CHAPTER 11

Quality of Service Management in Distributed Systems

David Hutchison, Geoff Coulson, Andrew Campbell and
Gordon S. Blair

Computing Department
Lancaster University
Lancaster LA1 4YR, UK
Email: mpg@comp.lancs.ac.uk

The integration of distributed multimedia systems support into a communications architecture including the new multiservice networks is important for realising the next generation of open systems standards: it is also a significant technical challenge. A key observation is that Quality of Service (QoS) provides a unifying theme on which the functions and facilities of the new integrated standards can be constructed. For future applications, especially highly interactive applications and those relying on the transfer of multimedia information, it is essential that QoS is guaranteed system-wide, including the distributed system platform, the transport protocol and the multiservice network. Enhanced communications protocol support such as end-to-end QoS negotiation, renegotiation, indication of QoS degradations and co-ordination over multiple related connections are required. This chapter examines the state of the art in QoS provision in current distributed systems architectures, reviews the layer-specific work that has been done, and highlights the need for the definition of a coherent framework that incorporates QoS interfaces, mechanisms and management across all the communications layers. The chapter introduces the new International Standardisation activity on Quality of Service Framework, and describes complementary research at Lancaster University (the QoS-A project) on an architectural approach to integrated QoS support for multimedia communications.

Lancaster University Report Number MPG-94-02; To be published as a chapter in a book on Distributed Systems Management edited by Morris Sloman, Imperial College London.

11.1. Introduction

Recent years have seen dramatic advances in computer communications technology. At the network level, new high-speed technologies such as FDDI, DQDB and, most recently, ATM-based networks are becoming available. These networks not only have the capability to transmit information at high speed, but also offer a range of properties including bounded delay, guarantees on throughput and isochronous communications.

Multimedia workstation technology for generating, compressing and displaying streams of digital video and audio is available in the marketplace in a range of price/performance categories, along with very high capacity storage systems such as CD-ROM.

These technological developments are complemented by new user perspectives. New classes of distributed application are being developed, such as distance learning, desktop video-conferencing and remote multimedia database access. These applications are characterised by their highly interactive nature and their significant use of multimedia information transfer. In these applications, communication requirements are extremely diverse, and demand varying levels of service in terms of parameters such as latency, bandwidth and jitter. Furthermore, for digital video and audio communications it is often a requirement that levels of service are *guaranteed*.

Other time critical distributed applications such as distributed real-time control systems are also growing in prominence: these may or may not involve multimedia information transfer, but also have stringent requirements for both reliability and guaranteed bounds on message latency.

These various applications share the need for *Quality of Service (QoS) management* in order to ensure that the requirements of the users are met. Until recently, the notion of QoS has been dealt with in a haphazard and inconsistent way in international standards, and barely at all in commercial systems or de facto standards.

This chapter examines QoS concepts and QoS management, considers the current state of the art and examines the impact of the new technological and application environment on QoS management in distributed systems. The approach of the chapter is, firstly, to set out a framework and terminology for thinking about QoS; secondly, to review and evaluate current research on QoS and, third, to highlight the need for a more integrated approach to QoS provision in system architectures. QoS in standards is covered briefly - this is not because it is unimportant (on the contrary, we argue that it will be highly significant), but rather because little work has yet been done. In fact, the research described in this chapter is intended to be fed into the work in progress on QoS standardisation.

11.2 Fundamental Concepts

11.2.1 Activities, Dimensions and Categories

As a first step in our analysis of QoS provision in distributed systems, we introduce the term *activity* to refer to those aspects of a system to which it is useful to ascribe quality of service characteristics. Examples of activities are processes, communications, or complete computer systems. One particularly important type of activity in the context of multimedia and real-time distributed applications is the notion of a *flow* which is defined as the production, transmission and eventual consumption of a single media stream as an integrated activity governed by a single statement of QoS [Partridge,90], [Topolcic,90], [Campbell,92]. Flows are always simplex but can be either unicast or multicast. They may carry a range of data types including both continuous media and control data such as RPC packets.

In characterising the QoS of activities, it is necessary to identify *dimensions* along which QoS can be measured and quantified. To take a familiar example, it is common to measure the QoS of a timesharing computer system along the dimensions of system throughput and user response time. It is also useful to group sets of QoS dimensions into QoS *categories* where each category contains dimensions pertaining to some logically identifiable aspect of QoS. As an example, we may define a ‘system reliability’ category which contains system related reliability dimensions such as mean time between failure (MTBF) or mean time to repair (MTTR). Another reliability category relating more particularly to the field of multimedia and real time distributed systems may contain dimensions relating to, for example, the permitted percentage of loss of media frames in a flow or the permitted bit error rate in ATM cells.

Other important QoS categories of relevance to the distributed multimedia application area are *timeliness* and *volume*. The timeliness category contains dimensions relating to the end-to-end delay of control and media packets in a flow. Examples of such dimensions of are *latency*, measured in milliseconds and defined as the time taken from the generation of a media frame to its eventual display, and *jitter*, also measured in milliseconds and defined as the variation in overall nominal latency suffered by individual packets on the same flow. The volume category contains dimensions that refer to the throughput of data in a flow. At the level of end-to-end flows, an appropriate QoS dimension may be video frames delivered per second. Alternatively, at the ATM layer, a typical volume QoS dimension would quantify throughput in terms of *peak-rate* throughput and *statistical* throughput measured in cells per second. These examples illustrate that certain dimensions are often only applicable at certain system layers and imply that a complete category should contain dimensions for each system layer involved in the support of that category.

