ ssthresh.

During slow start, TCP increases cwnd by at most SMSS bytes for each received ACK that acknowledges new data. The slow start phase ends either when cwnd exceeds ssthresh or when congestion is observed.

12.3.2 Congestion Avoidance

The congestion avoidance algorithm is used when cwnd $>$ ssthresh. When cwnd and ssthresh are equal, the sender may use either slow start or congestion avoidance. Congestion avoidance continues until congestion is detected. Within the congestion avoidance phase, cwnd is increased by 1 full-size segment after a number of ACKs corresponding to the value of cwnd/SMSS is arrived (each (cwnd/SMSS) ACKs $\rightarrow$ cwnd $=$ cwnd $+ 1 \cdot$ SMSS). One formula commonly used to update cwnd during congestion avoidance is given in (2). It contains the adjustment executed on every incoming non-duplicate ACK.

$$\text{cwnd} = \text{cwnd} + \text{SMSS} \times \text{SMSS}/\text{cwnd} \tag{2}$$

The implementations that maintain cwnd in units of full-sized segments find (2) difficult to use, and use another method to implement the general principle. Actually, the general principle on the base of Congestion Avoidance algorithm is that cwnd should be incremented

by 1 full-sized segment per round-trip time (RTT), which is defined as the time to get to the destination and to get back. Formula (2) allows approximating the general indication. When a TCP sender detects segment loss using the retransmission timer, the value of ssthresh must be set to no more than the value given in (3):

$$\text{ssthresh} = \max(\text{FligthSize}/2, 2 \times \text{SMSS}) \tag{3}$$

FlightSize is the amount of not yet acknowledged data in the network. It is important not to use cwnd rather than FlightSize. This mistake has characterized some TCP implementations in the past.

Furthermore, upon a timeout RTO, cwnd must be set to no more than the loss window, LW, which equals 1 full-sized segment, regardless of the value of IW. Therefore, after retransmitting the dropped segment, the TCP sender uses the slow start algorithm to increase the window from 1 full-sized segment to the new value of ssthresh, at which point congestion avoidance takes over again.

12.3.3 Fast Retransmit/Fast Recovery

A TCP receiver should send an immediate duplicate ACK when an out-of-order segment arrives. The purpose of this ACK is to inform the sender that a segment is received out-of-order and which sequence number is expected. From the sender's perspective, duplicate ACKs can be caused by a number of network problems (e.g. dropped segments, re-ordering of data and replication of data). Obviously, a TCP receiver will send an immediate ACK when the incoming segment fills all or part of a gap in the sequence space. The TCP sender uses the "fast retransmit" algorithm to detect and repair loss, which is based on incoming duplicate ACKs. The fast retransmit algorithm uses the arrival of 3 duplicate ACKs (4 identical ACKs without the arrival of any other intervening packets) as an indication that a segment has been lost. After receiving 3 duplicate ACKs, TCP performs a retransmission of what appears to be the missing segment, without waiting for the retransmission timer to expire.

After the fast retransmit algorithm sends what appears to be the missing segment, the "fast recovery" algorithm governs the transmission of the new data until a non-duplicate ACK arrives. The reason for not performing slow start is that the receipt of the duplicate ACKs not only indicates that a segment has been lost, but also that other segments are most likely to leave the network. For instance, if 3 duplicated ACKs reach the source, it means that 3 segments have reached the destination.

The fast retransmit and fast recovery algorithms are usually implemented together as follows.

- When the third duplicate ACK is received, the ssthresh value is set to the value given in (3).
- The lost segment is retransmitted and cwnd is set to ssthresh plus $3 \cdot \text{SMSS}$ (as already said, 3 duplicated acknowledgements mean that 3 segments have left the network).
- For each additional duplicate ACK received, cwnd is increased by SMSS.
- A segment is transmitted, if allowed by the new value of cwnd and the receiver's advertised window.

When the next ACK arrives that acknowledges new data, cwnd is set to ssthresh.

TCP researchers have suggested a number of loss recovery algorithms improving fast retransmit and recovery. Some of them are based on the TCP selective acknowledgement (SACK) option [RFC2018], which allows specifying exactly the sequence number of the missing segment.

Operatively and referring to a specific TCP implementation (a NewReno TCP under the 2.2.1 version of the Linux kernel), the TCP transmission begins with the slow start phase, where the congestion window (cwnd) is set to 1 segment ($IW = 1 \cdot SMSS$, in this implementation), and the slow start threshold (ssthresh) to a very high value (infinite). At each received acknowledgement (ACK), cwnd is increased by $1 \cdot SMSS$. If the value of cwnd is less than ssthresh, the system uses slow start. Otherwise, the congestion avoidance phase is entered. More precisely, cwnd is increased by $1 \cdot SMSS$ after receiving a number of "cwnd" acknowledgements. If there is a loss detected by 3 duplicated ACKs, the system enters the fast retransmit/fast recovery algorithm and performs a retransmission of the missing segment, without waiting for the retransmission timer to expire. ssthresh has been set to the maximum between FlightSize/2 and $2 \cdot SMSS$, where FlightSize is the measure (in bytes) of the amount of data sent but not yet acknowledged, i.e. the packets still in flight. cwnd is set to "ssthresh $+ 3 \cdot SMSS$". When the error is recovered, i.e. when the lost packets have been successfully retransmitted, the value of cwnd is set to ssthresh. The real transmission window (TW) is set, in any case, to the minimum between cwnd and the minimum between the TCP buffer dimension at the source and the receiver's advertised window (rwnd), which is half of the receiver TCP buffer length ($TW = \min\{cwnd, \min(source\ buff, rwnd)\}$). The receiver window rwnd has been measured to be 32 KB at the beginning of the transmission. It corresponds to half of the receiver buffer space, which is automatically set by the TCP to 64 KB. This numerical value is due to the TCP header (see [RFC793] and [Comer] for a detailed description) which uses a 16-bit field to report the receiver window size to the sender. Therefore, the largest window that can be used is 2^{16} bytes. To circumvent this problem, RFC 1323 [RFC1323] has defined a new TCP option, "Window Scale", to allow the use of larger windows. This option defines an implicit scale factor, which is used to multiply the window size value found in a TCP header to obtain the true window size. The "Window Scale" option is considered allowed in all the examples reported in the remainder of the chapter.

In a more schematic way, the procedures listed above may be summarized in Table 12.2; a C-like language is used for the description.

An important quantity is the "delay-bandwidth" product. As indicated in [RFC1323], the TCP performance does not depend upon the transfer rate itself, but rather upon the product of the transfer rate and the RTT. The "delay-bandwidth" product measures the amount of data that would "fill the pipe", as defined in Chapter 10, i.e. the amount of unacknowledged data, which TCP must handle in order to keep the pipe full. TCP performance problems arise when the delay-bandwidth product is large. In more detail, within a large delay-bandwidth product environment (as, for example, a geostationary satellite), the described acknowledgement mechanism takes a long time to recover errors. The propagation delay makes the acknowledgement arrival slow and cwnd needs more time to grow than in cable networks. If, for example, just 1 segment was sent, it takes at least 1 RTT to be confirmed. The throughput is very low, even in the slow start phase. This heavily affects performance of the applications based on TCP. The simple example in Table 12.3 may allow a better comprehension.

Table 12.2 TCP parameters

TW=min{cwnd, min(source buff, rwnd)}	
Slow start	$cwnd = 1 \cdot SMSS$; $ssthr = \infty$ $ACK \rightarrow cwnd = cwnd + 1 \cdot SMSS$
Congestion avoidance	$< cwnd/SMSS > ACKs \rightarrow cwnd = cwnd + 1 \cdot SMSS$
Fast retransmit/ recovery	$ssthr = \max\{FlightSize/2, 2 \cdot SMSS\}$; $cwnd = ssthr + 3 \cdot SMSS$; Duplicated $ACK \rightarrow cwnd = cwnd + 1 \cdot SMSS$; $cwnd = ssthr$

Table 12.3 Simplified TCP behaviour

Time (s)	Transmission Window (SMSS) RTT = 100 ms	Transmission Window (SMSS) RTT = 500 ms
0	1	1
0.1	2	1
0.2	4	1
0.3	8	1
0.4	16	1
0.5	32	2
0.6	64	2
0.7	128	2
0.8	256	2
0.9	512	2
1	1024	4

The control mechanism is simplified. The delayed ACK mechanism is not used and an ACK is sent for each full-size segment. No segment is lost and the slow start phase is never abandoned. In this simple example, the TW is considered limited not by the formula appearing in Table 12.2 but only by the value of cwnd. The example in the following is didactic and no real measures have been taken. The behaviour of the TCP (simplified as described above) when there is an RTT of 100 ms is compared with the behaviour when the RTT is 500 ms (the average RTT of a GEO satellite network). The value of the Transmission Window contains the number of SMSS (bytes) actually sent at the instant indicated in the first row. For example, if SMSS = 1500 bytes, the real number of bytes sent at the beginning (instant 0) is $1 \cdot 1500$, for both cases. Concerning the RTT = 100 ms case, after receiving the first ACK (i.e. after 0.1 s), cwnd is augmented by 1, and 2 segments may be sent; after $1 \cdot RTT$, 2 ACKs arrives substantially at the same time and 4 segments are allowed to leave the source (instant 0.2), and so on. The behaviour if RTT = 500 ms is exactly the same but at the time 0.2 the first ACK is not yet arrived and the second segment has not left the source. The result after 1 s is that 1024 SMSS has left the source if RTT = 100 ms and only 4 SMSS if RTT = 500 ms.

The problem is the delay of the network or, in more detail, the delay-bandwidth product of the network, which has a devastating effect on the acknowledgement mechanism used by the TCP. The effect would be similar if a TCP-like acknowledgement mechanism were used at application layer instead of at transport layer.

Figure 12.1 contains the throughput [KBps] really measured on the field if a file transfer of 675 KB is considered. The transfer is performed by using a standard TCP with a 64 KB buffer length at both the receiver and the source. Two RTT values have been set: 100 ms and 500 ms; the difference is outstanding. While the first case requires only 3 s, the other case needs more than 12 s. Table 12.4 contains the exact values of the overall transmission time and the throughput measured in the final phase of the connection (a movable window is used to measure the throughput).

The implications on applications are simple to imagine. It is sufficient to think of a tele-learning system where the student is waiting for the material (an image or a graph) or of a remote control system aimed at activating an alarm or a robot.

