Darmstadt University of Technology

Quality of Service - An Overview

Jens Schmitt, Lars Wolf

April 1997

Technical Report TR-KOM-1997-01

Industrial Process and System Communications (KOM)

Department of Electrical Engineering & Information Technology
Merckstraße 25 • D-64283 Darmstadt • Germany

Phone: +49 6151 166150
Fax: +49 6151 166152
Email: info@KOM.tu-darmstadt.de
URL: http://www.kom.e-technik.tu-darmstadt.de/

Quality of Service – An Overview

Jens Schmitt, Lars Wolf

Industrial Process and System Communications
Department of Electrical Engineering and Information Technology
Technical University of Darmstadt
Merckstr. 25 • D-64283 Darmstadt • Germany
{Jens.Schmitt, Lars.Wolf}@kom.th-darmstadt.de

Abstract:

In the last few years new types of distributed computer-based applications have appeared which have differing requirements in comparison to previously known applications. The most prominent of these applications types are multimedia applications. These deal with discrete media data (such as text and graphics) *and* continuous-media data (like audio and video) and request, due to the time-criticality of the audiovisual data, that retrieval, processing, transfer, and presentation of that data takes timing aspects into account. This means that all components involved in such operations must be capable of handling a well-defined quality of service (QoS). The most important QoS parameters are used to request (1) the required capacities of the involved resources, (2) compliance to end-to-end delay and jitter as timing restrictions, and (3) restriction of the loss characteristics.

Work on the notion of and mechanisms to provide QoS in general purpose computer systems has started around the late 1980 / early 1990 (mostly, with a few earlier exceptions) within the research community. Since that, interest and research effort has increased much and (partially) moved into development.

While the advent of QoS support in products has not been very broad by now, it can be expected that this will change soon because of gained experiences with such technology and also increased public interest in distributed multimedia applications.

The purpose of this paper is to give an overview about QoS – the notion, the issues which must be addressed, and approaches for QoS provisioning such as architectures and protocols. We describe aspects which have already been tackled with, as well as recently upcoming aspects which must be addressed in the future. To fulfill its purpose as an overview, the paper considers prominent models from research existing already for some time and currently followed approaches. The most active research area with respect to QoS is perhaps the field of communication systems. QoS is also under study in other areas, e.g., disk scheduling within video-servers. However, due to its importance and the research activity, communication system QoS aspects will be the main focus of this paper.

This work is supported in part by a grant of Volkswagen-Stiftung, D-30519 Hannover, Germany.

Table of Contents

List of Figures

1 Introduction

The advances in computer technology, e.g., processing speed and storage size, provide the ability to integrate audio and video data into computer systems, and makes them look more and more like communication devices rather than the pure 'number crunchers' of the former days. This integration allows for new application classes and enhanced user interfaces, for instance, video conferencing.

Human users perceive audio and video as continuously changing, therefore, these media are also referred to as *continuous media*. Because audio and video are time critical, the requirements of audiovisual data are different from data traditionally handled in computer systems. Furthermore, the processing demands of digital audio and video data are typically large, i.e., although the available capacity is sufficient it is not abundant for the integration of continuous media data.

One important new class of applications dealing with continuous media is the broad field of multimedia systems. By a multimedia system we denote the integrated manipulation (capturing, processing, communication, presentation, storage) of at least some information represented as continuous media data (such as audio and video) as well as some information encoded as discrete media data (such as text and graphics) [NS95a].

Multimedia applications must present their data with a certain quality to satisfy the user. Therefore, applications need a certain *Quality of Service (QoS)* from the system to fulfill their tasks. The required QoS depends on the used media (video, audio, etc.), the coding format used to encode the data, the application and the type of the application. For instance, the QoS of a video conference is different from that of a video retrieval application, since the dialogue-mode communication of a conference requires a short delay which is not as important for playback applications.

The notion of QoS is different for the various system layers, e.g., the description of QoS at the application layer is usually at a higher level than that at the network layer of a communication system. However, the QoS parameters, *bandwidth*, *delay*, and *loss* are present in all layers, sometimes used in conjunction with other parameters. Because of its layer-crossing character QoS is a difficult term to define and a difficult concept to realize, since it requires coordination between all system components.

Originally, the term QoS emerged in telecommunications to describe certain technical characteristics of data transmission which represented fixed values. These values could only be observed or measured but not be influenced by applications. Nowadays, applications are commonly regarded as the origin of QoS requirements. Therefore, applications must be able specify their requirements which, potentially after translation, are used to control the system parameters.

Resource management mechanisms provide the means to offer QoS to multimedia applications, e.g., so that the participants in a video conference do not experience large delays or low video frame rates during their interaction. These mechanisms administer and schedule system resources so that applications can get access to all necessary resources when needed.

In this paper we give an overview about the terms, issues and trends in the provision of QoS. The main focus is on communication system aspects and its capabilities to support QoS.

Chapter 2 gives a general introduction to the generic terms and mechanisms in the field of QoS support over all architectural layers of a system. In chapter 3 this generic discussion of QoS is substantiated by regarding several proposed models respectively architectures for QoS

support. Chapter 4 then concentrates on QoS provision on the communication system level by discussing concrete examples of signaling and reservation protocols which support QoS. In chapter 5 some more advanced topics with respect to QoS such as scaling, filtering and reservation in advance are introduced.

2 Introduction to QoS

New distributed applications expect configurable, realizable and maintainable QoS. Support for this must be provided by all resources involved in the overall tasks of the applications, i.e., those used by the applications for local processing, those to move the data to or from the transport system interface and those needed by the transport system to transfer the messages across the network.This, therefore, includes all the *system resources* through which a media stream passes which are

- bus bandwidth,
- I/O devices, including hard disks with file systems,
- network adapters and network resources to transfer packets from one node to another,
- CPUs to execute the application and the protocol software,
- buffer space to store the software and the data, e.g., communication packets, passing through the nodes.

These resources are often distinguished into *active* resources such as CPUs, buses, I/O systems, network adapters, transmission links and *passive* resources such as memory in hosts and intermediate systems like routers or switches. [CAH95], [VWHW97], [CCH94] (see Figure 1).

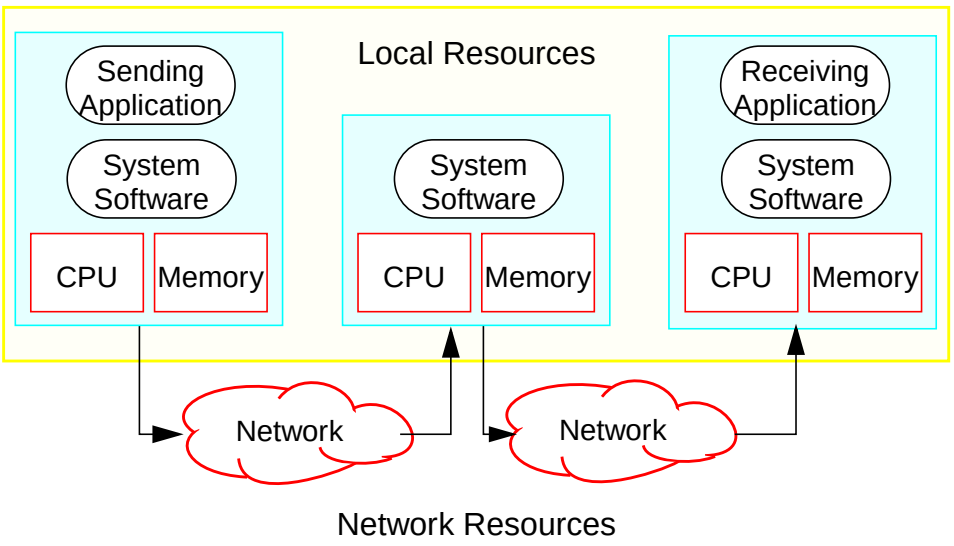


Figure 1: Resources Passed by a Multimedia Stream.

To fulfil these expectations of applications an integral and comprehensive view of QoS is inevitable, since eventually QoS has to be delivered *end-to-end*, from one application process to the other [MJV96].

These new applications, of which most fall into the category of *multimedia applications*, have very heterogeneous requirements on their deployed system components. Thus there is an urgent need of *parametrization of the services* offered by these system components as for example the operating system, the communication subsystem and the underlying network [Stil95].

What is required, is the possibility for an application to specify its requirements in a precise, but general way, such that this specification can be passed between the underlying system com-

ponents and layers, and be interpreted to map the application requirements correctly on the system resources while observing efficiency of resource utilization.

Resource management, which provides mechanisms to administer the available resources so that QoS requirements can be met, is commonly considered the only way to provide for QoS in an environment of scarce resources [ATWG90].

In order to deliver a particular level of QoS to an application, the system must not only have sufficient resources available, but these resources must be scheduled in such a way that they are available for the application when needed. While many of today's computer systems offer sufficient resources to handle, e.g., some continuous-media streams, the quantity and quality of such streams is limited since the resources are limited. This was illustrated by Anderson et al. as the "window of scarcity" [ATWG90] shown in Figure 2.

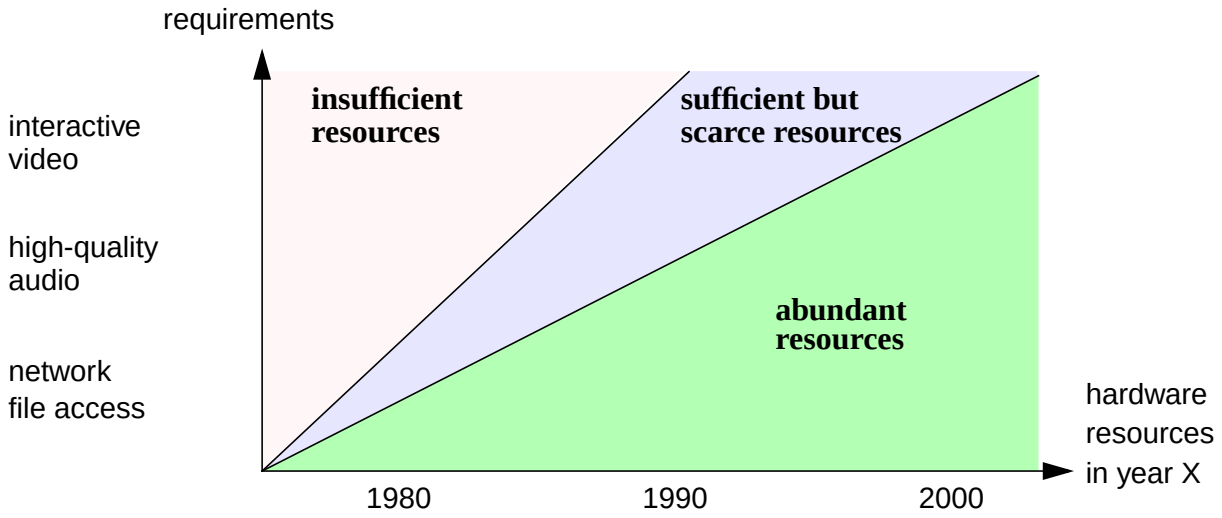


Figure 2: 'Window of Scarcity'.

At a specific time, for certain application types, the available resources are *insufficient* to provide acceptable service (left side of Figure 2). Due to ongoing improvements in technology, system resources become sufficient for new applications, however, the available resources are *scarce*, i.e., they must be administrated and scheduled carefully to offer the desired QoS (middle part of Figure 2). After further technology advances, resources are abundant with respect to a particular application, i.e., the service can be offered without specific management mechanisms (right side of Figure 2).

As the window of scarcity illustrates, at least in the near future computer systems will have only sufficient but scarce system resources available for, e.g., the processing of continuous-media streams. It can even be argued that they will never have abundant resources because the demands of applications will grow as well. Even in the future, should system resources available for processing and transmitting data be more than just sufficient, a time-critical application must be 'shielded' against non-real-time and other real-time applications so that they cannot inhibit its real-time processing. Additionally, providers of shared systems which offer services to several users simultaneously, e.g., video-on-demand servers, want as efficient resource use as possible, i.e., to serve as many clients using as few systems as possible. Thus, a means to manage the available system resources is necessary.

This is also emphasized by the fact that we still have not reached the area of abundant resources for many application types but the windows of scarcity and the claim that resource management is necessary has been made by Anderson et al. already several years ago. This also means that the area of scarce resources has a larger size than expected.

There are still different perspectives on QoS as observed by service providers and service users, whereby for the former efficiency of resource utilization is the focal point of QoS, while to the latter the completeness of parametrization of the service is most important. These different perspectives have led to different service models, which in the past did not allow for quantitative mappings between them [Stil95].

Hence a goal to be strived for must be an integral view of QoS for both, providers and users of a service. This is quite easily achieved if both components are integrated into one system as for example when the operating system offers services to the applications like process scheduling, but is very difficult when multiple parties have to cooperate as in the case of a network services provider and an application program, which naturally pursue very different objectives. While the application wants to be provided as many resources as possible for as little cost as possible, the network provider of course wants to minimize resource utilization while maximizing its profit.

Since it is perceived that these antagonistic goals cannot be resolved in a general way, mechanisms for conflict resolution in the specific case of a connection establishment are needed [NLP96] [VBDGHK94].

The architectural model to achieve this falls into several components [NS95a][CAH95]:

- QoS Specification,
- QoS Negotiation,
- QoS Translation and Mapping,
- QoS Control and Management.

While the first three of these components play their role during the establishment of a connection, the last component takes its actions during the data or media transfer.[1]

2.1 QoS Specification

What is required in the first instance is a specification of the QoS expected by the application. The QoS specification's purpose is on the one hand to enable the applications to formulate their QoS needs, while on the other hand the system components, which provide for QoS, accept the QoS specification as an order for a certain service. The QoS specification is *declarative* in nature, i.e. is given as a set of parameters. As the canonical parameter set is usually considered [NS95a] (see also Figure 3):

- *Throughput*: determined by the needed data rate of the application and also depending on the size of the data packets.
- *Delay*: distinguished into local (at the resource) and global (end-to-end) delay.
- *Jitter*: the maximum variance in the arrival of data at the destination.

1. However QoS negotiations may be repeated during the data transfer phase as so-called QoS renegotiations.

- *Loss Rate*: parametrizing respectively triggering the employed error detection and/or correction mechanisms.

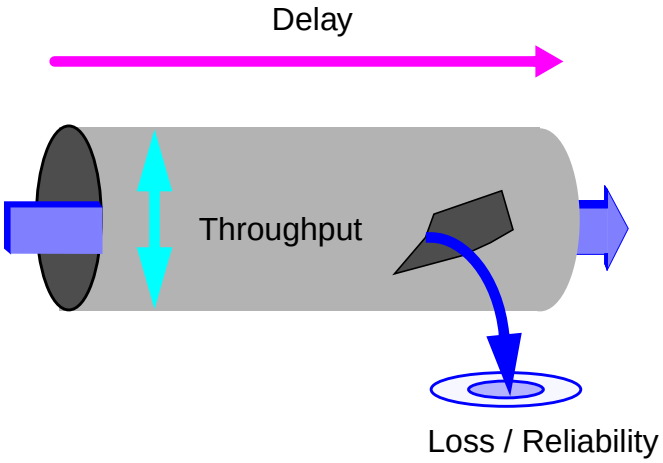


Figure 3: Generic QoS Parameters.

For the specification of requirements regarding these parameters several possibilities exist [Stil95]:

- they can be specified as a *single value* giving the desired level of this parameter,
- a *pair of values* giving the minimum acceptable and the on the average expected level of the considered service parameter or
- an *interval* defining the area between the minimally and maximally acceptable values for the parameter in consideration.

If an interval is used, the gain is to obtain a higher degree of freedom for the QoS negotiation and mapping procedures. Yet on the other hand, the accurateness of specification from the perspective of the application is decreasing.

In any case there should always exist a certain interval for each QoS parameter, in which an application is on the one hand operable with respect to the lower bound and on the other hand does not obtain better services as it is able to utilize usefully (see Figure 4).The goal for the application is to define its QoS inside this interval.

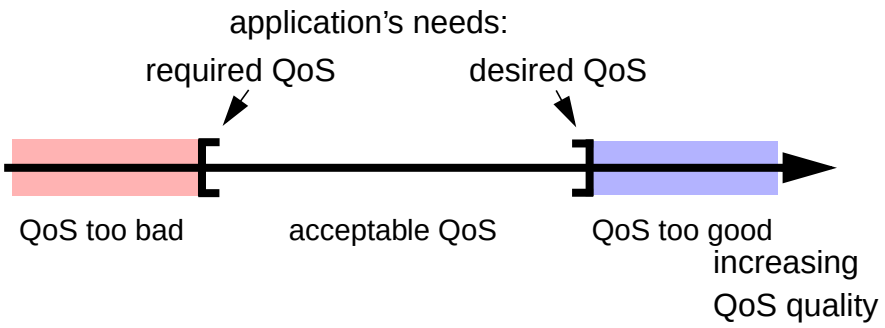


Figure 4: The Acceptable QoS Interval.

The QoS parameter specification or description is *layer-dependent* and may be qualitative or quantitative. For example delay, throughput, response time rate, data corruption, task and buffer specifications are quantitative parameters on different architectural layers whereas inter-stream synchronization, ordered delivery, error recovery, scheduling are qualitative parameters on different levels of abstractions.

Anyway it has to be observed that QoS layering is an important concept since the whole concept of QoS is too complex to be solved in a monolithic way. Therefore commonly *application-, communication subsystem- and network-QoS* are distinguished [Stil95].

Another QoS specification issue is the type of service guarantee offered by the service provider. Typical classifications distinguish between *guaranteed, statistical, predictive and best-effort QoS* [NS95a][Stil95][BCS94]. These service classes characterize the reliability of the commitment of the service provider to conform to the negotiated values of the service contract between service user and provider. While guaranteed service gives perfectly reliable guarantees on the contracted performance, the statistical service discipline provides the user only with statistical guarantees about the contracted parameter values. Even weaker guarantees are given by predictive services, which only support a certain number of qualitative levels of service commitment. The ultimatively weakest QoS support category is represented by the best-effort service discipline, which provides for no guarantees at all (and should hence better be called ‘no-effort’).

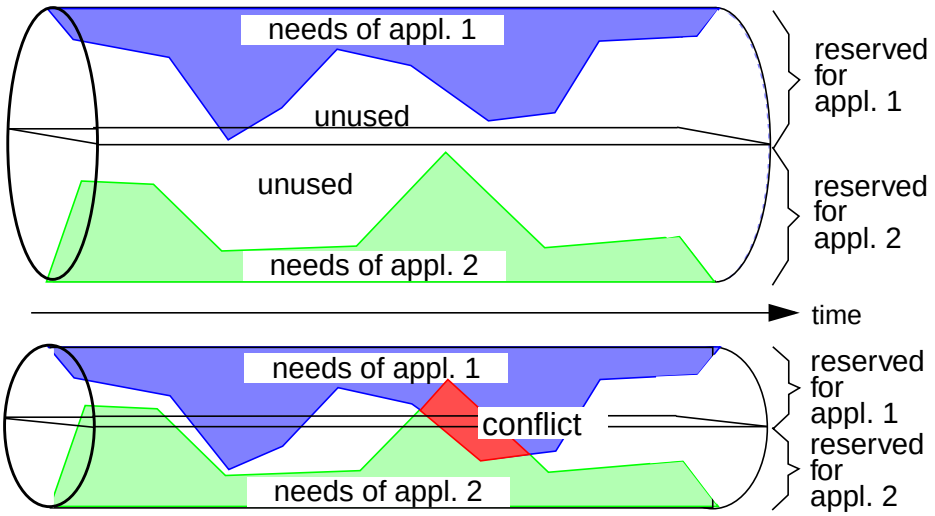


Figure 5: Deterministic Services vs. Statistical Services.

The motivation for providing more service varieties than just best-effort and deterministic services is from the service provider’s view efficiency, since statistical services allow to reserve less bandwidth than what is necessary for deterministic service disciplines (see Figure 5), while from the service user’s perspective the potentially lower costs of statistical services should be an incentive to use them.

Some QoS models like XRM [KW94], ATM [ITU95] or IntServ [BCS94] use an aggregation mechanism to summarize combinations of QoS parameters in so-called service classes arguing that otherwise the complexity in end-systems and network nodes would not be manageable. The other school of thought, as for example the Tenet approach [Ferr95], Lancaster’s

QoS-A [CCH94], the HeiTS/HeiRAT project [VWHW97] or the OMEGA end-point architecture [NS95c], leaves the multi-valued nature of QoS specifications in place in order to allow for the, in their perspective, necessary great flexibility in issuing QoS requests.

Another possible and desirable component of the QoS specification is the *QoS management policy* [CAH95], which determines what actions are to be taken in the event of a QoS violation. This encompasses at least in which cases of QoS degradation the application is to be informed, and may contain possible reactions to these events like adapting the traffic rate in case of a bandwidth problem or increasing the redundancy of data in case of a reliability problem. Figure 6 summarizes the components which are part of a QoS specification.

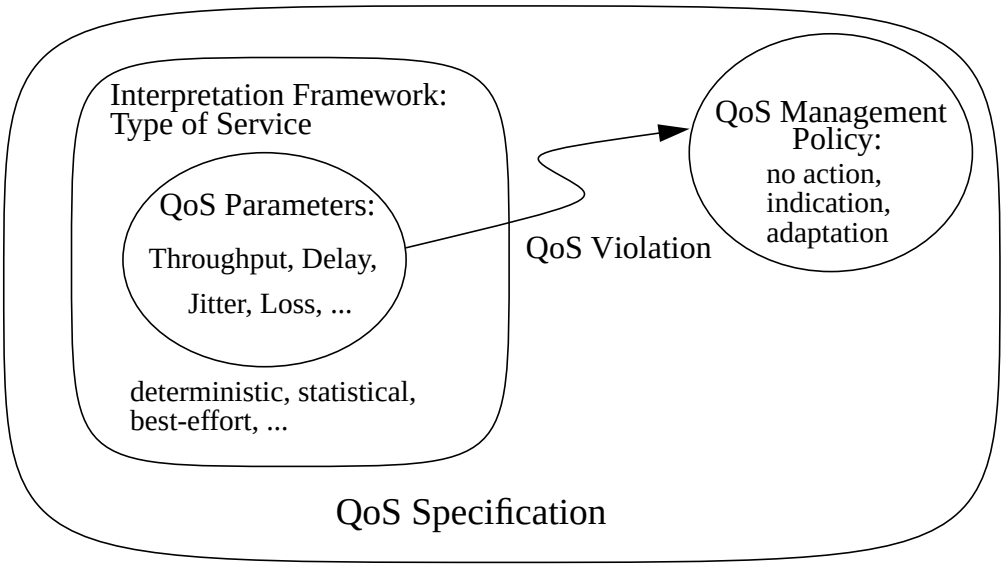


Figure 6: Components of a QoS Specification.

2.2 QoS Negotiation

Once the specification of desired QoS by the application is complete, the next step regardless of the considered abstraction level is the QoS negotiation between the entities taking part in the process of QoS provision and QoS availment. These negotiations take place either between peers, i.e. different entities on the same layer, or between adjacent layers in the system architecture.

Negotiations may be *unilateral*, which means that one entity issues a QoS request containing a certain QoS specification upon which the requested entity can only react by admitting or rejecting the QoS request, or they may have more sophisticated negotiation patterns, which have *bilateral* or *multilateral* characteristics [NS95a]. In the bilateral case the requested entity has the possibility to alter the QoS specification depending on the style used to specify the QoS parameters, i.e. whether a single value, pair of values or interval is used. The possibly altered QoS specification is then sent back to the requesting entity.

A further possibility is to iterate this behavior until agreement is achieved between the negotiating parties, however the increased delay in call establishment can be prohibitive [Stil95].

Another issue for QoS negotiation is whether it is done statically or dynamically. The options are either to have the communicating parties stick to the contract negotiated at the start of a connection or to allow for a *renegotiation* during the transmission of data.

An interesting approach to coordinate all the different negotiations taking place between peers and different layers is the QoS broker architecture [NLP96], in which the broker is a central point for all negotiations taking place (see Figure 7), thereby isolating the negotiation partners, which releases them of the burden of knowing all the details for communication between them.

