Architectural Perspectives on QoS Management in Distributed Multimedia Systems

Andrew Campbell, Cristina Aurrecoechea and Linda Hauw
Center for Telecommunication Research
Columbia University, New York, NY 10027, USA
e-mail: campbell@ctr.columbia.edu

Abstract

Over the past several years there has been a considerable amount of research within the field of quality of service (QoS) support for distributed multimedia systems. To date, most of the work has occurred within the context of individual architectural layers such as the distributed system platform, operating system, transport subsystem and network. Much less progress has been made in addressing the issue of overall end-to-end support for multimedia communications. In recognition of this, a number of research teams have proposed the development of QoS architectures which incorporate quality of service configurable interfaces and quality of service driven control and management mechanisms across all architectural layers. This paper examines the state-of-the-art in the development of QoS architectures. The approach taken is, initially, to present QoS terminology and a generalised QoS framework for understanding and discussing quality of service in the context of distributed multimedia systems. Following this, we consider current research in the area of layer specific quality of service support, and then, evaluate a number of QoS architectures that have recently emerged in the literature from the telecommunications, computer communications and standards communities.

Keywords

QoS specification, QoS mechanisms, QoS control and management, QoS architecture

1. Introduction

Meeting quality of service (QoS) guarantees in distributed multimedia systems is fundamentally an end-to-end issue, that is, from application-to-application. For example, consider the remote playout of a sequence of audio and video: in the distributed system platform, quality of service assurances should apply to the complete flow of media; from the remote server, across the network to the point/s of delivery. As illustrated in Figures 1-1, this generally requires end-to-end admission testing and resource reservation in the first instance, followed by careful co-ordination of disk and thread scheduling in the end-system, packet/cell scheduling and flow control in the network, and finally active monitoring and maintenance of the delivered quality of service. A key observation is that for applications relying on the transfer of multimedia, and in particular continuous media flows, it is essential that quality of service is configurable, predictable and maintainable system-wide, including the end-system devices, communications subsystem and networks. Furthermore, it is also important that all end-to-end elements of distributed systems architecture work together in unison to achieve the desired application level behaviour.

To date, most of the developments in the provision of quality of service support have occurred in the context of individual architectural layers. Much less progress has been made in addressing the issue of an overall *QoS architecture* for multimedia communications. There has been, however, considerable progress in the separate areas of Open Distributed Processing (ODP), end system and network support for quality of service. In end-systems, most of the progress has been made in the specific areas of scheduling, flow synchronisation and transport support. In networks, research has focused on providing suitable traffic models and service disciplines, as well as appropriate admission control and

resource reservation protocols. Many current network architectures, however, address quality of service from a providers point of view and analyse network performance, failing to comprehensively address the quality needs of applications. Until recently there has been little work on quality of service support in distributed systems platforms. What work there is has been mainly been carried out in the context of the Open Distributed Processing.

The current state of QoS provision in architectural frameworks can be summarized as follows [1]:

i) *incompleteness*: current interfaces (e.g., application programming interfaces such as Berkeley Sockets) are generally not QoS configurable and provide only a small subset of the facilities needed for control and management of multimedia flows;

ii) *lack of mechanisms to support QoS guarantees*: research is needed in distributed control, monitoring and maintenance QoS mechanisms so that contracted levels of service can be predictable and assured; and

iii) *lack of overall framework*: it is necessary to develop an overall architectural framework to build on and reconcile the existing notion of quality of service at different systems levels and among different network architectures.

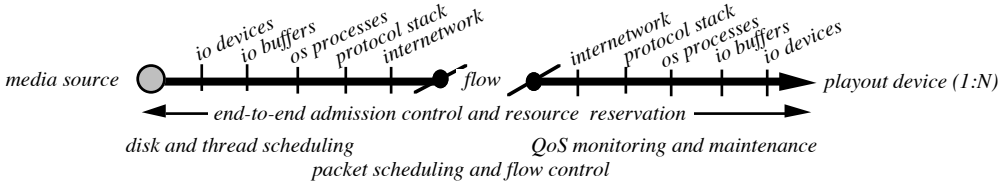

Figure 1-1. End-to-End QoS Scenario

In recognition of the above limitations, a number of research teams have proposed a systems architectural approach to QoS provision; we refer to these models as QoS architectures in this paper. The intention of QoS architecture research is to define a set of quality of service configurable interfaces that formalize quality of service in the end-system and network, providing a framework for the integration of quality of service control and management mechanisms.

The structure of this paper is as follows. We first present, in section 2, a *generalized QoS framework* and terminology[1] for distributed multimedia applications operating over multimedia networks with quality of service guarantees. The QoS framework is based on a set of principles that govern the behavior of QoS architectures. Following this, we review current layer-specific work on quality of service support (in section 3) considering the distributed systems platform layer, operating systems layer, and transport and network layers. In section 4, we evaluate three QoS architectures found in the literature that have been developed by the Telecommunications, Computer Communications and Standards Communities. Following this we present a short qualitative comparison and discussion in sections 5 and 6 respectively. Finally, in section 7 we offer some concluding remarks.

2. Generalised QoS Framework

In what follows, we describe a set of elements used in building quality of service into distributed multimedia systems. These include principles which govern the construction of a generalised QoS framework, QoS specification which captures application level quality of service requirements, and QoS mechanisms which realise desired end-to-end QoS behaviour.

2.1 QoS Principles

Five principles motivate the design of a generalised QoS framework. First, the *integration principle* states that quality of service must be configurable, predictable and maintainable over all architectural layers to meet end-to-end quality of service [2]. Flows[2] traverse resource modules (e.g., CPU, memory, devices, network, etc.) at each layer from source

1. Were appropriate we have adopted the standard terminology of the ISO QoS Working Group [73].

media devices, down through the source protocol stack, across the network, up through the receiver protocol stack to the playout devices. Each resource module traversed must provide QoS configurability (based on a QoS specification), resource guarantees (provided by QoS control mechanisms) and maintenance of on-going flows. The second principle of *separation principle* states that media transfer, control and management are functionally distinct architectural activities [3]. The principle states that these tasks should be separated in architectural frameworks; one aspect of separation is the distinction between signalling and media-transfer; flows (which are isochronous in nature) generally require a wide variety of high bandwidth, low latency, non-assured services with some form of jitter correction; on the other hand, signalling (which is full duplex and asynchronous in nature) generally requires low bandwidth, assured-type services with no jitter constraint. Next, the *transparency principle* states that applications should be shielded from the complexity of underlying QoS specification and QoS management such as QoS monitoring and maintenance. An important aspect of transparency is the QoS-based API [4] at which desired quality of service levels are stated (see QoS management policy in section 2.2). The benefits of transparency are three-fold: it reduces the need to embed quality of service functionality in applications; it hides the detail of underlying service specification from the application; and it delegates the complexity of handling QoS management activities to the underlying framework. Forth, the principle of *asynchronous resource management* [3] guides the division of functionality between architectural modules and pertains to the modeling of control and management mechanisms; it is necessitated by, and is a direct reflection of fundamental time constraints that operate in parallel between activities (e.g., scheduling, flow control, routing, QoS management, etc.) in distributed communications environments; the "state" of the distributed communication system is structured according to these different time scales. The communication system 'operating point' is arrived at via asynchronous algorithms that operate and exchange control data periodically among each other. The final principle is the *performance principle* which subsumes a number of widely agreed rules for QoS-driven communications implementation that guide the division of functionality in structuring communication protocols for high performance in accordance with Saltzer's systems design principles [5], avoidance of multiplexing [6], recommendations for structuring communications protocols such as application layer framing and integrated layer processing [7], and the use of hardware assists for protocol processing [8] [9].

2.2 QoS Specification

QoS specification is concerned with capturing application level quality of service requirements and management policies. QoS specification is generally different at each system layer and is used to configure and maintain QoS mechanisms resident at each layer. For example, at the distributed system platform level QoS specification is primarily user-oriented rather than system-oriented. Lower-level considerations such as tightness of synchronisation of multiple related flows, or the rate and burst size of flows, or the details of thread scheduling should all be hidden at this level. QoS specification is therefore declarative in nature: users specify what is required rather than how this is to be achieved by underlying QoS mechanisms. Quality of service specification encompasses the following:

- *flow synchronisation specification*, which characterises the degree (i.e., tightness) of synchronisation between multiple related flows [10]. For example, simultaneously recorded video perspectives must be played in precise frame by frame synchrony so that relevant features may be simultaneously observed. On the other hand, lip synchronisation in multimedia flows does not need to be absolutely precise when the main information channel is auditory and video is only used to enhance the sense of presence;
- *flow performance specification,* which characterises the user's flow performance requirements [11]; the ability to guarantee traffic throughput rates, delay, jitter and loss rates, is particularly important for multimedia communications. These performance-based metrics are likely to vary from one application to another; to be able to commit necessary end-system and network resources a QoS framework must have prior knowledge of the expected traffic characteristics associated with each flow before resource guarantees can be met;
- *level of service,* which specifies the degree of end-to-end resource commitment required (e.g, deterministic [12], predictive [13] and best effort). While the flow performance specification permits the user to express the required performance metrics in a quantitative manner, level of service allows these requirements to be

2. The notion of a flow is an important abstraction which underpins the development of QoS frameworks. Flows characterize the production, transmission and eventual consumption of a single media source (viz. audio, video, data) as integrated activities governed by single statements of end-to-end QoS. Flows are simplex in nature and can be either unicast or multicast. Flows generally require end-to-end admission control and resource reservation, and support heterogeneous QoS demands.

refined in a qualitative way as to allow a distinction to be made between hard, firm and soft performance guarantees. Level of service expresses a degree of certainty that the QoS levels requested at flow establishment or re-negotiation will actually be honored;

- *QoS management policy*, which captures the degree of QoS adaptation (continuous or discrete) that the flow can tolerate and scaling actions to be taken in the event of violations in the contracted QoS [14]. By trading-off temporal and spatial quality to available bandwidth, or manipulating the playout time of continuous media in response to variation in delay, audio and video flows can be presented at the playout device with minimal perceptual distortion. The QoS management policy also includes user-level selection of QoS indications in the case of violations in the requested quality of service, and periodic bandwidth, delay, jitter and loss notification (i.e., QoS signals) which are suitable for adaptive applications [24]; and
- *Cost of Service,* which specifies the price the user is willing to incur for the level of service; cost of service is very important factor when considering QoS specification. If there is no notion of cost of service involved in QoS specification, there is no reason for the user to select anything other than maximum level of service [15].

2.3 QoS Mechanisms

Quality of service mechanisms are selected according to user supplied QoS specification, resource availability and resource management policy. In resource management, QoS mechanisms are categorized as either static or dynamic in nature: *static resource management* deals with flow establishment and end-to-end QoS re-negotiation phases (which we describe as QoS provision), and *dynamic resource management* deals with the media-transfer phase (which we describe as QoS control and management). The distinction between the former and latter is due to the different time scales on which they operate and is a direct consequence of the asynchronous resource management principle; control distinguishes itself from management in that it operates on a faster timescale.