Moreover, within satellite environment, the problems are different if LEO (Low Earth Orbit), MEO (Medium Earth Orbit) or GEO (Geostationary Orbit) satellite systems are used [Maral]. Issues related to each environment are listed in [RFC2488].

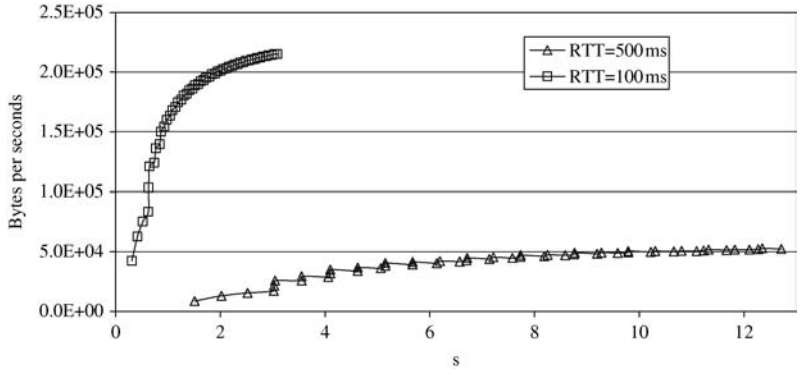


Figure 12.1 Throughput versus time, 675 KB file transfer

Table 12.4 Overall transmission time and throughput in the final phase of the connection

RTT (ms)	Time (s)	Throughput (Bps)
100	3.1	215,154
500	12.7	52,278

12.4 TCP over Satellite Networks

The following section is written for students and scientists who wish to know more details about the topic and to start a research activity about it. Actually, it is a list of references with few explanations. It should be considered as a guide for further studies. Its reading may also be postponed to the end of the chapter. The problem of improving TCP over satellite has been investigated in the literature for some years: see [Partridge97] for a first overview on the topic and [Lakshman97] for a more specific study in TCP/IP networks with high delay-bandwidth product and random loss. More recently, [Ghani99] provides a summary about improved TCP versions as well as issues and challenges in satellite TCP and possible enhancements at the link layer; [Henderson99] highlights the ways in which latency and asymmetry impair TCP performance; [RFC2760] lists the main limitations of the TCP over satellite and proposes many possible methods to act. Reference [Barakat00] is an excellent tutorial paper on the topic: various possible improvements both at the transport level and at the application and networks level are summarized and referenced; the paper focuses also on large delay-bandwidth product network and suggests possible modifications to TCP, as the buffer size. An issue of *International Journal of Satellite Communications* has been entirely dedicated to IP over satellite [Ephremides01]. In more detail, [Bharadwaj01] proposes a TCP splitting architecture for hybrid environments (see also [Zhang97]); [Kruse01] analyses the performance of Web retrievals over satellite and [Marchese01] shows an extensive analysis of the TCP behaviour by varying parameters as buffer size and initial congestion window. Reference [Goyal01] focuses also on buffer management but in an ATM environment. Also, International Standardization Groups as the Consultative Committee for Space Data Systems (CCSDS), which has already emitted a recommendation [CCSDS], and the European Telecommunications Standards Institute [ETSI], which is running its activity within the framework of the Satellite Earth Station – Broadband Satellite Multimedia (SES BSM) working group, are active on these issues. A recent paper by the author of this book is contained in [MarcheseIETE]. Much material in the following is taken from this reference.

The concept that the satellite portion of a network might be isolated and receive a different treatment and attention with respect to other parts of the network is also investigated in [Bharadwaj01]; methodologies such as TCP splitting [Bharadwaj01, Ghani99, Partridge97, Zhang97] and TCP spoofing [Partridge97, Zhang97] bypass the concept of end-to-end service by either dividing the TCP connection into segments or introducing intermediate gateways, with the aim of isolating the satellite link, as proposed in the previous chapters of this book. The drawback is losing the end-to-end characteristics of the transport layer. Reference [RFC3135] is dedicated to extend this concept by introducing Performance Enhancing Proxies (PEPs) intended to mitigate link-related degradations. Reference [RFC3135] is a survey of PEP techniques, not specifically dedicated to the TCP, even if emphasis is put on it. Motivations for their development are described as well as consequences and drawbacks. Most of the transport layer PEP implementations split the transport connection and, as a consequence, the end-to-end characteristic is lost.

Many national and international programmes and projects (listed extensively in [Marchese01]) in Europe, Japan and USA concern satellite networks and applications. In particular, some of them, or part of them, are aimed at improving performance at the transport level. NASA ACTS [Brooks99, Ivancic99], ESA ARTES-3 [ARTES] and CNIT-ASI [Adami01] deserve a particular attention, among many others.

12.5 TCP Parameters

In order to overcome TCP limitations, because of the large delay-bandwidth product, a first possibility is to parameterize the TCP protocol. The problem, as said, concerns the TCP congestion control summarized in Table 12.2.

Table 12.5 contains a proposal for the parameterization of TCP. The parameters IW and Th, along with the two functions $F(\cdot)$ and $G(\cdot)$ may be tuned following both the characteristics of the physical channel (delay, loss and bit error rate) and the network state (e.g. congestion). The function $F(\cdot)$ is aimed at regulating the size of the congestion window in the SLOW START phase. The characteristics of $F(\cdot)$ affect the increase of the window size and, as a consequence, the transmission speed and the protocol performance. The definition of $F(\cdot)$ is not trivial and many considerations may affect the decision. The increment in cwnd strictly depends on the current value of the cwnd itself and on the number of received acknowledgements, as indicated in Table 12.5. The choice allows tuning the behaviour of the protocol in dependence of the congestion window and to measure, at some extent, the network state represented by the arriving acknowledgements. The function $G(\cdot)$ is aimed at regulating the behaviour of the CONGESTION AVOIDANCE algorithm. The modification of the congestion avoidance scheme has not provided outstanding results over GEO channels but it might be very useful in LEO and radio-mobile environments.

A practical approach on the field has been carried out. Among the parameters indicated above such as the buffer length both at the source (source buff) and at the destination (it affects rwnd), the initial window (IW) and the function $F(\cdot)$ have been studied. The value of the parameter Th has been set to a very high value (infinite); the function $G(\text{cwnd}, \cdot)$ has been set to 1.

The study concerning the buffer length and the initial window partially derives from [Marchese01], the investigation of function $F(\cdot)$ partially from [MarcheseCC01] and from [MarcheseICC01].

12.5.1 The Real Test-bed

The used real test-bed is described in the following. Two remote hosts are connected through a satellite link by using IP routers. The experimentation, in practice, is concentrated over the

Table 12.5 Parameterized TCP

TW=min{cwnd, min(source buff, rwnd)}	
SLOW START [cwnd<ssthr]	cwnd = **IW** · SMSS ssthr = **Th** ACK → cwnd = cwnd + **F** (# of received acks, cwnd)·SMSS
CONGESTION AVOIDANCE [cwnd≥ssthr]	< cwnd/SMSS > ACKs → cwnd = cwnd + **G**(cwnd, •) SMSS
FAST RETRANSMIT / RECOVERY	ssthr=max{FlightSize/2, 2 · SMSS} cwnd = ssthr + 3 · SMSS Duplicated ACK → cwnd = cwnd + 1 · SMSS cwnd = ssthr

satellite portion (as shown in Figure 10.2). An average Round Trip Delay (RTD) of 511 ms has been measured. The TCP/IP protocol stack is used. The data link layer of the router uses HDLC encapsulation on the satellite side, where a serial interface is utilized, and Ethernet on the LAN side towards the hosts. A raw Bit Error Rate (i.e. BER with no channel coding) approximately of 10^{-2} has been measured; the utilization of a sequential channel coding with code rate 1/2, to correct transmission errors, has allowed reaching a BER of about 10^{-8}. As a consequence, the data link protocol "sees" a reliable channel. The system offers the possibility of selecting the transmission bit rate over the satellite link and a bit rate of 2048 Kbps has been used for the tests.

12.5.2 Test Application

The application used to get the results is a simple ftp-like one, i.e. a file transfer application located just above the TCP. It allows transferring data of variable dimension (DIM [bytes] in the following) between two remote sites. The designed application allows transferring a single file at a time, which is a case reported widely in the literature, both as a benchmark for working and as a configuration used in real environments. Two types of files have been utilized to perform the tests and to study the behaviour of the modified TCP: a file of relevant dimension of about 2.8 MB (2,800,100 bytes), indicated with DIM = 2.8 MB, and a small file of about 100 KB (105,100 bytes), indicated with DIM = 100 KB. The multiple connections case – reported to show the effect of a loaded network on the modified TCP – is obtained by activating a fixed number (NConn, in the following) of connections at the same time. Due to the high speed of the activation, the effect is loading the test-bed with NConn connections at the same time, even for transfers of short files. File transfers of 100 KB each are assumed for the multiple case. It should be highlighted that most of the Internet applications, as browsing, are mainly based on file transfer. A quick and secure file transfer guarantees a high quality of service to the users. Many network applications, e.g. distance learning, use massive file transfer.

12.5.3 Buffer Length and Initial Window (IW)

The analysis is dedicated to investigate the behaviour of TCP by varying the value of the initial window (IW is measured in bytes; i.e. the notation IW = 1 means IW = 1 · SMSS [bytes]) and of the buffer dimension. The latter is intended as the memory availability in bytes, for source and destination, which is kept equal; i.e. the buffer has the same length both for the source and for the destination. It is identified with the variable "buf" in the following.

Concerning the initial congestion window, the issue has also been treated in the literature. Simulation studies, though not specific for satellite environments [RFC2415], show the positive effect of increased IW for a single connection. Reference [RFC2414] clarifies the strict dependence of the performance on the application environment and suggests that "larger initial windows should not dramatically increase the burstiness of TCP traffic in the Internet

today". IW is set to 1 in the TCP version taken as reference. The performance improvement (i.e. the reduction of the time required for the whole transmission and the higher throughput) provided by varying the value of IW is shown in Figure 12.2 for a buffer (buf) of 64 KB, which is the value set by using TCP if no modification is imposed. A file transfer of 2.8 MB is performed in this case.

The buffer length is very important for the performance of the system; it rules the congestion window by imposing a bottleneck to its increment.