The above list of categories and dimensions is far from exhaustive. Other categories worth of mention are *criticality* which relates to the assignment of relative priority levels between activities, *quality of perception* which is concerned with dimensions such as screen resolution or sound quality, and *logical time* which is concerned with the degree to which all nodes in a distributed system see the same events in an identical order. *Cost* is another important category. This may contain dimensions such as the rental cost of an network link per month, the cost of transmitting a single media frame in a flow, or the cost of a multiparty, multimedia conference call. Cost considerations are also typically applied to the level of QoS provided in the various other QoS categories. A more complete selection of QoS categories can be found in [ISO,92].

11.2.2 QoS Management

Having established a framework for the measurement and quantification of QoS we now consider the extension of our conceptual framework to cover the *management* of QoS along the various dimensions. QoS dimensions are abstract characterisations of QoS requirements whereas QoS management is the concrete realisation of required levels of QoS in a real system. It is useful to consider the management of QoS under the headings of specification, mapping between layers, negotiation, resource allocation, admission control, performance maintenance, performance monitoring, policing and re-negotiation.

QoS Specification and Mapping

QoS specification is concerned with defining required levels of QoS which are interpretable by a system. QoS specifications are different at each system layer, but each layer is similar in consisting largely of a list of relevant QoS dimensions together with required values for those dimensions. Required values may be expressed in terms of advisory values, mandatory values, upper and lower limits or a variety of other forms. In addition to values of dimensions, a QoS specification may also contain information on actions to take if the requested QoS levels are violated [Campbell,93]. Often, particularly in the case of throughput QoS categories, a QoS specification is interpreted as a two-way *service contract* whereby the QoS provider undertakes to support a given level of QoS if and only if the traffic generator undertakes to supply its data at the agreed QoS.

QoS mapping performs the function of automatic translation between representations of QoS at different system levels and thus relieves the user of the necessity of thinking in terms of low level dimensions. For example, a user at the flow level may express a jitter dimension in terms of a statistical variance in arrival times of video frames and this could be translated at the lower layers into a requirement for an absolute bound on ATM cell jitter and a jitter smoothing buffer of a certain size. Note that QoS mapping is concerned simply with translating between representations: the resource allocation function (discussed next) is responsible for the actual instantiation of the ATM connection and buffers.

QoS Negotiation, Resource Allocation and Admission Control

The QoS negotiation function is responsible for analysing an activity onto components and finding a composite of the individual QoS levels supportable by those components which is necessary and sufficient to realise the QoS of the complete activity. In a continuous media flow, the system entities concerned would typically be media devices such as video codecs and frame buffers, operating system threads, transport protocol entities and network links. At the network layer *QoS based routing*, whereby a network route is chosen in accordance with the QoS supportable at each node, is an important consideration. When an activity is initiated, the negotiation function asks each component to state the level of QoS it is able to provide. Then, depending on the results, it assigns particular QoS levels to each component (or reports back to the initiator that the activity cannot be supported).

In order to support a given level of QoS it is usually necessary for the system components comprising an activity to dedicate certain resources to the activity. For example, a operating system thread acting as a component of a flow may require a certain number of CPU cycles per second to process video frames at a rate compatible with the flow’s throughput requirement. In addition, buffer space and network bandwidth will have to be allocated. In practice, resource allocation is often closely associated with QoS negotiation because it is often the case that low level system components can only determine whether or not they are capable of supporting a given level of QoS by actually requesting the necessary resources and noting the outcome of the request.

The admission control function is also intimately tied into the negotiation function. Admission control is responsible for comparing the resource requirement arising from the QoS levels associated with a new activity with the available resources in the system. The decision as to whether a new activity can be created depends on system policies as well as simple resource availability. For example, if the new activity has a high priority it may preempt resources currently dedicated to another activity.

QoS Maintenance, Monitoring and Policing

To maintain an agreed level of QoS it is often not sufficient to *statically* dedicate resources to an activity at QoS negotiation time as described above. Instead, *dynamic* QoS maintenance is frequently required to ensure that the required performance of individual system components is kept within bounds. The QoS maintenance function is particularly important in cases where resources must be statistically multiplexed among activities. Prime examples are CPU schedulers and ATM switch schedulers which must simultaneously meet the targets of QoS controlled activities while dealing with transient overloads and avoiding starvation of low priority activities.

QoS monitoring is used to allow each level of the system to track the ongoing QoS levels achieved by the lower layers and compare them with the initial requirement. It often takes the form of a feedback loop which monitors the QoS being achieved by the monitored component, compares it against the target and then performs fine-grained resource adjustments if necessary. At the interface of the top level activity the QoS monitoring function issues a *degradation indication* to the user when it determines that the lower layers have failed to maintain the QoS of the activity and nothing further can be done by the maintenance function. In response to such an indication the user can choose either to adapt to the reduced level of QoS, terminate the activity or attempt to *renegotiate* (see next section) the QoS of the activity.

QoS policing is used when a symmetrical service contract is in force. Policing can be viewed as the dual of monitoring: the latter 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. The action taken by the policing function can range from accepting violations and merely notifying the user, to shaping the incoming traffic to an acceptable QoS level to terminating the activity.

QoS Re-negotiation

Renegotiation is distinguished from the component level ‘adjustment’ referred to above in that the latter involves tuning the local QoS of an individual component whereas the former involves a global reconfiguration of all components of an activity through re-invocation of the QoS specification and negotiation functions. Note, however, that it is not necessarily the case that renegotiation can only be initiated by the top level user of an activity; activities themselves may form components of a hierarchically structured top level activity.