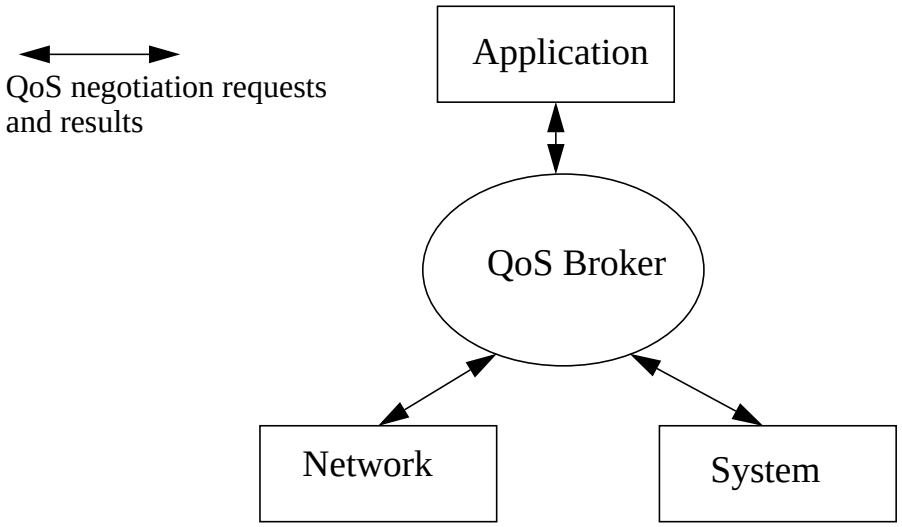


Figure 7: The QoS Broker Model for QoS Negotiation.

2.3 QoS Translation and Mapping

After the QoS level to be provided has been negotiated between application processes the contracted tuple of QoS values has to be translated into QoS description forms known to lower layers and system components, in order to initiate negotiation on these levels (see Figure 8).

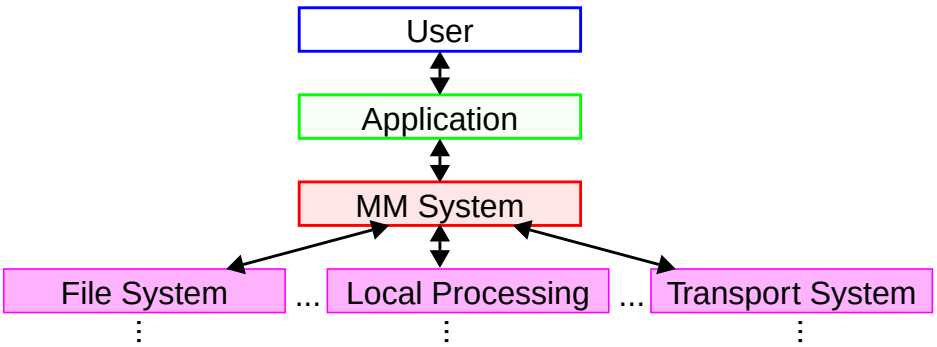


Figure 8: QoS at Various Levels with Differing Degree of Details.

For instance, the user just wants a video to be played, the transport layer is interested in

requirements such as throughput, delay and loss; the network layer is interested in these requirements as well yet at a fine granularity.

Different layers and system components have different terms of QoS specification, thus, this mapping of the parameters is no easy matter, yet a very important one, since it effects the efficiency of resource utilization to a major part. Especially in a heterogeneous environment, as for example if different network technologies lie on the path between source and destination, the mapping between the respective QoS parameters is a critical point with respect to the efficient use of resources and has so far not been resolved in a satisfying manner [Stil95].

As the process of negotiation of QoS and translation into lower layer constructs winds itself down to the lowest layer, parameters controlling the resources and thereby guaranteeing the specified and negotiated QoS levels are set.

During this process available resources are checked against requested resources and if sufficient resources exist, the request is admitted and the necessary resources are allocated.

If the resource in consideration is the network, then these tasks have to be done in a distributed fashion, which necessitates the existence of a *resource reservation protocol* as a vehicle to transport information about resource reservation requests between network nodes. However it is to be noted that the resource reservation protocol does not perform any reservations itself, but leaves this to the local resource management of the network nodes.

Resource reservation and allocation can be done in two ways [NS95a]:

- *optimistically* or
- *pessimistically.*

Pessimistic resource reservation avoids resource conflicts by making reservations for the worst case leading to guaranteed QoS, in contrast to optimistic resource reservation that reserves resources according to the average workload, which results in possible violations of the contracted QoS but delivers better resource utilization. Between these extreme cases lies a continuum of strategies that use a certain security margin above the average workload to do resource reservation, and therefore different trade-offs between statistical multiplexing and the strength of the provided QoS guarantees are being made.

2.4 QoS Control and Management

QoS control and management are two components of the overall architecture which in contrast to the other already presented components operate during the transfer of the data for which the QoS was requested. Hence two different *phases* with regard to the time of operation of QoS components can be distinguished (see Figure 9).

The distinction between control and management is with regard to the *time scale* on which they operate, while QoS control takes its action on the time scale of the data transfer, the management tasks are performed on a much slower time scale [CAH95].

The task of QoS control is to enforce the given QoS guarantees in all the components taking part in the data transfer. This includes CPU scheduling, buffer management and disk scheduling in the end systems and packet/cell scheduling and flow control in the network nodes in order to conform to delay and throughput guarantees, and furthermore error detection and/or correction mechanisms in both end systems and intermediate hosts to fulfill the reliability requirements of applications [CAH95].

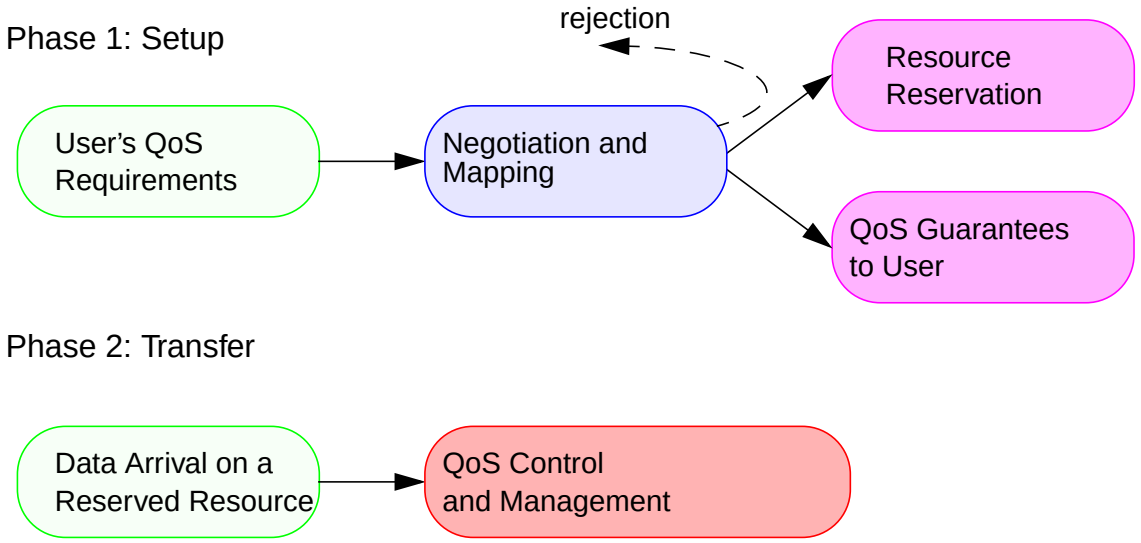


Figure 9: Different Phases in QoS Provision.

QoS management mechanisms are required to ensure that the contracted QoS is actually sustained by the provider, therefore its main task is to do monitoring [CAH95].

Another related task is that of *QoS adaptation* in case the observed QoS is worse than the originally contracted QoS. An application might have agreed to a kind of temporal QoS degradation in the service contract and must therefore be prepared to react to QoS degradation by adapting to the worse service. There are many classes of applications which have the ability to adapt to such changes in service provision, they will be happy to contract to such a kind of insecure provision of QoS if only they have to pay significantly less for this service in comparison to guaranteed service. Depending on whether the adaptation to QoS degradation takes place in the network or in the end system one speaks of *QoS filtering* respectively *QoS scaling*.

3 QoS Models and Architectures

Different QoS models and architectures have been proposed with partially different goals and scope in mind. We will review the most original and important ones to obtain an overview on different alternatives of QoS support and what seems to be common to these proposals.

We start with the *Tenet approach*, which was one of the first implementations of an architecture in support of QoS on the communication system level.

Then we consider *Lancaster's QoS architecture* which takes a broader view on QoS provision by also incorporating other system components besides the communication system.

In a similar vein the *HeiTS/HeiRAT* approach is to be seen, though it is the earlier of these two and therefore more original.

Next a model rather stemming from the telecommunications world, Columbia University's *XRM* model, is reviewed. The XRM model concentrates on architectural aspects of a QoS supporting system and pays less attention to the QoS modeling aspect in comparison to the other approaches.

In the last two sections we treat the Internet's service model *IntServ* and *ATM's service model* in more depth.

3.1 The Tenet Approach

The Tenet group from the University of California at Berkeley and the International Computer Science Institute (ICSI, also in Berkeley) designed and implemented a real-time communication services model with an emphasis on *network support for continuous media applications*, called the Tenet Approach [BFMMVZ96].

In this approach mathematically provable, but not necessarily deterministic performance guarantees, contractual relationships between client and server, general parameterized user-network interfaces with multiple traffic and QoS bounds definable over continuous ranges and large heterogeneous packet-switching networking environments are the main elements.

As the key mechanisms to achieve the realization of these elements are recognized [FBZ94]:

- *connection-oriented communication,*
- *per-connection admission control,*
- *channel rate control and*
- *priority scheduling*.

Multimedia communication is seen as a subset problem of real-time communication, furthermore it is assumed that a general architecture can solve the real-time communication problem without requiring specialized protocols for multimedia communications (like separate voice, video or movie protocols, etc.).

A realization of the Tenet approach's with respect to design compromises is in their terminology a *scheme* and the actual implementation is called a *suite*.There exist Scheme 0, 1 and 2 (numbered after their coming into existence), whereby these have a superset relation and are increasingly elaborated. Since scheme 0 was not implemented, there are only Suites number 1 and 2. For the same reason only Scheme 1 and 2 will be regarded without loss of information as they are supersets of Scheme 0 anyway.

3.1.1 Scheme 1

The central concept in Tenet's Scheme 1 is the *real-time channel* [Floy92], which is defined as a simplex unicast end-to-end connection with performance guarantees and restrictions on traffic shapes.

The reservation of resources on the route of a channel is the method to satisfy performance guarantees, whereby admission control tests are performed during the process of establishing a channel in order to control this reservation process and avoid over-reservation [FBZ94].

The Service Model

The service model of Scheme 1 is rather simple. The service interface allows for specification of traffic and performance parameters in a *contractual agreement between client (application) and server (network)*, which is negotiated unilaterally, since the network either accepts or rejects the requested connection, but does not adjust any of the specified parameters. The traffic specification consists of single values for [BFMMVZ96]:

- *minimum inter-message time,*
- *average inter-message time,*
- *averaging interval,*
- *maximum message size.*

The supported performance or QoS parameters are [BFMMVZ96]:

- *upper bound on end-to-end message delay,*
- *lower bound on probability of timely delivery,*
- *upper bound on delay jitter (optional),*
- *lower bound on probability of no loss due to buffer overflow.*

Both traffic and QoS parameters can be chosen from *continuous ranges* rather than from a limited menu of possibilities, which leads to a very general interface, that calls for a complex management by the network and a very large set of services for a client to choose from. However, it is argued that this flexibility is needed by applications and that on higher levels these values could be aggregated to common parameter combinations.

By the delay bound probability the hardness of the guarantee on the delay bound can be specified, therefore allowing for deterministic and statistical delay bounds, whereas for jitter and loss there are only deterministic bounds. Thus only a mixture of deterministic and statistical service commitment is available, that means service parameters and service commitmtents are not orthogonal to each other.

The Channel Setup Procedure

Due to the connection-oriented approach a channel set up procedure had to be designed in Scheme 1. It builds upon *traditional routing* to deliver routes[1], on which the global requirements as specified by the client are mapped onto local requirements for each node. At each of the nodes *admission tests* are performed, and, according to their result, reservations are made in a round trip: on the forward pass resources are reserved based on worst case assumptions,

1. A design simplification, however, the need for QoS-based routing mechanisms is recognized.

while on the backward pass these reservations are reduced in case QoS parameters were over-provided and relaxations are possible without compromising the client's requirements.

Due to the fact that packets always take the same route chosen during the channel setup (connection-oriented paradigm), the guarantees made by the network are never violated, assuming that the client adheres to its traffic specification (which is monitored by a rate control module or already part of the scheduling algorithm if rate-based scheduling is used) and there appears no network failure during the data transfer. Hence guarantees can be given a priori with respect to the data transfer phase.

The channel setup procedure as described above represents a very simple interaction of client and server, since the negotiation about the QoS parameters is essentially unilateral. If the network rejects a connection it gives back reasons for the failure but no hints how to form another, yet this time hopefully successful request.

The need for a pricing policy as means of preventing the clients from over-reservation is recognized but has not been implemented [BFMMVZ96].

3.1.2 Scheme 2

Scheme 2, the next Tenet scheme for real-time communication, builds on the concepts developed in Scheme 1 and has also been implemented in a protocol suite: Tenet Suite 2.The main focus of this work was to extend the basic real-time communication service provided by Suite 1 with respect to two important points [GHMS93]:

- *to provide abstractions and techniques for efficient multi-party communication and*
- *to make the client-service interface more flexible.*

These two issues had been neglected in Scheme 1 due to design simplifications [FBZ94].

Multi-Party Communication

The goals in providing multi-party communication were set to take an approach, which shall be *scalable* to hundreds of senders and/or receivers, and to provide for *dynamic group membership* by allowing to join and leave a communication group at any time [GHMS93].

As the basic unit of communication a 1xN simplex real-time multicast channel is chosen in contrast to the simplex unicast real-time channel in Scheme 1. This choice is motivated against a MxN model by the easy mapping onto lower level multicast services, by the easier implementation and the flexibility in having heterogeneous senders.

In this setting the traffic specification is given by the sender, while the QoS requirements are specified by the receivers thus allowing for *heterogeneous receiver requirements*. The key networking abstraction is the *target set*, which is similar to the host group in IP multicast with respect to separating senders from receivers, and which consists of a set of tuples indicating the address and QoS specification of each receiver. Receivers join target sets depending on their interest in a certain transmission (e.g., video from a given distributed conference), and channels are established from data sources to the members of the respective target sets. *Join* and *leave primitives* support dynamic membership in target sets, as well as the associated change in multicast channels.

Efficient utilization of network resources shall be achieved by the use of true multicast channels and resource sharing based on *sharing groups* [GHMN95], which allow a client to communicate application-specific information, such as an upper bound on the number of con-

currently active senders in the group to the network, which in turn uses this information to multiplex traffic from different senders on the same resources without violating performance guarantees.

Service Interface Flexibility

To improve the flexibility of the client-service interface, the values of the input parameters may be specified as a *pair*, representing the client's *desired value* and the *worst acceptable value*. This provision reduces the need for 'blind' negotiations as required in Scheme 1, because now the network does not have to reject a connection for which it cannot guarantee the desired service levels as long as it at can provide for at least the minimally acceptable values of the performance parameters [GHMS93].

It is suggested that the range of acceptable performance can also be used during the lifetime of a connection, if either a client requests for performance changes (e.g. in order to decrease the charges of the connection), or the network reacts to congestion on its links.

3.1.3 The Tenet Suites' Architecture - the Tenet Protocols

The Tenet suites were designed to coexist with the Internet protocols extending these by facilities for real-time communication.

The suites assume a data link layer capable of providing guaranteed performance services, thus the data transfer protocols being added to the standard TCP/IP protocols, which are still responsible for the transfer of best-effort traffic, are limited to the network and transport layer according to the TCP/IP model (see Figure 10 [BFMMVZ96]).

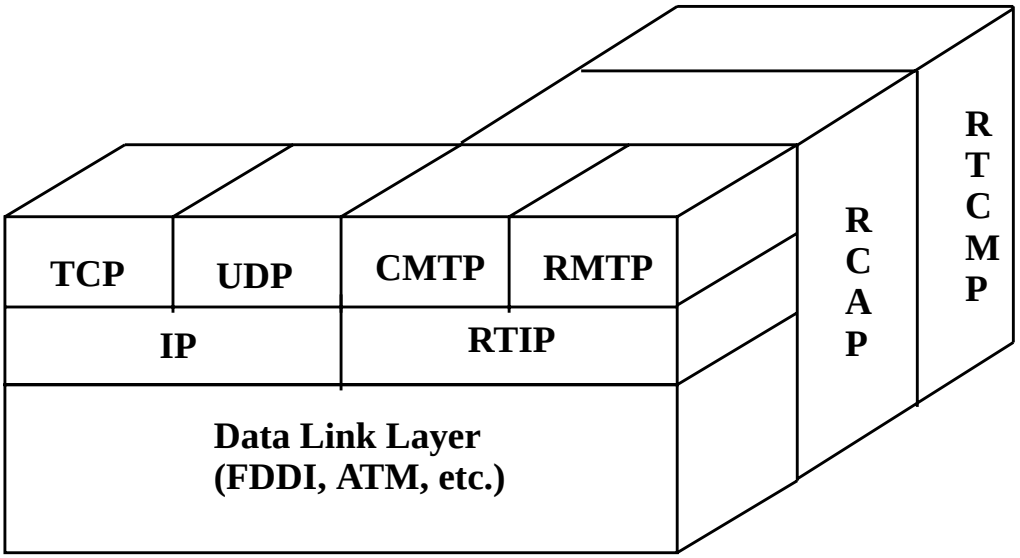


Figure 10: The Tenet protocols' architecture.

In the network layer for real-time communication a protocol called *RTIP* (Real-Time Internet Protocol) is operated in parallel to the traditional IP. RTIP performs rate-control, jitter control

and scheduling based on the QoS parameters of each connection, and thus offers an unreliable, simplex, guaranteed performance, in-order packet delivery service [ZVF92].

In the transport layer two protocols have been added: *RMTP* (Real-Time Message Protocol) and *CMTP* (Continuous Media Transport Protocol) [WM91]. The design philosophy of the Tenet suites is to provide one general-purpose protocols at each layer, which is obviously slightly violated by the two transport protocols (RMTP for transaction-oriented real-time traffic and CMTP for periodic, time-driven real-time traffic), however no media-specific protocols like video, audio or movie protocols are supplied.

Besides these protocols, which are responsible for data transfer, protocols for control functions were added: *RCAP* (Real-Time Channel Administration Protocol) and *RTCMP*. RCAP encompasses functions on data link, network and transport layer and is used for control under normal conditions and is mainly concerned with channel setup [BM91], while RTCMP's control functions relate to exceptional situations like network failures [BPF94].

3.2 Lancaster's Quality of Service Architecture

At the University of Lancaster an architecture of services and mechanisms for quality of service management and control of *continuous media flows* in multiservice networks has been developed: the so-called *QoS-A* model [CCH94]. This model was designed in support of *distributed multimedia systems*, which have diverse requirements on communication parameters like throughput, delay, jitter or loss rate and demand guarantees on the provided values for these parameters.

The implementation of QoS-A is based on an ATM network connecting heterogeneous workstations equipped with multimedia devices. The aim, QoS-A is targeting at, is the integration of QoS interfaces, management and mechanisms across all architectural layers and system components like end-systems, communication system and networks and therefore it is tried to provide QoS end-to-end system-wide.

3.2.1 Basic Terms

The basic terms around which the whole model is built are the notion of *flows, the concept of a service contract and the mechanisms of flow management* [CCH94]. A flow in QoS-A characterizes the production, transmission and eventual consumption of a single media stream, which may be unicast or multicast, but always simplex. By a service contract the QoS-A model introduces the need for *binding agreements* between service providers and users, hence only bilateral relationships are considered.

The term flow management refers to the mechanisms and principles by which the QoS guarantees are actually provided in the architecture, such as monitoring and maintaining the contracted QoS levels.

3.2.2 QoS-A Architectural Model

The QoS-A model is designed as a *layered architecture* [CCGHL93] similar to the OSI reference model, but besides layers different *planes* are used to structure the overall system and extend it in order to support QoS mechanisms for distributed multimedia applications (see Figure 11).

Apart from introducing multiple planes, the model also differs from the OSI reference model at the upper layers, where the distributed applications platform layer is supposed to give support for specific multimedia communications services like QoS configuration [CB93] and the orchestration layer shall provide for multimedia synchronization between multiple related flows [CCGH92]. The transport layer offers a suite of several protocols for different kinds of traffic like continuous media or constrained latency messages [Gar93]. As already mentioned the added planes purpose is the support of mechanisms to enable the system to provide end-to-end QoS.

There are three planes in the QoS-A model [CCH94]:

- The *protocol plane*, which is the traditional plane as known from the OSI reference model, however in the QoS-A model this plane is divided into two subplanes for data and control, in order to accommodate for the different communication demands of these traffic classes.

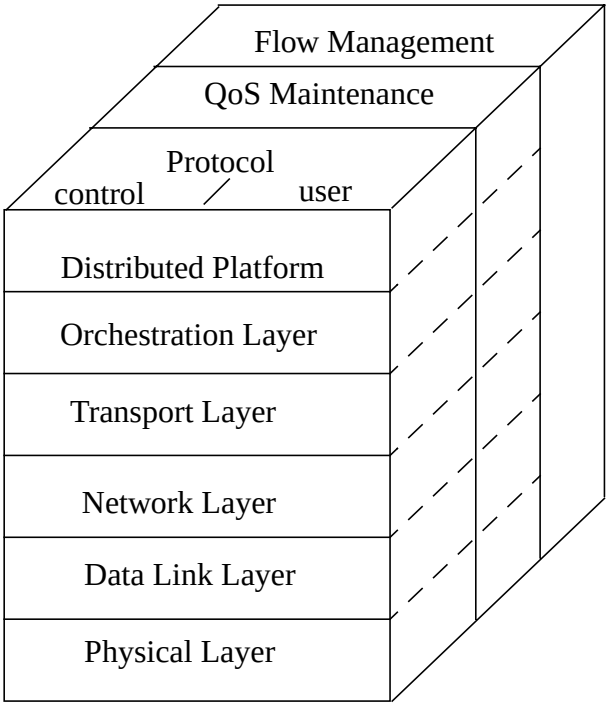


Figure 11: The QoS-A Protocol Stack.

- The *QoS maintenance plane*, which consists of a set of layer-specific QoS managers, which shall do fine-grained monitoring and maintenance of the contracted QoS levels (transparently from the user and in parallel to the media transfer).
- The *flow management plane*, whose tasks are the flow establishment (which encompasses flow admission control, reservation of resources, and QoS-based routing), QoS renegotiation, if either the service provider is no longer capable to maintain the contracted QoS level or the service user does no longer require the initially demanded QoS, QoS mapping between layers and QoS adaptation, which refers to coarse-grained actions in response to QoS violations by the provider and which need the invocation of the application and should therefore not be confused with the fine-grained control being performed in the QoS maintenance plane.

3.2.3 QoS-A Service Model

The QoS-A model retains the best-effort service model, but augments it with new service classes which require hard or soft guarantees about QoS parameters. For the new classes a set of mechanisms to provide performance monitoring, notification of QoS degradation and QoS renegotiation are proposed.The focal point in the QoS-A service model is the service contract between service user and service provider. The *service contract* encompasses [CCH94]:

- *QoS parameter requirements,*
- *the desired service commitment,*
- *the QoS adaptation strategy,*
- *the QoS maintenance policy,*

- *the connection type and*
- *the costs*.

The QoS parameters considered in general in the QoS-A architecture are *throughput, delay, jitter and loss*, of which each can be represented by a range of values expressing maximum, average and minimum requirements. The maximum, average and minimum values may also be bounded for QoS negotiation purposes [GHMY95]. If some QoS values do not play a role for an application “don’t care”-values may be attached to the respective parameters.

While the specification of the QoS parameters gives a quantitative characterization of the application requirements, the chosen service commitment shall determine how these parameters are interpreted by the service provider. The QoS-A model offers three different *service commitment classes:*

- *deterministic,*
- *statistical and*
- *best-effort.*

The deterministic service commitment shall be typically used for hard real-time applications, which are in need of totally reliable and timely service, while the statistical service allows for a certain and to be specified percentage of violations in the requested flow specification and is therefore suitable to continuous media which needs a timely service but can cope with some loss of data. The best-effort service is the traditional datagram service that gives no guarantees at all about QoS performance parameters.

A distinctive feature of the QoS-A model is the fact that each QoS parameter may have its separate service commitment [CCH94], thus allowing for a stream to have for example a deterministic bound on loss and only best-effort on throughput, delay and jitter, as would be appropriate, e.g. for a file transfer.

Another component of the service contract is the QoS adaptation strategy, that specifies in which manner should be reacted to a situation where the contracted QoS has to be degraded. The application has the choice between taking no action, disconnecting the flow, being indicated (and adapt to the available QoS level) and renegotiating the originally contracted QoS level [CCH95].