Figure 12.3 shows the throughput versus time for different values of buf (DIM = 2.8 MB). IW is set to 6.

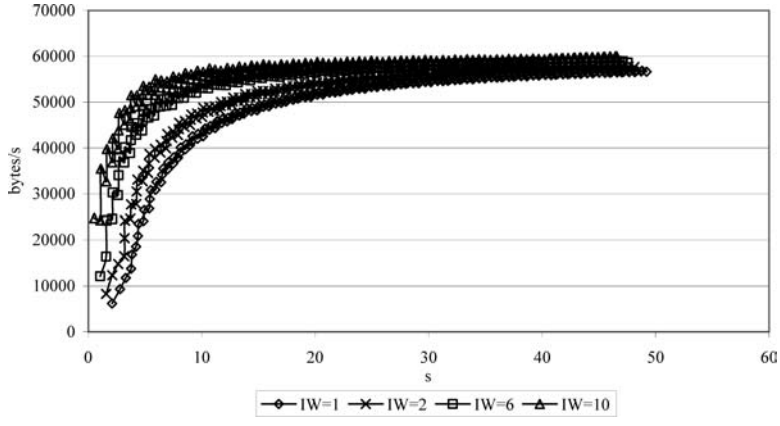


Figure 12.2 Throughput (KBps) versus time for different values of the initial congestion window (IW), buf = 64 KB, DIM = 2.8 MB

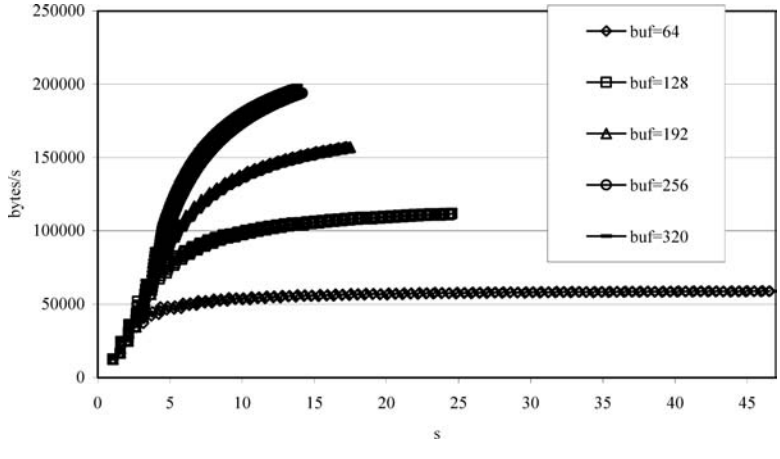


Figure 12.3 Throughput (KBps) versus time for different values of the buffer length, IW = 6, DIM = 2.8 MB

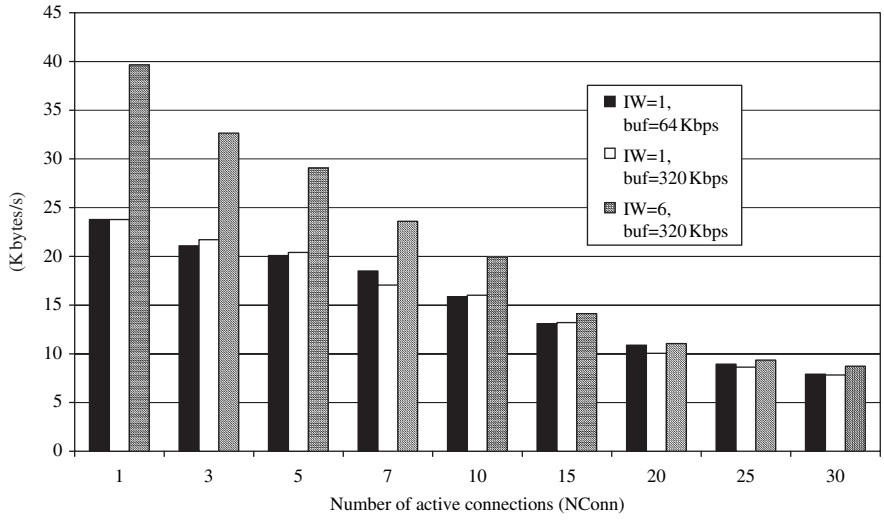


Figure 12.4 Throughput (KBps) versus the number of active connections (NConn), DIM = 100 KB

Figure 12.4 shows the average throughput per connection and compares the behaviour of different TCP configurations and the number NConn of active connections for a 100 KB transfer. The test is aimed at verifying the effect of the modifications proposed in the presence of a network loaded with multiple connections. The case considered is typical when several users, who try to download remote data, share a WWW site. Three configurations are taken into account: the reference configuration (IW = 1, buf = 64 KB), one configuration where only the buffer is varied (IW = 1, buf = 320 KB) and one configuration where both buf and IW are increased (IW = 6, buf = 320 KB).

It is important to note that the performance improvement due to modified values of IW and buf is not cancelled by the higher traffic load. The gain in throughput is evident up to 15 active connections; for larger values of N, the traffic load due to the number of connections in progress makes the TCP insensitive to the modifications introduced.

In this view, a parameterization, called "Multi Threshold Smoothed Increase" (MTSI), of the slow start increase function has been proposed in [MarcheseIETE] and briefly summarized here. The algorithm is based on the introduction of a different increasing function $F(\cdot)$, whose value when the N-th acknowledgement (Ack_N) is received depends on the current dimension of the congestion window (cwnd) and on the value of $F(\cdot)$ when the $(N-1)$-th acknowledgement is received (Ack_{N-1}).

cwnd is updated as follows,

$$\text{cwnd}(\text{Ack}_N) = \text{cwnd}(\text{Ack}_{N-1}) + F(\text{Ack}_N, \text{cwnd}(\text{Ack}_{N-1})) \cdot \text{smss} \tag{4}$$

The function $F(\cdot)$, introduced in [MarcheseICC01], is aimed at regulating the size of the congestion window in the slow start phase. The characteristics of $F(\cdot)$ affect the increase of the window size and, as a consequence, the transmission speed and the protocol performance. The definition of $F(\cdot)$ is not trivial and many considerations may affect the decision: the choice is aimed at increasing the transmission speed in the initial phase on the basis of the

experiments performed with an augmented value of IW without entering a congestion period. The function introduced in [MarcheseIETE] is called $F_{\mathrm{MTSI}}(\cdot)$. The choice allows tuning the behaviour of the protocol depending on the congestion window and to measure, at some extent, the network state represented by the arriving acknowledgements.

TCP sets the function

$$F(\mathrm{Ack}_N, \mathrm{cwnd}(\mathrm{Ack}_{N-1})) = 1 \tag{5}$$

$\mathrm{F}_{\mathrm{MTSI}}(\cdot)$ applies a linear increase which depends on the number N of received ACKs and on the current value of the congestion window, as shown in (6.a). In more detail, for the sake of completeness, within the application range of each threshold, $\mathrm{F}_{\mathrm{MTSI}}(\cdot)$ sets the function as in (6.b), where the increase is smoothed over time, depending on the threshold thr, linked to the dimension of the congestion window.

$$\mathrm{F}_{\mathrm{MTSI}}(\mathrm{Ack}_{\mathrm{N}}, \mathrm{cwnd}\ (\mathrm{Ack}_{\mathrm{N}-1})) = \begin{cases} \mathrm{N} \cdot \mathrm{K}_{\mathrm{thr}}(\mathrm{Ack}_{\mathrm{N}}) & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{\mathrm{N}-1}) < \mathrm{thr}_{\mathrm{n}} \\ 1 & \text{otherwise} \end{cases} \tag{6.a}$$

$$\mathrm{F}_{\mathrm{MTSI}}(\mathrm{Ack}_{\mathrm{N}}, \mathrm{cwnd}(\mathrm{Ack}_{\mathrm{N}-1})) = \begin{cases} \mathrm{F}_{\mathrm{MTSI}}(\mathrm{Ack}_{\mathrm{N}-1}, \mathrm{cwnd}(\mathrm{Ack}_{\mathrm{N}-2})) + \mathrm{K}_{\mathrm{thr}}(\mathrm{Ack}_{\mathrm{N}}) & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{\mathrm{N}-1}) < \mathrm{thr}_{\mathrm{n}} \\ 1 & \text{otherwise} \end{cases} \tag{6.b}$$

$$K_{\mathrm{thr}}(\mathrm{Ack}_N) = \begin{cases} K_{\mathrm{thr}_1} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_1 \\ K_{\mathrm{thr}_2} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_2 \\ K_{\mathrm{thr}_3} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_3 \\ . & . \\ . & . \\ K_{\mathrm{thr}_n} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_n \end{cases} \tag{7}$$

A variable number of thresholds (i.e. thr_n, where $n \in \mathbf{N}$) may be used. $F_{\mathrm{MTSI}}(\cdot)$ is aimed at adapting the protocol behaviour through the constants (K_{thr_n}, if n thresholds are used). Three thresholds have been heuristically estimated to be a proper number to increase the rate in the first phase of the transmission and to smooth it over time. The chosen function appears as in (8).

$$K_{\mathrm{thr}}(\mathrm{Ack}_N) = \begin{cases} K_{\mathrm{thr}_1} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_1 \\ K_{\mathrm{thr}_2} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_2 \\ K_{\mathrm{thr}_3} & \text{if } \mathrm{cwnd}(\mathrm{Ack}_{N-1}) < \mathrm{thr}_3 \end{cases} \tag{8}$$

The notation used to specify the threshold involved is $\mathrm{MTSI}(\mathrm{thr}_1 - \mathrm{thr}_2 - \mathrm{thr}_3)$; the value of the constant K_{thr_n}, with $n \in \{1, 2, 3\}$, represents the angular coefficient of the linear increase. Its value governs the speed of the increase and rules the protocol behaviour.

"TCP MTSI" is compared in Table 12.6 with:

- TCP (Initial Window IW = 1 and buffer set to 64 KB, recommended values for TCP implementations), identified as "TCP IW = 1 buf = 64 KB";

Table 12.6 Overall transfer time and average throughput, 2.8 MB, "TCP IW = 1 buf = 64 KB" as reference

TCP configuration	Transfer time (s)	Average throughput (KBps)
TCP IW = 1 buf = 64 KB	49.21	56.7
TCP IW = 6 buf = 320 KB	13.96	199.8
TCP MTSI	12.57	221.9

- TCP using an extended buffer length (320 KB) together with an extended initial window (IW = 6), identified as "TCP IW = 6 buf = 320 KB", extensively analysed in [MarcheseCC01].