Although renegotiation is commonly initiated in response to the receipt of a degradation indication, it is also requested in a number of other situations. For example a user may simple wish to conserve global system resources by downgrading a low priority activity. Alternatively, the user may wish to use the same communication channel successively for different purposes at different times. As an example of this latter possibility, a flow activity may be used for the transmission of full motion video interspersed with intervals of slow motion, or it may be upgraded from monochrome to colour video, or telephone quality to CD quality audio.

11.3 QoS in Standards

Because the effects of the new technological and application environment discussed in section 11.1 are only just making themselves felt, it is not surprising that current network architectures fail to comprehensively address the need for QoS support of distributed multimedia applications over high-speed networks. To give an impression of the degree of QoS support in present day systems, this section reviews the ISO's Reference Model for Open Systems Interconnection (OSI-RM) and the CCITT's I Series recommendations for ATM networking with regard to their support of QoS. The OSI reference model and the I Series recommendations are highlighted as the most prominent of the currently standardised network architectures.

11.3.1 ISO's Reference Model for Open Systems Interconnection

The International Standards Organisation (ISO) has developed a set of standards for computer communications in the form of the seven layer OSI-RM, and these standards are now mature and widely implemented. However, the OSI-RM evolved in an environment of data-only applications running over low speed networks, and the QoS support provided by the OSI-RM reflects the limited QoS requirements of this class of applications. QoS support in the OSI-RM is limited to statically defined parameters intended to be supported at the session and transport layers. To enable applications to access QoS facilities, the OSI upper layers (application and presentation layers) simply map QoS dimensions (parameters) through to the lower layers unchanged. At the transport layer, QoS parameters relate to each of the phases of the session; i.e. connection establishment, data transfer and connection release. The parameters are also classified as either *performance oriented* or *non performance oriented* [Henshall,88]. Non performance oriented parameters do not directly effect the performance of the communications but are concerned with protection, priority and cost QoS categories. The complete set of parameters together with their interpretations is given in Table 11.1 which lists the performance oriented parameters and Table 11.2 which lists the non-performance oriented parameters.

Parameter	*Description*

Throughput	The maximum number of bytes, contained in Service Data Units (SDUs), that may be successfully transferred in unit time by the service provider over the connection, on a sustained basis.
Transit Delay	The time delay between the issuing of a *data.request* and the corresponding *data.indication*. The parameter is usually specified as a pair of values, a statistical average and a maximum. Those data transfers where a receiving service-user exercises flow control are excluded. The computations are all based on a SDUs of a fixed size.
Residual Error Rate	The probability that an SDU is transferred with error, or that it is lost, or that a duplicate copy is transferred.
Establishment Delay	The delay between the issuing *connect.request* and the corresponding *connect.confirm*.
Establishment Failure Probability	The probability that a requested connection is not established within the specified maximum acceptable establishment delay as a consequence of actions that are solely attributable to the service-provider.
Transfer Failure Probability	The probability that the observed performance in respect to transit delay, residual error rate or throughput will be worse than the specified level of performance. The failure probability is, as such, specified for each measure of performance of data transfer, discussed above.
Resilience	The probability that a service-provider will, on its own, release the connection, or reset it, within a specified interval of time.
Release Delay	The maximum delay between the issuing of a *disconnect.request* primitive by the service-user and a corresponding *disconnect.indication* primitive issued by the service provider.
Release Failure Probability	The probability that the service-provider is unable to release the connection within a specified maximum release delay.

Table 11.1 OSI performance-oriented QoS parameters

Parameter	*Description*
Protection	The extent to which a service provider attempts to prevent unauthorised monitoring or manipulation of user data. The level of protection is specified qualitatively by selecting either *(i)* no protection; *(ii)* protection against passive monitoring; *(iii)* protection against modification, addition or deletion, a combination of *(i)* and *(ii)*.
Priority	High priority connections are serviced before lower ones. Lower priority connection packets will be dropped first before high priority packets, should the network become congested.
Cost Determinants	A parameter to define the maximum acceptable cost for a network connection. It may be stated in relative or absolute terms. Final actions on this parameter are left to the specific network providers

Table 11.2 OSI non-performance-oriented QoS parameters

11.3.2 CCITT I Series Recommendations

The CCITT have recognised the need for QoS configurability in the emerging standards for broadband integrated services digital networks (B-ISDN) which are to be based on asynchronous transfer mode (ATM) networking technology. As a result of this recognition they have issued a series of draft recommendations known as the I-series recommendations which defined a fairly comprehensive set of QoS dimensions at the ATM layer. QoS characterisation in ATM networks is applicable at three different levels. The *call control* and *connection levels* are concerned with the establishment and release of calls and the allocation of resources along a path of ATM switch nodes. The *cell control* level is concerned with the data transfer phase itself.

At the call control level, the available parameters are similar to those defined in OSI for analogous purposes: i.e. establishment delay, establishment failure probability, release delay etc. At the connection level the parameters outlined in Table 11.3 are applicable.