2.3.1 QoS Provision Mechanisms

QoS provision is comprised of three components:

i) *QoS mapping* performs the function of automatic translation between representations of QoS at different system levels (i.e., operating system, transport layer, network, etc.) and thus relieves the user of the necessity of thinking in terms of lower level specification. For example, the transport level QoS specification may express flow requirements in terms of level of service, average and peak bandwidth, jitter, loss and delay constraints. For admission testing and resource allocation purposes this representation must be translated to something more meaningful to the end-system scheduler. For example (as illustrated below), one function of QoS mapping is to derives the period, quantum (i.e., unit of work), and deadline times of the threads associated the from transport level flows [16] (see section 4.9 and 4.10).

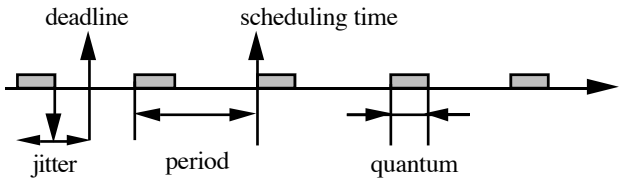

Figure 2-1. QoS Parameters derived by QoS Mapping

ii) *admission testing* is responsible for comparing the resource requirement arising from the requested QoS against the available resources in the system. The decision as to whether a new request can be accommodated generally depend on system-wide resource management policies and simple resource availability. Once admission testing has been successfully completed on a particular resource module, local resources are reserved immediately and then committed later if the end-to-end admission control test (i.e., accumulation of hop by hop tests) is successful.

iii) *resource reservation protocols* arrange for the allocation of suitable end-system and network resources according to the user QoS specification. In doing so, the resource reservation protocol interacts with QoS-based routing to establish a path through the network in the first instance; then, based on QoS mapping and admission control at each local resource module traversed (e.g. CPU, memory, I/O devices, switches, routers, etc.) end-to-end resources are allocated. The end result is that QoS control and management mechanisms such as network-level cell scheduler and transport-level flow monitors are configured appropriately;

2.3.2 QoS Control Mechanisms

QoS control mechanisms operate on timescales close to media transfer speeds. They provide real-time traffic control of flows based on requested levels of QoS established during the QoS provision phase. This is achieved by providing suitable traffic control mechanisms and arranging for time-constrained buffer management and communication protocol operation. The fundamental traffic control building blocks include the following:

- *flow shaping* regulates flows based on user supplied flow performance specifications. Flow shaping can be based on a simple fixed rate throughput (i.e., peak rate) or some form of statistical representation (i.e., sustainable rate and burstiness) the required bandwidth. The benefit of shaping traffic is that it allows the QoS framework to commit sufficient end-to-end resources and to configure flow schedulers to regulate traffic through the end-systems and network. It has been mathematically proven that the combination of traffic shaping at the edge of the network and scheduling in the network can provide hard performance guarantees. Parekh [17] has shown that if a source is shaped by a token bucket with leaky bucket rate control and scheduled by the weighted fair queueing service discipline [18], it is possible to achieve strong guarantees on delay;
- *flow scheduling* manages the forwarding of flows (chunks of media based on application layer framing) in the end-system [19][20][21] and network (packets and/or cells) in an integrated manner [22]. Flows are generally scheduled independently in the end-systems but may be aggregated and scheduled in unison in the network. This is dependent of the level of service and the scheduling scheme adopted;
- *flow policing* can be viewed as the dual of monitoring: the latter - usually associated with QoS management - observes whether QoS contracted by a provider is being maintained whereas the former observes whether the QoS contracted by a user is being adhered to. Policing is often only appropriate where administrative and charging boundaries are being crossed, for example, at a user-to-network interface [23]. A good flow shaping scheme at the source allows the policing mechanism to easily detect misbehaving flows. The action taken by the policing function can range from accepting violations and merely notifying the user, through to shaping the incoming traffic to an acceptable QoS level. We consider that policing flows in the end-system or network should be a function of the end-system or network level scheduling QoS mechanism;
- *flow control* includes both open-loop and closed loop schemes: open loop flow control is used widely in telephony and allows the sender to inject data into the network at the agreed levels given that resources have been allocated in advance; closed loop flow control requires the sender to adjust its rate based on feed-back from the receiver [24] or network [87]. Applications using closed loop flow control based protocols must be able to adapt to fluctuations in the available resources. Fortunately, many multimedia applications are adaptive [25][26] and can operate in such environments. Alternatively, multimedia applications which cannot adjust to changes in the delivered QoS are more suited to open loop schemes where bandwidth, delay and loss can be deterministically guaranteed for the duration of the session; and
- *flow synchronisation* is required to control the event ordering and precise timings of multimedia interactions. Lip-sync is the most commonly cited form of multimedia synchronisation (synchronisation of video and audio flows at a playout device); other synchronisation scenarios reported include: event synchronisation with and without user interaction, continuous synchronisation other than lip-sync, continuous synchronisation for disparate sources and sinks. All place fundamental QoS requirements on flow synchronisation protocols [27]. Dynamic QoS management associated with flow synchronisation is mainly concerned with the ‘tightness’ of synchronisation between flows.

2.3.3 QoS Management Mechanisms

To maintain agreed levels of QoS it is often not sufficient to just commit resources; in addition, QoS management is frequently required to ensure that the contracted QoS is sustained. QoS management of flows is functionally similar to QoS control. However, it operates on a slower time scale; that is, over longer monitoring and control intervals [28]. QoS management mechanisms include the following:

- *QoS monitoring* allows each level of the system to track the ongoing QoS levels achieved by the lower layer. It often plays an integral part in a QoS maintenance feedback loop which maintains the quality of service being achieved by the monitored resource modules. Monitoring algorithms operate over different timescales. For example, they can run as part of the scheduler (as a QoS control mechanism) to measure individual performance of on-going flows. In this case measured statistics can be used to control packet scheduling and for admission control. Alternatively they can operate as part of a transport level feedback mechanism [49];
- *QoS maintenance* compares the monitored quality of service against the expected performance and then exerts tuning operation (i.e., fine or coarse grain resource adjustments) on resource modules to sustain the delivered QoS. Fine grain resource adjustment counters QoS degradation by adjusting local resource modules (e.g., loss via the buffer manager, queueing delays via the flow scheduler and throughput via the flow regulator [2]);
- *QoS degradation* issues a QoS indication to the user when it determines that the lower layers have failed to maintain the QoS of the flow and nothing further can be done by the QoS maintenance mechanism. In response to such an indication the user can choose either to adapt to the available level of QoS or scale to a reduced level of service (i.e., end-to-end renegotiation);
- *QoS signalling* allows the user to specify the interval over which one or more QoS parameters (e.g., delay, jitter, bandwidth, loss, synchronisation) can be monitored and the user informed of the delivered performance via a QoS signal. Both single and multiple QoS signals can be selected depending the user requested QoS management policy; and
- *QoS scalability* comprises *QoS filtering* (which manipulates flows as they progress through the communications system) and *QoS adaptation* (which scales flows at the end-systems only) mechanisms. Many continuous media applications exhibit robustness in adapting to fluctuations in end-to-end quality of service. Based on the user supplied QoS management policy, QoS adaptation in the end-systems can take remedial actions to scale flows appropriately. Resolving heterogeneous quality of service issues is a particularly acute problem in the case of multicast flows. Here individual receivers may have differing capabilities to consume audio-visual flows; QoS filtering helps to bridge this heterogeneity gap while simultaneously meeting individual receivers' quality of service requirements.

3. Layer-specific QoS

In this section we selectively review layer-specific quality of service research considering the distributed systems platform, operating system, and transport and network layers in turn below; see [29] [30] for a more complete survey.

3.1 Distributed Systems Platform

There has been considerable research in the area of distributed systems platform over the past ten years [31]. Until recently, however, there has been very little work on quality of service support in such platforms. With the emergence of distributed multimedia applications, however, quality of service has become a major issue in distributed systems research. In a distributed system, there are three areas where quality of service is applicable: i) message passing services, which allow a programmer to explicitly send a message between two or more processes in a distributed system; ii) remote invocation, which allows operations in a server process to be invoked by a client process [32] [33]; and iii) stream services, which are connections that support the transmission of continuous media flows [34]. A number of experimental QoS-driven distributed systems platforms are now beginning to emerge. Researchers at Lancaster University have developed an extended version of ANSAware [32] featuring bounded invocations and QoS-controlled streams [35]. Similar work has also been undertaken at Cambridge University [36]. More recently, research on quality of service has centered on ODP standardization [34]. Ongoing research at CNET [37], and BBN and Rome Labs [38] are developing new languages to specify QoS for both operational and stream interface. The CNET work uses QoS

logic statements in the language to generate quality of service monitors. The BBN and Rome Lab research promotes object level QoS specification (i.e., methods per second) and not at the communication level (i.e.,bits per second). Both approaches allow quality of service to be negotiated, measured and enforced. For full details on the state of the art in distributed systems support for quality of service see [29].

3.2 Operating Systems

There has been considerable progress in operating systems support for quality of service with most progress having been made in the specific areas of communication protocols [40] and scheduling [20] [107]. There has been considerably less work on the integration of the various components into an overall operating system [16]. Communication protocol implementation involves predictability issues such as the need for correct scheduling of protocol activities and efficiency issues such as minimization of data copying, system calls, interrupt handling and context scheduling, an avoidance of multiplexing, the use of hardware assists for protocol processing and the importance of executing protocol code in a schedulable process rather than as a interrupt service routine. Much of the work has looked to maintain a level of compatibility with the de facto UNIX interface. Two main approaches can be identified: i) modifying existing UNIX implementations, and ii) completely re-implementing UNIX. In the first approach, alterations are made to the existing UNIX kernel to provide more predictable behaviour. In [41] Hanko describes a proposal for *time-driven resource management* which allows applications to signal their likely forthcoming resource requirements in terms of QoS parameters such as quantity deadline and priority. The system will not attempt to guarantee performance, but will instead bias available resources in the requested directions and concentrate on degradation of service, optionally accompanied by notification of degradation to the effected process. The second approach is to re-implement UNIX in terms of the mirco-kernel model. Examples of mirco-kernels capable of supporting UNIX interfaces are Chorus, Mach and Amoeba. Work has been undertaken at CWI, Amsterdam to support continuous media in an Amoeba-based UNIX environment [42]. Other significant work is being carried out using Mach [107] [21], Chorus[16], Peagus [43] and YARTOS [88] (and its rate-based extensions [89]) as the basis of a distributed system with end-to-end QoS support for continuous media. In the area of device management several research groups have suggested a new architecture for multimedia systems in which multimedia devices, that typically reside on the workstation I/O bus are placed on a high-speed ATM network. These architectures are typically referred to as Desk Area Network (DANs) [39] [90]. It is still too early to determine what if any new requirements DAN architecture places on future operating system design. For full details on the state of the art in operating systems support for quality of service see [1].