The service contract also specifies a *QoS maintenance policy*, which may be [CCH94]:

- *no maintenance at all,*
- *just monitor the actually delivered QoS, i.e. deliver periodically measurements of the QoS performance or*
- *maintain the QoS transparently by performing fine-grained actions to retain to the contracted QoS values.*

Furthermore the connection type being demanded by the application in the service contract is specified in the service contract. Here the choice is between a full end-to-end negotiated service, which means a setup phase before the media transfer, vs. a fast connect service, where the reservation and media transfer are performed in parallel. In addition the QoS-A model considers a *forward reservation mode*, that shall allow to have resources reserved not immediately, but for a specified time in the future. Yet, this mode has not been implemented.

The last component of the service contract is the associated *cost* of the demanded service. In the QoS-A model cost is mainly seen as a *backpressure method* to prevent users from demanding always the highest QoS and therefore degrading the overall system performance, since sub-

stantially less connections can be admitted. However, there is no such cost-based policy being implemented in the QoS-A model.

As described the QoS-A service model is very comprehensive, in particular, in providing for dynamic QoS by renegotiations which can be triggered by both the service provider and user without requiring to break up the existing connection. However considering the manageability in a large-scale networking environment, it might be too complex to be realized.

3.2.4 QoS-A Mechanisms

In addition to the augmented service model, which enables to specify QoS requirements, the QoS-A proposal also deals with the integration of a range of *QoS configurable protocols and mechanisms* into end-systems and network.

In end-systems the following functions are identified in the QoS-A model [CCH94]: thread scheduling, buffer allocation, jitter correction and synchronization of multiple related flows.

As main mechanisms to be incorporated in the network, reservation protocols and service disciplines are identified.

The required components to achieve these functions are ordered along the different planes in the QoS-A architectural model.

Protocol Plane

In the protocol plane the following components are identified as extensions to allow for the QoS support of the media transfer:

- a f*low regulator*, which shapes the flow generated by the application according to the flow specification in the service contract,
- a *flow scheduler*, which manages one queue for each service commitment class based on a hierarchical deadline scheduling,
- a *flow monitor*, whose task is to measure the actually provided QoS, and
- a *resource manager*, which depending on the commitment class and the considered QoS parameter invokes the necessary QoS mechanisms like buffer management, traffic regulation and scheduling.

QoS Maintenance Plane

The actions being taken in the QoS maintenance plane are subject to the same time domain as the media transfer and therefore execute control functions as opposed to *management functions* which operate on a slower time scale.

In case maintenance is a contracted service feature a *monitor-measure-adjust loop* is performed, which measures the receiver QoS against sender QoS and, if necessary, makes fine grained adjustments by adjusting loss via the buffer management, queueing delay via the flow scheduler and throughput via the flow regulator.

In particular, this plane is responsible for maintaining the contracted commitment in case of statistical services, where a certain percentage of the specified QoS level has to be achieved.

Flow Management Plane

The flow management plane exerts a range of static and dynamic QoS management functions, including the provision of a network signaling infrastructure, resource reservation, QoS adaptation (which means coarse grained dynamic QoS management as specified in the service contract's QoS adaptation strategy), QoS mapping between layers, support of forward connection mode and sampling of network load statistics.

In addition to these extensions of the communication system the QoS-A model implementation also extended a Chorus microkernel for QoS specification and real-time support through a real-time scheduling architecture [CB93].

3.3 HeiTS/HeiRAT

The *HeiProjects* [Herr94] at IBM's European Networking Center in Heidelberg were aimed at providing a distributed multimedia platform for PCs and workstations in an internetwork of LANs such as Token Ring and Ethernet. Among other things such as distributed multimedia applications, they included the development of *HeiTS* (the Heidelberg Transport System) for transporting multimedia streams across the network [WM91] and *HeiRAT* (the Heidelberg Resource Administration Technique) for providing a well-defined QoS for this transport [VHN93], [VWHW97].

HeiTS and HeiRAT have not only been developed for research purposes, yet, they are used in the Ultimedia Server/6000, a multimedia client-server system for audio and video retrieval. The applicability of HeiRAT is not confined to the HeiTS environment. Indeed, the HeiRAT approach for distributed QoS calculation can be extended to arbitrary chains of software modules (defined as *stream handlers* in [WM91]) and networks connecting sources to sinks. Here, for each individual stream handler or for sequences of adjacent stream handlers, the HeiRAT functions can be called to reserve appropriate resource capacities and to return QoS guarantees for the execution of these modules. In this scenario, the protocol stack of HeiTS could be one of the stream handlers. The QoS guarantees given for the individual stream handlers can then be accumulated in a similar fashion as done in a network with its routers and transmission links to yield end-to-end QoS guarantees. See [WM91] for more information.

3.3.1 HeiTS

HeiTS provides the ability to exchange streams of continuous-media data with quality of service (QoS) guarantees – where applications can specify the requirements they have for the transport service. To provide these QoS guarantees, the protocols of HeiTS are embedded into a processing environment which provides real-time techniques and resource management.

A sending application generates a continuous stream by reading data from an input device (e.g. disk, camera, microphone), possibly processing them (e.g. compressing video data) and then forwarding them across the transport layer interface to the transport system. The transport system possibly segments the stream, i.e., generates packets of a certain size, and transfers these packets through one or more networks and intermediate nodes to the target's host where they are reassembled and moved up to the application level. The application might do some further processing before the messages arrive at an output device.

Components

HeiTS provides protocols of transport, network, and data link layer. Apart from protocols, HeiTS contains also components for resource management, buffer management, and operating system abstraction (Figure 12).

The Layer 3 and Layer 4 protocols used within HeiTS are the network layer protocol ST-2 [Topo90] and the transport protocol HeiTP [DHHS92]. Both protocols are connection-oriented and enable the transfer of continuous-media data. At the bottom of HeiTS exists HeiDL (Heidelberg Data Link) providing the data link layer protocol and the interface to networks. HeiDL functions are used by the network layer to send and receive data, and to establish multicast groups. Data packets received are distributed to established ST-2 connections. Depending

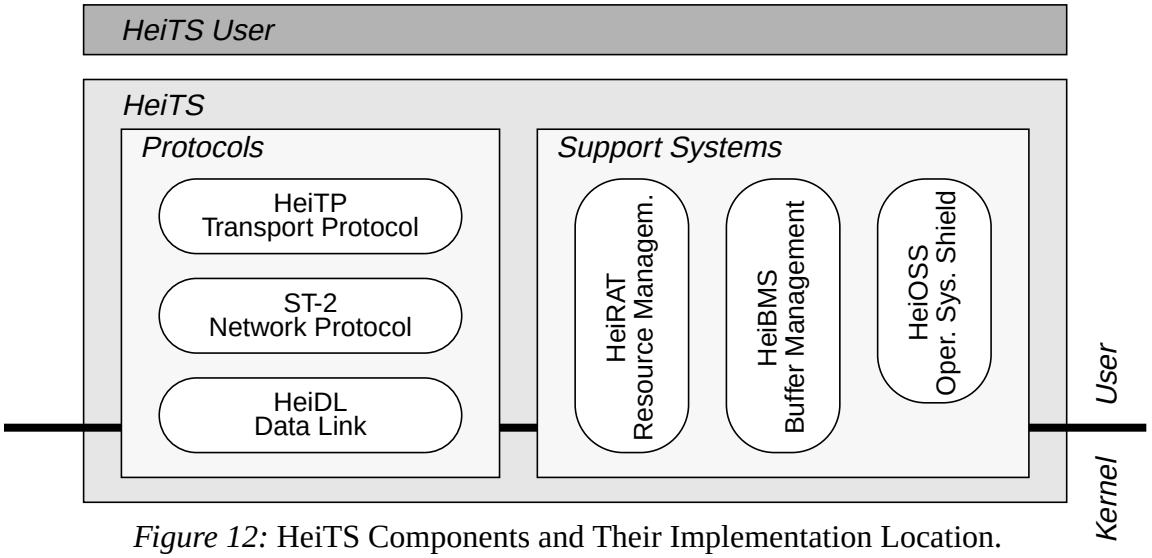


Figure 12: HeiTS Components and Their Implementation Location.

on the capabilities of the used network type (Token Ring, Ethernet, FDDI, B-ISDN,...), HeiDL transmits outgoing data packets with respect to their time-criticalness.

HeiTS has been implemented on different operating system platforms (mainly AIX and OS/2). To ease the porting of code between platforms an intermediate layer (HeiOSS) between the native operating systems and the transport system code has been used. Protocol packets sent across a network consist of several parts; the data an application wants to transfer and the protocol headers of each protocol layer. *Buffers* are the representations of packets in memory. The buffer management (HeiBMS) avoids data copy operations in the methods it provides to insert or remove data, to split a buffer, and to build a logical buffer out of multiple smaller memory pieces containing data and headers.

The resource management system HeiRAT is the focal point for resource allocation. Based on the QoS parameters delay, jitter, throughput, and reliability specified during connection setup it performs calculations to determine required system resources and administrates these resources.

Protocols

The network protocol of HeiTS is an enhanced version of ST-2. The basic ST-2 protocol stems from the Internet family, it is related to IP, and both protocols can coexist in one system. Many of the ST-2 enhancements made during the HeiTS development found their way into the follow-on version of that protocol, named ST-2+ [DB95], therefore, the latter resembles the HeiTS version of ST-2. This network layer protocol is used to set up streams from one sender to an arbitrary number of receivers. It implements the functions of bandwidth reservation and filtering in the HeiTS protocol stack. More information about ST-2 and ST-2+ will be given in Section 4.2. As part of the network layer activities of the HeiTS group, methods for QoS-based routing have been studied. A protocol for the exchange of according information was developed, results can be found in [VHKWW96].

The transport protocol of HeiTS is HeiTP. It implements media scaling and time-based error correction in the HeiTS protocol stack. HeiTP is designed to run on top of ST-2. Although ST-2 offers at the network layer a rich set of functions, there is still the need for a transport ser-

vice on top of it. Such a service needs to provide, for instance, functions to implement data segmentation and reassembly, and rate-based flow control. These basic functions, augmented by support for media scaling and time-based error correction schemes, are the core of the HeiTP services.

HeiTP supports connection-oriented communication only. This is considered as appropriate for multimedia streams because it facilitates applying resource reservation techniques. Further, media scaling and filtering both need the notion of a multimedia stream to be implemented as well. HeiTP streams are multi-destination simplex and they are univocally mapped to the underlying ST-2 streams. When building a new stream, the user provides a QoS specification (based on logical data units) which is translated into network layer terms.

HeiTP offers several reliability classes that can be selected by the user when a stream is being setup. These classes indicate whether the user intends to detect the presence of corrupted data and how the system should react upon such detection. Options include: a) discard the data, b) receive it with or without an indication, or c) attempt to correct the error. The latter leads to the provision of a reliable data transfer service. Offering reliable transmission is uncommon to multimedia communication systems, the reasons for also providing that service are that some audio and video decompression systems cannot tolerate loss, human perception is disturbed by loss of data, esp. of audio information, and that one cannot recover from an error that is included in the first recording of data. Since traditional reliability mechanisms have various flaws in the context of continuous-media data, HeiTP introduces the notion of *partially reliable streams* [DHHHST93]. Partially reliable streams suggest a weak concept of reliability: A receiver may request retransmission of one or more of the last few packets in a stream. By limiting the number of packets to be retransmitted, the sender never needs to store more than a certain number *n* of packets for potential retransmission. A lost packet not contained in these packets cannot be recovered; hence, the stream is only partially reliable. The value of *n* can be calculated from the timing constraints of the multimedia presentation, taking into account the reliability of the underlying networks.

A scalable stream can be seen as composed of various substreams (see also chapter 5 for further information about scaling and filtering). For example, one could use one substream for intra-coded frames and one or even several other streams for the remaining frames, which implies that there are streams of different degrees of importance. In the scaling implementation of HeiTS [DHHH94], the individual substreams are mapped onto different connections at the *network* layer, each with its own set of QoS parameters. The transmission quality can then be adjusted either with fine granularity within a connection (substream) or with coarse granularity by adding and removing connections (substreams). This can be considered as *continuous* and *discrete* scaling.

HeiDL, the data link layer part of HeiTS has been implemented for several subnetworks including Token Ring, Ethernet, FDDI, and ATM. Due to their varying characteristics, each of these provide different capabilities with respect to the two most important issues at this level: *bandwidth reservation* and *multicast*.

3.3.2 HeiRAT

The resource management system HeiRAT manages all the resources, on a path from source to sink(s), both in the local systems and the network, which are critical for the execution of con-

tinuous-media data processing: CPU time, network bandwidth, and memory. HeiRAT provides the functionality for QoS negotiation and for QoS enforcement.

In the QoS negotiation phase of a multimedia stream, applications specify their QoS requirements. These parameters are used for the throughput test and the QoS calculation which result either in a resource reservation or in the rejection of the stream establishment, the latter if the QoS cannot be met. In the transmission or QoS enforcement phase, after the successful establishment of a stream, the resources are scheduled with respect to the given QoS guarantees.

HeiRAT offers several options by which applications can specify their QoS requirements. QoS values are given in terms of maximum end-to-end delay, minimum throughput needed, and reliability class defining how the loss of data shall be treated. An application can select one of the QoS parameters for optimization by specifying an interval from *desired* to *worst-acceptable* values. For example, a video application might request a throughput between 15 and 30 video frames per second, indicating that video quality would not be acceptable with less than 15 frames, but that more than 30 frames are never needed. HeiRAT will then return the best QoS it can guarantee within this interval and make the corresponding reservation (or reject the request if even the lower bound cannot be supported). The HeiRAT throughput model is based on the *linear bounded arrival process (LBAP)* model as introduced by [Cruz91] and used by [Ande93]. The LBAP model assumes data to be sent as a stream of discrete units (*packets*) characterized by three parameters:

- S = *maximum packet size*,
- R = *maximum packet rate* (i.e., maximum number of packets per time unit), and
- W = *maximum workahead*.

The workahead parameter W allows for short-term violations of the rate R by defining that in any time interval of duration t at most $W + t*R$ packets may arrive on a stream. This is necessary to model input devices that generate short bursts of packets, for example disk blocks that contain multiple multimedia data frames, and also to account for any clustering of packets as they proceed towards their destination (for work conserving systems). Although it may be somewhat counter-intuitive, it is possible to use LBAPs for the management of variable bit-rate streams with varying bandwidth requirements as shown in [Vogt95].

In the transmission phase, data are processed and transmitted according to their urgency. Schedulers handle time-critical multimedia streams prior to time-independent data. They exploit properties of the underlying resources, for example, they are based on the operating system priority scheme for CPU scheduling or the MAC priority scheme of the network.

HeiRAT offers two types of QoS: *guaranteed* and *statistical*. For guaranteed QoS, the resource capacities reserved are for the maximum demand a stream may have during its lifetime. Reserving extensive amounts of capacities for such peak requirements can be rather costly and leads to the under-utilization of resources if there is a significant difference between peak and average data rate of a stream. A cheaper alternative is statistical QoS where resources are slightly overbooked. This implies that while QoS requirements will be met most of the time, occasional QoS violations may occur (and applications need to be ready to cope with them).

In addition to admission control and resource reservation, HeiRAT uses scheduling mechanisms to ensure that the negotiated QoS is provided. These are offered for CPU and network resources. The HeiRAT algorithms for CPU scheduling are based on classical approaches for

real-time processing, namely *earliest-deadline-first* and *rate-monotonic* scheduling [LL73]. In HeiRAT, these approaches have been extended to account for the two classes of guaranteed and statistical connections as well as for workahead packets. This extension is based on the method of deadline-workahead scheduling [Ande93] which dynamically classifies packets with respect to whether they are currently critical or workahead. A description and evaluation of the CPU scheduler is given in [WM91].

The network access scheduler influences the order in which packets are sent on the network. This means the access to the network adapter, i.e., the queue inside the computer in front of the network, as well as the time at which a packet is actually be transmitted via the network, especially in case of a shared-media network as most LANs are.

Within HeiRAT, modules for throughput test and QoS calculation have been implemented for various network technologies (Token Ring, FDDI, Ethernet, and partially for ATM) and local processing resources (CPU and buffer space). Especially for shared-media LANs, a *central bandwidth allocation* module was developed [KMR93]. Nodes which wish to reserve network resources contact this module and request a certain QoS. This module checks the available resource capacity and grants or denies the reservation. Since it has ‘global’ knowledge it can calculate better QoS values, additionally, it can serve as a central policy agent.

Finally, mechanisms for *Resource Reservation in Advance*, i.e., to setup a reservation for a future time interval, have also been added to HeiRAT [WS97].

3.4 The XRM Architecture

The COMET group at the University of Columbia is developing an Extended Integrated Reference Model (XRM) [KW94] as a modeling *framework for control and management of multimedia telecommunications networks*, which comprises multimedia computing platforms and broadband networks. The emphasis of the XRM model is rather on the architectural issues of a *system-wide end-to-end QoS provision* than on the definition of a comprehensive service model.

The XRM model assumes that the foundations for the operability of multimedia computing and networking devices are the same. They model both classes of devices as producers, consumers and processors of media, differing merely in the overall goal, that a group of devices is set to achieve, in the network or the multimedia platform.

If the XRM is restricted to the networking part of the model the so-called Integrated Reference Model (IRM) is the result. The IRM consists of five planes, which contain monitoring and real-time control, management, communication, and data abstraction primitives: the network and system management or N-plane, the resource control or M-plane, the data abstraction and management or D-plane, the connection management and binding or C-plane and the user information transport or U-plane (see Figure 13) [Laza94b].

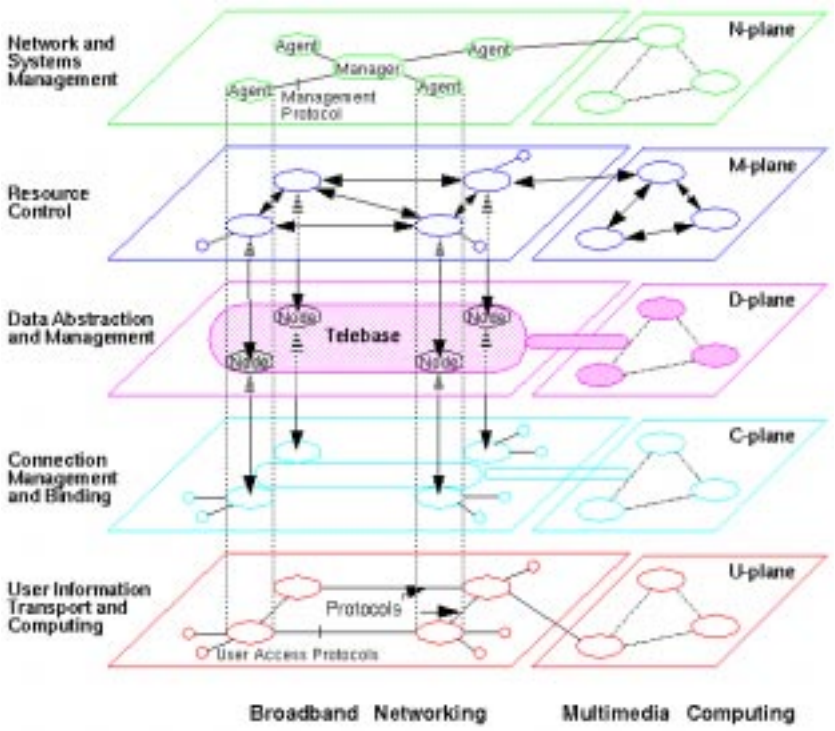


Figure 13: The XRM model.

If on the other hand the XRM is restricted to the multimedia computing platform a unit of similar functionality as the IRM can be viewed. In this partial model the N-plane caters for the system management functionality, and the M-plane encompasses process scheduling, memory

management, routing (when applicable), admission control and flow control. Objects modeling multimedia devices are found in the D-plane, the C-plane contains binding functionality, and the U-plane's task is the transport of user information within the Customer Premises Equipment (CPE).

For purposes of structuring, the XRM is furtherly divided into three (sub)models [Laz94a]. These models are defined by the functionality commonly associated with the broadband network and media processors, the multimedia network, and the applications and services network. These are called the R-, the G- and the B-models, or RGB for short. The RGB model is shown in Figure 14 [Laza94b].

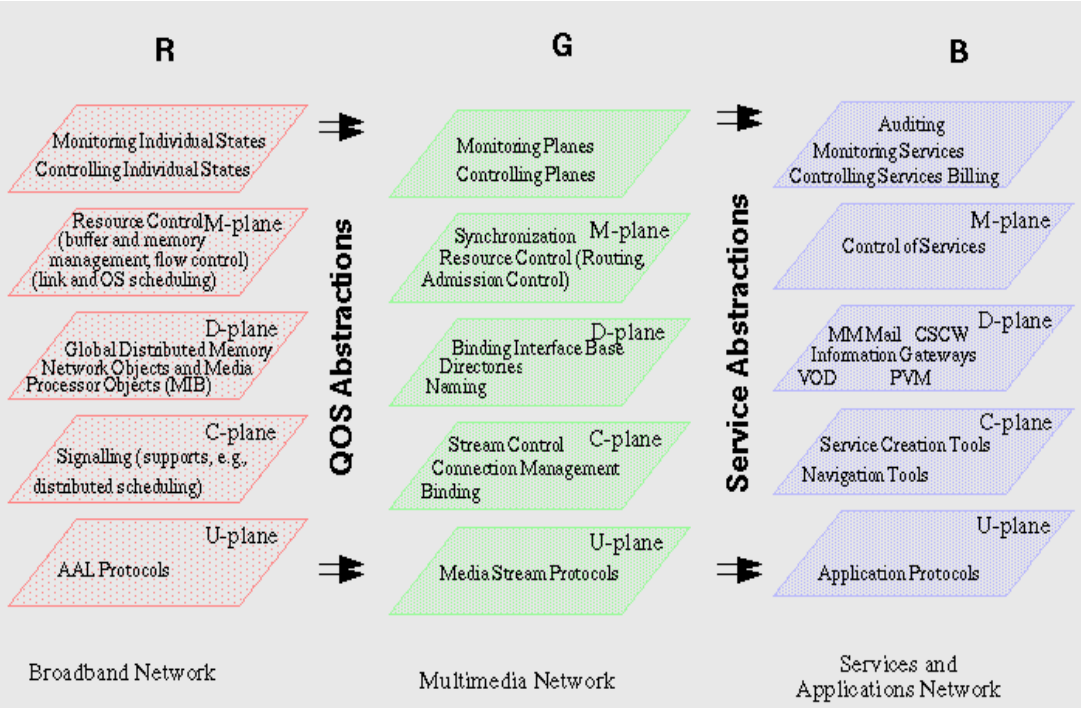


Figure 14: The RGB model.

These models interface to each other as defined by a set of abstractions or services. On the one hand *QoS abstractions* are provided by the broadband network and media processors to the multimedia network, while on the other hand *service abstractions* form the interface between the multimedia network and the services and applications network.

3.4.1 The Functionality of the Broadband Network and Media Processors (R-Model)

As already mentioned, the R-model, shown in the figure above, represents the functionality of the broadband network and the media processors. The task of the broadband network as perceived by the multimedia network, is to provide a service in terms of QoS abstractions.

The D-plane of the R-model abstracts the main network components, i.e., switches, multiplexers and media processors as a global distributed memory. In particular, communication links are viewed as FIFO memory, whereas switches and processors are modeled as random

access memory. These are again abstracted on a higher level and constitute eventually entities of a Management Information Base (MIB).

The N-plane's task is to perform network and system management and is about monitoring and controlling of individual states of the network and system components. These states might correspond to the status of a link, the temperature of an interface card, and so on and are represented as objects residing in the MIB. For monitoring and controlling purposes the N-plane manager uses as a primary mechanism a client-server interaction.

The M-plane models the resource control tasks. For example, the main resource functionality perceived at the switch or multiplexer level is buffer management and link scheduling. At the CPE level the same tasks appear as operating system scheduling and memory management. As another important resource control mechanism flow control is taken into account.

The exchange of state information among distributed buffer management and link scheduling entities is supported by the C-plane. Exchange of state information is in particular required in cooperative distributed scheduling. Thereby a larger networking capacity under QoS constraints can be achieved.

The U-plane provides for cell level adaptation protocols, which are used for segmentation and reassembly of cells as well as reliability checks.

3.4.2 The Functionality of the Multimedia Network (G-Model).

The functionality of the G-model is divided into the five planes of the XRM as well. From the perspective of the services and application network the multimedia network is a service provider of a well-defined set of service abstractions. As already mentioned in the previous section, these abstractions are realized by the multimedia network based on the QoS abstractions provided by the broadband network and media processors, hence a layered structure also between R-, G- and B-model is proposed.

A distributed repository containing information about entities that might participate in a binding process, the so-called Binding Interface Base (BIB) is contained in the D-plane. Services are regarded as a set of interconnected objects.