Parameter	*Description*
Peak arrival rate of cells	The maximum resources required by the application at peak load.
Peak duration	The average duration of the maximum load.
Average cell arrival rate	The average amount of network resources requested by the source. This is the number of cells measured during the duration of the connection divided by the duration.
Burstiness	The ratio between the peak cell rate and the average cell rate.
Cell loss ratio (CLR)	The ratio of number of lost cells to transmitted cells. This type of error usually occurs because of congestion in switches.
Cell insertion ratio (CIR)	This type of error occurs when the address field in the header is corrupted to another valid network address.
Bit error rate (BER)	Defined as the number of bits which are delivered erroneously divided by the total number transmitted. These sorts of errors are mainly caused by transmission system.

Table 11.3 CCITT QoS parameters

The first four parameters are intended to allow traffic to be characterised in advance so that the network can both allocate resources to support the desired traffic patterns and also police the traffic inserted at the network by the user to ensure that the user does not attempt inject data into the network at a higher QoS than that agreed to. It is also intended to use these parameters to support QoS renegotiation (known as *in-call renegotiation*). The remaining parameters control the degree of reliability expected by the user from the network.

As an example of traffic characterisation using the CCITT parameters, a characterisation of variable bit rate encoded video could be: peak rate = 50Mbps, average cell arrival rate = 25Mpbs, burstiness = 2 and the peak duration = 10ms.

11.3.4 Evaluation

It is clear that QoS support in the OSI standards is severely limited when related to the generality of the discussion in section 11.2. The current OSI service definitions do not provide for the specification of a full range of QoS dimensions such as jitter, criticality and cost. In addition, there is no support for QoS monitoring and no interface for renegotiation, and the precise semantics of responsibilities and guarantees are not clear. Even more limiting is the fact that at the protocol level there is no notion of QoS management in terms of QoS negotiation, mapping, resource allocation, and QoS maintenance. It is simply assumes that the underlying network provider will support the requested QoS levels.

Another important observation is that the OSI upper layers are not QoS aware. QoS parameters are simply mapped unchanged through to the transport layer. If users want to specify QoS they are forced to drop below the level of abstraction provided by the upper layer architecture and interact with layers that are intended to be hidden from applications.

The CCITT's ATM recommendations are more comprehensive and include a fairly detailed traffic characterisation model. The major service related limitation here is the lack of consideration of how traffic characterisation at the ATM layer can be derived from user QoS needs at the transport layer and above. Below the service interface, the current state of ATM standardisation suffers from a comparable lack of QoS management support to that found in the OSI field. There is currently no consensus on how resources will be allocated and how requested QoS levels will be maintained, policed and renegotiated in both networks and end-systems.

The essential characteristics of the current state of QoS provision in the major standards may therefore be summarised as follows:-

- *incompleteness*: current service interfaces typically provide only a small subset of the facilities outlined in section 11.2;
- *lack of mechanisms to support QoS guarantees*: research is needed in basic protocol and monitoring mechanisms so that contracted QoS levels can, in fact, be maintained;
- *lack of overall framework*: it is necessary to evolve an overall architectural framework to build on and reconcile the existing

notions of QoS at different systems levels and among different network architectures;

11.4 Layer Specific QoS

Most of the developments to date in QoS provision have occurred in the context of individual architectural layers. Much less progress has been made in addressing large scale architectural QoS requirements implied in the section above. This section reviews some of the layer specific work and attempts to extract some recurrent issues which have arisen in the research. We leave the question of large scale architectural issues until section 11.5 in which the progress achieved so far in this area is assessed.

The layers considered in this section are the ATM network, transport, operating system and distributed system platform. The scope of the network, transport and operating system layers is self explanatory. By distributed systems platform we have in mind operating system independent application support layers such as the computational model of the ISO's Open Distributed Processing (ODP) standards [ISO,92], the ANSA computational model [APM,92], the OSF's Distributed Computing Environment (DCE) [DCE,92].

11.4.2 Network Layer

Unlike existing telecommunications networks which dedicate a physical path (and thus a guaranteed QoS) to each connection, ATM networks operate by multiplexing fixed sized packets (known as cells) from different *virtual connections* over the same physical link. This multiplexing technique has the potential advantage of 'statistical gain' whereby network capacity can be over-committed because of the unlikelihood that all connections will use their maximum bandwidth allocation at any one time. At the present time, significant research is being undertaken in the ATM arena to design networks which are consistent with QoS provision in the face of such statistical behaviour. In outline, research in this area is concerned with the following topics.

QoS Categorisation

Several different ways of categorising QoS guarantees in packet switched networks have been identified. In [Clark,93] a distinction is made between three different service commitments: *(i)* guaranteed service for hard real-time applications; *(ii)* predicted service, which utilises the measured performance of delays and is targeted towards continuous media applications; and *(iii)* best effort service, where no QoS guarantees are provided. A unified traffic scheduling mechanism is also discussed which is based on a combination of weighted fair queuing and static priority algorithms. In the Lancaster QoS-A project (see section 11.5.2 below), commitment is supported both at the end-systems and in the network. The idea of QoS commitment introduced by Clark et al. is extended; that is, each individual QoS dimension can be configured to meet a specific level of service commitment.

Service Discipline

Means of providing QoS guarantees in high performance networks have been widely covered in the literature. for example, in [Lazar,90], an Asynchronous Time-Sharing network which provides a QoS capability is proposed. The switching provided by the network is novel in that the concept of QoS explicitly appears in the design specification at both the edge (at call level) and the core (at cell level) of the network. The network supports four well-defined traffic classes which support circuit emulation, voice and video, file transfer and network management flows. Following on from this work [Lazar,92] describes a joint scheduling and admission control mechanism used to guarantee QoS of each traffic class.