3.3 Transport Layer

A large number of research teams have investigated the provision of quality of service at the transport layer. Early work specifically addressed the provision of rate based protocols over high speed networks, e.g., XTP [8] and NetBlt [44]. More recently protocols have emerged which are designed specifically to meet the needs of continuous media. The Esprit OSI 95 project has proposed an enhanced transport service and protocol called TPX [45]. TPX provides support for connection-oriented services with sequenced delivery, QoS configurable and renegotiable QoS, and error notification. The enhanced connection-oriented service takes QoS parameters relating to throughput, delay, delay jitter, error selection policy and relative priority. Three transport quality of service semantics in addition to "best effort" are proposed for this service: compulsory, threshold and maximal QoS. The Tenet Group at the University of California at Berkeley have developed CMTP [46] which operates on top of RTIP [75] and provides sequenced and periodic delivery of continuous media samples with QoS control over throughput, delays and error bounds. Notification of all undelivered and/or corrupted data can be provided if the client selects this option. The HeiTS project [47] at IBM Heidelberg has developed a transport system which has concentrated on the integration of transport QoS and resource management (primarily CPU scheduling). HeiTS puts considerable emphasis on an optimized buffer pool which minimizes copying and also allows efficient data transfer between local devices. Other significant work has come from Schulzrinne, Casener and Van Jacobson who have developed RTP [48] for the Internet suite of multimedia tools [26]. Other work [49] reports on the development of a continuous media transport and orchestration service. For a full review of the state of the art in transport protocols and services see [40] [50].

3.4 Network Layer

The subject of providing quality of service guarantees in integrated service networks has been widely covered in the literature [51]. The multimedia networking community has developed sophisticated traffic models, control and management architectures for multimedia communications. Extensive work has considered flow specification, flow admission control, resource reservation, traffic shaping and queue management schemes. For researchers working on

multimedia networking, the primary aim is to provide performance bounds while exploiting statistical multiplexing of bursty sources to efficiently utilise bandwidth. Kurose [52] provides a good categorisation of the different approaches used in providing QoS guarantees found in the literature: (i) a *tightly controlled approach*, which is based on non-work conserving multiplexing service (queueing) disciplines (e.g., stop-and-go [53] and Tenet's EDD [12]), and which preserves the traffic shape guaranteeing delivered flow characteristics are the same as the source; (ii) an *approximate approach*, which as its names suggests is based on simple characterisation of the source model (e.g., equivalent capacity [54]) and which can provide approximate guarantees using simple service disciplines such as FIFO taking advantage of statistical multiplexing gain; (iii) a *bounding approach*, which takes into any account distortion of the flow as traverses work-conserving multiplexers (e.g., packetised generalised processor sharing [17] and weighted fair queueing [18]) resulting in mathematically provable performance bounds for statistical and deterministic service guarantees [55]; and finally (iv) an *observation-based approach*, which uses measured behaviour (e.g., COMET's approach [56] and Clark's predictive service [13]) of the aggregate traffic and the user supplied flow specification when making admission decisions.

The work on an integrated services Internet [57] is a significant contribution to providing QoS guarantees on a per-flow basis. The integrated service model comprises four components: (i) a *packet scheduler*, which is based on the CSZ scheduler [13] and Class Based Queueing (CBQ) [58], and which forwards packet streams using a set of queues and timers; (ii) a *classifier*, which maps each incoming packet into a set of QoS class (iii) an *admission controller*, which implements the decision control algorithm to determine whether a new flow can be admitted or denied; (iv) a *reservation setup protocol*, which is necessary to create and maintain flow-specific state in the end-systems and in routers along the path of the flow. There have been a number of significant contributions to reservation protocols in communication networks which have emerged over the past few years: ST-II [59] and SRP [78], and more recently RSVP [60], RCAP [61] and HieRAT [62] and UNI 3.0 [23]. For a full review of the state of the art in network support for QoS see [51] [52].

4. QoS Architectures

Until recently research in providing QoS guarantees has mainly focused on network oriented traffic models and service scheduling disciplines. These guarantees are not, however, end-to-end in nature. Rather they preserve QoS guarantees only between network access point that end-systems are attached to [63]. Work on QoS-driven end-system architecture needs to be integrated with network configurable QoS services and protocols to meet application-to-application requirements. In recognition of this, researchers have recently proposed new communication architectures which are broader in scope and cover both network and end-system domains. In this section we review[1] a number of distinct approaches which have recently emerged in the literature and which are born out of the telecommunications, computer communications and standards communities:

- *Extended Integrated Reference Model (XRM)*, which is being developed at Columbia University;
- *Quality of Service Architecture (QoS-A)*, which is being developed at Lancaster University;
- *OSI QoS Framework*, which is being developed by the ISO SC21 QoS Working Group;
- *Heidelberg QoS Model*, which is being developed at IBM's European Networking Center;
- *OMEGA Architecture*, which is being developed at the University of Pennsylvania;
- *TINA QoS Framework*, which is being developed by the TINA Consortium;
- *IETF QoS Manager (QM)*, which is being developed by the IETF Integrated Services Working Group;
- *Tenet Architecture*, which is being developed at the University of California at Berkeley;
- *MASI End-to-End Architecture*, which is being developed at Université Pierre et Marie Curie; and
- *End System QoS Framework*, which is being developed at Washington University.

1. See [95] for a full survey of QoS architectures.

4.1 The Columbia Extended Integrated Reference Model

The COMET group at Columbia University (New York) are developing an Extended Integrated Reference Model (XRM) [64] as a modeling framework for control and management of multimedia telecommunications networks (which comprise *multimedia computing platforms* and *broadband networks*). The COMET group argue that the foundations for operability (i.e., control and management) of multimedia computing and networking devices are equivalent; that is, both classes of devices can be modeled as producers, consumers and processors of media.The only difference is the overall goal that a group of devices has set to achieve in the network or end-system. COMET organizes the XRM into five distinct planes as illustrated in Figure 4-1:

- *management function*, which resides in the network management plane (N-plane), covers the OSI functional areas of network and system management;
- *traffic control function*, which comprises the resource control (M-plane) and connection management and control (C-plane) planes. Resource control constitutes cell scheduling, call admission, call routing in the network and, process scheduling, memory management, routing, admission control and flow control in the end-systems;
- *information transport function*, which is located in the user transport plane (U-plane), models the media protocols and entities for the transport of user information in both the network and the end- systems; and
- *telebase*, which resides in the data abstraction and management plane (D-plane), collectively represents the information, data, abstractions existing in the network and end-systems. The telebase implements the principles of data sharing (via asynchronous resource management) among all other XRM planes.

The subdivision of XRM into these different planes is motivated by a number of QoS principles: the separation and layering principles [3], and the asynchronous resource management principle [3]. The subdivision between the management and traffic control functions, on one hand, and the information transport functions on the other, is based on the principle of separation between control and communication. The separation between management and traffic control is due to the different timescales on which these planes operate; this is in turn motivated by the asynchronous resource management principle.

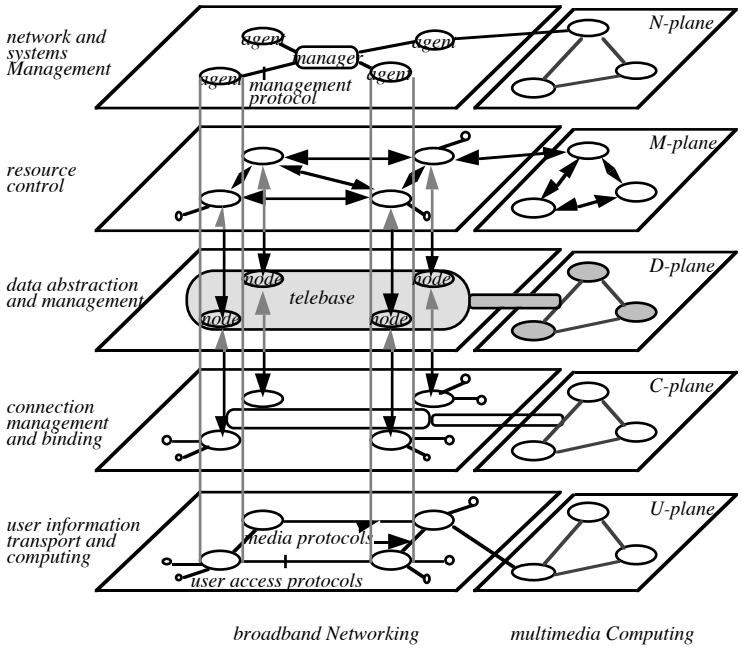

Figure 4-1. XRM

The XRM is built on sound theoretical work of guaranteeing QoS requirements in ATM networks and end-systems populated with multimedia devices. General concepts for characterising the capacity of network [65] and end-system [66] devices (e.g., disks, switches, etc.) have been developed. At the network layer, XRM characterises the capacity region of an ATM multiplexer with QoS guarantees as a *schedulable region*. Network resources such as switching bandwidth and link capacity are allocated based on four cell-level traffic classes (class I, II, III, and C) for circuit emulation, voice and video, data, and network management respectively. A traffic class is characterised by its statistical properties and QoS requirements. Typically QoS requirements reflect cell loss and delay constraints. In order to efficiently satisfy the QoS requirements of the cell level, scheduling and buffer management algorithms dynamically allocate communication bandwidth and buffer space appropriately.

The schedulable region represents the multidimensional capacity of the multiplexer; its dimensionality depends on the number of traffic classes and represents the stability region. The schedulable region is a resource abstraction that allows a separation of times scales: the time scales of cells and the time scale of call arrivals and departures. In [65] it is shown how separation of time scales is an appropriate tool for resolving admission control decisions. Based on a calculus of schedulable regions, the QoS in the network can be guaranteed. The three traffic classes in Figure 4-2 correspond to video, voice and data flows. Class I traffic is characterised by a frame duration of 62.5 ms and a peak rate of 10 Mbps, Class II traffic is modelled as an on-off source with constant arrivals with an exponentially distributed active period and 64 Kbps peak rate, and Class III traffic is modeled as a Poisson source with 1 Mbps average rate. The surface depicted in Figure 4-2 delimits the capacity region of the multiplexer. Any combination in the number of calls (i.e., active flows) below this surface has its QoS guaranteed.

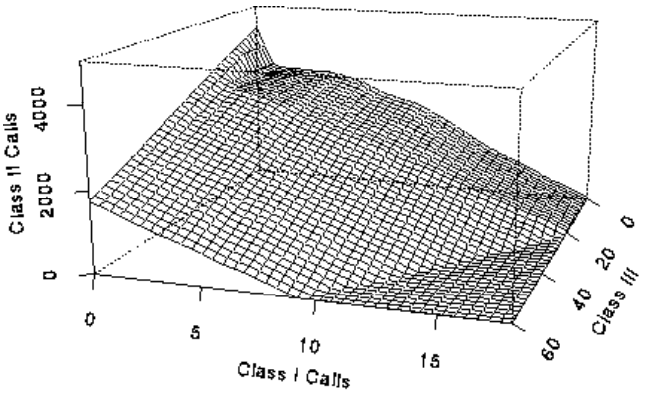

Figure 4-2. The Schedulable Region of a Multiplexer with Three Traffic Classes.