The *binding architecture* [LBL94] is the attempt to *open the signalling* of proprietary protocols and therefore enlarge their applicability. Currently, ATM LANs provide proprietary software in support of Q.2931 [ITU96c] or other closely related signalling protocols. Hence there is no facility to interface for non-proprietary software in order to access network resources. Thus a need is derived to devise a methodology for an open network access to broadband networking and media processor resources subject to the support of a rapid and flexible service creation, deployment and management strategy.

The N-plane's major task is the monitoring of the behavior of distributed systems. It should also provide for components management such as high speed manageable host interfaces.

The U-plane incorporates the support of a number of media stream protocols such as a native ATM stack and other real-time protocols. Also data protocols offering best-effort services like TCP/IP shall be supported and can therefore coexist with media stream protocols.

The M-plane offers orchestration as well as other resource allocation mechanisms. Routing and admission control are considered the most important parts of the resource control.

Stream control, connection management or more generally binding algorithms are supported by the C-plane. Stream control means the protocols required for remote control of mul-

timedia devices such as tape drives, multimedia on demand systems, etc. As connection management mechanisms unicast as well as multicast algorithms belong to the C-plane.

3.4.3 The Functionality of the Applications and Services Network (B-Model)

Again, the functionality, this time for the B-model, is divided among the five planes of the XRM architecture.

The management of services belongs to the N-plane. Management support for access, security, configuration, billing and auditing services are identified as possible tasks. The task of service admission control is shared between the N- and the M-plane.

A key requirement of the M-plane is the control of services and the negotiation of networking and computational resources.

The D-plane is supposed to be an object repository of services such as multimedia mail, computer supported cooperative work, video on demand, parallel virtual machines, etc.

Navigation and service creation tools are the object of the C-plane.

Finally, application protocols belong to the functionality of the U-plane.

3.4.4 Summarizing Considerations

The description of the XRM and its subdivision into three functional reference models implicitly suggests how the designers of XRM view the main problem areas in multimedia networking. Their key issue is [KW94]:

- the relationships with respect to service provision and interfaces between middleware (multimedia network) and broadband network and multimedia network and applications, or more specifically:
  (1) the definition of the interface between the multimedia network and the broadband network and media processors,
  (2) design and implementation of a QoS architecture for the multimedia network
  (3) design of programming tools for supporting the services at the interface between the multimedia network and the applications and services network.

Further points being made are that scaling and operability of the overall system should be considered major issues that will need to be dealt with extensively.

Scaling according to [Laz94b] refers to both time and space, and represents the ability to deal with more then ten orders of magnitude in time (from ns to minutes) and possibly global distribution of the network.

Operability refers to the overall ability to manage and control [Laza94b]. It implies the design of a system that gathers traffic statistics and responds rapidly to dynamically varying traffic loads, network status and fault conditions, and simultaneously provides different grades of service guarantees to different traffic types based on their fidelity requirements.

3.5 The IntServ Architecture

The need for support of new distributed applications mostly arising in the field of multimedia has also been recognized by the Internet community. Since the current Internet architecture is incapable of providing QoS, which is the main requirement of the newly emerging applications from a communications' point of view, the IETF (Internet Engineering Task Force) has been for some time examining, how the Internet architecture can be enhanced to provide such services [BCS94]. The proposed model for this is called the Internet Integrated Services, or briefly the *IntServ* model. This name already reveals that the new model shall be applicable to different classes of traffic, and hence support beyond others real-time traffic requirements like QoS provision.

The IntServ model is the attempt to merge the advantages of two different paradigms: d*atagram networks* and c*ircuit switched networks* [CSZ92].

Datagram networks like the Internet maximize network utilization by multiplexing multiple data streams. They also provide for multipoint communication and robustness by adapting to network dynamics. So far, however, datagram networks only provided a best-effort delivery service.

On the other hand, current circuit switched telecommunications and ISDN networks provide service guarantees, yet lead to inefficient use of network resources when sending bursty data traffic and have difficulties in adapting to link and intermediate node failures, furthermore they lack support for multipoint communications.

So the aim set by the IETF is to provide for robust, yet flexible services which allow for QoS support and multipoint communication while making efficient use of network resources [BCS94].

3.5.1 Basic Assumptions and Perspectives

Group communication is regarded as one of the main requirements of the new applications and therefore multicasting is considered vital for the new Internet infrastructure [BCS94].

In addition to the provision of QoS for real-time applications the *sharing of bandwidth* between different traffic classes is considered a desirable characteristic of a future Internet infrastructure,. Therefore, the IntServ model shall include besides best-effort and real-time services a so-called controlled-link service [Ferr95].

The approach taken to introduce these partially new services is not to design a new Internet architecture from the scratch, but to supplement and extend the existing Internet architecture with some new components.

The basic assumption of the IntServ approach is that resources must be explicitly managed, otherwise guarantees cannot be granted, no matter whether they are deterministic or statistical, strict or approximate.

Since the IntServ approach assumes flow-related state to do the resource reservation inside the network, this means a partial shift away from one of the basic principles in the Internet, the so called end-to-end argument. This claims that flow-related state should only be maintained in the end-systems [Clar88]. However, in order to relax this paradigm shift away from connectionless to connection-oriented services, the IntServ model operates on *soft-state* in the network, thereby trying to maintain the robustness characteristics of the Internet [ZBEHJ93]. By using soft-state a hybrid form between connectionless and connection-oriented services is used.

Since resource reservation leads to privileges, the IntServ model recognizes the need for *policy and administrative control*, which in turn will lead to authentication requirements as to who issued the reservation and whose packets receive the contracted QoS [Baker96].

Another fundamental assumption of the approach taken is the integration of real-time and non-real-time communication into a single Internet infrastructure, and thereby enabling *statistical sharing* between these two traffic classes. This is opposed to building an entirely new, parallel infrastructure for real-time communication, leaving the Internet architecture unchanged and dedicated to non-real-time traffic. Hence, a *unified protocol stack* for both real-time and non-real-time on the internet layer is envisaged in the implementation of the IntServ model.

3.5.2 Overview of the Architectural Components

In today's Internet, IP forwarding is totally egalitarian, i.e., all packets receive the same best-effort QoS and packets are typically forwarded using a strict FIFO queueing discipline. For integrated services a router must implement an appropriate QoS for each flow (in the IntServ model a simplex data stream with possibly multiple destinations), in accordance with the service model. The component that enables the routers to manage the resources in order to enforce the QoS of individual flows is called *traffic control* and consists of three components [SCZ93]:

- *Packet Classifier,*
- *Packet Scheduler,*
- *Admission Control.*

The fourth component of the IntServ framework is the *resource reservation protocol*, which shall hold together the local traffic control components of the routers by communicating the resource requirements of applications between the routers.

3.5.3 The Traffic Control Components

Packet Classifier

The packet classifier performs a mapping of incoming packets into classes, which are characterized by the fact that all packets of the same class get the same treatment from the packet scheduler [BCS94].

The possible classifications which are envisioned are, for example, all video flows or all flows attributable to a particular organization. However, also a single data flow can represent a whole class, this is supposed to be observed especially at the edge routers of the network, whereas the Internet core routers should rather be using aggregation mechanisms.

At the moment packet classification for routers is complicated by the fact that the destination address, which is the sole information for routing purposes in the Internet, is no sufficient information to select the class of service a packet must receive. This however, is expected to be alleviated by the use of the flow label field of IPv6 [Deer95], when the next generation IPv6 will be eventually widely employ in the Internet.

Packet Scheduler

The packet scheduler's task is the forwarding of different packet streams using a set of queues and possibly other mechanisms like timers [BCS94]. In order to achieve the desired QoS provision, the basic function of the scheduler is to reorder these queues, in the simplest case according to a priority scheme.

The packet scheduler is implemented at the output driver level, since this is the point where packets are queued. The output driver uses the appropriate link layer controls over the allocation of bandwidth - if such mechanisms exist - in order to enforce the given guarantees about reserved bandwidth.

A possible submodule of the packet scheduler might be a component called *estimator*, which samples statistics of traffic streams that control packet scheduling and admission control. Some perceive the estimator an independent module [SCZ93], but usually, if it is considered at all, it is viewed as a part of the packet scheduler.

Admission Control

In the IntServ model the admission control component implements the decision algorithm used by a router or a host to decide whether a new data flow can be granted its requested resources or not [BCS94].

In addition to ensuring that QoS guarantees are met, admission control also has the task of enforcing administrative policies on resource reservations and plays an important role in *accounting and administrative reporting* [JCSZ92].

The admission control is not to be confused with policy control, which is performed at the edge of the network to ensure that hosts do not violate their traffic characteristics and which is considered part of the packet scheduler [BCS94].

The traffic control components have to be realized in the routers of the Internet as shown in Figure 15.

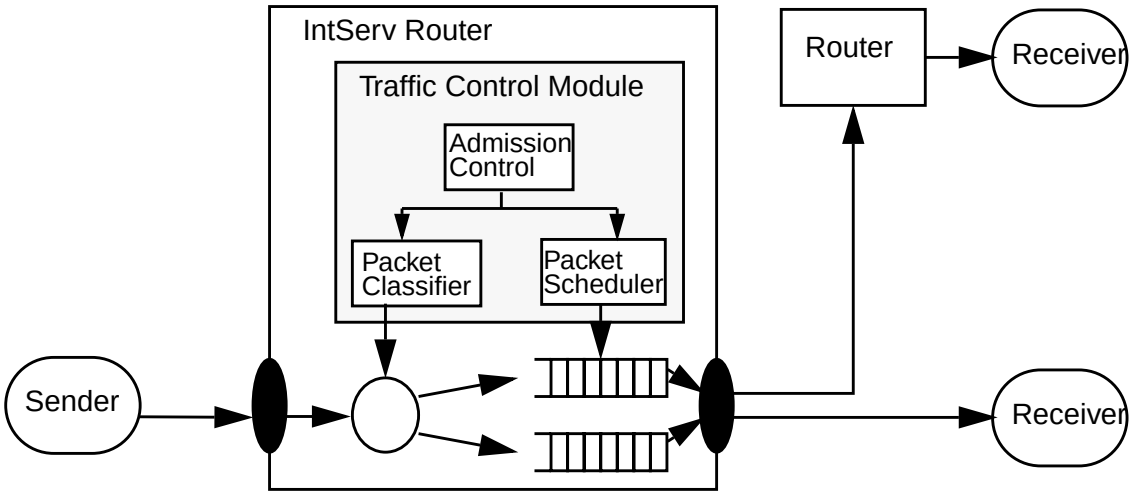


Figure 15: Traffic Control components in an IntServ router.

What is still missing is the component, which initializes the traffic control modules with the necessary parameters in a distributed fashion, by conveying the application needs from the end-users to the routers. This is achieved in the IntServ model by the use of a resource reservation protocol.

3.5.4 Resource Reservation Protocol

The resource reservation protocol creates and maintains flow-specific state in both, the end-systems and the routers along the path of a flow. At the moment, the IETF favors the *RSVP* protocol [ZBEHJ93][BZBHJ96] for resource reservation and advocates for its exclusive use in the future Integrated Services Internet, in order to avoid confusion by employing several incompatible protocols [BCS94].

The QoS requirements of applications are specified by so-called *FlowSpecs*, which are essentially lists of parameters [NS95b]. The subset of packets of a session that shall receive the requested QoS is specified by so-called *filter specs*, thereby achieving isolation between reserved resources and the packets that use these resources [ZBEHJ93].

These FlowSpecs and filter specs are transported by RSVP as opaque values and passed to the local traffic control of the routers along the path of the flow to be admitted or rejected by the admission control. If admission control succeeded, the filter spec is used to parameterize the packet classifier and the FlowSpec sets parameters in the packet scheduler module.

Another feature of RSVP is the support for several *reservation styles*, that determine in which manner the resource requirements of multiple receivers are aggregated in the routers.

Scalability and heterogeneity are among the design goals of RSVP. Therefore, a receiver-oriented reservation model is used, i.e. the receivers issue the reservation requests. They do so periodically during the data transfer establishing soft state in the routers [ZBEHJ93]. This as already mentioned tries to retain the connectionless paradigm of the traditional Internet architecture.

Figure 16 illustrates the above described architectural model of IntServ enhanced by RSVP. As can be seen, the traffic control module is working on the basis of the data being supplied by RSVP. More detailed information about RSVP will be given in Section 4.3.

3.5.5 The IntServ Service Model

The IntServ model recognizes the need that the service model must remain relatively stable over the long term, since it is intermediate between the applications requesting the services and the underlying network technology implementing the services and both will change with time. The service model shall therefore not be based on certain network technologies but on fundamental service requirements.

It proposes a core set of services for the Internet that is almost exclusively concerned with the time of delivery of packets. The model regards *network delay as the central quantity about which the network makes commitments*, further restricting its view to bounding the maximum and minimum delay. The degree to which application performance depends on low delay service varies widely, the IntServ views on the one end real-time applications while on the other end elastic applications are regarded [BCS94].

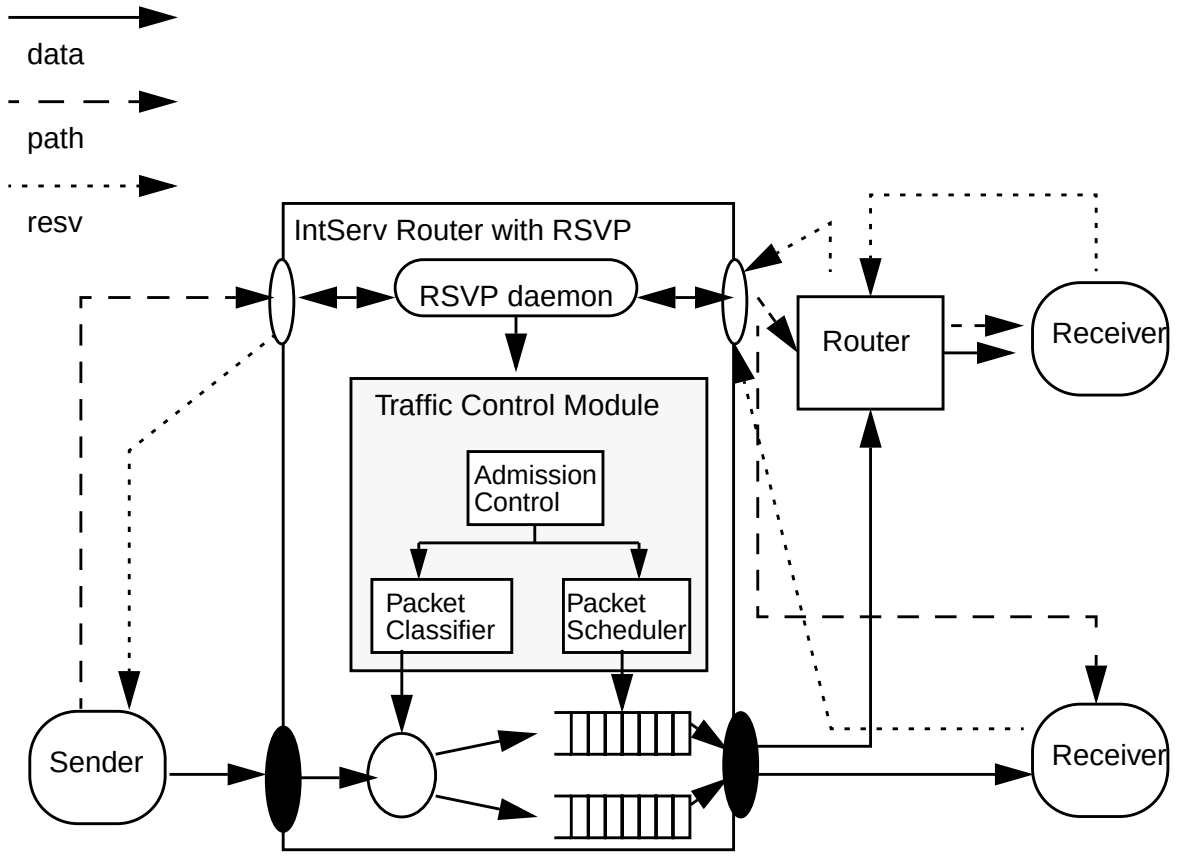


Figure 16: Incorporation of RSVP in an IntServ router.

Real-Time Applications

Playback applications are considered as main representatives by the IntServ model in this class. Such applications record some signal, packetize it and transmit the packets over the network to the receiver(s). The time when the signal is to be replayed at the receiver(s) is called the playback point, which has a fixed offset from the departure time from the sender. Data arriving after its playback point is useless in reconstructing the signal. To choose a reasonable playback point, an application needs an a priori characterization of the delay its packets will experience.

The model regards two criteria on which the performance of real-time applications is measured: *latency and fidelity.* In the category of real-time applications there exist different classes of applications regarding to their sensitiveness to delay and their tolerance to fidelity loss.

Intolerant applications like circuit emulation are supposed to be in need of a perfectly reliable upper bound on delay and therefore a guaranteed service class is introduced [SPG96], while tolerant applications can cope with some late packets and thus predictive services, which only give a fairly reliable bound on delay, are proposed [SCZ93].

The use of predictive services is motivated by efficiency considerations: the performance penalty in comparison to guaranteed services is supposed to be small for tolerant applications, but the overall efficiency gain is considered to be quite large. Of course, there has to be an

incentive to use predictive service instead of guaranteed service. This will be the cost of the chosen service [BCS94].

As a prerequisite for the provision of delay bounds the traffic characterization from the source and the existence of an admission control algorithm are considered.

The IntServ model also recognizes the existence of applications that can adapt to delay or rate variations, and therefore incorporates the notion of a network notification to these applications of a QoS degradation in order to allow them to change their traffic characterization.

Elastic Applications

Elastic applications, as they are perceived by the IntServ model, do not necessarily ignore delay variations but are characterized by the fact, that they will always wait for data to arrive rather than continue without them. Since these applications use incoming data immediately in contrast to real-time applications, it is argued that they do not require any a priori characterization of the service in order to fulfil their function. While real-time applications depend rather on the tail of the packet delay distribution, elastic applications care for the average packet delay [BCS94].

Elastic applications are further divided into interactive burst (e.g. telnet, X, NFS), interactive bulk (e.g. ftp), and asynchronous bulk transfer (e.g. mail).

The suitable service for these kinds of traffic shall remain the conventional best-effort or so-called *ASAP service*, although different classes reflecting the relative delay sensitivities and issues of fairness are being considered [BCS94].

Resource Sharing and its Integration in the Service Model

As already mentioned the IntServ model, besides providing QoS to individual flows, also incorporates resource sharing which is rather performed on a level of collected flows.

The quantity of primary interest in this context is the aggregate bandwidth of links as opposed to delay in case of the QoS provision aspect of the IntServ model.

The model envisions several styles of sharing like *multi-entity sharing*, if a link is to be jointly used by several organization, *multi-protocol sharing*, if a link divides up its capacity for different protocols in order to prevent them from interfering with each other, and *multi-service sharing*, where services obtain only a certain fraction of the total bandwidth available [BCS94]. These can of course be applied hierarchically. The service class being introduced for this kind of link-sharing is called controlled-load [Wroc96].

For this service class, too, an admission control will be necessary to ensure the conformance to the given commitments [JCSZ92].

Packet Dropping

An augmentation of the service model is the integration of a packet dropping mechanism, which allows for a preemptable packet service, in which some of the packets can be marked as preemptable. This is led by the observation that in many audio and video streams some packets are less important than other packets, and should therefore in a case of congestion be dropped first. It is therefore a means of delivering application-specific information to the network, in order to assist the network to make more reasonable decisions.

The IntServ model considers to go even one step further by introducing expendable packets, which in contrast to preemptable packets do not have to pass admission control any more, because the expectation for these packets is that many of them are to be dropped, while for preemptable packets still most should be delivered, and therefore they are considered a part of the flow subject to admission control.

3.6 The ATM Service Model

The ATM Forum service model, which is mainly contained in the UNI (User-Network Interface) TM (Traffic Management) [ATMF96a], the SIG (Signaling) 4.0 [ATMF96b], and the P-NNI (Private Network-Node Interface) specifications [ATMF96c], is more advanced than the ITU-T's model [ITU95][ITU96a][ITU96b] and can in most areas be regarded as a superset to it. Therefore, mainly the ATM Forum model will be considered, whereby some of the discrepancies between those two models will be given during this discussion.

The fundamental building block of the ATM Forum service model is the existence of multiple *service categories*, of which there are currently five [ATMF96a]:

- *Constant Bit Rate (CBR),*
- *Unspecified Bit Rate (UBR),*
- *real-time Variable Bit Rate (rt-VBR),*
- *non-real-time Variable Bit Rate (nrt-VBR) and,*
- *Available Bit Rate (ABR).*

The assumption is, that this set of service categories is sufficient to serve all the communication needs of different applications. This is in contrast to other approaches as for example the Tenet approach, which as already mentioned favors a continuum of services by allowing applications to choose from the whole set of parameters the values they regard as necessary for their goal to be achieved. While the Tenet approach postulates that this kind of flexibility is needed by applications, the ATM service model rather argues that their restricted set of QoS differentiation is the only possibility to retain the manageability of the QoS provision. They regard a qualitative change in the space of QoS parameters as represented by the service categories as necessary, since different functional mechanisms have to be employed, which do not allow for continual changes in QoS performance any more. Another argument by the ATM service model is the fact, that these service categories could be more elaborated and new ones could be added any time, as was the case in the past [Garr96].

The ATM service model and its service categories shall represent an abstraction to map applications to mechanisms provided by the network to achieve the necessary QoS for the application. By each of the service categories a set of important applications with certain common requirements and properties shall be represented.

Before regarding which applications the ATM Forum had in mind when designing their service model, a brief history of this model will be given. After that the provided QoS parameters, the service categories and the mechanisms used to achieve the QoS goals are treated in more depth.

3.6.1 Brief History

The ATM forum service architecture is largely based on the ITU-T's service architecture, which could also be regarded as its predecessor. The service categories in the ATM Forum model are similar to ITU-T's *ATM bearer capability (BC) classes* (there are four classes: A, B, C and X) as described in ITU-T recommendations I.356 [ITU95] and I.361[ITU96a]. The BC model built a class model along three properties:

- constant or variable source rate (CBR vs. VBR)

- whether time synchronization between source and receiver is required (real-time vs. non-real-time)
- connection-oriented vs. connectionless service above ATM layer

While the ATM Forum model keeps the CBR/VBR distinction, real-time is distinguished by the fact whether an application has explicit and quantifiable requirements regarding delay and jitter or not, which is a weaker real-time notion as in the ITU-T model. Finally, the connection-type indication was banned from the ATM Forum service model, since it was considered useless information in the ATM layer [Garr96].

The ATM Forum model incorporated from the beginning best-effort services, which were later (ATMF TM 4.0 [ATMF96a]) even split into two services: 'plain best effort' (UBR) and 'better best-effort' (ABR) [Garr96].

The VBR service retained from the ITU-T was then also divided along the categories of real-time and non-real-time.

When comparing the ATMF TM 4.0 against the respective ITU-T document, ITU-T Rec. I.371 [ITU96b], which also deals with traffic and congestion control, the following observations can be made:

- in I.371 CBR is called DBR (Deterministic Bit Rate), and VBR is called SBR (Statistical Bit Rate), however there is no distinction in real-time and non-real-time SBR, and no existence of a service comparable to UBR, furthermore ABR is not fully specified in I.371,
- with regard to the provided QoS by service categories it has to be observed, that while there is no negotiation of individual QoS parameters in the ITU-T model, but only QoS classes with fixed QoS parameter values, the ATM Forum provides for more flexibility by introducing individually specifiable and negotiable QoS parameters.

3.6.2 Realized Applications

As the ATM technology is intended to support a wide variety of services and applications, it is interesting to examine, which applications were realized, when the ATM Forum's service model was designed.

The applications taken into account can be structurally separated into classes depending on the type of medium they require to be transported, that means whether voice, video, image or data shall be transmitted, and on the interaction pattern between sender and receiver, i.e. whether it is a conversational/interactive, messaging, distribution or retrieval application.

From this classification the requirements for the service model were derived and the following traffic and QoS issues identified [Garr96]:

- CBR/VBR distinction,
- degree of burstiness,
- statistical multiplexing,
- real-time delay constraints,
- delay tolerance in non-real-time applications,
- interactiveness,
- loss tolerance,
- use of leftover bandwidth,
- multiresolution coding,
- QoS consistency,

- fairness.

While for many audio and video applications, which are summarized as *rate-oriented* applications [Garr96], CBR service was considered reasonable since no huge gains by statistical multiplexing may be obtained and real-time constraints can be accommodated quite easily, applications as in the field of traditional computer communications, which are so-called *unit-oriented* applications [Garr96], are often quite bursty and can therefore gain a lot from statistical multiplexing.

It was observed that such unit-oriented applications may have different degrees of delay sensitiveness with emphasis on minimizing the average delay, while the goal for rate-oriented applications was regarded as rather to minimize the tail of the delay distribution.