Reservation and admission algorithms

The area of resource reservation is fundamental in providing end-to-end QoS guarantees. There has been a number of significant contributions to resource allocation in communication networks which have emerged over the past few years. For example, ST-II [Topolcic,90] is a network layer resource reservation protocol designed specifically for packetised audio and video communications across the Internet. In contrast to ST-II which provides source initiated point-to-multipoint flows, RSVP [Zhang,93] provides receiver initiated reservation and multipoint-to-multipoint support. SRP [Anderson,92], also designed for the Internet, supports end-system and networks resource allocation. The Lancaster QoS-A flow reservation protocol is tailored for the local ATM environment and borrows heavily from ST-II and SRP. The QoS-A flow reservation service differs from the above ST-II, SRP and RSVP in that it supports a fast and forward reservation service.

11.4.3 Transport Layer

A large number of research teams have investigated the provision of QoS at the transport layer. As a significant example of such work, the recent Esprit OSI 95 project has proposed an enhanced transport service and protocol collectively described as TPX [Baguette,92]. TPX provides support for connection oriented services with sequenced delivery, configurable and re-negotiable QoS, and error notification. It also provides connectionless, connectionless multicast, and request/reply services. The enhanced connection oriented service takes QoS parameters relating to throughput, delay, delay jitter, error selection policy and relative priority. Three transport level QoS semantics are proposed in addition to ‘best effort’ service for this service: i.e. *compulsory, threshold* and *maximal* QoS. When a *compulsory* semantic is selected, the transport protocol commits to monitor the connection and will abort the service should the QoS drop below the requested value. The *threshold* QoS semantic, which is motivated by the needs of a multimedia service [Leopold,92], commits the service provider to monitor the on-going performance of the connection. In this case, however, a QoS indication informs the user should the QoS degrade below the requested value. The *maximal* QoS value deals with limiting the over utilisation of communications resources on a connection. In [Danthine,93] a number of negotiation rules are laid out for each of the QoS value. The threshold semantic is suitable for multimedia communications where applications may accommodate service fluctuations. The compulsory value, however, is not a suitable semantic for the multimedia communications as many applications prefer degraded service to no service. The OSI 95 transport service provides a set of QoS features which are suitable for a range of transport service user’s needs; however QoS maintenance, commitment and adaptation have not been addressed in any detail

The QoS-A project [Campbell,93] has also defined a QoS enhanced transport service interface. In this design, the QoS requirements of the user and the potential degree of service commitment of the network provider are unified and formalised in a *service contract* agreed by both parties. The service contract subsumes the well accepted performance parameters of jitter, error, delay and throughput, but also allows the specification of a wider range of options. These are characterised in terms of the following clauses.

- the *flow spec* characterises the user's QoS in terms of a set of values for a range of QoS dimensions;
- the *commitment* clause specifies the degree of resource commitment required from the lower layers (see below);
- the *adaptation* clause identifies actions to be taken in the event of violations to the contracted service;
- the *maintenance* clause selects the degree of monitoring and active QoS maintenance required of the QoS-A;
- the *connection type* clause selects from on demand, fast reservation, and forward reservation connection services;
- the *cost* clause specifies the costs the user is willing to incur for the services requested.

While the flow spec permits the user to express the required performance parameters in a quantitative manner, the commitment clause of the service contract allows these requirements to be refined in a qualitative way so as to allow a distinction to be made between hard and soft network performance guarantees. The commitment clause permits the selection of one of three possible commitment types [Ferrari,92]: *i) deterministic*, which is typically used for hard real-time performance applications; *ii) statistical*, which allows for a certain percentage of violations in the requested flow spec and is particularly suitable for continuous media applications; and *iii) best effort*, the lowest priority commitment and synonymous with a datagram service. Once a flow has been established the transport protocol actively monitors and maintains the flow based on the user supplies flow spec and service commitment identified in the service contract.

[Garcia,93] reports on the development of a transport service tailored for the support of continuous media communications. Multimedia synchronisation support is a novel feature of the transport service interface [Campbell,92]. Connections are represented as QoS configurable simplex streams which can be unicast or multicast. The protocol, which uses rate-based flow control, can detect lost or corrupt data. When this happens, the transport service user is informed via an error announcement.

The HeiTS project [Hehmann,91] has concentrated on the integration of transport QoS and resource management (primarily CPU scheduling). HeiTS puts considerable emphasis on an optimised buffer pool which minimises copying and also allows efficient data transfer between local devices. The 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. However, QoS monitoring, maintenance and commitment are not are not addressed in this work.

Other significant work on QoS provision at the transport layer has come from The Tenet Group at the University of California at Berkeley. This group have developed a family of protocols [Wolfinger,91] which run over an experimental wide area ATM network known as Aurora [Clark,91]. The protocol family includes the Real Time Internet Protocol (RTIP), the Continuous Media Transport Protocol (CMTP) and the Real Time Channel Administration Protocol (RCAP). The latter provides generic connection establishment, resource reservation, and signalling functions for the rest of the protocol family. CMTP [Wolfinger,92] 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 bounds. Notification can be provided of all undelivered and/or corrupted data if the client selects this option. The client interface to CMTP includes facilities to specify traffic characteristics in terms of burstiness, which is useful for variable bit rate encoding techniques, and workahead, which allows the protocol to deliver faster than the nominal rate if data is available. CMTP also permits dynamic QoS renegotiation by sending a special *on* TPDU which contains the new QoS parameters at the start of a new ‘stream’, and an *off* TPDU to signal the onset of a silent period.