XRM models the end-system architecture as a multiprocessor based multimedia workstation, comprising the following multimedia devices: (i) an *audio and video unit*, which is responsible for multimedia processing, and supports media processing tasks in a deterministic manner, and runs on a dedicated processor(s); (ii) an *input/output subsystem* is similarly modeled, separately through a disk storage unit, and is also run on a separate processor(s); (iii) a *main processor unit* runs the system tasks, both to increase speed and to remove external interrupts, as well as the other operating system overhead associated with application tasks. In the end-system, flow requirements are modeled through service class specifications with QoS constraints. For example, in the audio video unit the service class specification is in terms of JPEG, MPEG-I, MPGE-II video and CD audio quality flows with QoS guarantees. Quality of service for these classes is specified by a set of frame delay and loss constraints.The methodology of characterising network resources is extended to the end-system to represent the capacity of multimedia devices. Using the concept of a *multimedia capacity region* the problem of scheduling flows in the end-system becomes identical to the real-time bin packing exercise of the network layer. One significant difference between the schedulable region and the multimedia capacity region is the number of classes supported. The number of service classes at the user level is expected to far exceed the number of traffic classes at the multiplexer. A number of service classes, however, can be mapped onto a single traffic class of the multiplexer, and therefore, supporting a large number of service classes will not require an increase in the number of traffic classes.

The implementation of XRM including key resource abstractions such as the schedulable and multimedia capacity region is currently being realised as part of a *binding architecture* [67]. The binding architecture achieves seamless binding between networking and multimedia devices. The building blocks of the architecture consist of a set of interfaces, methods and primitives. The former abstract the functionalities of multimedia networking devices and are organised into a *binding interface base (BIB)*. The methods and primitives are invoked for implementing binding applications. Communication between the interfaces of the architecture is supported by OMG's CORBA [69]. Binding requirements arise in each of the planes of the XRM. Dynamic binding requirements, however, are particularly demanding in the C and M planes of the XRM. The binding architecture resides in the M, D and C-planes of the XRM. Specifically, the binding interface base resides in the D-plane and the binding algorithms execute from within the M and C-planes. The binding architecture represents a software environment on top of which binding applications execute. Examples of binding applications arise in connection set up for broadband networks, distributed systems implementing flow synchronisation protocols, resource allocation protocol such as those intended for the Internet [60], multimedia computing platforms, etc. New binding applications can be added without changing the underlying binding architecture.

4.2 The Lancaster Quality of Service Architecture

Over the last three years the QoS-A Project at Lancaster University [2] has been developing a Quality of Service Architecture (QoS-A) in co-operation with ATM switch manufacture GDC (formally Netcomm Ltd). The QoS-A promotes the idea of integrated QoS, spanning the end-systems and network, and takes the support of audio and video flows as its primary design goal. The QoS-A is a layered architecture of services and mechanisms for quality of service (QoS) management and control of continuous media flows in multiservice networks. The architecture incorporates the following key notions: *flows*, which characterise the production, transmission and eventual consumption of single media streams (both unicast and multicast) with associated QoS; *service contracts*, which are binding agreements of QoS levels between users and providers; and *flow management*, which provides for the monitoring and maintenance of the contracted QoS levels. The realisation of the flow concept demands active QoS management and tight integration between device management, end-system thread scheduling, communications protocols and networks. The QoS-A is based on a set of principles that govern the realisation of end-to-end QoS in a distributed systems environment: the integration, separation, transparency and performance principles.

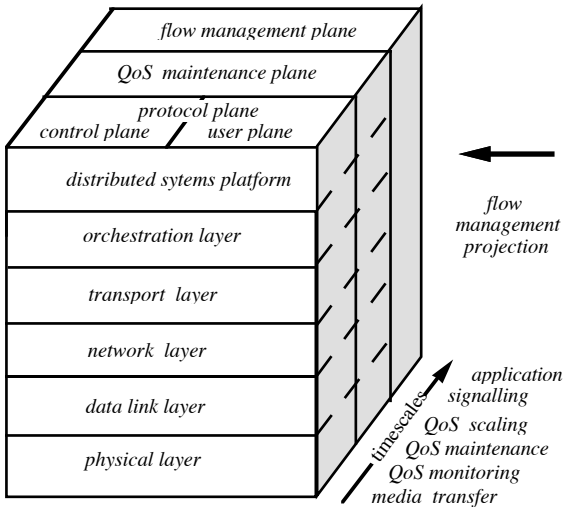

Figure 4-3. QoS-A

In functional terms, the QoS-A (as illustrated Figure 4.3) is composed of a number of layers and planes. The upper layer consists of a distributed applications platform augmented with services to provide multimedia communications and QoS specification in an object-based environment [16]. Below the platform level is an orchestration layer which provides jitter correction and multimedia synchronisation services across multiple related application flows [49]. Sup-

porting this is a transport layer which contains a range of QoS configurable services and mechanisms. Below this, an internetworking layer and lower layers form the basis for end-to-end QoS support. For full details on the QoS-A see [68].

QoS management is realised in three vertical planes in the QoS-A. The protocol plane, which consists of distinct user and control sub-planes, is motivated by the principle of separation. QoS-A uses separate protocol profiles for the control and media components of flows because of the essentially different QoS requirements of control and data. The QoS maintenance plane contains a number of layer specific QoS managers. These are each responsible for the fine grained monitoring and maintenance of their associated protocol entities. For example at the orchestration layer, the QoS manager is interested in the tightness of synchronisation between multiple related flows. In contrast, the transport QoS manager is concerned with intra-flow QoS such as bandwidth, loss, jitter and delay. Based on flow monitoring information and a user supplied service contract, QoS managers maintain the level of QoS in the managed flow by means of fine grained resource tuning strategies. The final QoS-A plane pertains to flow management, which is responsible for flow establishment (including end-to-end admission control, QoS based routing and resource reservation), QoS mapping (which translates QoS representations between layers) and QoS scaling (which constitutes QoS filtering and adaptation for coarse grained QoS maintenance control).

Recent work on the QoS-A has concentrated on realising the architecture in an environment comprising an enhanced Chorus micro-kernel [16], and an enhanced multimedia transport service and protocol [68] in the local ATM environment. While the model is end-to-end in design, the main contribution of the work is end-system architecture. This includes a *multimedia enhanced transport system (METS)*, a transport service contract and associated operating systems support. At the transport layer, the support of QoS is dependent on interactions between the transport protocol, transport QoS manager, the flow management plane and the network layer. The transport service contract subsumes the well accepted QoS parameters of jitter, loss, delay and throughput, but also allows the QoS specification of a wider range of options. These are characterised in terms of the following six contractual clauses:

i) *flow specification* characterises the user's quantitative traffic performance requirements in terms of token bucket characterisation of throughput, jitter, delay and loss, and media characterisation in terms of a flow-id and media type;

ii) *QoS commitment* specifies the degree of resource commitment required from the lower layers; three classes of service are provided: best effort, adaptive [14] and deterministic;

iii) *QoS scaling policy* identifies the QoS adaptation [83] and QoS filtering [82] options in addition to actions to be taken in the event quality of service violations in the contracted service;

iv) *QoS maintenance* selects the degree of monitoring and active QoS maintenance required;

v) *resource reservation* provides either on-demand, fast reservation or advanced reservation services; and

vi) *cost* specifies the price the user is willing to incur for the service requested.

It is the responsibility of the transport protocol to share communications resources in end-systems among flows with widely different QoS commitments. To meet this need the protocol incorporates buffer sharing, rate regulation, scheduling, and basic flow monitoring modules as illustrated in Figure 4-4. Also included is a resource management component responsible for overseeing the allocation and adaptation of the protocol resources. The buffer management scheme is structured to avoid copies across layers and uses separate pools for each commitment type. The transport protocol and transport-level QoS manager are tightly coupled to operate in the same time domain. In essence, the transport protocol monitors a flow's on-going performance and the transport QoS manager maintains it. The transport protocol's monitoring mechanism is able to build up a statistical representation of the end-to-end QoS using the performance data supplied in the transport control messages. The resulting flow statistics represents the actual end-to-end QoS experienced by the receivers. The transport QoS manager uses this information and a user supplied flow spec for fine grained QoS tuning. The flow management plane is responsible for a number of static and dynamic QoS control and management functions. The major functions consist of the provision of network signalling infrastructure, end-to-end admission control, and QoS scaling for course grained QoS management based on a user supplied QoS scaling policy [14]. The flow management plane also performs other management functions such as the mapping of QoS rep-

resentation between layers, the support of the on demand, fast and advanced reservation services. For full details of QoS mapping, end-to-end admission control testing and resource reservation in the context of the Lancaster Chorus and ATM environment see [16].

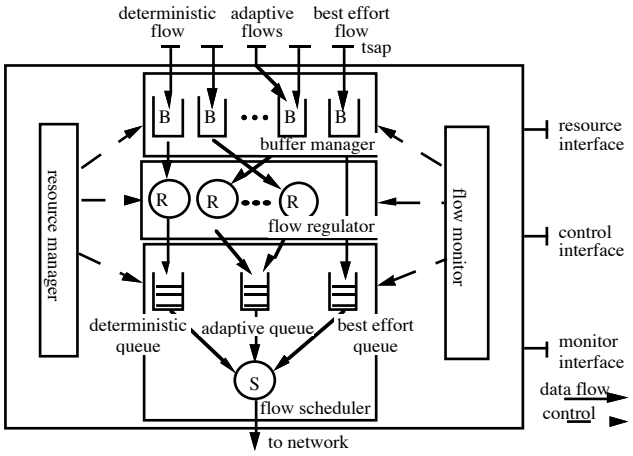

Figure 4-4. User Plane Transport-level QoS Mechanisms and Interfaces

In [14] the QoS-A work is extended by populating the QoS management planes of the architecture with a framework for the control and management of multi-layer coded flows operating in heterogeneous multimedia networking and multicast environments. Two key techniques are proposed: an end-to-end rate shaping scheme which adapts the rate of MPEG-coded flows to the available network resources while minimising the distortion observed at the receiver, and an *adaptive network service*, which offers "hard" guarantees to the base layer of multi-layer coded flows, and "fairness" guarantees to the enhancement layers based on a bandwidth allocation technique called *weighted fair sharing*.