Special considerations were given to so-called *semi-interactive applications*, which on one side of the conversation treat the communication as interactive and on the other do not, as for example, audio/video messaging or retrieval.

Some fundamental observation were made [Garr96]: Messaging applications are very tolerant to delay; similarly distribution applications can be delayed substantially as well, yet delay variation has to be bounded because buffers are necessary to synchronize the playback at the receiver; less tolerant to delay are retrieval applications which may need a delay bound, but in general a very generous one not comparable to a real-time delay constraint.

It was also noted that with regard to loss the relationship between non-real-time and real-time traffic is exactly the other way around when being compared to delay: computer communications are in general sensitive to loss, while audio and video can cope with some loss during transmission, however their accompanying synchronization information is sensitive to loss, and, if layered (multiresolution) codings are used, some of the video data are more important, and therefore more sensitive to loss than others as well. Furthermore the loss tolerance of real-time traffic was regarded as dependent on the application, as for example video in videoconferencing and entertainment have very different loss rate requirements.

As statistical multiplexing is applied in some of the services, fairness of resource utilization by different connections and consistent QoS across applications were considered important issues, that should also be dealt with in the service model.

The above described traffic and QoS issues were considered the *driving forces* for the development of the ATM Forum's service model, which will be described below in more detail.

3.6.3 ATM's Traffic and QoS Parameters

The parameters specifying the characteristics of a certain connection are divided into traffic and QoS parameters.

Traffic Parameters

Traffic parameters describe the traffic characteristics of a source. They are grouped into a *source traffic descriptor* which is part of a *connection traffic descriptor* (which comprises additionally the Cell Delay Variation Tolerance (CDVT) and a conformance definition that is used to unambiguously specify the conforming cells of the connection).

The connection traffic descriptor together with the desired QoS parameters is the basis for negotiations between application and network. The eventually negotiated characteristics of a connection are called the *traffic contract*.

The following traffic parameters are specified in the ATM Forum service model [ATMF96a]:

- *Peak Cell Rate* (PCR), which is the highest cell rate the source will generate during the duration of the connection,
- *Sustained Cell Rate* (SCR), which is the average cell rate the source will generate during the duration of the connection,
- *Maximum Burst Size* (MBS), which is the maximally expected period of time during for a burst phase of a rate-variable source,
- *Minimum Cell Rate (MCR),* which is a minimum of bandwidth guaranteed to a connection.

QoS Parameters

The QoS parameters are part of the traffic contract and describe the performance guarantees of the network to the application. They c*an be specified individually* for each direction of a connection and must subsequently be negotiated between network and end-systems. Besides negotiated QoS parameters there are also unnegotiated ones, which are set by the network without negotiations [ATMF96a].

The negotiated QoS parameters are:

- *maximum Cell Transfer Delay* (maxCTD), which is the maximum allowable end-to-end transmission delay for a cell to arrive and not to be considered late (and therefore for real-time applications lost),
- *peak-to-peak Cell Delay Variation* (CDV), which corresponds approximately to jitter, and is the difference between the earliest possible arrival of a cell and the latest allowed arrival (maxCTD) of a cell,
- *Cell Loss Rate* (CLR), which is the ratio of lost cells and totally transmitted cells.

The unnegotiated parameters are:

- *Cell Error Ratio* (CER), which is the ratio of errored cells and the sum of successfully transferred and errored cells,
- *Severely Errored Cell Blocks Ratio* (SECBR), which is the ratio of severely errored cell blocks and totally transmitted cell blocks, where a cell block is a sequence of N cells transmitted consecutively,
- *Cell Misinsertion Ratio* (CMR), which is the rate of misinserted cells.

The interpretation of the parameters and which parameters are applicable for a certain service is given by the service categories.

3.6.4 Simple Services for Real-Time and Non-Real-Time: CBR and UBR

As the most important distinction between the service categories the support of real-time or non-real-time services is regarded [Garr96]. The simplest services for theses two categories are CBR for real-time and UBR for non-real-time data transmission.

CBR

The CBR service is a very *simple, reliable, guaranteed channel* [ATMF96a]. It was designed for hard real-time applications, which cannot adapt to QoS violations even for short time peri-

ods. Hence the QoS commitments being given are always met, i.e. this is a *deterministic service* for real-time constrained application.

The allocated rate is defined by PCR and therefore a static amount of bandwidth that is continuously available during the connection lifetime is reserved. Thus this service category is ideal for video and audio which are not very bursty. For bursty audio or video sources, CBR still would be a viable service, yet resource efficiency could be poor depending on the degree of burstiness.

UBR

UBR is a service for non-real-time applications, i.e. applications which have no delay or delay variation constraints to be met [ATMF96a].

No quantitative numerical commitments are being made (not even CLR) and fairness across connections cannot be assumed, since FIFO is used as a service discipline.The PCR is signaled, but serves only informational purposes for the network.

UBR's application field are *traditional computer communications* like file transfer, email, etc. and messaging applications. Both have very bursty traffic characteristics that allow for considerable efficiency gains by statistical multiplexing, and both are not sensitive to delay but may be to loss. Since, as already mentioned, a FIFO service discipline is used in conjunction with large buffers in the network, delay is traded off against loss.

UBR shall hence represent exactly the service model of the current Internet [Garr96].

3.6.5 More Complex Services: rt-VBR, nrt-VBR, ABR

Since CBR and UBR do not completely or efficiently satisfy all features applications might ask for [Garr96], these are tried to be accommodated by the more complex service categories of rt-VBR, nrt-VBR and ABR. The relation between these service categories of the ATM Forum service model are shown in Figure 17.

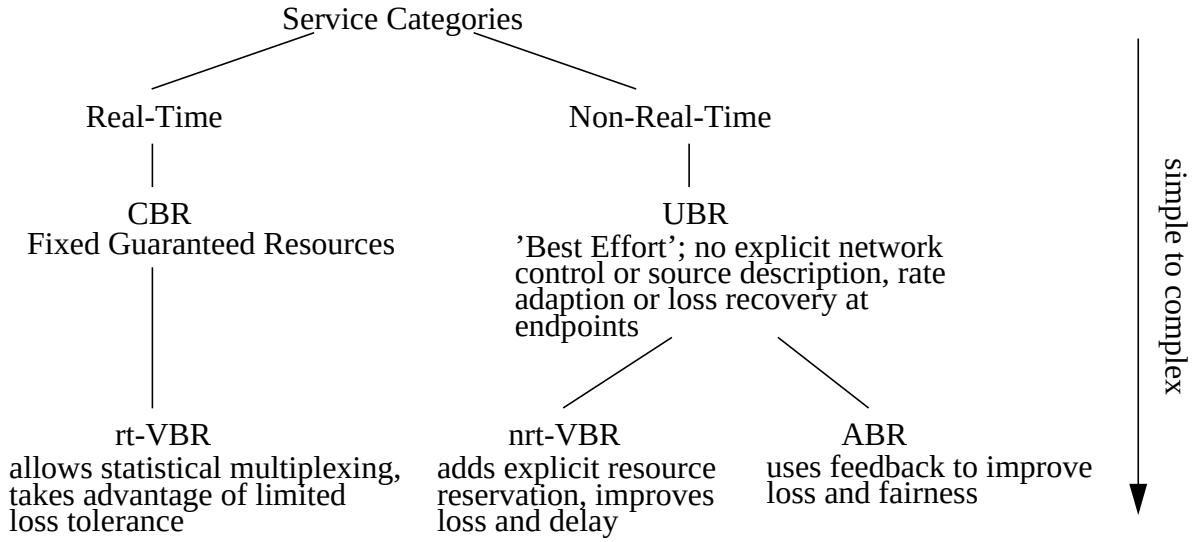


Figure 17: Relation Between the ATM Service Categories.

The rt-VBR service category was designed, as its name says, for real-time applications, i.e applications with delay and delay variation constraints, with a time-varying source rate [ATMF96a].

As it is assumed that some non-real-time applications can benefit substantially by improved fairness, loss, and/or delay, these features, which were not provided by the simple UBR service category, were tried to be accommodated by nrt-VBR and ABR.

rt-VBR

For efficiency reasons some rather bursty real-time applications should employ a VBR service, thereby allowing for some degree of statistical multiplexing between such source-rate variable real-time media streams. Statistical multiplexing introduces of course some loss, which must be accommodated by the coding, so that loss-sensitive applications should not use this service category but rather the reliable CBR service, which of course could waste some of the resources being reserved for a variable source.

For rt-VBR service the SCR, MBS and PCR traffic parameters are used to describe the varying source rate. As QoS parameters maxCTD, peak-to-peak CDV and CLR are supplied.

Sometimes two flavors of rt-VBR are distinguished [Garr96], yet they are not separated in the standards: *PVBR* (Peak-VBR) and *SMVBR* (Statistically Multiplexed VBR).

For PVBR the traffic rate varies, but the QoS is always consistent, because sufficient resources are always guaranteed by allocating resources for the peak rate of the media stream. This scheme leads to the situation, that only lower priority traffic such as UBR and ABR can make use of leftover bandwidth. PVBR can therefore be regarded as CBR with recovery of leftover bandwidth.

SMVBR on the other hand actually reserves less than the peak rate would demand and may therefore lead to loss and delay, which may violate the given QoS guarantees, which thus have no longer a deterministic but rather a statistical nature. However, by this less conservative bandwidth allocation strategy statistical multiplexing also between different SMVBR streams is possible, and should therefore lead to a more efficient use of resources.

nrt-VBR

The nrt-VBR service shall support non-real-time application with bursty traffic characteristics and shall improve the loss and delay characteristics of a UBR connection, which can give no service guarantees about these quantities [ATMF96a].

This is achieved by providing the application with the PCR, SCR and MBS traffic parameters and the QoS parameter CLR. So while the desired loss characteristics can be specified, no delay bounds are given. However by reserving some bandwidth according to the traffic parameters, the delay should not be excessively large, a qualitative statement, which is assumed to be sufficient for delay-sensitive non-real-time applications.

ABR

The ABR service was developed for non-real-time applications, which have no delay or delay variation bounds, but which desire *good loss characteristics and fairness* across all ABR connections operating at the same time [ATMF96a].

It is a service category for which the transfer characteristics may change during the data transfer phase. It is based on a flow control mechanism to adapt source rate in response to changing network characteristics, in order to prevent congestion in the network, and thereby avoiding excessive loss rates.

The goal is to minimize loss and maximize fairness by this rate-based feedback flow control protocol. Fairness is of course provided only to those connections, that really adapt according to the flow control protocol.

ABR does not use a traffic descriptor and is therefore closer to UBR than nrt-VBR [Garr96]. It has no signaled cell loss rate, delay or jitter, but like UBR an informational PCR and an optional MCR, which may not be fallen short of, which means that some reservations are necessary for ABR as well, in case MCR is chosen greater than zero.

To summarize, in Figure 18 the service categories and their valid traffic and QoS parameters are shown.

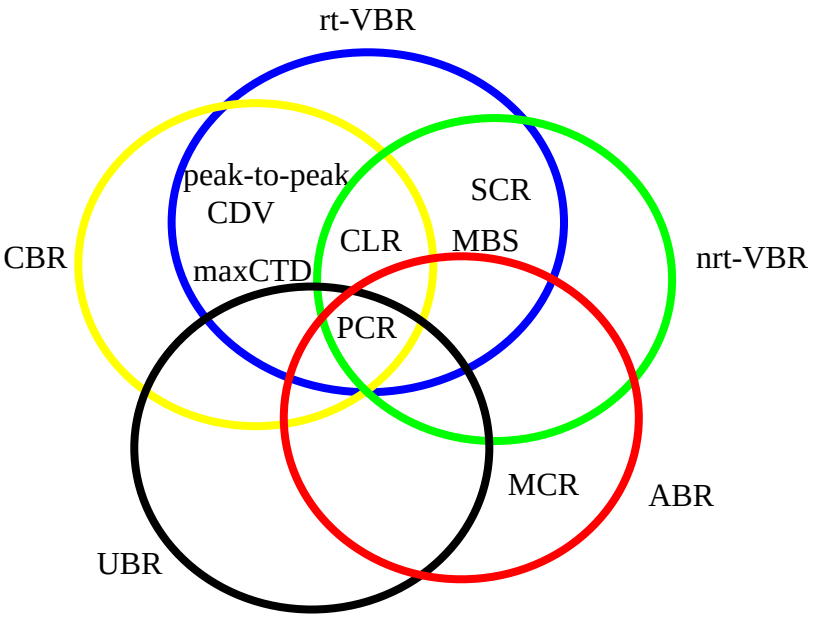


Figure 18: Applicability of Traffic and QoS Parameters with Respect to Service Categories.

3.6.6 Components of the ATM Service Architecture

To support the wide variety of applications which are addressed by the different service classes, an ATM network must provide for appropriately differentiated mechanisms.

The set of mechanisms and techniques for managing and controlling traffic and congestion in ATM networks is called traffic management. It comprises the following functions which shall enable the network to provide the contracted QoS to compliant connections, i.e. connections which send data according to the traffic contract [ATMF96a]:

- *Connection Admission Control (CAC),*
- *Usage Parameter Control (UPC),*
- *Selective Cell Discarding,*
- *Traffic Shaping,*
- *Explicit Forward Congestion Indication (EFCI).*

The primary role of traffic management is to avoid congestion, it further aims at efficient use of network resources.

CAC

The CAC in the ATM Forum service architecture is a set of actions, which determine whether a connection can be admitted or has to be rejected. It is done either at connection setup for Switched Virtual Circuits (SVC) or by the network management during Permanent Virtual Circuit (PVC) establishment. It is a local function, that decides whether an incoming connection request shall be progressed to the next switch, based on the service category, the traffic contract, the required QoS of the new request and on the commitments to existing connections.

If the call is admitted then the required resources are allocated by the CAC.

UPC

The UPC's purpose is to control and monitor traffic as generated by the user in order to protect network resources from malicious as well as unintentional misbehavior, which could degrade the QoS of other connections [ATMF96a]. Hence the UPC function is to support the QoS objectives of compliant connections. Whenever a traffic contract is violated, policing actions are executed on the cells which were sent in excess to the contracted traffic parameter values. These policing actions taken by the UPC may be *cell tagging* or *cell discarding*. Cell tagging means to indicate that the cell may be discarded in case of a congestion (by setting the CLP bit) since it is non-conforming, whereas cell discarding is the ultimative action of not letting the cell pass to the next switch.

Selective Cell Discard

A network element may selectively discard cells which either do belong to a *non-compliant connection* (which will be detected by UPC as described above) or which have been marked with the CLP (Cell Loss Priority) bit set to 1. The CLP bit may be set either by the user or by the network as for example as part of the UPC policing actions. When set by the user these cells will usually be part of less essential data which may tolerate some loss, thereby allowing critical data to be transmitted reliably. This technique shall give the network a hint which cells to discard in case of congestion.

Another related technique called frame discard discovers when one cell of a frame is discarded and subsequently discards all other cells of this frame (a data unit at the AAL layer), in order to avoid waste of resources by already known useless cells.

Traffic Shaping

Traffic shaping changes the characteristics of a stream of cells of a connection, in order to increase network efficiency while maintaining the QoS objectives, or to ensure conformance of a stream on the following interface [ATMF96a].

Example techniques of traffic shaping are peak cell rate reduction or burst length limiting.

EFCI

A network element, which is congested or endangered to be congested soon, may set the EFCI bit in the cell header, so that an end-system, which receives this cell, can react to this situation and possibly lower its data rate by using a protocol, that adapts to congestion or impending congestion situations.

4 Resource Reservation and Signaling Protocols

In this section we will concentrate on communication system aspects of QoS provisioning. The QoS mechanisms at the communication system level have to be realized in a distributed fashion, i.e. as protocols. We therefore review the most prominent reservation and signaling protocols which were designed for QoS provision by the communication system.

First we discuss general issues of reservation protocols. Then, before we regard the Internet community's currently favorite reservation protocol RSVP, we describe the ST-2 protocol and its successor ST-2+. These protocols have been developed before the advent of RSVP and originated also from the IETF. But they are no longer considered the first choice for a reservation protocol in the Internet. Lastly, we treat the relevant parts of the ATM signaling procedures for QoS provision.

4.1 General Issues of Reservation Protocols

4.1.1 Reservation of Resources

The decision whether an incoming reservation can be accepted is made within the admission control module of the resource management. The resource management must internally maintain the overall state of the node's resources as well as store sufficient information of the original requests to be able to release and modify reservations later (on request, in case of preemption, or in case of an unannounced breakdown). While requests dealing with only one data stream can be handled by the admission control of the network node, the interaction of more than one stream is subject to policy control where precedence among streams are considered, e.g., streams may be preempted in presence of resource requests of higher priority.

The node must map the functions of the supported reservation protocols onto the internal representation of the network nodes resource management system. Furthermore, it needs such functions as activity monitoring, extracting or adding information from the routing module or adding, removing an end-system to an active stream or modifying its characteristics. Parts of these are protocol-dependent functions, e.g., adding an end-system to a stream.

Resource reservation and scheduling in general are protocol-independent activities. Thus, it could be considered to support various reservation protocols within a node. However, workload and reservation models influence the approaches for service, reservation and schedulability. Since reservation protocols are tied somehow to such models, the support of varying reservation protocols is only partially possible, best if they employ similar models.

4.1.2 Reservation Styles

Reservation is applicable in many areas of communication. Depending on the particular application, the reserved stream may involve two or more parties which can be organized in one of the following approaches:

- *Single Sender / Single Receiver (Unicast)*
- *Single Sender / Multiple Receivers (1:n Multicast)*
- *Multiple Senders / Multiple Receivers (m:n Multicast)*

The latter two approaches are combined with the notion of group concept. The single 1:n multicast leads to a multicast tree. If m different systems want to send data, m different multicast trees must be established, leading to considerable effort in the network nodes and communication paths. If now only a few of these m sources actually transmits data, say k, only resources for k instead of m trees are needed. Hence, the m:n multicast allows for a better sharing of resources. A similar sharing could be achieved by a stream grouping scheme, where during the tree establishment the membership within a particular group must be announced.

A major difference among current reservation protocols is the *direction in which the reservation occurs* [DHVW93]. This reservation direction can be *sender-oriented, receiver-oriented or neither.* Sender-oriented means that the sender of the data is the entity that initiates the reservation setup. Therefore, the sender must know the target addresses for the reservation setup. In contrast to this, in the receiver-oriented approach the receivers must know the address of the data source; the sender has no knowledge of the participating receivers which might be useful for some applications. Furthermore, information about the resources that are required for the data transmission must be given to the receivers to enable them to perform adequate reservations. Finally, the reservation might be considered as a network management issue done by a third party operating as a mediator between the senders and receivers; this entity must be informed about senders, receivers and QoS specifications. This is a rather uncommon approach.

Sender-oriented reservation can lead to substantial management workload at the sender if a large number of receivers participate in the transmission and generate control messages to be processed by the sender such as ’join’ or ’leave’ operations. This problem restricts the *scalability* of sender-oriented reservations to small-to-medium size. In the receiver-oriented approach only in a few cases the reservation request shall actually be transferred as far as to the sender(s) [MESZ94].

As a solution to the sender-oriented scalability limitation, sender- and receiver-oriented reservations can be combined, e.g., the reservation starts with a reservation for some receivers, later, additional receivers who want to join the transmission can be added at routers without involvement of the sender [DHHS94].

A reservation protocol can be tightly coupled to a specific protocol used for data transmission and require that reservation setup precedes data transmission. Alternatively, the transmission may be independent of reservation setup. The former approach is a typical connection-oriented communication. The latter approach promises more flexibility, yet, it must nevertheless be possible to extract information from the packets which can then be used to determine the reserved resources for scheduling purposes. This approach is followed by RSVP which exploits information from IP packets, yet, IP can also be used without any reservation made by RSVP.

4.1.3 QoS Driven Routing

Resource reservation protocols must be supported by QoS driven routing algorithms to allow for the establishment of connections with QoS guarantees. QoS driven routing algorithms focus on the problem of searching optimal paths for a given set of QoS requirements in a meshed network of resources. The selection of paths for multimedia streams have been left by existing network and resource reservation protocols to specific routing mechanisms. The core of QoS driven routing is the provision of one or more suitable paths to a given target consider-

ing a given QoS requirement. To improve the quality of routing for multimedia streams any QoS driven routing algorithm should consider:

- Current loads of resources
- QoS requirements of the connections
- Resource load after the routing decision

Local and network resource managers can reject connections because of resource overloads. Therefore it is required to consider the actual loads of resources. The routing should also regard the QoS required by the connection to find a route best-suited for this QoS. For instance, a video playback application accepts a higher delay than an video conferencing application but the bandwidth requirement is larger. Third, a QoS driven routing algorithm should consider the load of resources after the connection establishment. For example, routes should be preferred if the acceptance of the new stream does not consume the majority of resources on this route.

QoS routing is currently still in its infancy. The necessity of such methods, the principle ability to perform QoS routing, and the proposed approaches are under highly controversial discussion at the moment.

4.2 ST-2 and ST-2+

The Stream Protocol Version 2 [Topo90] is a connection-oriented protocol designed to co-exist with IP. ST-2 defines a sender-oriented, unreliable multicast protocol that provides QoS negotiation and resource reservation in all intermediate network nodes.

ST-2 defines actually two closely associated protocols: ST for the data transfer and SCMP for control operations such as connection setup and maintenance. ST mainly supports the unidirectional delivery of data from the initiator of the connection to all targets of the connection.

4.2.1 Basic Operation Model

At connection setup time, the QoS is negotiated among the initiator of the connection, all intermediate network nodes and the targets of the communication as illustrated in Figure 19. The

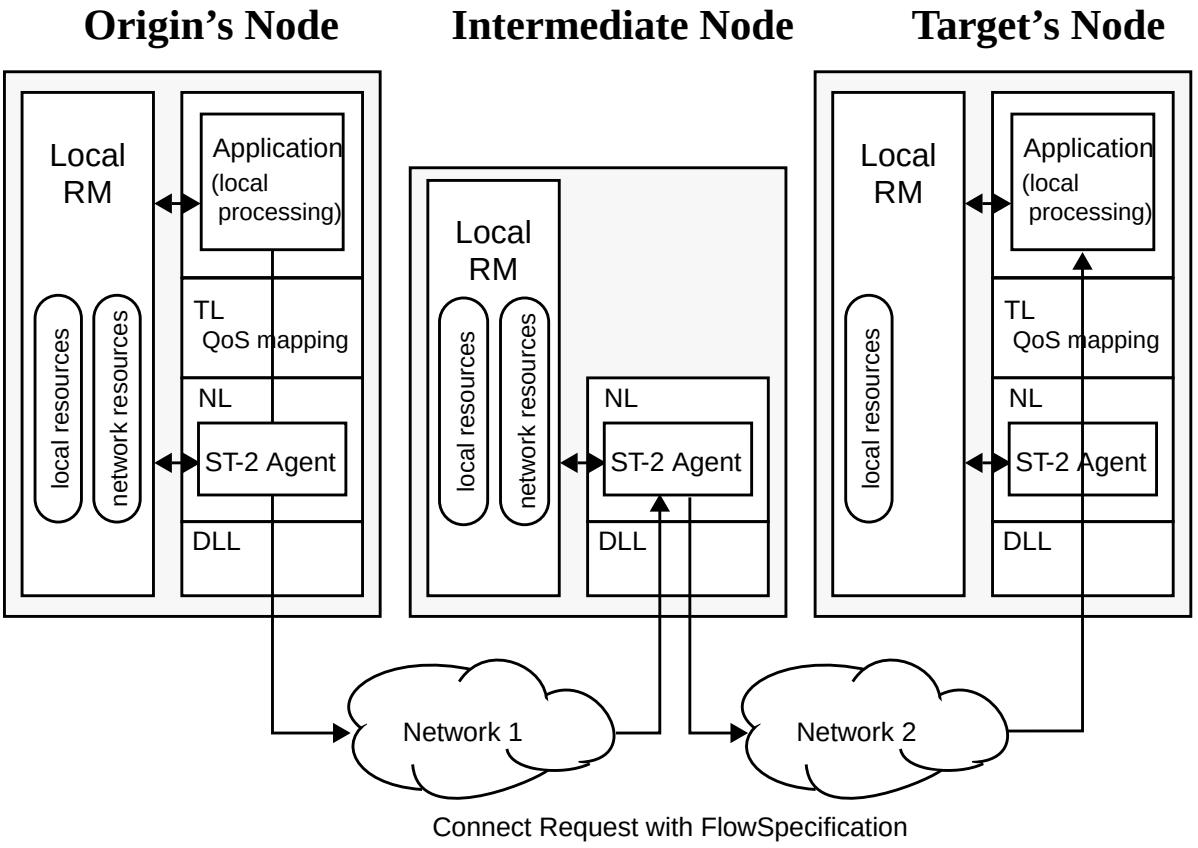


Figure 19: Distributed QoS Computation in ST-2.

connection, called 'stream', is established by the SCMP `CONNECT` message that is issued by the sender, also called origin, and that specifies all initial targets and a 'FlowSpec' which describes the requested QoS. This QoS request is based on the application-layer QoS requested, possibly mapped by the transport layer on a QoS request in terms of network layer units due to packet segmentation. This connection establishment message is sent hop-by-hop on the paths to the targets. Each router being part of the path towards a target receives the `CONNECT`. Its local resource manager ("local RM") checks the available resources at the node,

computes the QoS its resources can provide and reserves the corresponding resource capacities.