The satellite channel provides 2 Mbps. A file of 2.8 MB is transferred in the mono-connection case.

The comparison with "TCP IW = 1 buf = 64 KB" allows to have a global idea about the improvement guaranteed by the new increase strategy, while the comparison with TCP implementing only extended buffer and initial window allows to focus on the gain guaranteed by the slow start modification independently of the contribution given by buffer and initial window. A detailed performance comparison with other schemes is reported in [MarcheseIETE].

TCP MTSI is used in the "TCP MTSI 20-30-40" configuration.

12.6 Complete Knowledge TCP

The TCP modifications reported in section 12.5 help understand the problems introduced by a large delay-bandwidth product environment and possible interventions limited to parameters and algorithms tuning. Nonetheless, the benefits that these modifications are able to bring are limited since they do not rely on the detailed knowledge of the specific environment and of the protocols acting below TCP. From this viewpoint, it is immediate to realize that such TCP parameterizations operate essentially blindly, hence yielding a performance gain with respect to common TCP implementations but without ensuring the achievement of the maximum allowed by the specific environment. For this reason, making use of the knowledge of the environment characteristics (e.g. available bandwidth and propagation delay) along with the pecularities of the underlying protocols (as explained in Chapter 10) is expected to further improve the performance. As highlighted in Chapter 10, it is possible to exploit the "Complete Knowledge", and consider a more effective transport layer protocol, tailored to the satellite environment, which is henceforth referred as "Complete Knowledge-Satellite Transport Protocol" (CK-STP).

Actually, shortcomings deriving from the use of TCP over satellite networks, as summarized and pointed out previously, are mainly due to the ineffective reaction of TCP to link errors and to the high delay-bandwidth product that prevents TCP from using as much available bandwidth as possible. Thus, CK-STP is supposed to fix these two points, by taking advantage of the knowledge offered by the lower layers, specifically in terms of available satellite bandwidth and propagation delay experienced on the link. Information about the available bandwidth is determined on the basis of resource allocation operations performed at the data link layer and of the bandwidth assignment policies implemented at the network layer aimed at ensuring specific QoS levels to specific classes of services (e.g. on DSCP basis with a DiffServ approach), as widely highlighted in previous chapters.

As far as the propagation delay is concerned, its evaluation is, in general, not simple because RTT fluctuations can arise when either the communication nodes are mobile or the times to traverse buffers (of the network and data link layers) cannot be considered negligible. These observations, however, can be relaxed in the case of geostationary satellite networks, which are used in the case study reported in this chapter.

Moving directly to the CK-STP design, the initial window is set to the delay-bandwidth product and the congestion window value is no longer increased when acknowledgments arrive. It is straightforward that the TCP buffers on the receiver and sender sides are also properly tuned in order to make the modifications indicated above effective.

Also, the TCP recovery mechanism has been modified, by entering the "Fast Retransmission" once a duplicated acknowledgement arrives. The main improvement of this proposal is that, at the end of the recovery phase, the congestion window value is not reduced and, as a consequence, the transmission rate is the maximum possible as set at the opening of the connection in order to "fill" the bandwidth pipe. Also, timeout management has been slightly modified. In particular, when the timer expires, the congestion window is not reduced to 1 segment but, once terminated the retransmission phase, the data communication continues with the maximum possible rate. A short summary of the CK-STP proposal is shown in Table 12.7.

In order to assess the performance offered by CK-STP deeply, it is convenient to consider different satellite configurations and applications. In this perspective, tests are performed by

Table 12.7 CK-STP specification

CK-STP	
Transmission algorithms/events	
"Normal Transmission"	Setup: IW =<delay-bandwidth product> cwnd = IW ssthresh = Infinity MAX_RTO=120 SYN_RETRIES=10 TCP_RETRIES2=15 Data communication Acknowledgment arrival TW=min(source buff, min (cwnd, rwnd))
"Fast Retransmit/Fast Recovery"	<retransmit data> <transmit new data if possible>
1 Duplicated Acknowledgment	"Fast Retransmit" entered
"Fast Recovery" terminated	"Normal Transmission" entered
Timeout	Backoff algorithm: RTO=min (2RTO, MAX_RTO) <retransmit data> "Normal Transmission" entered

varying the bandwidth availability (500 Kbps, 1 Mbps and 2 Mbps) as well as the dimension of the file transfer, so that the characteristics of the new solution can be investigated depending on both of them. Considered file dimensions are listed in the following, precise values are reported in brackets: 10 KB (10,136 bytes), 100 KB (102,808 bytes), 1 MB (1,048,352 bytes), 3 MB (3,145,056 bytes) and 10 MB (10,486,416 bytes).

"TCP MTSI" (which applies, again, an Initial Window of 6·SMSS, a source/receiver buffer length of 320 KB and a function F(·) utilized with $thr_1 = 20 - K_{thr_1} = 4$, $thr_2 = 30 - K_{thr_2} = 2$, $thr_3 = 40 - K_{thr_3} = 1$) and CK-STP (which applies a source/receiver buffer length of 320 KB and transmission window tuned to the delay-bandwidth product equal to 21, 42 and 84 SMSS for bandwidth values of 500 Kbps, 1 Mbps and 2 Mbps, respectively) are compared with:

- TCP (Initial Window IW = 2 and buffer set to 64 KB, used within the more recent operating system protocol implementations), identified as "TCP IW = 2 buf = 64 KB". Over the satellite test-bed, the performance is substantially equivalent to IW = 1 for large file transfers (there is an improvement of 2.2 % in case of a file of 2.8 MB, but the improvement is relevant for short transfers – 14.9 % in case of a 100 KB file).
- TCP (Initial Window IW = 6 and buffer set to 320 KB), identified as "TCP IW = 6 buf = 320 KB". The aim of this configuration is to isolate exactly the real contribution of the new scheme, not considering the advantage already given by the extension of the buffer length and initial window.

Table 12.8 contains the overall transmission time (measured in seconds) for the different solutions at the transport layer identified above.

Table 12.8 Overall transmission time (s) for different transport layer implementations, by varying bandwidth available and file dimension

		TCP IW=2 buf=64 KB	TCP IW=6 buf=320 KB	CK-STP	TCP MTSI
Bandwidth (bps)	File (bytes)	Time (s)	Time (s)	Time (s)	Time (s)
500 K	10 K	1.66	1.09	0.68	1.09
	100 K	4.28	3.18	2.22	2.64
	1 M	21.19	18.89	17.89	18.33
	3 M	58.7	53.71	52.64	53.16
	10 M	189.96	175.63	174.32	175.08
1 M	10 K	1.61	1.07	0.60	1.07
	100 K	3.95	2.86	1.36	1.83
	1 M	20.22	10.76	9.20	9.7
	3 M	56.16	28.04	26.58	27.13
	10 M	181.67	89.25	87.42	88.3
2 M	10 K	1.58	1.05	0.55	1.05
	100 K	3.8	2.71	0.94	1.69
	1 M	19.75	7.27	4.86	5.73
	3 M	54.91	16	13.54	14.46
	10 M	177.54	46.55	43.96	44.99

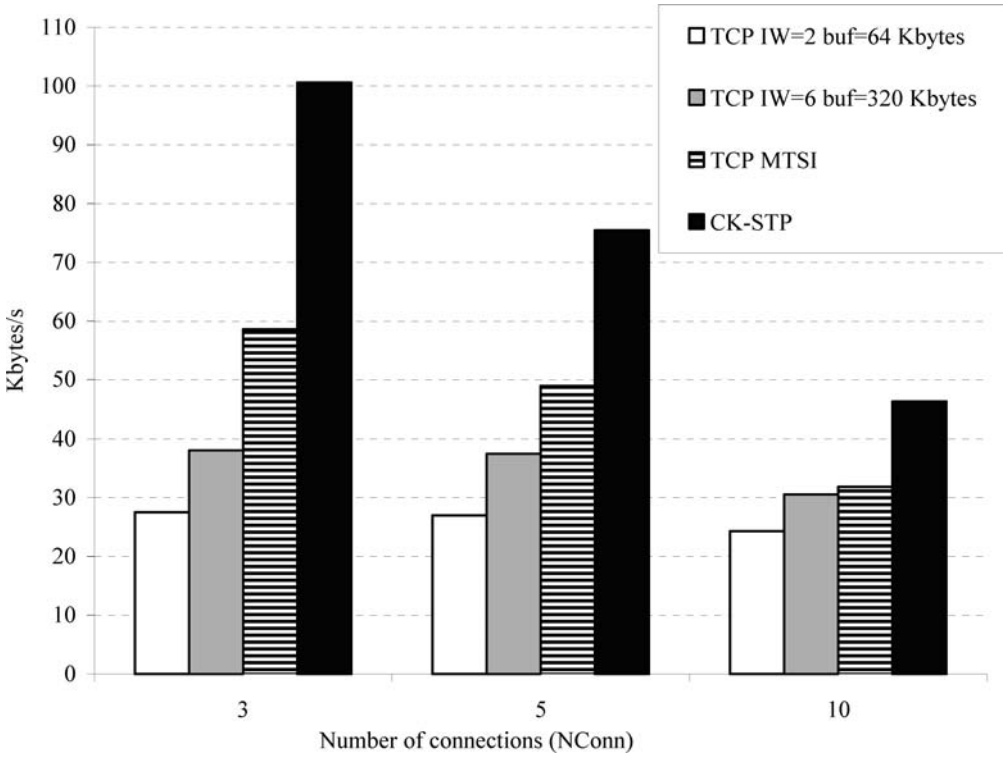


Figure 12.5 Throughput per connection versus number of connections, multiple connection case, 625 KBps

Figures 12.5 and 12.6 contain the throughput per connection and the overall throughput by varying the number of connections, respectively, in the presence of multiple connections and 625 KBps of available bandwidth.

As far as "CK-STP" is concerned, the results are quite obvious: it outperforms all the solutions since it behaves ideally in absence of errors.

12.7 Further Improvement of the Performance

Having in mind a scenario in which QoS-PRN gateways are responsible for routing reliable data and matching pre-defined QoS requirements (i.e. SLSs/SLAs), it is immediate to think of these gateways as acting as PEP nodes, thus implementing a relay layer defining advanced communication functionalities. From this standpoint, it is straightforward to give a look at what a PEP architecture really is, and how it can be suited to the scenario investigated throughout the book.