4.3 The OSI QoS Framework

One early contribution to the field of QoS-driven architecture is the *OSI QoS Framework* [73] which concentrates primarily on quality of service support for OSI communications [74]. The OSI framework broadly defines terminology and concepts for QoS and provides a model which identifies objects of interest to QoS in open system standards. The QoS associated with objects and their interactions is described through the definition of a set of QoS characteristics. The key QoS framework concepts include:

- *QoS requirements*, which are realized through QoS management and maintenance entities;
- *QoS characteristics*, which are a description of the fundamental measures of QoS that have to be managed;
- *QoS categories*, which represent a policy governing a group of QoS requirements specific to a particular environment such as time-critical communications; and

QoS management functions, which can be combined in various ways and applied to various QoS characteristics in order to meet QoS requirements.The OSI QoS framework (illustrated in Figure 4-5) is made up of two types of management entity that attempt to meet the QoS requirements by monitoring, maintaining and controlling end-to-end QoS:

i) *layer-specific entities*: The task of the policy control function is to determine the policy which applies at a specific layer of the open system. The policy control function models any priority actions that must be performed to control the operation of the layer. The definition of a particular policy is layer-specific and therefore cannot be generalized. Policy may, however, include aspects of security, time-critical communications and resource control. The role of the QoS control function is to determine, select and configure the appropriate protocol entities to meet layer-specific QoS goals.

ii) *system-wide entities*: The system management agent is used in conjunction with OSI systems management protocols to enable system resources to be remotely managed. The local resource manager represents end-system control of resources. The system QoS control function combines two system-wide capabilities: to tune performance of protocol entities and to modify the capability of remote systems via OSI systems management. The OSI systems management interface is supported by the systems management manager which provides a standard interface to monitor, control and manage end-systems. The system policy control function interacts with each layer-specific policy control function to provide an overall selection of QoS functions and facilities.

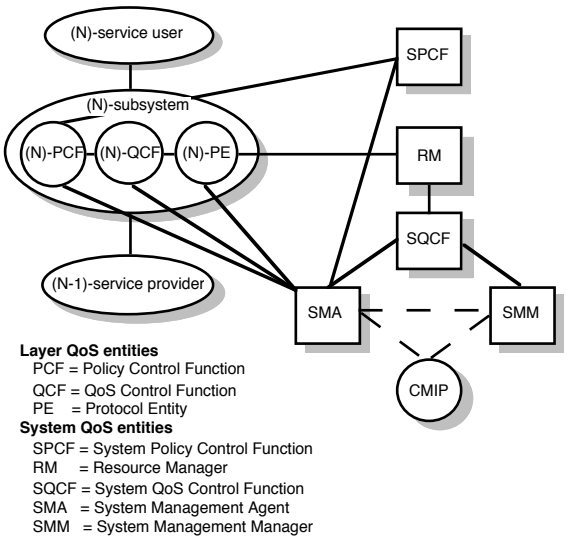

Figure 4-5. OSI QoS Framework

4.3.1 The Heidelberg QoS Model

The HeiProject at IBM's European Networking Center in Heidelberg have developed a comprehensive QoS model which provides guarantees in the end-systems and network [62]. The communications architecture includes a continuous media transport systems (HeiTS/TP) [47] which provides QoS mapping and media scaling [77] as illustrated in Figure 4-6. Underlying the transport is an internetworking layer based on ST-II which supports both guaranteed and statistical levels of service; in addition the network supports QoS-based routing (via a QoS finder algorithm) and QoS filtering. Key to providing end-to-end guarantees is *HieRAT (resource administration technique)*: what is based on initial work in [78]. HeiRAT comprises a comprehensive QoS management scheme which includes QoS negotiation, QoS calculation, admission control and QoS enforcement, and resource scheduling [62]. The HeiRAT scheduling policy used in the supporting operating system is a rate-monotonic scheme whereby the priority of an operating system thread performing protocol processing is proportional to the message rate accepted.

The Heidelberg QoS model has been designed to handle heterogeneous QoS demands form individual receivers in a nulticast group and to support QoS adaptivity via flow filtering and media scaling respectively. Media scaling [85] and codec translation [48] at the end systems, and flow filtering [82] [83] [62] and resource sharing [60] in the network are fundemental to meeting heterogeneous QoS demands. Media scaling matches the source with the receivers' QoS capability by manipulating flows at the network edges. In contrast, filtering accommodates the receivers' QoS capability by manipulating flows at the core of the network as they traverse bridges, switches and routers. Both schemes compensate for variation in network load/performance by re-scaling or filtering the delivered QoS respectively. Potentially this includes manipulating hierarchical flows; for example, delivering the I frames of an MPEG encoded flow and dropping the P and B frames to match the end system or network QoS constraints. Network level filtering looks very promising when used in conjunction with multicast protocols for dissemination of continuous media in support of heterogeneous receivers; for example, Pasquale et. el. [82] suggest that several receivers having disparate QoS commu-

nication requirements, and needing to access the same video flow simultaneously can be supported by a propagating filter scheme which deliver the appropriate QoS to each receiver. This scheme promotes efficient use of network resources, and as the literature suggests, reduces the likelihood of the on set of congestion

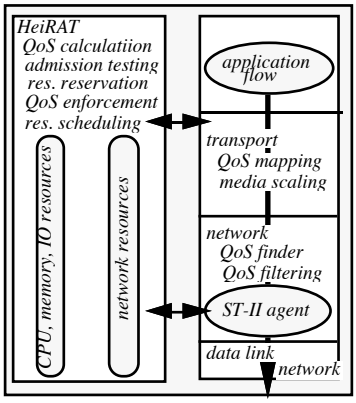

Figure 4-6. Heidelberg QoS Model

4.4 Other QoS Architectures

4.4.1 The OMEGA Architecture

Over the last three years the University of Pennsylvania has been developing a end-point architecture called the OMEGA architecture [84]. OMEGA is the result of an interdisciplinary research effort that is examining the relationship between application QoS requirements (which make stringent resource demands) and the ability of local (the operating system) and global resource management (combining communication and remotely managed resources) to satisfy these demands. The OMEGA architecture assumes a network subsystem which provides bounds on delay, errors and can meet bandwidth demands, and an operating system which is capable of providing run time QoS guarantees. The essence of the OMEGA architecture is resource reservation and management of end-to-end resources. Communications is preceded by a call setup phase where application requirements, expressed in terms of QoS parameters, are negotiated, and guarantees are made at serveral logical levels, such as between applications and the network subsystem, applications and the operating system, network subsystem and operating system. This establishes customized connections and results in the allocation of resources appropriate to meet application requirements and operating system/ network capabilities. To facilitate this resource management process the University of Pennsylvania have also developed a brokerage model [80] which incorporates QoS translation, and QoS negotiation and renegotiation (see [81] for full details on similar work on QoS negotiation protocol at University of Montreal).

4.4.2 The TINA Quality of Service Framework

The TINA QoS Framework [70] describes a framework for specifying QoS aspects of distributed telecommunications within the context of the Computing Architecture. The QoS framework addresses the computational and engineering viewpoints of distributed telecommunications applications. It is governed by the separation between telecommunication applications and the *Distributed Processing Environment (DPE)* in the first instance; that is multimedia services offered by a provider utilise the DPE and underlying computing and communications capabilities.The TINA QoS framework is partly based on work in the literature (e.g., ANSA QoS Framework [71] and CNET Framework [37]). In the computational viewpoint, QoS parameters required to provide guarantees to objects are stated declaratively as *service attributes*. In the engineering model, QoS mechanisms employed by resource managers are considered. By stating QoS requirements declarative, applications are relieved of the burden of coping with complex resource management mechanisms needed for ensuring QoS guarantees; this is motivated by the principle of QoS transparency.

4.4.3 The IETF QoS Manager

In [79] Clark introduces some early work on a *Quality of Service Manager (QM)* for an integrated services Internet suite of protocols. The QM (illustrated in Figure 4-7) presents an abstract management layer designed to isolate applications from underlying details of specific services provided in an QoS-driven Internet [57]. One motivating factor behind the introduction of a QM is that applications can negotiate desired QoS without needing to know the details of a specific network service; in this case, the QM provides a degree of transparency whereby applications express desired levels of QoS in user-oriented language rather than using communication specifics. The QM is responsible for determining what QoS management capabilities are available on the application's communication path, and choosing the path best suited to the application. A number of benefits are gained by migrating specific services knowledge from the application to the QM:

- *heterogeneity* is supported; the QM can match application needs to the underlying QoS capability;
- *transparency* is provided; applications will not need to be aware of the details of specific QoS management capability; and
- *extensibility* is supported; new QoS capabilities can be more easily deployed in the Internet, because applications need not be modified as new services become available.

The initial thrust of the work will be to map application specific needs to one of the new set integrated services (e.g., [79]) and provide some support for monitoring of performance. In the future, however, the interface between the application and QM may cover more general issues such as cost of service, as well as more technical matters such as delay and bandwidth. In related work, Partridge [79] presents a multimedia-based Berkeley Sockets specification which includes support for flows in terms of a flow-spec, and QoS management aspects of Clark's QM.

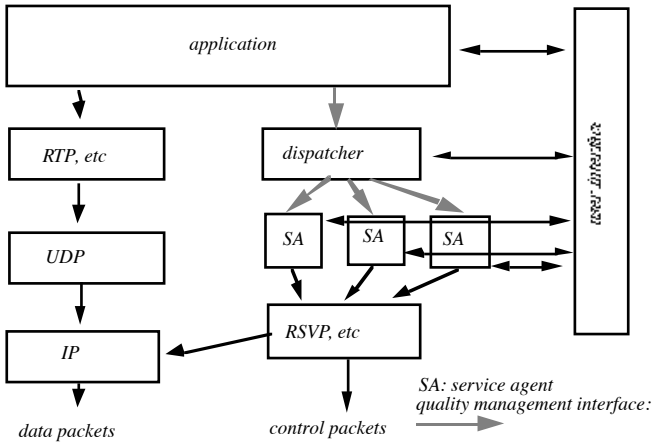

Figure 4-7. IETF QoS Manager

4.4.4 The Tenet Architecture

The Tenet Group at the University of California at Berkeley have developed a family of protocols [75] [76] which run over an experimental wide area ATM network. The protocol family includes a Real Time Channel Administration Protocol (RCAP) [61] in addition to Real Time Internet Protocol (RTIP), Continuous Media Transport Protocol (CMTP) [46]. The former provides generic connection establishment, resource reservation and signaling functions for the rest of the protocol family. RCAP spans the transport and network layers for overall resource reservation and flow setup. CMTP is explicitly designed for continuous media support. It is a lightweight protocol which runs on top of RTIP and provides sequenced and periodic delivery of continuous media samples with QoS control over throughput, delays and error boundswhich is being developed at the University of California at Berkeley. The Tenet Group [12] make a distinction between deterministic and statistical guarantees for hard real-time and continuous media flows respectively.

In the deterministic case, guarantees provide a hard bound on the performance of all cells within a session. Statistical guarantees promise that no more than x% of packets would experience a delay greater than specified, or no more that x% of cells might in a session might be lost.