If the reservation fails (either due to resource overload or insufficient resources for a given QoS requirement), an SCMP `REFUSE` message is sent back to the origin releasing all reservations made so far. Otherwise, the ST-2 agent updates the FlowSpec (e.g. keeping track of the accumulated delay) and passes the connection establishment message downstream.

On each branch of the routing tree, the final FlowSpec, i.e., the end-to-end QoS of the branch, is communicated to the target. The target may base its accept/reject decision on this FlowSpec. If it accepts the connection, the FlowSpec is propagated back to the origin. The origin can then compute the QoS of a multiple target connection from the QoS returned from the individual branches.

After the connection has been established, the origin may send a CHANGE message to the targets modifying the QoS guarantees and thus adjusting the reservations. This is especially useful when the QoS computed in the connection establishment phase exceeds the QoS requirements of the origin. Here one could, e.g., lower the throughput of the individual resources or loosen the delay bounds.

The multicast tree can be modified dynamically, a target can leave the tree or the sender may drop a target; it is also possible to add more targets to the multicast tree, yet, in ST-2 this can only be performed by the sender.

When finally a connection is terminated, the corresponding resource reservations are released.

The specification of ST-2 in RFC1190 does not determine how resource managers compute the QoS and make reservations and how resources are scheduled according to these reservations. Furthermore, ST-2 provides only a preliminary definition of the FlowSpec which can be altered. However, the protocol assumes these mechanisms and data structures as its basis. A concrete example for such mechanisms are the HeiRAT [VHN93], [VWHW97] components developed as part of the HeiTS system[WM91].

4.2.2 ST-2+

The follow-on version ST-2+ [DB95] simplifies the overall structure and adds several extensions to the basic ST-2 protocol. The most important extension is the mechanism for receiver-oriented stream joining. To join, a candidate receiver sends a `JOIN` message in the general direction to the origin (using reverse routing). This `JOIN` message reaches a router already participating in the stream. Here a distinction must be made depending on the particular stream: the origin of a stream specifies whether joining at routers is prohibited, needs a notification to the origin, or is free-to-all. In the free-to-all case, the router just adds the requester as new target to the stream and forwards any future data to it as well. In the second case, the router adds the target and sends a notification to the origin of the stream. Finally, the join at an intermediate system may be prohibited which means that all requests have to travel to the origin of the stream and are handled there as in ST-2.

To provide different QoS to single targets, a filtering approach as discussed in Section 5.1.2 could be used; yet, this is not defined in the basic protocol specification.

The ST-2+ specification defines several FlowSpecs, the mandatory FlowSpec provides for QoS definition by precedence, QoS class, and QoS parameters. The precedence of a stream is defined as its priority in comparison with other streams. If a sender with a high precedence

value makes use of this value at stream setup or stream change time, and if the resources in a hop are not sufficient to support both this stream and also one of a lower precedence, then the stream of lower precedence can be preempted (partially torn down).

The QoS classes distinguishes among ‘predictive’ and ‘guaranteed’ reservations. For a predictive reservation, intermediate hops reserve sufficient resources to guarantee data delivery for the average data rate. For guaranteed reservation requests, intermediate hops reserve sufficient resources to guarantee timely packet forwarding in all cases, but the sender is never allowed to exceed these limits.

The QoS parameters provided by the initiator of the stream specify both optimal and acceptable values for the message size, the message rate and the end-to-end delay.

The notion of groups of streams has been defined, yet, the specification for actual group mechanisms is only limited. Groups can share bandwidth, routes, subnetwork resources or ‘fate’ (which means that if one stream is preempted in a node, all other should also be terminated). For the setup of a group, a stream may provide a group id in the connect message. If this group has already been established, the new stream joins this group, otherwise a new group is created. If streams are in the same group, only some more resources have to be reserved in addition to the resources that have already been allocated for the other streams of this group. It can also be requested that all streams of a group will share their routes between senders and receivers as far as possible. This is especially useful to reserve resources for various streams of a conferencing application that have common maximum resource requirements and for which path sharing, partially in reverse directions of sender and receiver, is beneficial.

4.3 The RSVP Protocol

RSVP has been and still is being designed as the Internet's resource reservation protocol to support integrated services [BZBHJ96]. However, it is general enough to be employed in other environments as well. For example, in [NLP96] an architecture for a multicast service over ATM using RSVP as QoS negotiation protocol has been proposed.

4.3.1 Design Goals, Principles and Properties

By the use of the RSVP protocol hosts are enabled to request a specific QoS from the network. RSVP propagates the QoS request to all the routers along the path and additionally maintains state information within routers and hosts to provide for the requested service. It can therefore be regarded as a *state establishment and maintenance protocol* [ZBEHJ93].

The protocol is placed on top of IP, thus not requiring any routing mechanisms in RSVP itself, but using unicast and multicast routing mechanisms provided by the network layer. RSVP does not transfer application data but operates as a control protocol only.

From the beginning the protocol was designed with *group communication* in mind, and so many design objectives are due to situations arising in multicast data transfers.

The designers of RSVP had several goals in mind [ZBEHJ93]:

1. *heterogeneous receivers,*
2. *dynamic membership,*
3. *aggregation of resource reservations,*
4. *channel changing feature,*
5. *adaptation to network dynamics,*
6. *tunable and controllable protocol overhead,*
7. *independence of other architectural components.*

Before regarding the operation of RSVP, these design goals and the underlying principles and properties of RSVP will be studied.

Heterogeneous Receivers

As the RSVP protocol claims to be scalable to large multicast groups, this consequently leads to the problem of handling heterogeneous receivers, since in a wide area internetwork, such as the Internet, receiving hosts as well as the paths used to reach these hosts can have very different properties from one another. Therefore a major design goal of RSVP is to accommodate heterogeneous receivers.

This shall be achieved by using a *receiver-initiated reservation style*, since the receivers know best about their QoS requirements, and should hence be made responsible for initiating and keeping the reservation active as long as they want to receive data.

Dynamic Membership

The presence of multiple receivers raises another issue: there will be receivers joining and leaving the multicast session at any time during the session. This necessitates a mechanism for dealing gracefully with dynamic membership, particularly in the case of large multicast groups, as the dynamics increase with multicast group size.

This issue again is tried to be tackled by the receiver-oriented approach and by the fact that the data transfer is handled separately from the control by RSVP, thereby enabling receivers to join and leave the QoS distribution tree installed by RSVP at any time during the data transmission. RSVP is therefore no call setup protocol, which makes it fit more neatly into the connectionless datagram service of the Internet.

Aggregation of Resource Reservations

A multicast group, e.g., used for large-scale video-conferencing, may often have multiple senders. Therefore, a desirable characteristic for a resource reservation protocol is that resource reservations for multiple senders are shared and that there should not be a different distribution tree for every sender if the group communication pattern allows only for a limited set of senders to send simultaneously.

This facility is tried to be accommodated in RSVP by introducing different reservation styles that allow to specify to which extent intermediate switches should aggregate reservation requests from receivers in the same multicast group. Three *reservation styles* have so far been implemented in the RSVP protocol [BZBHJ96]:

- *Wildcard-Filter*: a single resource allocation is made for all traffic directed to the receiver which initiated the resource reservation, thus allowing for one sender at a time only.
- *Fixed Filter*: the receiver can specify a set of sources for each of which a certain amount of resources is reserved, thus allowing for a fixed set of simultaneously transmitting senders.
- *Shared-Explicit Filter*: in contrast to the fixed filter reservation style the shared-explicit filter style shares one reservation between a specified set of senders.

Another reservation style that has been proposed [ZBEHJ93], but has not yet been implemented, is the *dynamic filter style*, which is similar to the fixed-filter style in that it reserves multiple channels, however allows for changing the set of senders dynamically during the session.

Channel Changing Feature

By implementing the dynamic filter reservation style another design goal of RSVP would be met: the channel changing feature. This means the ability to dynamically switch between senders, which could of course be met by reserving resources for all senders, yet this is inefficient if the receiver only ‘listens’ to a limited number of senders and discards the data of all other senders. This discarding should better be done inside the network, which is exactly the reason for having dynamic filter style reservations.

Adaptation to Network Dynamics

RSVP is not a routing protocol and merely uses the services of a given unicast or multicast routing protocol. This way, network dynamics can lead to a situation where data is transferred on routes for which no reservation has been made. These dynamics can be caused by either network failures or route changes. RSVP’s remedy to recover from situations where data transfer takes place over ‘un-reserved’ routes is in principle the soft-state approach to resource reservation, which will ultimately lead to the setup of reservations along the new route when the refreshing of resource reservations takes place. However this might lead to an unacceptably long period of possibly degraded service quality. Hence in the case of route failures a *local*

repair mechanism triggered by the routing protocol to recover from the route failure was proposed [BZBHJ96]. In the case of a route change the situation is a bit curious, since the routing protocol finds a better route than the one previously being used, and yet the service quality might be decreased by using this new 'un-reserved' route. A proposed solution to this problem is to use a technique called *route pinning* that allows to fix a route during the lifetime of a flow [ZB96] which basically fixes the soft state along the path and leads essentially to connection-oriented communication. Route pinning has been discussed several times within the RSVP working group and is currently not provided. Therefore, a hard guarantee that QoS can be provided cannot be given by RSVP.

Tunable and Controllable Protocol Overhead

A more technical design goal is to keep the protocol overhead tunable and controllable. This shall be achieved by the *refresh period parameter* in the protocol, which controls how often the soft-state in the routers and hosts has to be refreshed. The overhead incurred by the refresh messages has to be weighed against the accurateness and actuality of the state information.

Independence of other Architectural Components

The last design goal is a rather general matter of modular design: RSVP shall be designed independently of other architectural components like flow specification, admission control, packet classification, packet scheduling and routing. This shall ensure the independent evolution of these components. However, particularly in the case of routing this design trades off optimality and efficiency against engineering principles, since the choice of a route can depend on the quality of service being requested and should therefore be integrated with resource reservation.

4.3.2 RSVP Operation

In RSVP a data stream is modeled as a simplex distribution tree rooted at the source and extending to all receivers. A source application, of which there can be many, begins participating in a group by sending a path message containing a flow specification to the destination address, be it unicast or multicast. The `PATH` message serves two purposes:

- to distribute the flow specification to the receivers,
- to establish path state in intermediate RSVP agents to be used in propagating reservation requests toward specific routes.

RSVP does not restrict a source from transmitting data even when no receiver has installed a reservation to it, however service guarantees are not enforced. Also, there may be some best-effort receivers, while other receivers may use reserved resources for 'better' services as for example guaranteed delay.

Before establishing a reservation, each receiver must first join the associated multicast group in order to begin receiving path messages, yet this is a function of the multicast routing protocol and therefore outside the scope of RSVP.

Each receiver may use information from path messages and any local knowledge (computing resources available, application requirements, cost constraints) to determine its QoS requirements. It is then responsible for initiating its own reservation. For that, it generates a

RESV message which travels towards the sender along the reverse path of the PATH message. This can be done because the intermediate RSVP agents, which reserve network resources along the subnet leading toward the receiver, can use the established path state to propagate the reservation request toward the group sender(s). Reservation message propagation ends as soon as the reservation encounters an existing distribution tree with sufficient resources allocated to meet the requested QoS, i.e. until the reservation request can be merged into an existing reservation. As already mentioned, this receiver-initiated reservation style shall enable RSVP to accommodate heterogeneous receiver requirements and by the merging process it shall be made scalable for use in wide-area networks such as the Internet.

The RESV message contains information about the desired QoS and also a filter specification and determines in which out of the three above described styles the reservation is made. On their way up to the sender, reservation requests have to pass local admission control tests in the routers lying on their path. If the reservation is too demanding for one of these intermediate systems, it is rejected and the receiver that issued the reservation request, obtains an indication of the reservation failure. This is essentially a one-pass or unilateral method of negotiation of the service characteristics, however it is enhanced in the RSVP protocol by a mechanism called advertising.

The overall approach to QoS negotiation in RSVP is called *One-Pass With Advertising* (OPWA) [SB95]. Sources of data flows periodically send so-called advertisement messages which are actually contained in the path messages of the sender as *ADSPEC* objects. These are used to advertise (beforehand) the end-to-end service that would result from any given service request. On their way down to the (potential) receivers the advertisement messages accumulate information about quantities such as delay, bandwidth and hop count in each router on the path for several categories of service, thereby giving the receivers an idea of what kind of service level they might expect to be successful in admission control tests. Thus the receivers task of forming reasonable reservation requests is simplified by the OPWA mechanism.

Since RSVP sends the path and reservation messages periodically it maintains soft state in the intermediate nodes. While path refreshes serve the automatic adaptation to changes in the multicast distribution tree and install path state in any new branches of the tree, reservation refreshes maintain established reservations and incorporate changed receiver reservations thereby accommodating for dynamic QoS changes, i.e. changes of the service characteristics during data transfer. It has to be observed that RSVP is no call setup protocol because reservation requests are issued in parallel to the data transfer, and can hence be made at any time during the data transfer phase.

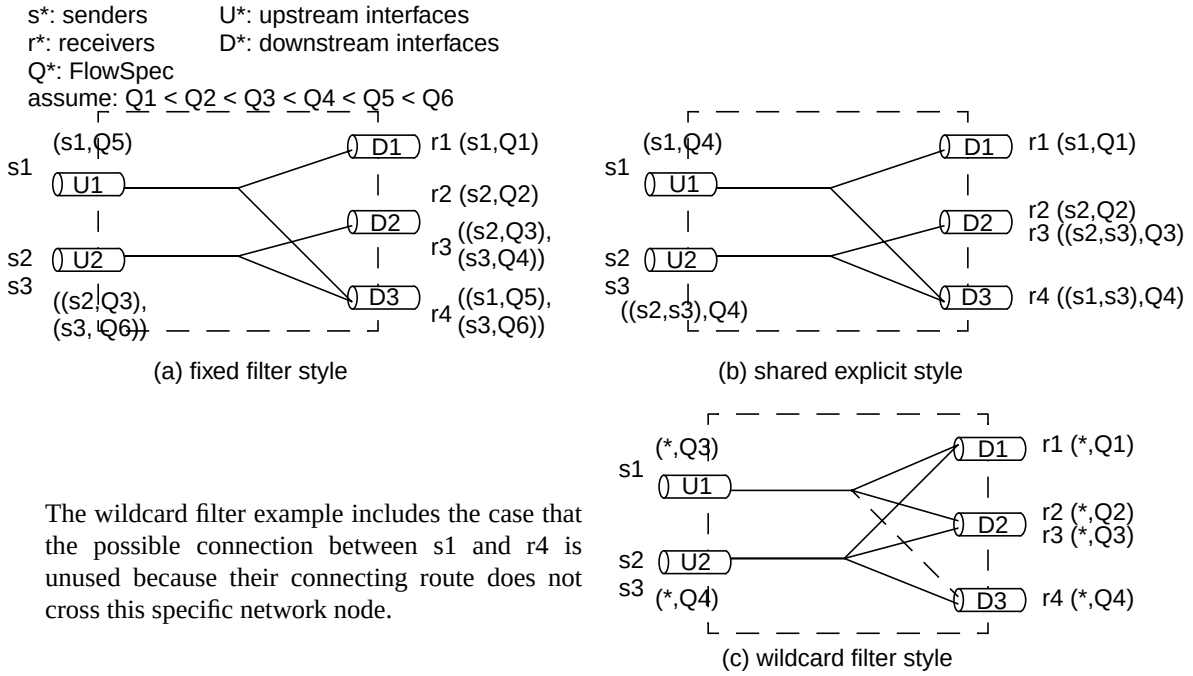
This refresh based mechanism allows orphaned reservations and state to be automatically timed out and recovered. However the proper choice of the refresh interval is still an open matter. This choice affects, of course, on the one hand the protocol overhead and on the other hand the responsiveness of the protocol regarding network dynamics and changing receiver requirements.

4.3.3 RSVP Filters at Work

As described above, the following three reservation styles are defined:

- the fixed filter style, which can address a fixed number of senders and contains a specific FlowSpec for each sender,

- the shared explicit style, which can address a fixed number of senders and contains a FlowSpec for all of these senders,
- the wildcard filter style, which address all senders for a flow and contains a FlowSpec that is the same for all of these senders.



The wildcard filter example includes the case that the possible connection between s1 and r4 is unused because their connecting route does not cross this specific network node.

Figure 20: RSVP Reservation Styles.

Figure 20 shows the results of the three different filter styles on the establishment of filters in a node, and the forwarding of FlowSpecs based on the merging of FlowSpecs in the node. Each service that is supported by RSVP for its reservation protocol must specify a function that allows a network node to compute a FlowSpec that is larger than both of two given FlowSpecs. In an imaginary service that defines its QoS exclusively by throughput, for example, a simple comparison operation “greater than” can be applied. For a service that defines its QoS by a pair (throughput, delay), a FlowSpec created by merging two FlowSpecs (t1,d1) and (t2,d2) would be (max(t1,t2),min(d1,d2)). In the example, we assume a total order Q1 < ... < Q6 for simplicity.

If the fixed filter style is applied (Figure 20 a), each receiver specifies a FlowSpec for each sender from which it wants to receive data. In the network node, the upstream interface for each sender is known (because of the PATH message). The newly arriving FlowSpecs along with all established FlowSpecs for the same sender are compared and a new FlowSpec that is greater than all individual FlowSpecs is computed. Assume that in the case of (a), the reservations for r1, r2 and r4 have been established. The forwarded FlowSpecs are (s1,Q5) on the interface D1 and ((s2,Q2),(s3,Q6)) on the interface D2. The arrival of the reservation request ((s2,Q3),(s3,Q4)) from r3 does not involve the interface U1, but the FlowSpec to U2 and the filters at D2 have to be recomputed. The filter at D2 must be opened to allow packets according to the less restrictive FlowSpec Q3 to pass, and it must be opened for packets from sender s3. Two new FlowSpecs must be calculated for the interface U2 to s2 and s3. The flow requested from s3 by r4 is “greater” than that requested by r3, so no change is necessary in that case. The flow

requested from s2 by r3 is the new “greatest” flow, so the upstream information must be updated by forwarding a RESV message with the FlowSpec (s2,Q3) on U2.

In the shared explicit style case (Figure 20 b), a receiver can specify a single FlowSpec for all senders that it is interested in. As in the fixed filter style, the downstream filter is calculated from the “greatest” FlowSpec of all FlowSpec that were requested on that downstream interface. Assume that in (b), the reservations for r1, r2 and r3 are already established. The filters at the downstream interfaces are set according to Q1 for the interface D1, to Q3 for the interface D2, and closed for the third interface. When the RESV message with the reservation request ((s1,s3),Q4) arrives at the interface D3, the downstream filter at that interface is set to Q4. The FlowSpecs at the upstream interfaces are calculated, and Q4 is computed as being “greater” than all previous FlowSpecs. The RESV message (s1,Q4) is forwarded on the interface U1 and the RESV message ((s2,s3),Q4) is forwarded on the interface U2.

In the wildcard filter style case (Figure 20 c), no sender information is included in a RESV message. Instead, the maximum of all applicable FlowSpecs is forwarded on the upstream interfaces. It could be falsely concluded that this results in identical FlowSpecs on all upstream interfaces of a node. The example shows that this is not necessarily so. Assume that r4 receives the data from s1 on a route that does not cross the node presented in the example. In that case, the RESV message that carries the FlowSpec from r4 is not included in the computation of the maximum FlowSpecs which is forwarded on interface U1 which leads only to s1. Since Q4 is the “greatest” of all FlowSpecs, the FlowSpecs that are forwarded on U1 and U2 are different.

Obviously, the amount of information that needs to be stored in the routers decreases from the first to the last style. Generally, network nodes are required to store a large amount of dynamic information and data about RSVP flows:

- the (running) timers and counters for each receiver of each flow which indicate when a reservation expires and implicit teardown must be initiated,
- the FlowSpecs of all receivers along with the incoming interface to calculate changes in a receiver’s FlowSpec or to release resources in case of timeout or teardown,
- the merged FlowSpecs for each outgoing interface to identify when a change in a FlowSpec has an effect on an upstream network node,
- the filtering information which consists of a list of requested packet origins (by sender and port) for each receiver by which a source selection within a flow is supported.

4.4 ATM Signaling: QoS Negotiation

In this section, the parts of ATM signaling described in the UNI SIG 4.0 [ATMF96b], PNNI [ATMF96c] and B-ICI [ATMF96d] specifications (which are all based on ITU-T's recommendation Q.2931 [ITU96c]), which are relevant for QoS parameter negotiations during a connection setup are discussed. After a brief view on the ATM signaling mechanisms for connection set up and an overview of the generic process of negotiation between the end-system and the ATM network are presented.

4.4.1 ATM Signalling During Connection Setup

In Figure 21 the usual ATM signaling procedure to set up a connection is depicted.

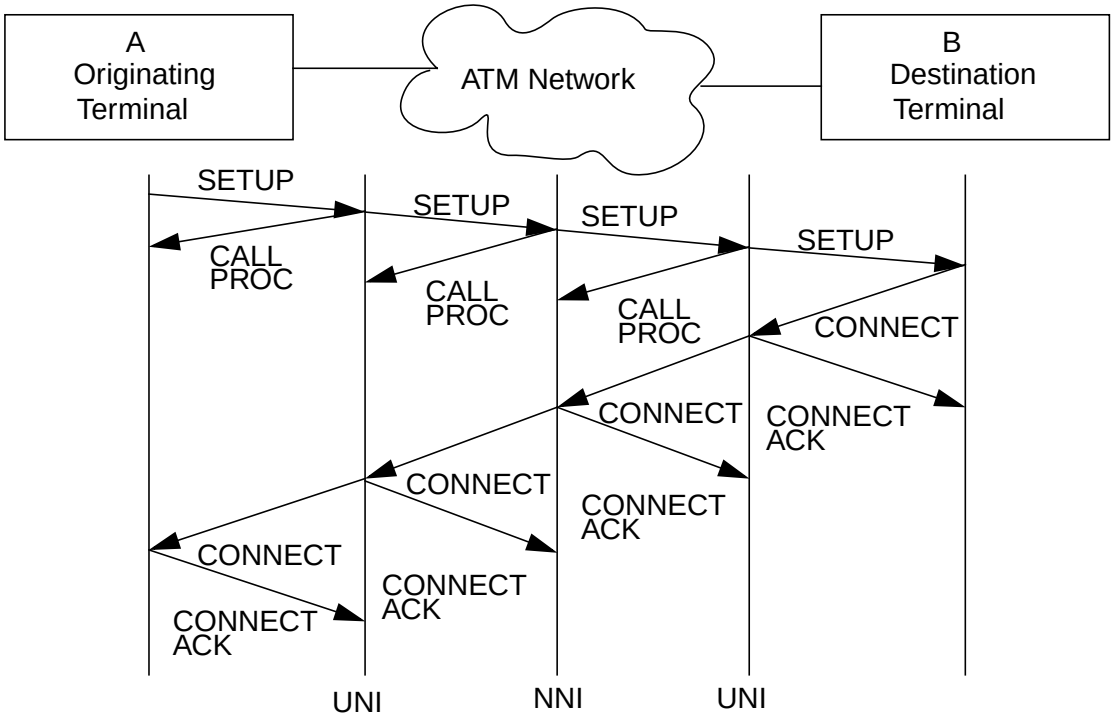


Figure 21: ATM Connection Setup Procedure.

If the end-system A wants to setup a connection to the end-system B it sends a SETUP message to its User Network Interface (UNI) essentially containing the ATM address of B, the desired traffic characteristics/parameters and the desired QoS parameters. Upon receiving the SETUP message, the UNI returns a CALL PROCEEDING message to A and forwards the SETUP to either the UNI servicing end-system B or, if multiple networks are interconnected, to the Network-Network-Interface (NNI), which connects the network of the UNI servicing A to the network containing the UNI servicing B (of course there could be several connected networks between A and B). On every interface lying on the path from A to B this process of forwarding SETUP messages and backwarding CALL PROCEEDING messages is repeated and in addition resources are allocated, and VC tables are built according to the information elements being passed with the SETUP messages. Eventually the SETUP message arrives at the

called user B, which, if it admits the call sends back a CONNECT message to its UNI, which acknowledges by a CONNECT ACKNOWLEDGE message. The CONNECT message travels back the whole way the SETUP message was sent and triggers on its way back several CONNECT ACKNOWLEDGE messages on every interface lying on this path. When the CONNECT message finally arrives, the signalling phase is completed and the data transfer phase can begin.