Essentially, a PEP architecture is composed of specialized middle boxes, acting as gateways among domains built on different protocols and media technologies, implementing advanced networking and communication functions so as to overcome hazards introduced by the environment, which would impair the protocol performance. From this viewpoint,

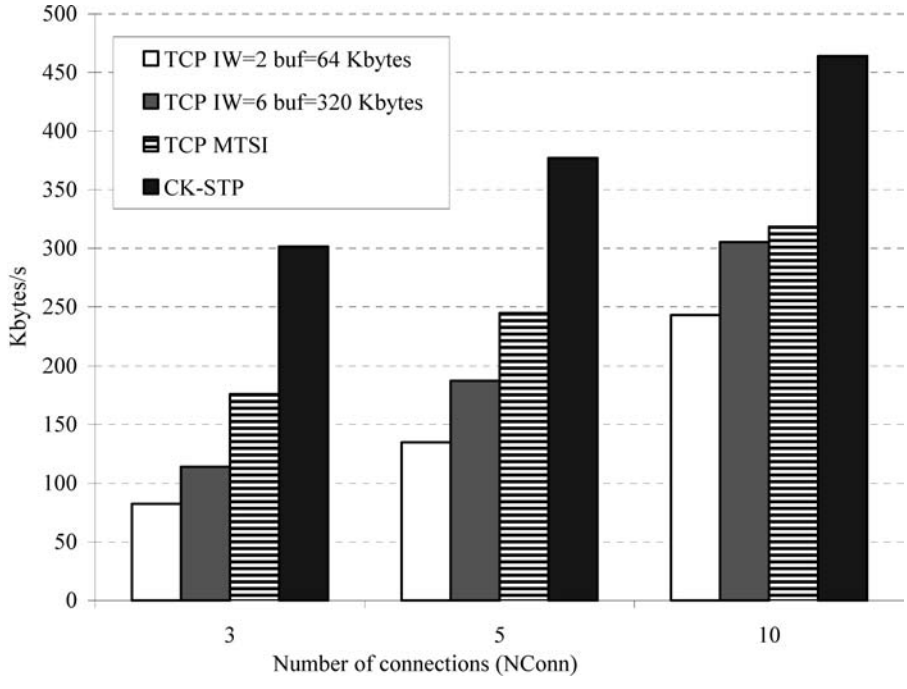


Figure 12.6 Overall throughput versus number of connections, multiple connection case, 625 KBps

it is immediate to recognize different issues that can be addressed by a PEP architecture: achieving throughput maximization, matching QoS users' requirements and dealing with handoff/handover events efficiently, just to mention some open problems. A characteristic of PEPs is that, dependently on the scenario peculiarities and the performance expectations, the implementation can include different layers of the protocol stack. In more words, two possible approaches can be identified: "layer by layer" and "integrated". In the former, the PEP implementation actually resides in a specific layer (e.g. transport) and, consequently, is aimed at mitigating specific impairments that can affect the performance of the protocol working at that layer. On the other hand, the latter approach consists in the distributed implementation of the PEP architecture across different layers, thus exploiting the benefits of cross-layer mechanisms properly defined to transfer the "knowledge" necessary to optimize the performance metrics among the layers. Obviously, in this case, a more complex control plane is expected to manage PEP functionalities implemented within each layer and, accordingly, the cross-layer mechanisms.

As far as "layer-by-layer" implementation is concerned, different protocol layers can be considered for the PEP implementation: application, transport, network and data link [RFC3135].

Application Layer. Today, different kinds of application layer proxies are widely used in the Internet. Such proxies include Web caches and relay Mail Transfer Agents (MTA) and they typically try to improve performance, service availability and reliability in general, but they do not necessarily include any optimizations specific for environment characteristics. Application layer PEPs, on the other hand, can be implemented to improve application

protocol as well as transport layer performance with respect to a particular application over a particular type of link. An application layer PEP may have the same functionality as the corresponding regular proxy for the same application (e.g. relay MTA or Web caching proxy) but extended with link-specific optimizations of the application protocol operation. Some application protocols employ extraneous round trips, excessively verbose headers and/or inefficient header encoding, which may have a significant impact on performance, in particular, with long delay and slow links. This unnecessary overhead can be reduced in general or for a particular type of link, by using an application layer PEP in an intermediate node. Some examples of application layer PEPs introduced to improve performance on slow wireless WAN links are described in [Liljeberg96] and [Chang97].

Transport Layer. Transport layer PEPs operate at the transport level. They may be aware of the type of application carried by the transport layer but, at most, only use this information to influence their behaviour with respect to the transport protocol; they do not modify the application protocol in any way, but let the application protocol operate end-to-end. Most transport layer PEP implementations interact with TCP. Such implementations are called "TCP PEPs". For example, in environments where ACKs may bunch together causing undesirable data segment bursts, a TCP PEP may be used simply to modify the ACK spacing in order to improve performance. On the other hand, in environments with a large delay-bandwidth product, TCP PEPs may be used to alter the behaviour of the TCP connection by generating local acknowledgements to TCP data segments in order to improve connections' throughput. The term "TCP spoofing" is sometimes used as a synonymous for TCP PEP functionality. However, the term "TCP spoofing" more accurately describes the characteristic of intercepting a TCP connection in the middle and terminating the connection as if the interceptor were the intended destination. This is a characteristic of many TCP PEP implementations, but not of all TCP PEP implementations.

Network Layer. PEP nodes working at the network layer are actually designed to mitigate performance degradation due to mobility issues such as handover/handoff events that may prevent from routing data correctly. To this aim, a number of solutions have been provided in the years, mostly addressing routing inefficiencies. Besides, middle boxes implementing advanced QoS functionalities (e.g. QoS mapping and resource reservation schemes) can reasonably fall in this category.

Data link Layer. PEP nodes working at the data link layer implement specific functionalities aimed at contrasting transmission link impairments, by using solutions very similar to those adopted at the transport layer within the transport layer PEPs. A particular attention may be given to SNOOP and SNOOP-ELN (Error Loss Notification) solutions, designed to overcome link degradations, common in satellite and radio environments. Basically, they rely upon the concept of snooping (i.e. intercepting and processing) data packets; afterwards, depending on the specific implementation, either retransmissions are performed or transmission is frozen, followed by the notification of link errors.

As introduced before, another possibility is represented by the use of "integrated" PEP, exploiting advanced communication and networking functionalities implemented at each layer. Actually, this approach allows introducing to optimization frameworks, as said in Chapter 10. What is evident here is that an integrated PEP may consider typical issues for transport layer scientists, such as throughput maximization, and also resource reservation and QoS management, which are typically addressed at the network, data link and physical layer. In this

light, it is immediate to forecast a QoS-gateway, whose implementation follows the QoS-PRN philosophy, and it is also able to tune transport/application layers actions according to the network portion characteristics and the data traffic traversing the network.

In particular, the relay layer at the top layer of the QoS-PRN gateway protocol stack should implement also typical transport layer PEP mechanisms, such as connection splitting, spoofing and ACK filtering/pacing. This possible implementation, obviously, leads to divide virtually the Relay Layer into two sublayers (network and transport), as shown in Figure 10.4. In this perspective, the transport layer should be responsible to intercept the incoming TCP connections and to implement suitable schemes to improve performance, such as splitting connection and spoofing. Furthermore, the need of tuning automatically the transport layer protocol should be taken into account.

In more detail, a list of hot research topics should be addressed for the relay layer design within QoS gateways:

- Definition of an ad hoc transport protocol, working on a peculiar side (i.e. radio and satellite) and automatic tuning of its transmission algorithms.
- Definition of TCP spoofing mechanisms, needed to process the incoming TCP packets.
- Mapping TCP connections onto network layer QoS classes (depending on the QoS paradigm used, e.g. DiffServ and IntServ), as shown in Figure 10.4.
- Managing buffers, implemented at the transport layer, in order to prevent overflow events when congestion situations or link disruptions are likely to occur.
- Managing security issues that can be hardly addressed in the case of TCP PEPs.
- Implementing freezing schemes to suspend the data communication, when operative conditions do not allow timely recovery operations.

These issues, which are only partially connected to QoS management, are not covered in this book and are left to the reader for further investigation and study.

References

[Adami01] D. Adami, M. Marchese, L. S. Ronga, "TCP/IP based Multimedia Applications and Services over Satellite Links: Experience of an ASI/CNIT Project", *IEEE Pers. Comm.*, vol. 8, no. 3, June 2001, pp. 20–27.

[Adami02] D. Adami, M. Marchese, L. S. Ronga, "An Applied Research Study for the Provision of a QoS-Oriented Environment for Voice and Video Services over Satellite Networks", *Com. Comm. Journal*, Special Issue on Advances in Performance Evaluation of Computer and Telecommunications Networking, vol. 25, no. 11–12, July 2002, pp. 1113–1124.

[AGAVE] http://www.ist-agave.org/

[Allan] D. Allan, N. Bragg, A. McGuire, A. Reid, "Ethernet as Carrier Transport Infrastructure", *IEEE Comm. Mag.*, vol. 44, no. 2, February 2006, pp. 134–140.

[ARTES] ARTES 3, "The ESA Multimedia Initiative", Home Page, http://www.estec.esa.nl/artes3.

[ATM-Forum96] ATM Forum. "Traffic Management Specification Version 4.0". AF-TM-0056.000, April 1996.

[ATMSignalling] H. Brandt, *ATM*, John Wiley & Sons, Chichester, England, 2001.

[Awduche99] D. O. Awduche, "MPLS and Traffic Engineering in IP Networks," *IEEE Comm. Mag.*, vol. 37, no. 12, Dec. 1999, pp. 42–47.

[Barakat00] C. Barakat, E. Altman, W. Dabbous, "On TCP Performance in a Heterogeneous Network: A Survey", *IEEE Comm. Mag.*, vol. 38, no. 1, January 2000, pp. 40–46.

[Bertsekas] D. Bertsekas, R. Gallager, *Data Networks*, Prentice Hall, Englewood Cliffs, NJ, 1987.

[Bertsekas01] D. Bertsekas, *Dynamic Programming and Optimal Control*, 2nd Ed., Athena Scientific, Belmont, MA, 2001.