4.4.5 The MASI End-to-End Architecture

The CESAME Project [72] at Laboratoire MASI, Université Pierre et Marie Curie, is developing an architecture for multimedia communications which takes end-to-end QoS support as it main objective. As with the Lancaster QoS-A the MASI architecture offers a generic QoS framework to specify and implement the required QoS requirements of distributed multimedia applications operating over ATM-based networks. The CESAME Project considers end-to-end resource management which span the host operating system, host communication subsytem and ATM networks. The research is motivated by i) the need to map QoS requirements from the ODP layer to specific resource modules in a clean and efficient manner; ii) the need to resolve multimedia synchronisation needs of multiple related ODP streams [34]; and iii) the need to provide suitable communication protocol support for multimedia serviceswhich is being developed at Université Pierre et Marie Curie;

4.4.6 The Washington University End System QoS Framework

End System QoS Framework, Other significant work at Washington University by Gopal and Purulkar [63] has developed a QoS framework for providing QoS guarantees within the end-system for networked multimedia applications. There are four components of the Washington University end system QoS framework: QoS specification, QoS mapping, QoS enforcement and protocol implementation. QoS specifications are at a high level and use a small number of parameters to allow applications greater ease in specifying their flow requirements. Based on QoS specification, QoS mapping operations derive resource requirements for each end-to-end session of the application. Important resources considered are the CPU and network connectionwhich is being developed at Washington University. The third component of framework is QoS enforcement. QoS enforcement is mainly concerned with providing real-time processing guarantees for media transfer. A real-time upcall (RTU) facility [85] has been developed to for structuring protocols. RTUs are scheduled using a rate monotonic policy with delayed pre-emption that takes advantage of the iterative nature of protocol processing to reduce context switching overhead and increase end-system scheduling efficiency. The final component of the framework is an application level protocol implementation model. Protocol code is structured as RTUs with attributes that are derived from high level specifications by QoS mapping operations.

5. Comparison

In this section we present a simple qualitative comparison of QoS architectures survey in section 4. We use the elements of the generalised QoS framework (descided in section 2) as the basis for the comparison shown in Table 1. The legend for the comparison is is as follows:

QoS Model [Ref]	QoS Provision			QoS Control					QoS Management	
QoS Mechanism	QoS Mapping	Adm. Control / Resource allocation	E2E Coordination	Flow Scheduling	Flow Shaping	Flow Control	QoS Filtering	Flow Synchronization	Monitoring / Alerts	QoS Maintenance
XRM [67]	E N	E N	(E) N	(E) N	-	N	-	-	N	-
QoS-A [68]	E N	E (N)	E N	E (N)	E	(E)	(E) N	E	E Sig D	E N R S
ISO [73]	(E) (N)	E N	E N	-	-	-	-	-	E N	E N
H'berg [62]	(E) (N)	E N	E N	E (N)	(E)	(N)	N	-	E D	E R S
TINA [70]	(E)	(N)	N	-	-	-	-	(N)	(N)	-

Table 1: Comparison of QoS Models

QoS Model [Ref]	QoS Provision			QoS Control					QoS Management	
QoS Mechanism	QoS Mapping	Adm. Control / Resource allocation	E2E Coordination	Flow Scheduling	Flow Shaping	Flow Control	QoS Filtering	Flow Synchronization	Monitoring / Alerts	QoS Maintenance
IETF [79]	E N	-	E	-	-	-	-	-	E N	E N R
Tenet [75]	E N	N	N	N	N	(E)	N	-	E D	E R S
MASI [72]	E (N)	E (N)	E	E	-	-	-	E	E	E
Omega [80]	E, (N)	E, (N)	E (N)	E (N)	E	E	-	-	E	E R
WashU [63]	E	E		E	E	-	-	-	-	E R

Table 1: Comparison of QoS Models

-	"not addressed"
E/N	"addressed in detail in the end-system/network"
(E)/(N)	"mentioned only in the end-system/network"
R	"QoS renegotiation addressed in detail"
(R)	"QoS renegotiation mentioned only"
S	"QoS scaling addressed in deatil"
D	"QoS degradation addressed in detail"
(D)	"QoS degradation mentioned only"
Sig	"QoS signalling in detail"

The term "E2E coordination" refers to the coordination of end-system and network resources for flows. This could be provided by a resource reservation protocol (al la RSVP [60]), connection setup protocol (al la RCAP [61] [93]) or signalling protocol (al la Q.2931 [23]).

6. Discussion

All of the QoS architectures cited in this paper, with the exception of the IETF QM model (which only presents an interface for QoS management to the application) consider extending the classic end-to-end protocol argument from the network to include the end-system too. The QoS architectures reviewed in section 4 differ in serveral ways. This may be simply a product of the particular communities which have developed these architectures. For example, the XRM emerged from telecommunication community, the QoS-A and Heidelberg QoS model from the computer communications community, and the ISO QoS framework and IETF QoS manager from the standard communities. Therefore, it would be inappropriate to declare one approach "better" than another.

Even architectures emanating from the same community differ widely. For example, XRM is network centric and is based on a deep understanding of teletraffic theory while, in contrast, the TINA QoS Framework is strong on the application of distributed systems technology to resolve the end-to-end QoS problem but does not quantitatively address end-to-end resource management issues. While commonalities exits between QoS architectures described in section 4 a comprehensive comparison of all architectures is beyond the scope of this paper. In this section, however, we discuss serveral important open issues that emerged during the comparison.

6.1 QoS Specification

All QoS architectures consider QoS specification (e.g., contracts, flow specification, and service and traffic classes, etc.) to be fundamental in capturing user level QoS requirements. Some architectures consider QoS specification at different logical layers or planes in the end-system and network. In this case QoS mapping is used to translate QoS

specifications between logical layers/planes. Although there is broad consensus on the need for a flow specification, which captures quantitative performance requirements, there exist two schools of thought on what it should be. On the one hand, XRM and ATM [23] solutions are based on a flow specification that is made up of one or two QoS parameters that identify traffic class and average bandwidth. On the other hand, the Tenet, QoS-A and OMEGA architectures adopt a multi-valued flow spec (cf. RFC1633, ST-II, RSVP, HieTS).While both of these proposals seem similar, philosophically they are rather different. The COMET group [67] argue that by limiting flows to a set of well define services (in the end-system) and traffic classes (in the network) complexity in the end-system and network is manageable. In contrast, Tenet, QoS-A and Omega architectures consider such an approach unnecessarily limiting. These groups argue that by defining a set of discrete QoS classes applications may be unduly constrained to conform to a QoS class which may not meet its desired QoS requirements. In summary, the first school of thought holds that all flows fall into a small set of general service and traffic classes with well defined delay, burstiness and loss characteristics. The other schcool holds that flows are application specific and that traffic classes will change constantly as an when new applications are developed [61]. In [76] Ferarri counter argues that the latter approach can emulate the former: for example, it is easy to provide a menu of traffic classes above ST-II or Tenet suite of protocols, present it to the user, and extend it whenever needed. It remains unclear, however, whether networks can manage the complexity introduced when a continuum of choice is made available to applications - as advocated by the second school of thought.

6.2 Level of Service

Level of service (section 2.2) expresses the degree of certainty that the QoS levels specified in a flow spec will be honoured. Each architecture offers a different set of services to applications. Terminology used to describe level of service in the literature includes: service class, traffic class, QoS commitment, application class, QoS class, etc. For example, the Washington University QoS Framework supports three application classes which it maps applications level flows into; these include: i) *an isochronous class*, which is suitable for continuous media flows; ii) *a burst class*, which is suitable for bulk data transfer; and iii) *a low delay class*, which is suitable for applications that require a small response time such as an RPC request. The Washington QoS Framework assumes that all applications fall into one of these three general classes. In contrast, the QoS-A supports three levels of service (viz. best effort, adaptive, guaranteed) called QoS commitment. While all architectures provide services based on both hard (i.e. guaranteed service) and soft (i.e., best effort) QoS guarantees it is difficult to determine which set optimally covers the application base. Additional services found in the literature include the predicted service (IEFT), statistical service (Tenet, XRM and Heidelberg) and the available bit rate service (ATM Forum). The extend to which these services are sufficient to cover present and future applications is too early to say. What is encouraging, however, is that multimedia services can be provided using soft bounds provided by a best effort delivery system. This is best illustrated by the MBONE [91] suite of multimedia tools (e.g., VIC and VAT [25]) which are adaptive in nature (i.e., network conscious applications [92]) and which have proved highly successful.

6.3 Soft v Hard State

Most QoS architectures consider both static QoS management (in terms of QoS mapping, admission control and resource reservation) and dynamic QoS management (in terms of monitoring, scaling and maintenance). With the exception of the IETF work (which uses RSVP maintained state) all architectures consider connection oriented or 'hard state' solutions to network level QoS provision; that is, they couple path establishment and resource reservation. Work in the IETF on and Integrated Services Architecture (using RSVP and IPv6 flows) has shown experimentally that network level QoS guarantees can be obtained using a 'soft state' approach; that is, no explicit connection is established but flows traverse intermediate routers on paths that are temporarily (i.e., network state is timed out and periodically refreshed) established. In this instance path establishment and resource reservation are decoupled. It is argued that a soft state approach provides better scalability, robustness, and eradicates the round-trip call setup time found in connection oriented approaches [23] [61]. In [94] Turner suggest a hybrid approach called *ATM-soft* which benefits form the use of soft state in a native ATM environment. It is still too early to determine which approach is more suitable for future QoS architectures given the need to support both high-end (e.g., telesurgery and time critical applications) and low-end (e.g., video conferencing and audio tools) multimedia applications.

6.4 End-System and Network Commonalities

Commonalities exists between QoS control and managements mechanisms found in the end-system and network: e.g, admission control, resource management, scheduling mechanisms. The extend to which network level QoS mechanisms are applicable in the end-systems (and vice versa) remains an open issue. The COMET group argue end-system

and network devices can be modelled in a similar way, and that the only real difference is the overall goal that end-system or network devices are set to achieve. XRM models the end-system as a virtual switch [67], and a set of configurable multimedia devices based on a DAN architecture. The XRM approach endorses the notion of commonality between the network and end-system components. Furthermore, it is evident that commonality exists between scheduling strategies found in switches/routers and end-system operating systems (e.g., fair share techniques can be found in the end-system and network switches/routers). This seem encouraging in the first instance, however, a counter argument is that end-systems have fundamentally different scheduling goals than routers and switches. End-systems schedule a wide variety of both isochronous (e.g., continuous media flows) and asynchronous (e.g., RPCs) work whereas switches and routers are mainly involved with switching/routing of cells or packets respectively. This means that in the end-system application execution time (i.e., quantum [16] of work in Figure 2-1) can vary widely (e.g., uncompressing a video flow is more computationally intensive than displaying to a screen). In contrast, switch and router schedulers are generally moving packets/cells from queues to ports or vice versa - and are optimised for that task. Therefore techniques resident in switches (such as HRR [22]) may be inappropriate in host operating systems. The impact of any duality is unclear.

6.5 QoS Mapping and Heterogeneous QoS Demands

Work on QoS mapping is still in its infancy. What work there is primarily focuses on deriving appropriate QoS parameters [80] for memory, CPU processing (e.g., threads requirements) and network connections in a rather static, architecturally specific manner. It is still not clear to what extent QoS mapping must cater to higher layer (e.g., distributed systems platform) requirements where services other than flows are apparent. Currently there is no single comprehensive study of QoS mapping in the literature.