11.4.4 Operating System Layer

The Need for QoS in Operating Systems

Traditionally, work on QoS has concentrated on the network and communications infrastructure. Recently, however, the topic has become increasingly important in an end system context because of the interest in operating system support for multimedia applications. It is becoming recognised in the multimedia community that classes of application exist which must actively manipulate real-time continuous media data in an operating system environment (as opposed to merely controlling and supervising the flow of media in specialised hardware). In order to support multimedia applications, operating systems must provide a degree of quality of service support to uphold the real-time isochronous nature of continuous media data types such as audio and video.

There has already been considerable research in operating systems support for more traditional real-time applications [Stankovic,88]. However, such real-time operating systems tend to be tailored towards specific areas such as factory automation or robotics. Consequently, they tend to assume a *static* environment where the number of processes and their resource requirements are fixed and known in advance. Unfortunately, this assumption of static resource needs is not valid in the multimedia field where it is important that real-time applications co-exist with existing applications and run on general-purpose workstations. What is required are quality of service driven operating systems which can support existing applications and simultaneously offer predictable performance in a dynamic and unpredictable environment.

One solution to this problem is to provide adequate resources to comfortably meet all anticipated demand for processing capacity, disk bandwidth, etc. In practice, however, this is unlikely to be achievable given the heavy resource needs of continuous media data [Davies,91]. Instead, it is important that *resource management strategies* are developed to more effectively exploit limited resources and provide some guarantees of predictability. This task, although challenging, is eased by the inherently ‘soft’ real-time nature of many multimedia applications - for example, it is often acceptable to downgrade picture quality in a videophone application if insufficient resources are available to provide adequate audio quality. Resource management strategies are required for all areas of operating system management including processor scheduling, communications, device management and memory management. There is also a need for new abstractions and interfaces which capture the notion of quality of service driven resources.

State of the Art

To date, most of the work on QoS in the operating system community has focused on *processor scheduling*. It is now recognised that currently used scheduling policies, primarily priority based scheduling, are too static and coarse grained for the support of multiple isochronous sessions in a dynamic environment. In such sessions, each information unit has an implicit deadline and, hence, it is natural to schedule the processes handling these units on the basis of such deadlines. This has led to the adoption of *earliest deadline first* scheduling [Liu,73] as an attractive policy for multimedia. Recent research has proposed the use of *split level scheduling* for the processing of continuous media [Govindan,91]. In such schemes, the application programmer is presented with the abstraction of multiple user level threads in a single address space (the use of user level threads has the advantage of minimising context switches in a multi-threaded application). Responsibility for scheduling of user level threads is split between a user level scheduler and a kernel level scheduler. The two schedulers communicate through the use of shared memory, to ensure that scheduling decisions are globally valid.

The issue of QoS driven *communication protocols* has already been dealt with in some detail in the previous section. However, the *implementation* of protocols in an operating system environment remains to be discussed. Protocol implementation involves predictability issues such as the need for correct scheduling of protocol activities, and efficiency issues such as the minimisation of data copying, system calls and context switches. Efficiency considerations have increased in importance as the communications bottleneck has shifted from the network to the end-system in the new communications environment. The current situation is that a number of widely agreed principles for quality of service driven protocol implementation are beginning to emerge including the avoidance of multiplexing [Tennenhouse,90], the use of hardware assists for protocol processing and the importance of executing protocol code in a schedulable process rather than as an interrupt service routine.

In the area of *device management*, most of the research has concentrated on the development of storage techniques for continuous media. Examples of research in this area are the Video-on-Demand service designed at the University of California at San Diego [Vin,93], the Continuous Media File System developed at the University of California at Berkeley [Anderson,92], and the Lancaster Continuous Media Storage Server [Lougher,93]. Most of this research has concentrated on disc layout and disc head scheduling. The aim is to optimise the layout of continuous media data on the disc to minimise disc head movements and guarantee transfer rates. The technique of *disc striping* is commonly used whereby successive segments of a continuous media stream are stored on separate discs arranged in an array. This technique enables N discs to provide a throughput approaching N times that of a single disc. So far, however, these techniques have failed to produce fully scaleable solutions for the simultaneous retrieval of large numbers of continuous media streams.

Finally, there has been some work carried out on the overall architecture of QoS-driven operating systems. Most of this work has attempted to maintain a level of compatibility with the de facto standard 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. For example, a range of projects is currently under way at SUN Microsystems in this area. Their proposal is for *time driven resource management* [Hanko,91] which allows applications to signal their likely forthcoming resource requirements in terms of QoS dimensions such as quantity, deadline and priority. The second approach is to preserve the standard UNIX interface, but re-implement it in terms of the *micro-kernel* model. Examples of micro-kernels capable of supporting UNIX interfaces are Chorus [Bricker,91], Mach [Accetta,86] and Amoeba [Tanenbaum,88]. Work has been undertaken at CWI, Amsterdam to support continuous media in an Amoeba based UNIX environment [Bulterman,91]. Work is also being carried out at Lancaster University using Chorus as the basis of a distributed system with end-to-end quality of service support [Coulson,93a]. Finally, work has been carried out in a Mach environment to provide processor scheduling appropriate to continuous media [Govindan,91], [Tokuda,93].