[Bharadwaj01] V. G. Bharadwaj, J. S. Baras, N. P. Butts, "An Architecture for Internet Service via Broadband Satellite Networks", *Int. J. Sat. Comm.*, Special Issue on IP, vol. 19, no. 1, January–February 2001, pp. 29–50.

[Bisio] I. Bisio, M. Marchese, "Analytical Expression and Performance Evaluation of TCP Packet Loss Probability Over Geostationary Satellite", *IEEE Comm. Letter*, vol. 8, no. 4, April 2004, pp. 232–234.

[Bisio06] I. Bisio, M. Marchese, "QoS-Constrained MOP-based Bandwidth Allocation over Space Networks", In *Proc. Global Communications Conference 2006 (GLOBECOM06)*, San Francisco, CA, November–December 2006.

[BisioMarchese06] I. Bisio, M. Marchese, "Minimum Distance Bandwidth Allocation over Space Communications", *IEEE Comm. Letter*, vol. 11, no. 1, January 2007, pp. 19–21.

[Bocci03] M. Bocci, J. Guillet, "ATM in MPLS-based Converged Core Data Networks", *IEEE Comm. Mag.*, vol. 41, no. 1, January 2003, pp. 139–145.

[Bolla] R. Bolla, F. Davoli, M. Marchese, "Adaptive Bandwidth Allocation Methods in the Satellite Environment", In *Proc. Conference on Computer Communications (ICC)*, Helsinki, Finland, June 2001, pp. 3183–3190.

[BollaDavoliMarchese] R. Bolla, F. Davoli, M. Marchese, "Bandwidth Allocation and Admission Control in ATM Networks with Service Separation",", *IEEE Comm. Mag.*, Special Issue on Bandwidth Allocation in ATM Networks, vol. 35, no. 5, May 1997, pp. 130–137.

[BollaMarcheseZap] R. Bolla, M. Marchese, S. Zappatore, "A Congestion Control Scheme for Multimedia Traffic in Packet Switching 'Best-Effort' Networks", *Proc. ECMAST'97*, May 1997, Milan, Italy, *Lecture Notes in Computer Science*, pp. 523–536.

[Boucadir05] M. Boucadair, "QoS-Enhanced Border Gateway Protocol", <draft-boucadair-qos-bgp-.spec-01.txt>, IETF Internet Draft, July 2005.

[BriscoeCAC] B. Briscoe, P. Eardley, D. Songhurst, F. Le Faucheur, A. Charny, J. Babiarz, K. Chan, S. Dudley, G. Karagiannis, A. Bader, L. Westberg, "An Edge-to-Edge Deployment Model for Pre-Congestion Notification: Admission Control over a DiffServ Region", <draft-briscoe-tsvwg-cl-architecture-03.txt>, IETF Internet Draft, 26 June, 2006, work in progress.

[BriscoePCN] B. Briscoe, P. Eardley, D. Songhurst, F. Le Faucheur, A. Charny, V. Liatsos, J. Babiarz, K. Chan, S. Dudley, G. Karagiannis, A. Bader, L. Westberg, "Pre-Congestion Notification marking", <draft-briscoe-tsvwg-cl-phb-03.txt>, IETF Internet Draft, 10 October, 2006.

[Brooks99] D. E. Brooks, C. Buffinton, D. R. Beering, A. W. William, D. Ivancic, M. Zernic, D. J. Hoder, "ACTS 118x Final Report, High Speed TCP Interoperability Testing", NASA/TM – 1999-209272, July 1999.

[Byungsuk02] Byungsuk Kim, Isil Sebüktekin, "An Integrated IP QoS Architecture – Performance", *IEEE Military Communications Conference – MILCOM 2002*, Anaheim, CA, USA, October 2002, pp. 1189–1193.

[Cassandras03] C. G. Cassandras, G. Sun, C. G. Panayiotou, Y. Wardi, "Perturbation Analysis and Control of Two-Class Stochastic Fluid Models for Communication Networks", *IEEE Trans. Automat. Contr.*, vol. 48, no. 5, May 2003, pp. 23–32.

[CCSDS] Consultative Committee for Space Data Systems (CCSDS), *Space Communications Protocol Specification-Transport Protocol*, CCSDS 714.0-B-1, Blue Book, May 1999.

[Celandroni03] N. Celandroni, F. Davoli, E. Ferro, "Static and Dynamic Resource Allocation in a Multiservice Satellite Network with Fading", *Int. J. Sat. Comm.*, vol. 21, no. 4–5, July–October 2003, pp. 469–488.

[Chang97] H. Chang, C. Tait, N. Cohen, M. Shapiro, S. Mastrianni, R. Floyd, B. Housel, D. Lindquist, "Web Browsing in a Wireless Environment: Disconnected and Asynchronous Operation in ARTour Web Express", *Proc. MobiCom'97*, Budapest, Hungary, September 1997, pp. 260–269.

[ChaoGuo02] H. J. Chao, X. Guo, *Quality of Service Control in High-Speed Networks*, John Wiley & Sons, Chichester, England, 2002.

[Ciscodocumentation] http://www.cisco.com/public/support/tac/documentation.html.

[Combes04] S. Combes, L. Goegebeur, N. Sanier, M. Fitch, G. Hernandez, A. Iuoras, S. Pirio, "Integrated Resources and QoS Management in DVB-RCS Networks", *AIAA International Communications Satellite Systems Conference, ISSCS04*, May 2004, Monterey, CA, USA.

[Comer] D. E. Comer, *Internetworking with TCP/IP*, Volume I: *Principles, Protocols, and Architecture*, Prentice Hall International Editions, Englewood Cliffs, NJ, 1991.

[Cortese03] G. Cortese, R. Fiutem, P. Cremonese, S. D'Antonio, M. Esposito, S. P. Romano, A. Diaconescu, "CADENUS: Creation and Deployment of End User Services in Premium IP Networks", *IEEE Comm. Mag.*, vol. 41, no. 1, January 2003, pp. 54–60.

[DoD] DoD GIG. Proposed DSCP Strawman for Phase I. Version 10, DoD GIG QoS/CoS Working Group, 8 November 2003.

[Elangovan] A. Elangovan, "Efficient Multicasting and Broadcasting in Layer 2 Provider Backbone Networks", *IEEE Comm. Mag.*, vol. 43, no. 11, November 2005, pp. 166–170.

[Engel03] T. Engel, H. Granzer, B. F. Koch, M. Winter, P. Sampatakos, I. S. Venieris, H. Husmann, F. Ricciato, S. Salsano, "AQUILA: Adaptative Resource Control for QoS Using an IP-based Layered Architecture", *IEEE Comm. Mag.*, vol. 41, no. 1, January 2003, pp. 46–53.

[Ephremides01] A. Ephremides, Special Issue on IP, Guest Editor, *International Journal of Satellite Communications*, vol. 19, no. 1, January–February 2001, pp. 1–2.

[EthernetTransport] S. S. Gorshe, G.Parsons, M. Truskowski and I. Busi, "Ethernet Transport over Public Wide Area Networks", Guest Editorial, *IEEE Comm. Mag.*, vol. 43, no. 11, November 2005, pp. 134–135.

[ETSI-ETR003] ETSI. Network Aspects (NA); General Aspects of Quality of Service (QoS) and Network Performance (NP). ETSI Technical Report, ETR 003, Second Edition, October 1994.

[ETSI-TR101] ETSI. Satellite Earth Stations and Systems (SES). Broadband Satellite Multimedia, IP over Satellite. ETSI Technical Report, TR 101 985, V1.1.2, November 2002.

[ETSI-TR102] ETSI. Satellite Earth Stations and Systems (SES). Broadband Satellite Multimedia IP. IP Interworking over Satellite; Performance, Availability and Quality of Service. ETSI Technical Report, TR 102 157 V1.0.0, March 2003.

[ETSI-TS102] ETSI. Satellite Earth Stations and Systems (SES). Broadband Satellite Multimedia. Services and Architectures; BSM Traffic Classes. ETSI Technical Specification, TS 102 295 V1.1.1, February 2004.

[ETSI-TS-102462] ETSI. Satellite Earth Stations and Systems (SES), Broadband Satellite Multimedia (BSM) Services and Architectures, QoS Functional Architecture, TS 102 462 V0.4.2, Janaury 2006.

[ETSI-TS-102463] ETSI. Satellite Earth Stations and Systems (SES), Broadband Satellite Multimedia (BSM) Services and Architectures, Interworking with RSVP-based QoS (IntServ), TS 102 463 V0.4.2, October 2006.

[ETSI-TS-102464] ETSI. Satellite Earth Stations and Systems (SES), Broadband Satellite Multimedia (BSM) Services and Architectures, Interworking with DiffServ QoS, TS 102 464 V0.4.1, September 2006.

[Fineberg02] V. Fineberg, "A Practical Architecture for Implementing End-to-End QoS in an IP Network", *IEEE Comm. Mag.*, vol. 40, no. 1, January 2002, pp. 122–130.

[Floyd93] S. Floyd, V. Jacobson, "Random Early Detection gateways for Congestion Avoidance", *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, August 1993, pp. 397–413.

[Forouzan] B. A. Forouzan, "TCP/IP Protocol Suite," Third Edition, Mc Graw Hill, Boston, USA, 2006.

[Ghani99] N. Ghani, S. Dixit, "TCP/IP Enhancement for Satellite Networks", *IEEE Comm. Mag.*, vol. 37, no. 7, July 1999, pp. 64–72.

[GII] R. H. Brown, L. Irving, A. Prabhakar, S. Katzen, "The Global Information Infrastructure: Agenda for Cooperation", http://www.ntia.doc.gov/reports/giiagend.html.

[Giordano03] S. Giordano, S. Salsano, S. Van den Berghe, G. Ventre, D. Giannakopoulos, "Advanced QoS Provisioning in IP Networks: The European Premium IP Projects", *IEEE Comm. Mag.*, vol. 41, no. 1, January 2003, pp. 30–37.

[Girard] A. Girard, *Routing and Dimensioning in Circuit-Switched Networks*, Addison-Wesley, Reading, MA, 1990.

[Goyal01] R. Goyal, R. Jain, M. Goyal, S. Fahmi, B. Vandalore, S. Kota, N. Butts, T. vonDeak, "Buffer Management and Rate Guarantees for TCP over Satellite-ATM Networks", *Int. J. Sat. Comm.*, Special Issue on IP, vol. 19, no. 1, January–February 2001, pp. 93–110.