Most QoS architectures reviewed in this paper were either sender or receiver oriented; the exception to this being the OMEGA architecture which supports both options. The Tenet, Heidelberg and QoS-A architectures support heterogeneous QoS demands from individual receivers in multicast groups. Supporting such flexibility is important considering heterogeneity exist in applications, communications systems and media format. Resolving heterogenous QoS demands requires the use of advance techniques such as QoS filtering (in the network) and QoS scaling (at the network edges). The success of such an approach is still unclear. Most experimental work on QoS filtering and scaling has been carried out in local area only. One drawback of such an approach is the additional state required at switches/routers to establish and maintain filters. Whether such filtering techniques transfer to the wide area and gain broad support remains unresolved.

6.6 Architectural Comparison

It is evident that there are many architectural similarities in the work cited in this paper. For example, there are several functional similarities between the XRM and QoS-A architectures. In broad terms the QoS-A signalling, control and management planes can be easily mapped to the XRM framework. First, the QoS-A control and user planes are equivalent to the C-plane and U-plane of the XRM respectively. In implementation the QoS-A user plane is populated with a multimedia enhanced transport system, IP++ and AAL5++ stack. In contrast, the XRM user plane only consist of AAL5 in the first instance. This leads us to describe QoS-A as being considered “user-plane centric” in relation to the XRM. This view is further reinforced when we attempt map the XRM’s M-Plane to the QoS-A. The M-Plane includes cell scheduling, call admission control and flow control. Taking these three functions in turn: scheduling and flow control are a function of the QoS-A user plane, and admission control is a function of the QoS-A flow management plane. The XRM’s N-Plane, which represents network and system management, does not easily map to QoS-A. The reason for this is because network management is not considered within the scope of QoS-A research. Instead QoS-A management is primarily associated with the monitoring and maintaining on-going flows. The QoS mechanisms that realised these functions reside in the QoS maintenance plane of the QoS-A. Another architectural difference is that XRM *explicitly* models end-system and network “state” as a telebase (D-Plane). The telebase collectively represents the information, data, abstractions in the systems. There is no such functional equivalent in the QoS-A. In summary, QoS-A is considered to be end-system centric and XRM network centric. Both architectures include QoS mapping, admission control and resource reservations. Both architectures are connection-oriented.

7. Conclusion

In this paper we have argued that multimedia systems designers should adopt an end-to-end approach to meet application level QoS requirements. To meet this challenge we have proposed a generalised QoS framework that is motivated by five design principles; that is, the principles of integration, separation, transparency, asynchronous resource management and performance. Elements of our generalised framework include QoS specification, and static and dynamic QoS management. We summarised and evaluated key research in QoS support for distributed multimedia applications. We started by describing layer-specific QoS work and then reviewed more broader architectural work that we described as QoS architectures. We briefly compared these architectures and then discussed some open issues that emerged during the comparison. While the area of QoS research in multimedia networking is mature, work on QoS architectures remains in its early stages of development with no substantial performance results having been published to verify the validity of the approach. Given that, the work presented in this paper contributes towards a qualitative understanding of the key principles, services and mechanisms needed to build quality of service into networked multimedia systems.

8. References

[1] Hutchison, D., Coulson G., Campbell, A., and G. Blair , “Quality of Service Management in Distributed Systems”, M. Sloman ed., Network and Distributed Systems Management, Addison Wesley, chapter 11, 1994.

[2] Campbell, A., Coulson, G., García, F., Hutchison, D., and H. Leopold, “Integrated Quality of Service for Multimedia Communications”, Proc. IEEE INFOCOM’93, pp. 732-739, San Francisco, USA, April 1993.

[3] Lazar, A.A., “A Real-time Control, Management, and Information Transport Architecture for Broadband Networks”, Proc. International Zurich Seminar on Digital Communications, pp. 281-295, 1992.

[4] Bansal, V., Siracusa, R.J, Hearn, J. P., Ramamurthy and D. Raychaudhuri, “Adaptive QoS-based API for Networking”, Fifth International Workshop on Network and Operating System Support for Digital Audio and Video, Durham, New Hampshire, April, 1995.

[5] Saltzer, J., Reed, D., and D. Clark, "End-to-end Arguments in Systems Design", ACM Transactions on Computer Systems, Vol. 2., No. 4., 1984.

[6] Tennenhouse, D.L., "Layered Multiplexing Considered Harmful", Protocols for High-Speed Networks, Elsevier Science Publishers (North-Holland), 1990.

[7] Clark, D., and D.L. Tennenhouse, "Architectural Consideration for a New Generation of Protocols", Proc. ACM SIGCOMM ‘90, Philadelphia, 1984.

[8] Chesson, G., “XTP/PE Overview”, Proc. 13th Conference on Local Computer Networks, Pladisson Plaza Hotel, Minneapolis, Minnesota, 1988.

[9] Zitterbart, M., Stiller, B., and A Tantawy,“A Model for Flexible High-Performance Communication Subsystems”, IEEE JSAC, May 1992.

[10] Little, T.D.C, and A. Ghafoor, “Synchronisation Properties and Storage Models for Multimedia Objects”, IEEE Journal on Selected Areas on Communications, Vol. 8, No. 3, pp. 229-238, April 1990.

[11] Partridge, C., “A Proposed Flow Specification”, Internet Request for Comments, no. 1363, Network Information Center, SRI International, Menlo Park, CA, September 1990.

[12] Ferrari, D. and Verma D. C., “A Scheme for Real-Time Channel Establishment in Wide-Area Networks”, IEEE JSAC, 8(3), 368–77, 1990.

[13] Clark, D.D., Shenker S., and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism", Proc. ACM SIGCOMM’92, pp. 14-26, Baltimore, USA, August, 1992.

[14] Campbell, A., Coulson G. and D. Hutchison, “Supporting Adaptive Flows in a Quality of Service Architecture”, Multimedia Systems Journal, November, 1995.

[15] Kelly, F. P., "On Tariffs, Policing and Admission Control for Multiservice Networks", Proc. Multiservice Networks '93, Cosener's House, Abingdon, July 1993, and Internal Report, Statistical Laboratory, University of Cambridge, England, 1993.

[16] Coulson, G., Campbell, A and P. Robin, "Design of a QoS Controlled ATM Based Communication System in Chorus", IEEE Journal of Selected Areas in Communications (JSAC), Special Issue on ATM LANs: Implementation and Experiences with Emerging Technology, May 1995.

[17] Parekh, A. and R. G. Gallager, "A Generalised Processor Sharing Approach to Flow Control in Integrated Service Networks - The Multiple Node Case", Proc. IEEE INFOCOM'93, pp.521-530, San Francisco, USA, April 1993.

[18] Keshav, S., "On the Efficient Implementation of Fair Queueing", Internetworking: Research and Experiences, Vol. 2, pp 157-173, 1991.

[19] C. Liu, J. Layland, "Scheduling Algorithms for Multiprogramming in Hard Real Time Environment", JACM, 1973.

[20] Stankovic et al., "Implications of Classical Scheduling Results for Real-Time Systems", IEEE Computer, Special Issue on Scheduling and Real-Time Systems, June 1995.

[21] Tokuda H. and T. Kitayama, "Dynamic QOS Control Based on Real-Time Threads", Proc. Fourth International Workshop on Network and Operating System Support for Digital Audio and Video, Lancaster University, Lancaster LA1 4YR, UK, 1993.

[22] H. Zhang, S. Keshav, "Comparison of Rate-Based Service Disciplines", ACM SIGCOMM, 1991.

[23] ATM Forum, ATM User-Network Interface Specifications, Version 3.0, Prentice-Hall, 1993.

[24] Shenker, S., Clark, D., and L. Zhang, (1993) "A Scheduling Service Model and a Scheduling Architecture for an Integrated Service Packet Network", Working Draft available via anonymous ftp from parcftp.xerox.com: /transient/service-model.ps.Z.

[25] Jacobson, V., (1994) "VAT: Visual Audio Tool", vat manual pages, Feb 1993.

[26] Kanakia, H., Mishra, P., and A. Reibman, (1993) "An Adaptive Congestion Control Scheme for Real Time Packet Video Transport", Proc. ACM SIGCOMM '93, San Francisco, USA, October 1993.

[27] Escobar, J., Deutsch, D. and C. Partridge, "Flow Synchronisation Protoco", IEEE GLOBECOM'92, Orlando, Fl., December 1992.

[28] Pacifici, G., and R. Stadler, "An Architecture for Performance Management of Multimedia Networks", Proc. IFIP/IEEE International Symposium on Integrated Network Management, Santa Barbara, May 1995.

[29] Vogel, A.,Bochmann, G. v., Dssouli, R., Gecsei, J. and B. Kerherv, "Distributed Multimedia Applications and Quality of Service - A Survey", IEEE Multimedia, 1994.

[30] Miloucheva, I. "Quality of Service Research for Distributed Multimedia Applications", ACM Pacific Workshop on Distributed Multimedia Systems, 1995.

[31] Mullender S., ed. (1993). Distributed Systems, 2nd end., Addison-Wesley.

[32] APM Ltd , "ANSAware 3.0 Implementation Manual", APM Ltd, Poseidon House, Castle Park, Cambridge CB3 0RD, UK, 1991

[33] Open Software Foundation, "Distributed Computing Environment", 11 Cambridge Center, Cambridge, MA 02142, USA, 1992.

[34] ODP, "Draft recommendations X.903: basic reference model of open distributed processing", ISO/IEC JTC1/SC21/WG7, International Standards Organisation, 1992.

[35] Coulson G., Blair G. S., Davies N. and Williams N." Extensions to ANSA for Multimedia Computing", Computer Networks and ISDN Systems, 25(11), 305–23, 1992.

[36] Nicolaou, C., "An Architecture for Real-Time Multimedia Communication Systems", IEEE Journal on Selected Areas in Communications, Vol. 8, No. 3, April 1990.

[37] Hazard, L., Horn, F., and J. B. Stefani, "Towards the Integration of Real-Time and QoS Handling in ANSA", CNET Report CNET.RC.ARCADE.01, June 1993.

[38] Zinky, J., Bakken, D., R. Schantz, "Overview of Quality of Service for Distributed Objects" , Technical Report, BBN Systems

and Technologies, Cambridge, 1995.

[39] Hayter, M. and D. McAurely, “The Desk Area Network”, ACM Operating Systems Review, Oct 1991.

[40] Feldmeier, D.,“A Framework of Architectural Concepts for High Speed Communication Systems”, Computer Communication Research Group, Bellcore, Morristown, May 1993.

[41] Hanko, J. G., Keurner E. M., Northcutt J. D. and Wall G. A., “Workstation support for time critical applications”, Proc. Second International Workshop on Network and Operating System Support for Digital Audio and Video, Heidelberg, Springer Verlag, 1991.