Outstanding Issues

There has been considerable progress in operating system support for QoS with most progress having been made in the specific areas of communications and scheduling. There has been considerably less work on *integration* of the various components into an overall operating system design. Most significantly, work is required to integrate techniques for communications and scheduling. For example, to implement an audio connection with a given quality of service, it is necessary to achieve the desired quality of service in the transport protocol *and* to schedule threads at the desired rate to deal with the arrival of audio data. Such integration should also eventually extend to areas such as device management and memory management.

In addition, work is required on the integration of the abstract quality of service management functions described in section 11.2.2 into the operating system environment. Functions such as negotiation protocols, admission control, and graceful degradation have a wider scope than individual resource managers and are important for the correct operation of the system as a whole. However, as yet almost no work has been carried out in this area.

A number of other outstanding issues remain:

i) It is not clear where protocol processing should be performed in future operating systems: some researchers still advocate traditional kernel implementations; others advocate the use of specialised hardware to implement protocol processing [Arnould,89]; a third group suggest that protocols should be implemented in user space [Forin,90] (this is consistent with trends in micro-kernel design to move functionality out of the kernel and into user space).

ii) It is recognised that earliest deadline first policies do not operate well in overload situations. Some researchers therefore propose extensions to earliest deadline first policies either by adding resource reservation or by aborting certain threads [Tokuda,93]. It is not clear at this stage whether such extensions are required in a generalised QoS support environment.

iii) The real-time synchronisation of related continuous media streams in the operating system environment remains an unsolved problem. Multimedia applications need related streams to be tied together for such purposes as maintaining ‘lip-sync’ between audio and video streams without, at the same time, violating the temporal integrity of the individual streams. This problem has been addressed at a specification level (e.g. [Little,90]), but there has been little work on realising the specified behaviour in the operating system environment.

iv) More work is required on abstractions for QoS driven operating systems. For example, many application platforms are structured in terms of objects, but it is not yet clear what level of support at the operating system level is needed to support objects with real-time behaviour. In addition, multimedia applications typically use the concept of structured objects to represent multimedia documents. It is not clear how much support is required for such abstractions in multimedia file servers. Most experimental file servers in existence today deal only with individual streams.

v) As discussed above, it is necessary in the emerging application environment for applications with varying needs and assumptions to co-exist in an operating system environment able to simultaneously satisfy all their various requirements. For example, few real systems have been built with the ability to effectively run batch mode applications, interactive applications and continuous media applications.

11.4.5 Distributed Systems Platform

The Need for QoS in Distributed Systems

The role of the distributed systems platform is to provide a network and computer independent programming environment for the development of distributed applications. There has been considerable research in this area over the past ten years [Mullender,93]. However, until recently, 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. There has also been some relevant research in the more specialist area of distributed real-time systems [Stankovic,88].

Quality of service in distributed systems platforms is fundamentally an *end-to-end* issue, i.e. from application to application. For example, consider remote access to a storage sequence of video. In the distributed systems platform, QoS specification should apply to the complete flow of information from the remote server, across the network and to the point of application. This requires careful co-ordination of disk scheduling, thread scheduling and the various layers of communications protocols. QoS specification at this level should also be *user-oriented* rather than system oriented. In other words, the QoS dimensions provided should be meaningful to the end-user, and lower level considerations such as the rate and burst size of a transport connection should be hidden. Finally, it is important that QoS specifications are *declarative*, i.e. users should be specifying what is required rather than how this is to be achieved.

In a distributed system, there are three areas where quality of service might apply.

i) Message Passing Services: these allow a programmer to explicitly send a message between two processes in a distributed system. Normally, message passing in distributed systems is asynchronous, i.e. there is a delay between the sending and the receiving of the message. Such message passing services are provided in, for example, distributed operating systems such as Chorus [Bricker,91] and Mach [Accetta,86]. QoS support in this context is concerned with bounding the latency of the message. It is also desirable though to specify whether delivery should be reliable or unreliable. Finally, if facilities are offered to multicast messages to a group of processes, it is important to be able to specify the constraints on the ordering of message arrivals [Birman,91].

ii) Remote Invocation: this allows operations in a server process to be invoked by a client process. The results of this invocation are then returned directly to the client. This style of interaction of often referred to as the remote procedure call paradigm and can be found in platforms such as ANSAware [APM,93] and DCE [DCE,92]. QoS specification on remote invocations is more abstract than with message passing and is defined over the whole interaction between client and server (involving a number of message exchanges). Crucial QoS dimensions in this category include the round-trip latency and the semantics of the remote operation (e.g. at least once, at most once, or exactly once).

iii) Stream Services: these are connections which support the transmission of continuous media data such as audio or video. Such services have only recently been developed and are not yet available in commercial distributed systems. Quality of service in this context is concerned with managing the ongoing flow of data between a continuous media source and sink and is therefore defined in terms of parameters such as required throughput, latency, jitter and error characteristics [Hehmann,90].

State of the Art

A number of experimental QoS driven distributed systems platforms are now beginning to emerge. For example, researchers at Lancaster University have developed an extended version of ANSAware [APM,92] featuring bounded invocations and QoS controlled streams [Coulson,92]. Similar work has also been undertaken at Cambridge University [Nicolaou,90].

More recently, research on quality of service has centred on Open Distributed Processing (ODP) standardisation. ODP is an ISO sponsored activity to develop international standards for distributed computing in potentially heterogeneous environments. The ODP community have now recognised the importance of quality of service and have proposed extensions to the ODP Computational Language to support QoS specification.