[Gozdecki03] J. Gozdecki, A. Jajszczyk, R. Stankiewicz, "Quality of Service Terminology in IP Networks", *IEEE Comm. Mag.*, vol. 41, no. 3, March 2003, pp. 153–159.

[Guerin91] R. Guérin, H. Ahmadi, M. Naghshineh, "Equivalent Capacity and Its Application to Bandwidth Allocation in High-Speed Networks", *IEEE J. Select. Areas Commun.*, vol. 9, no. 7, September 1991, pp. 968–981.

[Handel91] R. Händel, M. N. Huber, "Integrated Broadband Networks", Addison-Wesley, Wokingham, England, 1991.

[Hardy01] W. C. Hardy, *QoS Measurement and Evaluation of Telecommunications Quality of Service*, John Wiley & Sons, Chichester, England, 2001.

[Henderson99] T. R. Henderson, R. H. Katz, "Transport Protocols for Internet-Compatible Satellite Networks", *IEEE J. Select. Areas Commun*., vol. 17, no. 2, February 1999, pp. 326–344.

[Howarth06] M. P. Hoarth, M. Boucadair, P. Flegkas, N. Wang, G. Pavlou, P. Morand, T. Coadic, D. Griffin, A. Asgari, P. Georgatsos, "End-to-End Quality of Service Provisioning Through Inter-provider Traffic Engineering", *Com. Comm*., vol. 29, no. 6, March 2006, pp. 683–702.

[ID-Chakravorty] S. Chakravorty, "IPv6 Label Switching Architecture", <draft-chakravorty-6lsa-02>, IETF Internet Draft, October 2006.

[ID-Polk01] J. M. Polk, "Multi-Level Precedence and Preemption over IP", IETF Internet Draft, February 2001.

[ID-Vasseu03] R. Zhang, J. Vasseu, "MPLS Inter-AS Traffic Engineering requirements", <draft-ietf-tewg-interas-mpls-te-req-03.txt>, IETF Internet Draft, December 2003.

[IEEE802.1ah] IEEE 802.1 Working Group, Active Task Group Interworking, 802. 1ah-Provider Backbone Bridges, http://www.ieee802.org/1/pages/802.1ah.html.

[IEEE802.1p] IEEE 802.1 Working Group. Traffic Class Expediting and Dynamic Multicast Filtering published in 802.1D. IEEE Standards for MAC bridges. IEEE 802.1p, 1998, http://www.ieee802.org/1/pages/802.1D.html, last access June 2004.

[IEEE802.1Q] IEEE 802.1 Working Group. Virtual LANs. IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks. IEEE 802.1Q, 2003, http://www.ieee802.org/1/pages/802.1Q.html, last access June 2004.

[Intermon] http://www.ist-intermon.org/

[IPJ-June05] T. Suthar, "IPv6 – A Service Provider View in Advancing MPLS Networks", *The Internet Protocol Journal*, vol. 8, no. 2, June 2005, pp. 2–12.

[ISO94] Information Technology – Open Systems Interconnection – Basic Reference Model: The Basic Model, ISO/IEC 7498-1, International Organization for Standardization, 1994.

[ITU-TE.800] ITU-T Recommendation. Terms and Definitions Related to Quality of Service and Network Performance Including Dependability. ITU-T Recommendation E.800, August 1994.

[ITU-T-G.707] ITU-T Recommendation. Network Node Interface for the Synchronous Digital Hierarchy (SDH). ITU-T Recommendation G.707, March 1996.

[ITU-T-H.323] ITU-T Recommendation. Packet-based Multimedia Communications Systems. ITU-T Recommendation H.323, June 2006.

[ITU-T-I.113] ITU-T Recommendation. Vocabulary of Terms for Broadband Aspects of ISDN. ITU-T Recommendation I.113, June 1997.

[ITU-T-P.59] ITU-T, Artificial Conversational Speech, ITU-T Recommendation P.59, March 1993.

[ITU-T-Y.1241] ITU-T Recommendation. Support of IP based Services Using IP Transfer Capabilities. ITU-T Recommendation Y.1241, March 2001.

[ITU-T-Y.1540] ITU-T Recommendation. IP Packet Transfer and Availability Performance Parameters. ITU-T Recommendation Y.1540, November 2002.

[ITU-T-Y.1541] ITU-T Recommendation. Network Performance Objectives for IP-Based Services. ITU-T Recommendation Y.1541, February 2003.

[Ivancic99] W. D. Ivancic, D. Brooks, B. Frantz, D. Hoder, D. Shell, D. Beering, "NASA's Broadband Satellite Networking Research", *IEEE Comm. Mag*., vol. 37, no. 7, July 1999, pp. 40–47.

[Jia01] Y. Jia, M. Chen, "A New Architecture of Providing End-to-End Quality-of-Service for Differentiated Services Network", *IEEE Military Communications Conference-MILCOM 2001*, October 2001, pp. 1451–1456.

[Kingston00] J. L. Kingston, "Dynamic Precedence for Military IP Networks", *IEEE Military Communications Conference–MILCOM 2000*, Los Angeles, CA, USA, October 2000, pp. 475–479.

[Kleinrock] L. Kleinrock, *Queueing Systems*, Volume I: *Theory*Wiley, New York, 1975.

[Kota03] S. Kota, M. Marchese, "Quality of Service for Satellite IP Networks: A Survey", *Int. J. Sat. Comm*., vol. 21, no. 4–5, July–October 2003, pp. 303–349.

[Kruse01] H. Kruse, M. Allman, J. Griner, D. Tran, "Experimentation and Modelling of HTTP over Satellite Channels", *Int. J. Sat. Comm*., Special Issue on IP, vol. 19, no. 1, January–February 2001, pp. 51–69.

[Kushner] H. J. Kushner, G. G. Yin, *Stochastic Approximation Algorithms and Applications*, Springer-Verlag, New York, NY, 1997.

[Lakshman97] T. V. Lakshman, U. Madhow, "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss", *IEEE/ACM Trans on Networking*, vol. 5, no. 3, June 1997, pp. 336–350.

[Liljeberg96] M. Liljeberg, H. Helin, M. Kojo, K. Raatikainen, "Mowgli WWW Software: Improved Usability of WWW in Mobile WAN Environments", *IEEE Global Internet 1996 Conference*, London, UK, November 1996.

[Liu04] N. Liu, J. Baras, "Modeling Multi-Dimensional QoS: Some Fundamental Constraints", *Int. J. Comm. Sys*., vol. 17, no. 3, April 2004, pp. 193–215.

[Lorenz00] P. Lorenz, "Quality of Service and New Architectures for Future Telecommunications Networks", *IEEE Military Communications Conference – MILCOM 2000*, Los Angeles, CA, USA, October 2000, pp. 695–698.

[Lutz99] E. Lutz, H. Bischl, J. Bostic, C. Delucchi, H. Ernst, M. Holtzbock, A. Jahn, M. Werner, "ATM-based Multimedia Communication via Satellite", *Europ. Trans. on Telecomm*., vol. 10, no. 6, November–December 1999, pp. 623–636.

[Maral] G. Maral, M. Bousquet, *Satellite Communications Systems – Systems, Techniques and Technology*, 3rd Ed., John Wiley & Sons, Chichester, 1998.

[Marchese01] M. Marchese, "TCP Modifications over Satellite Channels: Study and Performance Evaluation", *Int. J. Sat. Comm*., Special Issue on IP, vol. 19, no. 1, January–February 2001, pp. 93–110.

[Marchese05] I. Bisio, M. Marchese, "E-CAP-ABASC versus CAP-ABASC: Comparison of Two Resource Allocation Strategies in Satellite Environment", *Int. J. Space Comm*., vol. 19, no. 3–4, December 2005, pp. 171–182.

[Marchese06] M. Marchese, M. Mongelli, "On-line Bandwidth Control for Quality of Service Mapping over Satellite Independent Service Access Points", *Com. Net*., vol. 50, no. 12, August 2006, pp. 2089–2111.

[MarcheseCC01] M. Marchese "Performance Analysis of the TCP Behavior in a GEO Satellite Environment", *Com. Comm. J.*, Special Issue on the Performance Evaluation of Telecommunication Systems: Models, Issues and Applications, vol. 24, no. 9, May 2001, pp. 877–888.

[MarcheseICC01] M. Marchese, "Proposal of a Modified Version of TCP Adapted to Large Delay Satellite Channels", *IEEE International Conference on Communications (ICC 2001)*, Helsinki, Finland, June 2001.

[MarcheseIETE] M. Marchese, "Modifications of the Slow Start Algorithm to Improve TCP Performance Over Large Delay Satellite Channels", *IETE J. Res.*, Special Issue on Protocols for Resource, Link and Mobility Management for Wireless and Satellite Communication Networks, vol. 52, no. 2–3, March–June 2006, pp. 121–137.

[McDysan99] D. McDysan, D. Spohn, *ATM*, McGraw-Hill, New York, USA, 1999.

[Meddeb] A. Meddeb, "Why Ethernet WAN Transport?", *IEEE Comm. Mag.*, vol. 43, no. 11, November 2005, pp.136–141.

[Mellia02] M. Mellia, A. Carpani, R. Lo Cigno, "Measuring IP and TCP Behavior on an Edge Node", *IEEE Global Communication Conference Globecom 2002*, Taipei, Taiwan, November 2002, pp. 2540–2544.

[MESCAL] http://www.mescal.org/

[Miettinen] K. Miettinen, *Nonlinear Multi-Objective Optimization*, Kluwer Academic Publishers, Boston, 1999.

[Montanez02] M. Montanez, "Deploying QoS in the Enterprise", *Packet – Cisco System Users Magazine*, vol. 14, no. 4, 4th quarter 2002, pp. 30–34.

[MOS] Mean Opinion Score, Networking Definitions, http://searchnetworking.techtarget.Com/sDefinition/0,,sid7_gci786677,00.html.

[Muthoo] A. Muthoo, *Bargaining Theory with Applications*, Cambridge University Press, Cambridge, UK, 1999.

[Mykoniati03] E. Mykoniati, C. Charalampous, P. Georgatsos, T. Damilatis, D. Godersi, P. Trimintzios, G. Pavlou, D. Griffin, "Admission Control for Providing QoS in DiffServ IP Networks: the TEQUILA Approach", *IEEE Comm. Mag*., vol. 41, no. 1, January 2003, pp. 38–44.