[42] Bulterman D. C. and van Liere R., “Multimedia synchronisation and UNIX”, Proc. Second International Workshop on Network and Operating System Support for Digital Audio and Video, Heidelberg, Springer Verlag, 1991.

[43] Leslie, I.M., McAuely, D., and S.J. Mullender, “Pegasus - Operating Systems Support for Distributed Multimedia Systems,” Operating Systems Review, Vol. 27, No. 1, 1993.

[44] Clark, D.D., Lambert, M.L., and L. Zhang, “NETBLT: A High Throughput Transport Protocol”, Computer Communications Review, Vol. 17, No. 5, 1987.

[45] Danthine, A., Baguette Y., Leduc G., and L. Leonard, “The OSI 95 Connection-Mode Transport Service - Enhanced QoS”, Proc. 4th IFIP Conference on High Performance Networking, University of Liege, Liege, Belgium, December 1992.

[46] Wolfinger, B. and M. Moran, “A Continuous Media Data Transport Service and Protocol for Real-time Communication in High Speed Networks”, Second International Workshop on Network and Operating System Support for Digital Audio and Video, IBM ENC, Heidelberg, Germany, 1991.

[47] Hehmann, D.B., Herrtwich R.G., Schulz W., Schuett, T., and R. Steinmetz, “Implementing HeiTS: Architecture and Implementation Strategy of the Heidelberg High Speed Transport System”, Second International Workshop on Network and Operating System Support for Digital Audio and Video, IBM ENC, Heidelberg, Germany, 1991.

[48] Schulzrinne, H. and S. Casner, “RTP: A Transport Protocol for Real-Time Applications”, Work in Progress, Internet Draft, <draft-ietf-avt-rtp-05.ps>, 1995.

[49] Campbell A., Coulson G., Garcia F. and Hutchison D., “A Continuous Media Transport and Orchestration Service”, Proc. ACM SIGCOMM ‘92, Baltimore, Maryland, USA, 99–110, 1992.

[50] W. Doeringer, D. Dykeman, M. Kaiserswerth, B. Meister, H. Rudin, R. Williamson, “A Survey of Light-weight Transport Protocols for High-speed Networks”, IEEE Transactions on Communications, November 1990.

[51] Keshav, S., “Report on the Workshop on Quality of Service Issues in High Speed Networks”, ACM Computer Communications Review, Vol 22, No 1, pp 6-15, January, 1993.

[52] Kurose, J.F., “Open Issues and Challenges in Providing Quality of Service Guarantees in High Speed Networks”, ACM Computer Communications Review, Vol 23, No 1, pp 6-15, January 1993

[53] Golestani, S.J., “A Stop and Go Queueing Framework for Congestion Management”, Proc. ACM SIGCOMM’90, San Francisco, June 1990.

[54] Guerun, R., Ahmadi, H., and M. Naghshineh,”Equivalent Capacity and its Application to Bandwidth Allocation in High Speed Networks” , IEEE Journal on Selected Areas in Communications, Vol. 9, No. 7, Sept. 1991.

[55] Cruz, R., “A Calculus for Network Delay: Part I: Network Elements in Isolation”, IEEE Transactions on Info. Theory, Vol. 37. No. 1, Jan. 1991.

[56] Hyman, J., Lazar, A., and G. Pacifici, “Real-Time Scheduling with Quality of Service Constraints”, IEEE Journal on Selected Areas in Communications, Vol. 9. No. 7, April 1990.

[57] Braden R., Clark, D., and S. Shenke, “Integrated Services in the Internet Architecture: an Overview”, Request for Comments, RFC-1633, 1994.

[58] Floyd, S., “Link-Sharing and Resource Management Models for Packet Networks”, Draft available via anonymous ftp from ftp.ee.lbl.gov: link.ps.Z, September 1993.

[59] Topolcic, C., “Experimental Internet Stream Protocol, Version 2 (ST-II)”, Internet Request for Comments No. 1190 RFC-1190, October 1990.

[60] Zhang, L., et. al., “RSVP Functional Specification”, Working Draft, draft-ietf-rsvp-spec-07.ps, 1995.

[61] Benerjea, A. and B. Mah, “The Real-Time Channel Administration Protocol”, Second International Workshop on Network and Operating System Support for Digital Audio and Video”, Heidelberg, November 1991.

[62] Volg, C., Wolf, L., Herrtwich, R. and H. Wittig, “HeiRAT - Quality of Service Management for Distributed Multimedia Systems”, Multimedia Systems Journal, November 1995.

[63] Gopalakrishna, G., and G. Parulkar, “Efficient Quality of Service in Multimedia Computer Operating Systems”, Department of computer science, Washington University, Report WUCS-TM-94-04, August 1994.

[64] Lazar, A. A., “Challenges in Multimedia Networking”, Proc. International Hi-Tech Forum, Osaka, Japan, Februray 1994.

[65] Hyman, J., Lazar, A., and G. Pacifici, “Joint Scheduling and Admission Control for ATS-based Switching Node”, Proc. ACM SIGCOMM ‘92, Baltimore, Maryland, USA, August 1992.

[66] Lazar, A. A., Ngoh, L.H. and A. Sahai, “Multimedia Networking Abstraction with Quality of Services Guarantees”, Proc. SPIE Conference on Multimedia Computing and Networking, San Jose, February 1995.

[67] Lazar, A. A., Bhonsle S., Lim, K.S., “A Binding Architecture for Multimedia Networks”, Proceedings of COST-237 Conference on Multimedia Transport and Teleservices, Vienna, Austria, 1994.

[68] Campbell, A., Coulson, G. and D. Hutchison, “A Quality of Service Architecture”, ACM Computer Communications Review, April 1994.

[69] OMG, (1993), “The Common Object Request Broker: Architecture & Specification”, Rev 1.3., December 1993.

[70] TINA-C, “The QoS Framework”, Internal Technical Report,1995.

[71] Guangxing, “An Model of Real-Time QoS for ANSA” , Technical Report APM.1151.00.04, APM Ltd, Cambrigde, UK, March 1994.

[72] Besse, L., Dairaine L., Fedaoui, L., Tawbi, W., and K. Thai, “Towards an Architecture for Distributed Multimedia Application Support”, Proc. International Conference on Multimedia Computing and Systems, Boston, May 1994.

[73] ISO-QoS, “Quality of Service Basic Framework - Qutline”, ISO/IEC JTC1/SC21/WG1 N1145, International Standards Organisation, UK, 1994.

[74] Sluman, C., “Quality of Service in Distributed Systems”, BSI/IST21/-/1/5:33, British Standards Institution, UK, October 1991.

[75] Ferrari, D., Ramaekers J. , and G. Ventre, “Client-Network Interactions in Quality of Service Communication Environments”, Proc. 4th IFIP Conference on High Performance Networking, University of Liege, Liege, Belgium, December 1992.

[76] Ferrari, D., “The Tenet Experience and he Design of Protocols for Integrated Services Internetworks”, Multimedia Systems Journal, November 1995.

[77] Delgrossi, L., Halstrinck, C., Hehmann, D.B., Herrtwich R.G., Krone, J., Sandvoss, C., and C. Vogt, “Media Scaling for Audiovisual Communication with the Heidelberg Transport System”, Proc. ACM Multimedia ‘93, Anaheim, August 1993.

[78] Anderson, D.P., Herrtwich R.G., and C. Schaefer. “SRP: A Resource Reservation Protocol for Guaranteed Performance Communication in the Internet”, Internal Report , University of California at Berkeley, 1991.

[79] Slides from IETF meeting 31, Integrated Service Working Group, ftp://mercury.lcs.mit.edu/pub/intserv, 1995.

[80] Nahrstedt K. and J. Smith, “The QoS Broker”, IEEE Multimedia, Spring 1995.

[81] Vogel, A., G. v. Bochmann, R. Dssouli, J. Gecsei, A. Hafid and B. Kerherve, “On QoS Negotiation in Distributed Multimedia Application”, Proc. Protocol for High Speed Networks, April 1994.

[82] Pasquale G., Polyzos E., Anderson E. and Kompella V. The multimedia multicast channel. Proc. Third International Workshop on Network and Operating System Support for Digital Audio and Video, San Diego, USA, 1992.

[83] Yeadon, N., Garcia, F., Campbell, A and D. Hutchison, "QoS Adaptation and Flow Filtering in ATM Networks", Second International Workshop on Advanced Teleservices and High Speed Communication Architectures, Heidelberg, 1994.

[84] Nahrstedt K. and J. Smith, "Design, Implementation and Experiences of the OMEGA End-Point Architecture", Technical Report (MS-CIS-95-22), University of Pennsylvania, May 1995, (submitted to JSAC).

[85] Gopalakrishna, G., and G. Parulkar, "A Real-time Upcall Facility for Protocol Processing with QoS Guarantees", (Poster) 15th ACM Symposium on Operating Systems Principles, December. 1995.

[86] Govindan, R., and D.P. Anderson, "Scheduling and IPC Mechanisms for Continuous Media", Thirteenth ACM Symposium on Operating Systems Principles, Asilomar Conference Center, Pacific Grove, California, USA, SIGOPS, Vol 25, pp 68-80, 1991.

[87] Jain, R., "Congestion Control and Traffic Management in ATM Networks: Recebt Advances and a Survey", Computer Networks and ISDN Systems (to appear)

[88] Jeffay, K., "The Real-Time Producer/Consumer Paradigm: A Paradigm for Construction of Efficient, Predictable Real-Time Systems," Proc. 1993 ACM/SIGAPP Symposium on Applied Computing, Indianapolis, IN, February 1993.

[89] Jeffay, K and D. Bennet, "A Rate-Based Execution Abstraction For Multimedia Computing," Proc. Fifth International Workshop on Network and Operating Systems Support for Digital Audio and Video,", Durhan, NH, April 1995.

[90] Tennenhouse, D. L., Adam J.F., Carver, D., Houh, H.H., Ismert M., Linblad, C.J., Stasior, W., Wetherall, D., Bacher D., and T. Chang, "A Software OrinetedApproach to the Design of Media Processing Environmnents," Proc. IEEE International Conference on Multimedia Computing and Systems, Boston, 1994.

[91] Macedonia, M. R., and D. P. Brutzman, "MBONE Provides Audio and Video across the Internet", IEEE Computer, Qpril 1994.

[92] Dabbous, W. and C. Diot, "High Performance Protocol Architectures", Technical Report, INRIA, Sophia Antipolis, France, 1995

[93] Keshav and Saran, "Semantics and Implementation of a Native-Mode ATM Protocol Stack", Bell Labs Technical Memorandum, http://www.cs.att.com/csrc/keshav/papers.html, 1994.

[94] Turner, J. "ATM-Soft: A Mini-Proposal for ATM Network Control Using Soft State", Technical Note, Washington University, 1995.

[95] Aurrecoechea, C., Campbell, A. and L. Hauw, "A Survey of Quality of Service Architectures", Technical Report, Lancaster University, ftp://ftp.comp.lancs.ac.uk/public/mpg/MPG-95-18.ps.Z, 1995.