After this brief presentation of the protocol mechanisms when setting up an ATM connection, a deeper discussion of the negotiation process between the end-systems and the network regarding the characteristics of the connection, i.e. its traffic and QoS parameters, will follow.

4.4.2 Overview of the Negotiation Process

The purpose of the negotiation is to select a value of each parameter that can on the one hand be provided by the network and on the other hand meets the needs of both end-systems.

Since call establishment is accomplished in one round trip (i.e. there are two relevant messages), negotiations must be tied to a single round trip exchange.

At the highest level, the calling end-system proposes a requested value and an acceptable value. The network tries to meet the requested value, or at least some value between the requested and the acceptable value, and if it cannot meet the acceptable value, it clears the call. Otherwise, it offers the call to the called end-system with the better of the requested value and the best value it can support.

The Negotiation Object: Traffic and QoS Parameters

The protocol specifies a range of values for the respective parameters. Typically, the range is integer valued (although real or enumerated types are possible). In some cases, the range may be a choice of two or more discrete values. One end of the range is designated as the most desirable value, and the other is the least desirable value. If the most desirable value is greater than the least desirable, the negotiation proceeds downwards, i.e., decreases in value. Otherwise negotiation is upwards, i.e. increases in value. For example a high cell rate is considered more desirable than a lower one and hence negotiation proceeds downwards, while a low CTD is more desirable than a higher one, so negotiation proceeds upwards.

The protocol also determines whether the objective of the negotiation is to *‘meet’* or to *‘meet or exceed’* the parameter value indicated by the calling end-system. If the objective is to meet the value, it is usually because the parameter relates to a resource (e.g. bandwidth) that can be divided with relatively fine granularity among calls, with a penalty to the network attached to giving the connection a greater value to the connection than that indicated. If the objective is to ‘meet or exceed’, the network may have alternative values available, not necessarily over a continuous range, and it cannot benefit from giving the connection exactly the value indicated rather than the next most desirable value. For example peak cell rate is a ‘meet’ parameter, since it can be divided fairly precisely among connections, if there is 10 Mbit/s of capacity available and the connections needs 3 Mbit/s, then that quantity will be assigned and the remainder will be available for other connections. Delay on the other hand is a ‘meet or exceed’ parameter: if a connection needs 50 ms and routes are available that offer 20, 40 and 60 ms, respectively, then the 20 ms or 40 ms route will be selected (all other things being equal), and there will be no reason to artificially prolong the delay to 50 ms.

A parameter is characterized as either a *metric* or an *attribute*. A metric is a parameter that requires the contribution of each topological component (i.e. link or node) to be accumulated to determine the value of the parameter for the connection. An attribute considers each topological component individually, such that the least desirable contribution of one or more links constitutes the parameter value for the connection. For example, delay is a metric, while peak cell rate is an attribute. In general, path calculation is facilitated by minimizing the number of metrics that can apply to the connection.

The Negotiation Algorithm

The negotiation for each of the traffic and QoS parameters is again described only generically, for the details see [ATMF96b][ATMF96c].

In the initial call establishment message, the SETUP message, the calling system may indicate a requested value, which is the value of the parameter that it would like to have, and also indicates either a highest acceptable value (for upward negotiation) or a lowest acceptable value (for downward negotiation) for each of the parameters. The latter value is the worst (either highest or lowest) value of the parameter that the calling end-system is willing to accept, that means, it would rather have the call being rejected than to have it admitted with a worse value. If the requested value is not specified (either because the calling end-system chooses not to include it, or because it is not supported in the signaling protocol), it is assumed to be the same as the highest or lowest acceptable value.

The switch servicing the calling end-system makes a preliminary determination as to whether it can meet the requested or at least the highest or lowest acceptable value for each of the parameters.This might involve for example tests against administrative limits or similar checks against fundamental restrictions for the necessary QoS provision. If the network cannot satisfy at least the highest or lowest acceptable value, it rejects the call. These determinations may also be repeated at administrative boundaries between networks, for example at B-ICI interfaces or interfaces between public and private networks.

If the regarded parameter to be negotiated is an attribute, the SETUP message forwarded by the interfaces (either UNI or NNI) carries an indicated value of the parameter in addition to the requested and lowest or highest acceptable value already contained in the received SETUP message. If the parameter is a metric the SETUP message forwarded by the interfaces (either UNI or NNI) carries a cumulative value of the parameter, again in addition to the requested and the highest or lowest acceptable value already contained in the received SETUP message. The accumulation function is part of the specific negotiating behavior for the parameters.

Each successive switching system determines an outgoing link over which it will progress the SETUP message, and determines the value of each parameter that applies to that link. For ‘meet’ parameters it selects the appropriate value for the parameter. For ‘meet or exceed’ parameters it might or might not have the choice among several values, and if so, it selects one according to switch implementation specific criteria.

If the parameter is an attribute, the value of the parameter that applies to the outgoing link is compared with the lowest or highest acceptable value in the initial call establishment message. If it is less desirable than the highest or lowest acceptable value, i.e. either higher than the highest acceptable value or lower than the lowest acceptable value, the call is rejected. If it less desirable than the indicated value carried in the incoming SETUP message received from a preceding switching system, or if the initial call establishment message was received from an

end-system, then the indicated value carried in the outgoing initial call establishment is set to the value that applies to the outgoing link.

If the parameter is a metric, the accumulation function is applied to the value of the parameter that applies to the outgoing link and the cumulative parameter in the incoming SETUP message, resulting in the new cumulative value. If the new cumulative value is less desirable than the highest or lowest acceptable value, the call is rejected (or a procedure known as cranckback occurs, see [ATMF96c]). The new cumulative value is carried in the outgoing SETUP message.

When the SETUP message is received at the called end-system, it contains the requested, either highest or lowest acceptable and either indicated or cumulative values of the parameter. The called end-system determines whether or not the value is acceptable, and either accepts or rejects the call. For some parameter negotiations, as e.g. PCR, where the parameter value affects resources in the called end-system, these may select a less desirable value for the parameter, as long as it is not less desirable than the highest or lowest desirable value. In this case, the CONNECT message sent by the called end-system contains the agreed upon value, which is always between the indicated or cumulative value and the highest or lowest acceptable value contained in the received initial call establishment message.

The CONNECT message progresses through the network and eventually arrives at the sending end-system with the agreed upon values of the parameters being negotiated. These values then apply for the whole duration of the call, i.e. there is no renegotiation without shutting down the old connection and setting up a new connection with the newly desired characteristics.

In Figure 22 the adjustment process of the parameters is presented for a single parameter (traffic or QoS) and a connection crossing just one switch (i.e. there is no NNI). The parameter

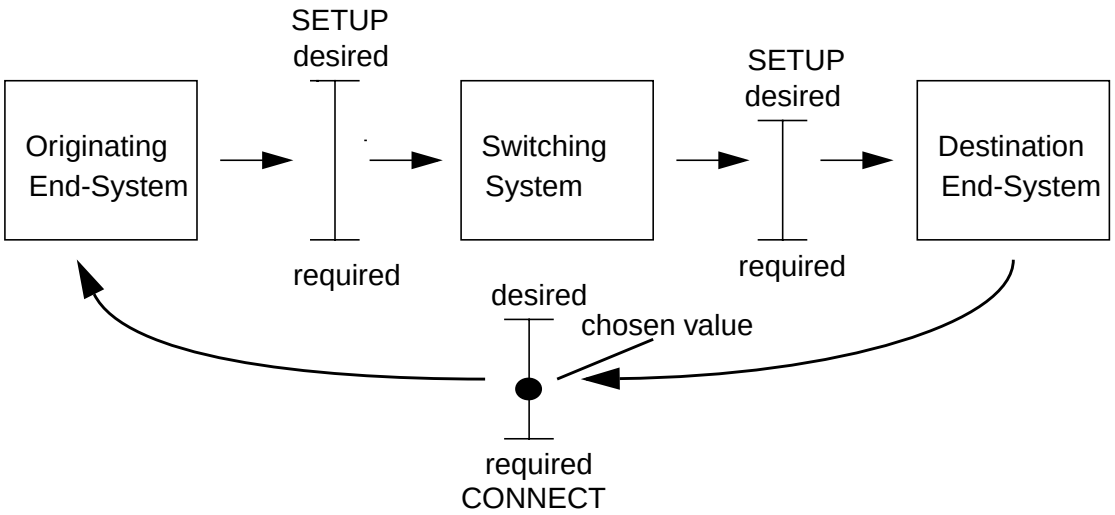


Figure 22: Distributed QoS Computation in ATM.

value range of the requested/desired value and the worst acceptable/required value is symbolized by the bars, which are contained in the SETUP and CONNECT messages being passed between end-systems and switch. Of course, the CONNECT message from the destination end-system is passed to the switch first, however the switch forwards it without modifications, so that the figure shows no participation of the switch.

5 Advanced QoS Techniques

5.1 Adaptive Mechanisms

As described in the previous sections, resource reservation mechanisms can provide QoS guarantees. However, for several system and network components, such mechanisms have not been deployed yet and, perhaps, might never be existing for some of these. Thus, at least for some end-to-end applications and scenarios, QoS cannot be provided via reservation mechanisms. Furthermore, QoS support for continuous-media streams with a variable bitrate encoding is only partially available nowadays. And for 'non-hard guaranteed' QoS classes, there is and will always be a probability for congestion. This means that there is interest in methods for handling such situations. This can be achieved by changing the amount of data which is transmitted, processed and presented by using adaptive methods.

5.1.1 Scaling

Scaling methods are mechanisms where the amount of data generated at and transferred from the origin to the target is changed. To perform the adaptation, a feedback control loop is introduced as shown in Figure 23 – the load state of network and local end-system resources must be monitored and if significant changes occur, e.g., the network is overloaded leading to large delay and high loss, appropriate actions must be taken, e.g., the generated load must be reduced by dropping parts of the data or by using a coarser coding of the input data. This reduction can be achieved in various ways, by explicit communication between receiver and sender (the receiver informs the sender to slow down), completely in the network on a hop-by-hop basis or by feedback from congested network nodes to the sender.

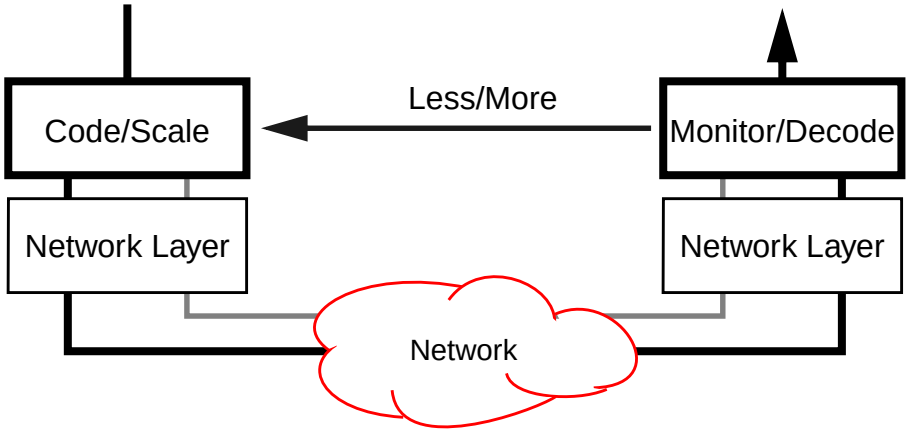


Figure 23: Feedback Control Loop to Allow for Scaling Operations.

Scaling provides two kinds of benefit. First, it has the potential to increase the number of streams a system can support simultaneously in comparison to systems using hard guarantees. This is due to its ability to handle and resolve a system's overload situation. Second, scaling keeps media streams meaningful to the user which would break during overload situations. Instead of interrupting the service for a stream when an overload situation is encountered, the quality of the stream is degraded when the resource load situation reaches a critical state. Since

scaling is a technique that dynamically takes actual resource load into account, it can easily adapt to changing situations and can keep the system in a range of optimal load.

Multiple systems have been developed for scaling, especially in the unicast transmission case. For instance, [GHMN95] and [KMR93] regulate the sender codec to adapt the amount of transmitted data. Jeffay et al. [JSTS92] describe a special queuing mechanism to adapt the bandwidth taken by video sent across packet-switched networks. Hoffmann et al. address in [HSF93] network feedback to the sender to avoid congestion in networks that cannot properly be supported by resource management. Tokuda et al. have implemented a dynamic QOS management for local area networks [CT92], [TTCM92].

Implementing scaling mechanisms within each single application forces programmers to construct their own mechanisms. Besides duplicating work, this approach leads to interworking problems between applications, because it cannot address the behavior of the system in case of several streams being scaled simultaneously raising the question of fairness and balancing between streams. These problems can be solved by the 'middleware' approach where media scaling methods are integrated into a general system support for multimedia in order to simplify the implementation of scalable applications [KMR93]. This way, down- and upscale operations of all scalable streams can be coordinated and hence a balanced sharing of resources can be performed.

5.1.2 Filtering

Sending feedback from receiver back to sender informs the latter about the requirements of the receiver. Several distributed multimedia applications such as video conferencing or video lectures must support multiple receivers. For applications with many participating receivers and for applications which transmit their data across a wide geographical range, there exists a need to support receivers and intermediate transmission paths with different capabilities. Hence, we have to cope with a situation where several multicast receivers require different amount or different encoding of data from the sender which can be the case, for example, if the network or the processing capacities of the receivers vary.

An approach to support heterogeneous receivers in multimedia applications is to use *filter* mechanisms where only a subset of the full information is presented to the end user on the receiving side. Data which are not presented are stripped off from the original data stream at some intermediate agents, for example routers. Thus, the source always emits a full stream, but the stream is possibly scaled to a stream with lower quality.

In [PPAK92], [Pasq93] Pasquale has introduced filters as a general concept, this idea has also been taken up by others, e.g. at Lancaster University. His filters would allow a system to perform arbitrary operations on multimedia data in any part of the network. They can, for example, be used to transform one encoding format to another, e.g., from ADPCM to PCM audio coding. Although the generality of the model is appealing, it can lead to several problems: long processing times may increase communication delays, security aspects may prohibit users from down-loading code for arbitrary filters into routers, and not all intermediate nodes, for example ATM switches, may be suited to provide the required processing capabilities.

In the following, we consider the use of filters for the purpose of packet discarding only, i.e., we understand filtering in the true sense of the word. Then, an intermediate node within the network changes the amount of transmitted information by removing parts of the data and for-

warding only a subset of the full information, which is finally presented to the end user on the receiving side. This filtering takes place in the network layer since only on this layer can all intermediate nodes in a communication path be known. Other “filtering” operations such as mixing audio streams can be accomplished in higher layers using mechanisms such as, e.g., provided by RTP.

Encoding

Deciding which parts of a multimedia information stream to forward and which to filter out, can only be done with respect to the data encoding scheme used. The encoding of the full information at the origin can follow one of two approaches:

- In *independently* coded streams, higher-quality parts are substitutions for lower-quality parts. For example, one substream S_1 may contain complete images of the size a*b and another substream S_2 may contain complete images of the size 2a*2b. To choose a different quality means to choose a different substream.
- In *hierarchically* encoded multimedia streams, higher-quality parts are additions to the lower-quality parts. For example, one substream S_1 may contain images of the size a*b and another substream S_2 may contain all *additional* pixels that extend the format to 2a*2b. To present data in the highest quality, all substreams must be presented.

The independent coding of streams can lead to inefficiency due to the resulting overhead of transmitting ’similar’ data multiple times (often called simulcast). The hierarchical coding avoids this, yet, it can be more complex because several parts must be combined to have the full information.

Hierarchically encoded streams will play an important role in the future of multimedia systems. Data formats such as MPEG-II use hierarchical encoding to achieve different levels of presentation quality. These levels result from scaling the original video data in several dimensions. For instance, Gonzales and Viscito describe the following techniques for this purpose [GG91]:

- *Spatial scaling:* a multiplicity of spatial resolutions.
- *Rate scaling:* a multiplicity of picture rates. This is already part of MPEG-I via the division into *I* (intra-frame coded), *P* (predicted coded), and *B* (bidirectional predicted coded) frames. This kind of scaling is often also referred to as *temporal* scaling.
- *Amplitude scaling:* multiple versions of a picture with varying fidelity at the same spatial and temporal resolution. This is also specified as *frequency* scaling, either via data partitioning, in which the discrete cosine transformation (DCT) coefficients are separated into several regions and the regions are handled differently, or via signal-to-noise-ratio (SNR) scaling, in which the least significant bits of the DCT coefficients are separated from the most significant bits and separately handled.

For any of these scaling methods, all substreams together yield an image of the best quality. There is a well-defined order to filter out substreams, should it become necessary.

An example for the use of a hierarchically-coded stream is shown in Figure 24. There, the full stream S, coded with a hierarchical scheme, consists of 3 parts, the base stream S_0, and the additional streams S_1 and S_2. By combining S_0, S_1, S_2, it is possible to build three different quality streams to be presented:

1. Stream Q_0: (low quality, contains S_0 only, requires lowest bandwidth).
2. Stream Q_1: (medium quality, contains S_0 & S_1, requires medium bandwidth).
3. Stream Q_2: (high quality, contains S_0, S_1 & S_2, requires largest bandwidth).

The hosts H_2, H_3, and H_5 participating in the transmission of the streams receive the data from the source H_0 through routers H_1 and H_4. In this example, each destination decides to receive a stream of different quality, corresponding to different portions of the data. While the target at host H_2 would like to receive the full quality stream Q_2 (thus, all available parts), targets H_3 and H_5 need not so high a quality, and therefore they use S_0 & S_1 (yielding Q_1), and only S_0 (resulting in Q_0), respectively.

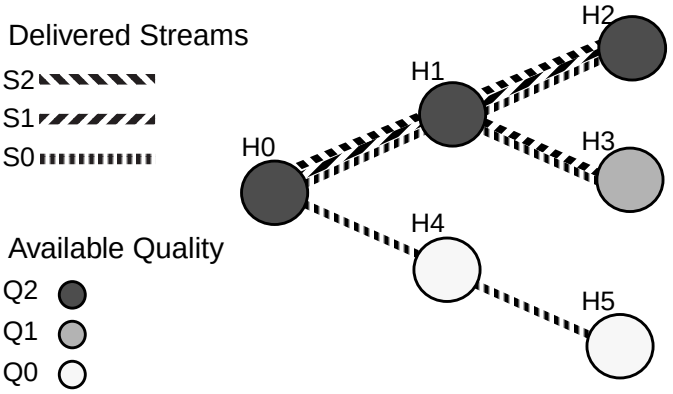


Figure 24: Hierarchically-Coded Stream and Filtering.

Data Transmission for Filtering

For the hierarchical coding, two approaches can be followed after splitting the hierarchically coded data into parts:

(1) independent streams
(2) substreams

In the first approach, the application splits the data into streams and sends them independently, as has been described in [DHHH94] where ST-2 is additionally used to reserve resources for the base layers, or as in [BTSW94] and [CG96] where a similar approach using an IP multicast group for each layer has been taken. A refinement of this is the use of error detection within the receiver-driven layered multicast approach [MESZ94] where receivers start out to receive the base layer and add further enhancement layers until they have either subscribed to all layers or they experience packet loss. In the latter case, they remove the less important layers again. This way, receivers search for the optimal level of subscription. And, due to the pruning mechanisms of IP multicast, if there has no receiver a subscription for a specific layer in a particular area of the internet, this layer must not be forwarded to that. The receiving applications must, of course, know which layers are existing in order to join. This mechanisms can but must not be combined with resource reservation, e.g., reservations using RSVP could be established for some of the layers.

The independent transmission of the layers can lead to differing delays among the layers (e.g., due to the use of different routes). Yet, since the informations contained in the layers are dependent on each other such an approach results in synchronization problems.

In the second approach, substreams are constructed by describing the relationship among the parts and transmitting them as one stream consisting of substreams. This approach avoids the resynchronization problems which occurs with independently transmitted streams. With substreams, the application transmits one stream (sets up one flow/connection) yet describes the structure of the stream (e.g., as part of the FlowSpec) which can then be used by routers to specify filters which strip off information not to be transmitted towards that receiver [WDSSW95]. Two issues must be addressed to use this substream approach, packets must be identifiable and the streams and their relationship must be specified.

To make the decision which packets to drop, routers need information about the content of the data packets. This can be specified by tagging the packet, i.e., the source assigns an appropriate value to a field of the packet's header, or by using pattern matching, i.e., the source describes patterns and the router which analyses a packet checks if it matches the given pattern and decides based on that what to do with the packet. The pattern matching approach is very flexible, on the other hand, it has also some inconveniences, because of the longer time required by the source to specify the different patterns and by the routers to match them.

The QoS requirements must be specified for each substream so that all the routers and targets know which lower-quality substreams can be derived from the full stream. This can be done by providing substream FlowSpecs specifying values per substream for QoS class, bandwidth, and reliability requirements; other parameters such as delay are common for all substreams. Based on the substream description and the FlowSpecs, resources can be reserved to provide QoS and filters can be created (and updated, e.g., by merging if additional receivers occur) at intermediate systems.

5.2 Resource Reservation in Advance

The resource management systems described above, offer functions which allow the reservation of resources for a time interval which starts with the reservation attempt and which lasts for an unspecified time. For several application scenarios this model of immediate reservations is not appropriate. Consider, for instance, a virtual meeting room (conferencing) scenario supported by multimedia systems, where perhaps weeks in advance of the actual 'meeting', it must be ensured that sufficient resources to hold the conference are available. To support these 'virtual meeting room' scenarios the resource reservation system must offer mechanisms to reserve in advance the resources needed for the conference, i.e., certain capacities of networks, routers, and end-system resources. Resource Reservation in Advance (ReRA) is not only needed for conferencing but for other scenarios such as video-on-demand as well. In general, if resource reservation is needed, then ReRA must be provided as well. However, several difficult issues must be resolved before ReRA will find widespread support; here we can only address some of them.

5.2.1 Characterization and Model

Reservations can be classified based on two key factors [WDSSW95]: (1) whether the resources are exploited at reservation time, and (2) whether the reservation duration is known at reservation time. Traditional resource management systems (non-ReRA) assume that the resources are immediately used after they have been successfully reserved and no assumptions are made on the duration of the reservations. A ReRA scheme, on the contrary, is characterized by deferred resource usage and reservations of known duration (which might possibly be extended). In case of immediate usage and known duration, either scheme can be realized.

Table 1: Classification of Reservation Schemes.

		Reservation Duration	
		Known	Unknown
Resource Usage	Immediate	non-ReRA / ReRA	non-ReRA
	Deferred	ReRA	

Then the ReRA scheme consists of two parts (see also Figure 25):

(1) the resource reservation in advance and
(2) the usage of the reserved resources.

In the first part, the client specifies its request, i.e., it gives a *workload specification* and defines the *begin* and *duration* of the reservation. The second phase begins shortly before the client intends to exploit its reservation. The client contacts the service provider to demand the previously reserved resources. Then the client exploits its reservation by making use of the reserved resources. Once a session is established, the participants may either finish earlier (than previously reserved) or they may want to extend the time. The first case is simple; resources can be freed and made available for other applications. However, in the second case, if the application duration is to be extended, the system may or may not have a sufficient amount of resources to

serve the application with the necessary QoS. If enough resources are available, the service should not be interrupted and the application should be provided with the means to extend its previous reservation. If insufficient resources are available, the system may still attempt to serve the application on a best-effort basis with a degradation in the QoS.

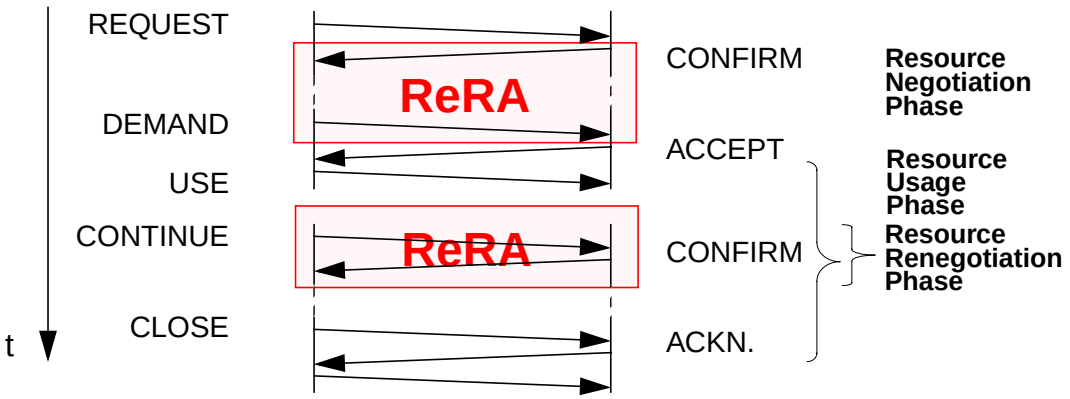


Figure 25: Reservation in Advance Primitives.