[Nash] J. Nash, "The Bargaining Problem", *Econometrica*, vol. 18, 1950, pp. 155–162.

[Onvural94] R. Onvural, *Asynchronous Transfer Mode Networks Performance Issues*, Artech House, Boston, USA, 1994.

[OPWA95] S. Shenker, L. Breslau, "Two Issues in Reservation Establishment", *ACM SIGCOMM '95*, Cambridge, MA, August 1995, pp. 14–26.

[OSPF-TE02] D. Katz, D. Yeung, K. Kompella, "Traffic Engineering Extensions to OSPF Version 2", IETF Internet Draft, <draft-katz-yeung-ospf-traffic-09.txt>, October 2002.

[Partridge97] C. Partridge, T. J. Shepard, "TCP/IP Performance over Satellite Links", *IEEE Network*, vol. 11, no. 5, September–October 1997, pp. 44–49.

[Pattavina98] A. Pattavina, *Switching Theory*, John Wiley & Sons, Chichester, England, 1998.

[Pongpaibool04] P. Pongpaibool, H. S. Kim, "Providing End-to-End Service Level Agreements across Multiple ISP Networks", *Comp. Net*., vol. 46, no. 1, 16 September 2004, pp. 3–18.

[RFC1323] V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", IETF RFC 1323, May 1992.

[RFC2018] V. Jacobson, R. Braden, "TCP Extensions for Long-Delay Paths", IETF RFC 2018, October 1988.

[RFC2205] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, "Resource ReserVation Protocol (RSVP) – Version 1 Functional Specification", IETF RFC 2205, September 1997.

[RFC2210] J. Wroclawski, "The Use of the Resource Reservation Protocol with the Integrated Service", IETF RFC 2210, September 1997.

[RFC2211] J. Wroclawski, "Specification of the Controlled-Load Network Element Service", IETF RFC 2211, September 1997.

[RFC2212] S. Shenker, C. Partridge, R. Guerin, "Specification of Guaranteed Quality of Service", IETF RFC 2212, September 1997.

[RFC2215] S. Shenker, J. Wroclawski, "General Characterization Parameters for Integrated Service Network Elements", IETF RFC 2215, September 1997.

[RFC2386] E. Crawley, R. Nair, B. Jajagopalan, H. Sandick, "A Framework for QoS-based routing in the Internet", IETF RFC 2386, August 1998.

[RFC2414] M. Allman, S. Floyd, C. Partridge, "Increasing TCP's Initial Window", IETF RFC 2414, September 1998.

[RFC2415] K. Poduri, K. Nichols, "Simulation Studies of Increased Initial TCP Window Size", IETF RFC 2415, September 1998.

[RFC2474] K. Nichols, S. Blake, F. Baker, D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", IETF RFC 2474, December 1998.

[RFC2475] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An Architecture for Differentiated Services", IETF RFC 2475, December 1998.

[RFC2488] M. Allman, D. Glover, L. Sanchez, "Enhancing TCP over Satellite Channels using Standard Mechanism", IETF RFC 2488, January 1999.

[RFC2581] M. Allman, V. Paxson, W. S. Stevens, "TCP Congestion Control", IETF RFC 2581, April 1999.

[RFC2597] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, "Assured Forwarding PHB Group", IETF RFC 2597, June 1999.

[RFC2746] A. Terzis, J. Krawczyk, J. Wroclawski, L. Zhang, "RSVP Operation Over IP Tunnels", IEFT RFC 2746, January 2000.

[RFC2747] F. Baker, B. Lindell, M. Talwar, "RSVP Cryptographic Authentication", IETF RFC 2747, January 2000.
[RFC2760] M. Allman, S. Dawkinks, D. Glover, J. Griner, T. Henderson, J. Heidemann, S. Ostermann, K. Scott, J. Semke, J. Touch, D. Tran, "Ongoing TCP Research Related to Satellites", IETF RFC 2760, February 2000.
[RFC2996] Y. Bernet, "Format of the RSVP DCLASS Object", IETF RFC 2996, November 2000.
[RFC2998] Y. Bernet, P. Ford, R. Yavatkar, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, E. Felstaine, "A Framework for Integrated Services Operation over Diffserv Networks", IETF RFC 2998, November 2000.
[RFC3135] J. Border, M. Kojo, J. Griner, G. Montenegro, Z. Shelby, "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations", IETF RFC 3135, June 2001.
[RFC3168] K. Ramakrishnan, S. Floyd, D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", IETF RFC 3168, September 2001.
[RFC3206] R. Gellens, "The SYS and AUTH POP Response Codes", IETF RFC 3206, February 2002.
[RFC3209] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", IETF RFC 3209, December 2001.
[RFC3246] B. Davie, A. Charny, J. C. R. Bennet, K. Benson, J. Y. Le Boudec, W. Courtney, S. Davari, V. Firoiu, D. Stiliadis, "An Expedited Forwarding PHB (Per-Hop Behavior)", IETF RFC 3246, March 2002.
[RFC3261] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, "SIP: Session Initiation Protocol", IETF RFC 3261, June 2002.
[RFC3312] G. Camarillo, W. Marshall, J. Rosenberg, "Integration of Resource Management and Session Initiation Protocol (SIP)", IETC RFC 3312, October 2002.
[RFC3726] M. Brunner, "Requirements for Signalling Protocols", IETF RFC 3726, April 2004.
[RFC4023] T. Worster, Y. Rekhter, E. C. Rosen, "Encapsulating MPLS in IP or Generic Routing Encapsulation (GRE)", IETF RFC 4023, March 2005.
[RFC4080] R. Hancock, G. Karagiannis, J. Loughney, S. Van den Bosch, "Next Steps in Signaling (NSIS): Framework", IETF RFC 4080, June 2005.
[RFC4094] J. Maner, X. Fu, "Analysis of Existing Quality of Service Signalling Protocols", IETF RFC 4094, May 2005.
[RFC4271] Y. Rekhter, T. Li, S. Hares, "A Border Gateway Protocol 4 (BGP–4)", IETF RFC 4271, January 2006.
[RFC4411] J. Polk, "Extending the Session Initiation Protocol (SIP) Reason Header for Preemption Events", IETF RFC 4411, February 2006.
[RFC4412] H. Schulzrinne, J. Polk, "Communications Resource Priority for the Session Initiation Protocol (SIP)", IETF RFC 4412, February 2006.
[RFC4594] J. Babiarz, K. Chan, F. Baker, "Configuration Guidelines for DiffServ Service Classes", IETF RFC 4594, August 2006.
[RFC4655] A. Farrel, J. P. Vasseur, J. Ash, "A Path Computation Element (PCE)-Based Architecture", IETF RFC 4655, August 2006.
[RFC793] Information Sciences Institute, University of Southern California, "Transmission Control Protocol – Darpa Internet Program – Protocol Specification", IETF RFC 793, September 1981.
[RFC904] D. L. Mills. "Exterior Gateway Protocol Formal Specifications", IETF RFC 904, April 1984.
[Ross] K. W. Ross, "Multiservice Loss Models for Broadband Telecommunication Networks", Springer-Verlag, London, 1995.
[Sarangan2006] V. Sarangan, J.-C. Chen, "Comparative Study of Protocols for Dynamic Service Negotiation in the Next-Generation Internet", *IEEE Comm. Mag*., vol. 44, no. 3, March 2006, pp. 151–156.
[Schmitt03] J. Schmitt, "Translation of Specification Units between IP and ATM Quality of Service Declarations", *Int. J. Comm. Sys*., vol. 16, no. 4, April 2003, pp. 291–310.

[Schwartz88] M. Schwartz, *Telecommunications Networks*, Addison-Wesley, Reading, Massachusetts, USA, 1988.

[Soldatos05] J. Soldatos, E. Vayias, G. Kormentzas, "On the building blocks of quality of service in heterogeneous IP networks", *IEEE Communications Surveys & Tutorials,* vol. 7, no. 1 First Quarter 2005, pp. 70–89.

[Sorteberg05] I. Sorteberg, O. Kure, "The Use of Service Level Agreements in Tactical Military Coalition Force Networks", *IEEE Comm. Mag*., vol. 43, no. 11, November 2005, pp. 107–114.

[Tacoms05] Tacoms Post 2000 Project (TACOMS), TACOMS STANAG 4637, "TACOMS HEAD STANAG" (draft ed. 1), NATO Steering Committee, http://www.tacomspost2000.org, NSA, 2005.

[TE-QoS01] G. Ash, "Traffic Engineering & QoS Methods for IP-, ATM-, & TDM-Based Multiservice Networks", IETF Internet Draft <draft-ietf-tewg-qos-routing-04.txt>, Oct. 2001.

[Trimintzios03] P. Trimintzios, G. Pavlou, P. Flegkas, P. Georgatsos, A. Asgari, E. Mykoniati, "Service-Driven Traffic Engineering for Intradomain QoS Management", *IEEE Network*, vol. 17, no. 3, May–June 2003, pp. 29–36.

[Vali04] D. Vali, S. Paskalis, L. Merakos, A. Kaloxylos, "A Survey of Internet QoS Signalling", *IEEE Communications Surveys & Tutorials*, vol. 6, no. 4, Fourth Quarter 2004, pp. 32–43.

[Video-Traces] http://www-tkn.ee.tu-berlin.de/research/trace/trace.html.

[Wardi02] Y. Wardi, B. Melamed, C. G. Cassandras, C. G. Panayiotou, "Online IPA Gradient Estimators in Stochastic Continuous Fluid Models", J. Optimiz. Theory App., vol. 115, no. 2, November 2002, pp. 369–405.

[Wright01] D. J. Wright, *Voice over Packet Networks*, John Wiley & Sons, Ltd, Chichester, England, 2001.

[Yaïche] H. Yaïche, R. R. Mazumdar, C. Rosemberg, "A Game Theoretic Framework for Bandwidth Allocation and Pricing in Broadband Networks", *IEEE/ACM Trans. on Net*., vol. 8, no. 5, October 2000, pp. 667–678.

[Zhang97] Y. Zhang, D. De Lucia, B. Ryu, Son K. Dao, "Satellite Communication in the Global Internet", http://www.wins.hrl.com/people/ygz/papers/inet97/index.html.

Index