5.2.2 Distribution of Announcement Information

In addition to the information about stream characteristics which are exchanged via resource reservation protocols such as RSVP and ST-2, information about the date of the stream start and even basically the knowledge about its existence time must be distributed as well. Such information is today usually distributed via other means than those later used for the application; for example, the invitation to join a multi-user phone conference is given to the potential participants by contacting each person independently via a point-to-point phone call. In the Internet, the sd or sdr programs are often used for such notifications if the event is open and can be joined by anyone who is interested.

The information about announcements can be handled by a ‘user agent’ which is similar to the user agent of a mail system. It provides the interface for the user to handle resource reservations in advance. An incoming invitation to a multimedia application (to be started sometime in the future) is presented to the user, who can acknowledge or reject the invitation. Using this agent, users can also start reservation attempts themselves. The user agent should provide the ability to automatically start the application at the scheduled starting time of the data stream, i.e., just before the conference begins. ([WS97])

5.2.3 Failure Situations

With ReRA, in addition to the handling of failures in the negotiation phase and the usage phase, care must be taken of failures that may occur between these phases, i.e., after a reservation has been made but before it is used. First, the reservation state stored at end-systems and intermediate nodes might be needed for long time periods. State information must be stored in non-volatile storage. This is not only necessary as a protection against failures, but also because any node may be restarted regularly between the phases, e.g., for maintenance.

Furthermore, as opposed to failures occurring during data transmission, no client is running when a node notices a failure. The failure itself might, however, not be detected at the failing

node but only at a neighbor which has only partial information about the reservation state stored at the node. Means to inform the clients explicitly about the failure situation and whether or not it can be resolved in time must be provided, respectively the application must be able to query the correctness and availability of the reservation before it starts its usage phase.

5.2.4 Modifications to Support Advance Reservations

Various components of resource management systems have to be modified to support ReRA scenarios.

- The interfaces of resource management systems need in addition to the QoS parameters now also specifications of the time parameters (begin and duration).
- These time values must be contained in the flow specification that is distributed via the resource reservation protocols to all affected network nodes.([Rein94], [Rein95])
- The database of existing reservations must represent time slices (e.g., [FGV95]). For each time the set of existing or reserved streams with their QoS parameters and the free resources must be known.
- The admission control algorithms must take the time parameters into account. An example for such an algorithm for predictive service is given in [DKPS95].
- Additional failure handling mechanisms and means to save state information in permanent storage are necessary.

Furthermore, the reservation protocols must be enhanced. New PDU types to support the additional states and transitions and to handle failure situations and notify neighbor nodes about them are needed.

6 Conclusion and Outlook

The provision of QoS support in distributed computer systems is coupled with a wide range of aspects – from QoS definition and modeling, via QoS translation and QoS calculation to QoS enforcement. This makes clear that the term 'Quality of Service' is not a simple term, but a rather complex one with multiple facets. Furthermore, QoS is not 'single-faced' but comes in different flavors: guaranteed QoS, statistical QoS, predicted QoS, best-effort/no-effort QoS, …

Typically, techniques which support for, respectively perform resource reservation are considered as QoS mechanisms. Other methods such as adaptive mechanisms are often not subsumed by this term, however, they can be considered as QoS techniques as well since they provide for a best-possible presentation quality at the user interface (except the mechanisms purely directed towards congestion control).

It might be considered that with growing system and network bandwidth, the need for QoS support is disappearing. Yet, we believe that there will still be a role for careful QoS calculation and reservation techniques: As bandwidth goes up, so does demand. Furthermore, network and system providers want to serve as many customers as possible with as few resources as necessary to reduce their costs and increase their profit.

The main driving force behind the development of QoS techniques has been, and probably will be, the support of distributed multimedia applications such as video conferencing, retrieval systems and video-on-demand. Such application, for which the successful delivery of audiovisual and other time-critical data with a well-defined QoS is a crucial issue, will address all network types, LANs (e.g., in-house information systems), MANs (e.g., city information systems, campus networks) and WANs (e.g., distributed lectures).

Resource reservation and scaling mechanisms have been an active research area with increasing dedication already during the last years. The work on Tenet at the ICSI in Berkeley, on the QoS-A at Lancaster University, and on HeiTS and HeiRAT at IBM ENC in Heidelberg showed that QoS provisioning within computer systems and networks is feasible.

Currently, the Internet does not support QoS on a wide scale. This will probably change in the near future due to the support of RSVP, accompanying admission control and scheduling mechanisms. Their commercial success will depend on the proof of the respective suitability for

- a large scale use, i.e. a huge amount of concurrent flows & numbers of participants in a flow,
- the support of all, or at least a majority of the most important, types of applications.

The success of ATM, in general and for multimedia communication applications using it in particular, depends on the successful standardization of its signaling mechanisms, its ability to attract the development of native ATM applications and the integration of ATM with other communication systems.

QoS support by resource reservation inserts state information into systems and network nodes. The routing algorithms which determine which path the transmitted data follows while traveling through the network must take resource availability on the various possible routes into account when making its decision about the data forwarding path. Otherwise, the reservation setup becomes merely a trial and error approach. Only few work has been performed on this topic of 'QoS routing' yet. Recently the interest in that issue has risen and it can be expected that research results will follow in the future.

If reservation is needed at all (what the authors are convinced of) then this applies also to ReRA. Yet ReRA will require modifications, add complexity to protocols and network nodes,

and furthermore, requires that state information is kept in the network for quite some time. These requirements lead to questions about stability and scalability.

An important issue for the future success of distributed multimedia applications will be the costs for data transmission with or without QoS. This will influence the overall success of distributed multimedia applications at least as much as one technical issue. However, now there exist only some basic approaches (or far too complex algorithms) to cope with pricing. Investigations on accounting policies and mechanisms as well as the willingness of users to accept a certain QoS for a specific price have started recently.

Another major issue is network integration. The integration of the various network infrastructures into a global, ubiquitous network capable of providing suitable support for QoS provided communications must address, e.g., current Internet technology, ATM, and mobile systems.

Secure communication must be possible for audiovisual data to find widespread usage between companies over a global network. This requires, in addition to suitable algorithms for encryption, also firewalls which provide appropriate mechanisms to support such data flows.

References

[Ande93] D.P. Anderson: *Metascheduling for Continuous Media*. ACM Transactions on Computer Systems, Vol. 11, No. 3, 1993.

[ATMF96a] The ATM Forum: *Traffic Management (TM) Specification 4.0 (Draft)*, April 1996.

[ATMF96b] The ATM Forum: *User-Network-Interface (UNI) Signalling Specification*, July 1996.

[ATMF96c] The ATM Forum: *Private Network-Node Interface (PNNI) Signalling Specification*, February 1996.

[ATMF96d] The ATM Forum: *Broadband-Inter-Carrier Interface(BICI) Signalling Specification 2.0 (Draft)*, February 1996.

[ATWG90] D.P.Anderson, S.Tzou, R.Wahbe, R.Govindan: *Support for Continuous Media in the DASH System*, Proc. of the 10th International Conference on Distributed Computer Systems, Paris, May 1990.

[Baker96] F.Baker: *RSVP Cryptograhic Authentication*, Internet Draft, February 1996.

[BCS94] R.Braden, D.Clark, S.Shenker: *Integrated Services in the Internet Architecture*, Internet Request for Comment 1633, 1994.

[BFMMVZ96] A.Banerjea, D.Ferrrari, B.A.Mah, M.Moran, D.C.Verma, H.Zhang: *The Tenet Real-time Protocol Suite: Design, Implementation, an Experiences*, IEEE/ACM Transactions on Networkng, February 1996.

[BM91] A.Banerjea, B.Mah: *The Real-Time Channel Administration Protocol*, Proceedings Second International Workshop on Network and Operating System Support for Digital Audio and Video, November 1991.

[BPF94] A.Banerjea, C.Parris, D.Ferrari: *Recovering Guaranteed Performance Service Connections from Single and Multiple Faults*, Proc. GLOBECOM, November 1994.

[BTSW94] J.C. Bolot, T. Turletti, S.Schröder, I. Wakeman: *Scalable Feedback Control for Multicast Video Distributionin the Internet*, Proceedings of SIGCOMM ‘94.

[BZBHJ96] R.Braden, L.Zhang, S.Berson, S.Herzog, S.Jamin: *Resource Reservation Protocol (RSVP) - Version 1 Functional Specification*, Internet Draft, November 1996.

[CAH95] A.Campbell, C.Aurrecoechea, L.Hauw: *A Survey of Quality of Service Architectures*, Technical Report Lancaster University, 1995.

[CB93] G.Coulson, G.Blair: *Micro-Kernel Support for Continuous Media in Distributed Systems*, Technical Report Lancaster University, 1993.

[CCGH92] A.Campbell, G.Coulson, F.Garcia, D.Hutchison: *A Continuous Media Transport and Orchestration Service*, Proc. ACM SIGCOMM '92, August 1992

[CCGHL93] A.Campbell, G.Coulson, F.Garcia, D.Hutchison, H.Leopold: *Integrated Quality of Service for Multimedia Communication*, Proc. IEEE INFOCOM '93, April 1993.

[CCH94] A.Campbell, G.Coulson, D.Hutchison: *A Quality of Service Architecture*, ACM Computer Communications Review, April 1994.

[CCH95] A.Campbell, G.Coulson, D.Hutchison: *Supporting Adaptive Flows in a Quality of Service Architecture*, Multimedia Systems Journal, November 1995.

[CG96] N.Chaddha, A.Gupta: *A Frame-Work for Live Multicast of Video Streams over the Internet*, Proceedings of the IEEE International Conference on Image Processing, 1996.

[Clar88] D.D.Clark: *The Design Philosophy of the DARPA Internet Protocols*, Proc. ACM SIGCOMM '88 Conference, August 1988.

[Cruz91] R.L. Cruz: *A Calculus for Network Delay, PART I: Network Elements in Isolation*, IEEE Transactions on Information Theory, Vol.37, No. 1, January 1991.

[CSZ92] D.D.Clark, S.Shenker, L.Zhang: *Support for Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanisms*, Proc. ACM SIGCOMM '94 Conference, October 1992.

[CT92] S.T.-C. Chou, H. Tokuda: *System Support for Dynamic QOS Control of Continuous Media Communication.* Third International Workshop on Network and Operating System Support for Digital Audio and Video, San Diego, 1992.

[DB95] L. Delgrossi, L. Berger, *Internet Stream Protocol Version 2 (ST2) Protocol Specification - Version ST2+*, August 1995.

[Deer95] S.Deering: *Internet Protocol Version 6 Specification*, Internet Request for Comment 1883, 1995.

[DHHH94] L.Delgrossi, C.Halstrick, D.Hehmann, R.Herrtwich, O.Krone, J.Sandvoss, C.Vogt: *Media Scaling in a Multimedia Communication System*, ACM Multimedia Systems Journal, Vol. 2, No. 4, 1994, pp. 172-180.

[DHHS92] L. Delgrossi, C. Halstrick, R.G. Herrtwich, H. Stuettgen: *HeiTP: A Transport Protocol for ST-II*. GLOBECOM‘92, Orlando, 1992

[DHHS94] Luca Delgrossi, Ralf Guido Herrtwich, Frank Oliver Hoffmann, Sybille Schaller: *Receiver-Initiated Communication with ST-2*, ACM Multimedia Systems Journal, Vol. 2, No. 4, pp. 141-149, 1994.

[DHHHST93] Luca Delgrossi, Christian Halstrick, Ralf Guido Herrtwich, Frank Oliver Hoffmann, Jochen Sandvoss, Barbara Twachtmann: *Reliability Issues in Multimedia Transport*, 2nd IEEE Workshop on the Architecture and Implementation of High-Performance Communication Subsystems HPCS’93, Williamsburg, Virginia, (USA), September 1993.

[DHVW93] Luca Delgrossi, Ralf Guido Herrtwich, Carsten Vogt, Lars C. Wolf: *Reservation Protocols for Internetworks: A Comparison of ST-II and RSVP*, 4th International Workshop on Network and Operating System Support for Digital Audio and Video, Lancaster (United Kingdom), October 1993.

[DKPS95] M. Degermark, T. Köhler, S. Pink, O. Schelén: *Advance Reservation for Predicted Service*, Proceedings of Fifth International Workshop on Network and Operating System Support for Digital Audio and Video, Durham, NH, USA, April 19-21, 1995, Springer-Verlag LNCS 1018.

[FBZ94] D.Ferrari, A.Banerjea, H.Zhang: *Network Support for Multimedia: A Discussion of the Tenet Approach*, Computing Networks ISDN Systems, Special Issue on Multimedia Networking, 1994.

[Ferr95] D.Ferrari: *The tenet experience and the design of protocols for integrated services internetworks*, Multimedia Systems Journal, 1995.

[FGV95] D. Ferrari, A. Gupta, G. Ventre: *Distributed Advance Reservation of Real-Time Connections*, Fifth International Workshop on Network and Operating System Support for Digital Audio and Video, Durham, NH, USA, April 19-21, 1995, Springer-Verlag LNCS 1018.

[Floy92] S.Floyd: *Issues in Flexible Resource Mangement for Datagram networks*, Proc. of the Third Workshop on Very High Speed Networks, March 1992.

[FV90] D.Ferrari, D.C. Verma: *A Scheme for Real-Time channel Establishment in Wide-Area Networks*, IEEE Journal on Selected Areas of Communication, April 1990.

[Garc93] F.Garcia: *A Continuous Media Transport Service*, PhD Thesis Lancaster University, 1993.

[Garr96] M.W.Garret: *A Service Architecture for ATM: From Applications to Scheduling*, IEEE Network, May 1996.

[GHMN95] A.Gupta, W.Howe, M.Moran, Q.Nguyen: *Scalable Ressource Reservation for Multi-Party Communication*, Proc. INFOCOM, April 1995.

[GHMS93] A.Gupta, W.Heffner, M.Moran, C.Szyperski: *Network Support for Real-time Multiparty Applications*, Proc. Fourth International Workshop on Network and Operating System Support for Digital Audio and Video, November 1993.

[GHMY95] F.Garcia, D.Hutchison, A.Mauthe, N.Yeadon: *QoS Support dor Distributed Multimedia Communications*, Technical Report Lancaster University, 1995.

[GG91] M.Gilge, R.Gusella: *Motion video coding for packet-switching networks - an integrated approach*, Proceedings of the SPIE Conference on Visual Communications and Image Processing, 1991.

[GV93] C. Gonzales, E. Viscito: *Flexibly Scalable Digital Video Coding, Signal Processing: Image Communication*, Vol. 5, No. 1-2, February 1993.

[Herr94] R.G. Herrtwich: *Distributed Multimedia Solutions from the HeiProjects*. In: J.L. Encarnacao, J.D. Foley (Eds): Multimedia – System Architectures and Applications, Springer, 1994.

[HSF93] D. Hoffman, M. Speer, G. Fernando: *Network Support for Dynamically Scaled Multimedia Data Streams*, Fourth International Workshop on Network and Operating System Support for Digital Audio and Video, Lancaster, UK, 1993.

[ITU95] ITU-T Recommendation I.356: *B-ISDN ATM layer cell transfer performance*, ITU-T Study Group 13, 1995.

[ITU96a] ITU-T Recommendation I.361: *B-ISDN ATM Layer Specification*, ITU-T Study Group 13, 1996.

[ITU96b] ITU-T Recommendation I.371(Draft): *Traffic Control and Congestion Control in B-ISDN*, ITU-T Study Group 13, 1996.

[ITU96c] ITU-T Recommendation Q.2931: *B-ISDN- Digital Subscriber Signalling No.2 (DSS 2) - User Network Interface Layer 3 Specification for basic Call/ Connection Control*, ITU-T Study Group 11, 1996.

[JCSZ92] S.Jamin, D.Clark, , S.Shenker, L.Zhang: *An Admission Control Algorithm for Predictive Real-Time Services*, Proc. Third Workshop on Network and Operating System Support for Digital Audio and Video, November 1992.

[JPV93] D. Jordaan, M. Paterok, C. Vogt: Layered Quality of Service Management in Heterogeneous Networks. IBM European Networking Center, TR 43.9304, Heidelberg, 1993.

[JSTS92] K. Jeffay, D.L. Stone, T. Talley, F.D. Smith: *Adaptive Best-Effort Delivery of Digital Audio and Video Across Packet-Switched Networks,* Third International Workshop on Network and Operating System Support for Digital Audio and Video, San Diego, USA, 1992.

[KMR93] H.Kanaki, P.P.Mishra, A. Reibman: *An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport*, Proceedings of SIGCOMM '93.

[KW94] T. Kaeppner, L. Wolf: *Media Scaling in Distributed Multimedia Object Services*, Proc. Second International Workshop on Advanced Teleservices and High-Speed Communication Architectures, September 1994, Heidelberg, Germany.

[Laza94a] A.A. Lazar: *Challenges in multimedia networking*, in Proc. of the International Hi-Tech Forum, Osaka '94, 1994.

[Laza94b] A.A. Lazar: *A Research Agenda for Multimedia Networking*, position paper at the Workshop on Fundamentals and Perspectives on Multimedia Systems, International Conference Center for Computer Science, Dagstuhl Castle, Germany, July 4-8, 1994.

[LBL94] A.A.Lazar, S.Bhonsle, K.S.Lim: *A Binding Architecture for Multimedia Networks*, Proceedings of the Multimedia Transport and Teleservices, Vienna, Austria, November 14-15, 1994.

[LL73] C.L. Liu, J.W. Layland: *Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment*, Journal of ACM, Vol. 20, No. 1, 1973.

[MESZ94] D.J.Mitzel, D.Estrin, S.Shenker, L.Zhang: *An Architectural Comparison of ST-II and RSVP*, Proc. of IEEE INFOCOM '94, June 1994.

[MJV96] S.McCanne, V.Jacobson, M.Vetterli: *Receiver-driven Layered Multicast*, Proceedings ACM SIGCOMM'96, October 1996.

[Nahr95] K.Nahrstedt: *Challenges in Providing End-to-End QoS Guarantees in Networked Multimedia Systems*, Technical Report University of Pennsylvania,1995

[NLP96] L.H.Ngoh, H.Y.Li, H.K.Pung: *A Direct Multicast Service with Quality of Service Guarantees*, Proc. IEEE Multimedia, 1996.

[NS95a] K.Nahrstedt, R.Steinmetz: *Multimedia: Computing, Communication & Applications*, Prentic Hall, 1995.

[NS95b] K.Nahrstedt, J.Smith: T*he QoS Broker*, IEEE Multimedia, Spring 1995.

[NS95c] K.Nahrstedt, J.Smith: *Design, Implementation, and Experiences of the OMEGA end-point architecture*, Technical Report University of Pennsylvania, 1995.]

[Part92] C.Partridge: *A Proposed Flow Specification*, Internet Request for Comments 1363, July 1992.

[PPAK92] J. Pasquale, G. Polyzos, E. Anderson, V. Kompella: *The Multimedia Multicast Channel*, Proc. Third International Workshop on Network and Operating System Support for Digital Audio and Video, San Diego, November 1992.

[Pasq93] J. Pasquale: *Filter Propagation in Dissemination Trees: Trading off Bandwidth for Processing in Continuous Media Networks*, Proc. Fourth International Workshop on Network and Operating System Support for Digital Audio and Video, Lancaster University, November 1993.

[Rein94] W. Reinhardt: *Advance Reservation of Network Resources for Multimedia Applications*, Proceedings of the Second International Workshop on Advanced Teleservices and High-Speed Communication Architectures, Heidelberg, Germany, September 26-28, 1994, Springer-Verlag LNCS 868.

[Rein95] W. Reinhardt, *Advance Resource Reservation and its Impact on Reservation Protocols*, Proceedings of Broadband Island'95, Dublin, Irland, September 1995.

[SB95] S.Shenker, L.Breslau: *Two Issues in Reservation Establishment*, Proc. ACM SIGCOMM ’95, August 1995.

[SCZ93] S.Shenker, D.Clark, L.Zhang: *A Scheduling Service Model and a Scheduling Architecture for an Integrated Services Packet Network*, ACM/IEEE Transaction on Networking, November 1993.

[SPG96] S.Shenker, C.Partridge, R.Guerin: *Specification of Guaranteed Quality of Service*, Internet Draft, August 1996.

[Stil95] B.Stiller: *Quality of Service*, TAT, 1995.

[Topo90] C. Topolcic: *Experimental Internet Stream Protocol, Version 2 (ST-II)*, RFC 1190, October 1990.

[TTCM92] H. Tokuda, Y. Tobe, S.T.-C. Chou, J.M.F. Moura: *Continuous Media Communication with Dynamic QOS Control Using ARTS with an FDDI Network.* ACM SIGCOMM 92, Baltimore, 1992.

[VBDGHK94] A.Vogel, G.v.Bochmann, R.Dssouli, J.Gecsei, A.Hafid, B.Keherve: *On QoS Negotiation in Distrbuted Multimedia Applications*, Proc. Protocols for High Speed Networks, 1994.

[VHKWW96] R. Vogel, R.G. Herrtwich, W. Kalfa, H. Wittig, L. Wolf: *QoS-Based Routing of Multimedia Streams in Computer Networks*; IEEE Journal on Selected Areas in Communications, Vol. 14, No. 7, September 1996.

[Vogt95] C. Vogt: *Quality-of-service management for multimedia streams with fixed arrival periods and variable frame sizes*. ACM Multimedia Systems, Vol. 3, No. 2, 1995.

[VHN93] C. Vogt, R.G. Herrtwich, R. Nagarajan: *HeiRAT: The Heidelberg Resource Administration Technique – Design Philosophy and Goals*, Proceedings of Kommunikation in Verteilten Systemen, Munich, Germany, March 3-5, 1993, Springer-Verlag.

[VWHW97] C. Vogt, L.C. Wolf, R.G. Herrtwich, H. Wittig: *HeiRAT – Quality-of-Service Management for Distributed Multimedia Systems*, to appear in ACM Multimedia Systems Journal – Special Issue on QoS Systems, 1997.

[WBV96] L.C. Wolf, W. Burke, C. Vogt: *Evaluation of a CPU Scheduling Mechanism for Multimedia Systems*, Software - Practice and Experience, Vol. 26, No. 4, 1996.

[WDSSW95] L.C. Wolf, L. Delgrossi, R. Steinmetz, S. Schaller, H. Wittig: *Issues of Reserving Resources in Advance*, Fifth International Workshop on Network and Operating System Support for Digital Audio and Video, Durham, NH, USA, April 19-21, 1995, Springer-Verlag LNCS 1018.

[WH94] Lars C. Wolf, Ralf Guido Herrtwich: *The System Architecture of the Heidelberg Transport System*, ACM Operating System Review, Vol. 28, No. 2, April 1994.

[WHD95] L.C. Wolf, R.G. Herrtwich, L. Delgrossi: *Filtering Multimedia Data in Reservation-Based Internetworks*, Proceedings of Kommunikation in Verteilten Systemen, Chemnitz, 1995, Springer-Verlag.

[WM91] B.Wolfinger, M.Moran: *A Continuous Media Data Transport Service and Protocol for Real-Time Communication,* in High Speed Networks, Proceedings Second International Workshop on Network and Operating System Support for Digital Audio and Video, November 1991.

[Wolf96] L.C. Wolf: *Resource Management for Distributed Distributed Multimedia Systems*, Kluwer Publ., Boston 1996.

[Wroc96] J.Wroczlawski: *Specification of the Controlled-Load Network Element Service*, Internet Draft, August 1996.

[WS97] Lars C. Wolf, Ralf Steinmetz: *Concepts for Resource Reservation in Advance*, Special Issue of Journal of Multimedia Tools and Applications (Kluwer), The State of the Art in Multimedia Computing, May 1997.

[ZB96] D.Zappala, R.Braden: *Interdomain Multicast Routing Support for Integrated Services Network*, Internet Draft, December 1996.

[ZBEHJ93] L.Zhang, R.Braden, D.Estrin, S.Shenker, D.Zappala: *RSVP: A New Ressource Reservation Protocol*, IEEE Network, 1993.

[ZVF92] H.Zhang, D.C.Verma, D.Ferrari: *Design and Implementation of the Real-Time Internet Protocol*, Proc. of the IEEE Workshop on the Architecture and Implementation of High Performance Communication Subsystems, September 1991.