The ODP Computational Language defines the basic concepts necessary to develop computer and network independent designs of distributed applications. The language is object-based and features concepts such as computational objects, operational interfaces, stream interfaces and operation invocation [ISO,92]. QoS specification is included in the form of annotations on ODP interfaces. In addition, extensions have recently been proposed to enable the creation of explicit *bindings* between interfaces. Bindings are abstractions over communication between objects and can only be established if both type and QoS specifications match. Bindings are themselves objects; this provides a placeholder and access point for the management of QoS of the binding.

Ongoing research in this area is being carried out at CNET in France [Stefani,93]. The aim of this research is to develop a more complete Computational Language for both real-time and multimedia applications. In the CNET approach, quality of service annotations are written in a real-time logic called QL. The logic statements are then used to generate quality of service monitors written in the real-time control language Esterel. This work is also being extended, in a collaboration with Lancaster University, to include engineering support for the enhanced Computational Language [Coulson,93b].

In summary, there has historically been little work on QoS support in the distributed systems community. Recently, though, there have been some significant developments particularly in the context of ODP standardisation. However, it is clear that further research is needed to provide a complete framework for QoS management incorporating the various functions identified in section 11.2.2. Furthermore, work is required to integrate the distributed system view of quality of service with the other layers discussed in this section.

11.5 Towards an Integrated View of QoS

In recognition of the QoS limitations of current systems a number of research teams have recently proposed a *systems architectural* approach to QoS provision in distributed systems. The intention is to extend current systems by defining a set of service interfaces which generalise and formalise the existing OSI and CCITT QoS services and also provide a framework for the integration of new QoS management mechanisms. It is hoped that a systems approach can help avoid duplication of functions across layers and maximise efficient QoS management by providing a global framework for QoS specification and management extending from the distributed application platform through the transport subsystem and the network.

One significant pioneering contribution to the provision of QoS in the OSI domain is being made by the “Quality of Service Framework” New Work Item project, the aim of which is to enable the future extension of OSI standards in the direction of QoS provision by defining a reference architecture and standard terminology. The intention of this project at this stage is *not* actually to develop new standards for QoS in the OSI-RM.

Another project which is attempting to build a framework for QoS provision in distributed systems is the UK SERC funded Quality of Service Architecture (QoS-A) project at Lancaster University. This project is more pragmatically biased than the JTC 21 initiative. It is the aim of the QoS-A project to design and implement a QoS framework in a specific distributed systems environment consisting of multimedia applications running over an ODP platform which, in turn, is supported by newly developed QoS configurable transport services running over a local, campus wide, ATM network.

This remainder of this section describes the above two projects in more detail.

11.5.1 SC21 QoS Framework

11.5.2 Quality of Service Architecture

The Lancaster Quality of Service Architecture (QoS-A) [Campbell,93] is a layered architecture of services and mechanisms for QoS management in an environment based on the ISO's Open Distributed Processing standards and ATM networks. The architecture is geared mainly to the support of continuous media applications and uses the concept of a flow described in section 11.2.1 above.

In functional terms, the QoS-A illustrated in Figure 11.1 is broadly divided into a number of layers and planes. The upper layer consists of a *distributed applications platform* provided by an ODP compatible distributed systems platform with services to provide multimedia communications and QoS configuration in an object-based environment [Coulson,93]. Supporting this is a *transport layer* which contains a range of QoS configurable protocols. For example, separate protocols are provided for continuous media and constrained latency message protocols.

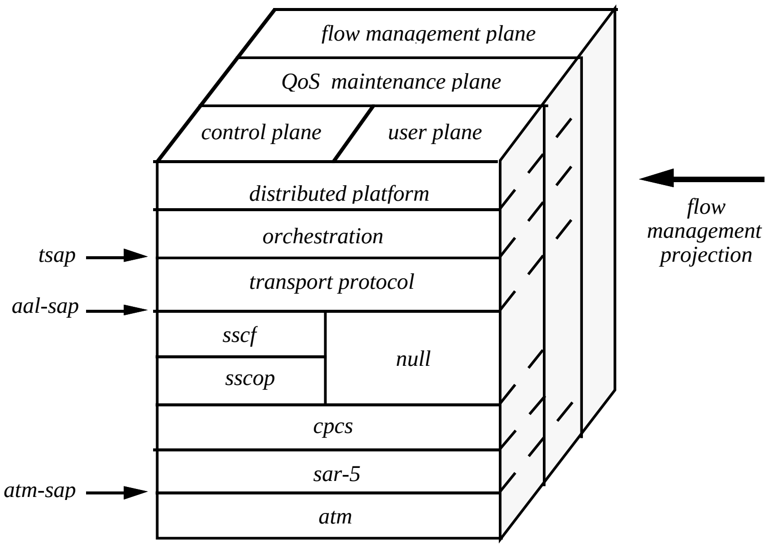

Figure 11.1 QoS-A (remove ATM detail, orchestration and projection)

The vertical planes in the QoS-A, of which there are three, are as follows:-

- The Protocol Plane: this consists of a *user plane* and a *control plane*. Separate protocol profiles are used for the control and data components of flows because of the essentially different QoS requirements of control and data. Control generally requires a low latency full duplex assured service whereas multimedia data generally requires a range of non assured, high throughput simplex services.

- The QoS Maintenance Plane: this contains a number of layer specific QoS managers. These are each responsible for the fine grained monitoring and maintenance of their associated protocol entities. 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 Flow Management Plane: this is responsible for *flow establishment* (including flow admission control, resource reservation and QoS based routing), *QoS renegotiation*, *QoS mapping* (which translates QoS representations between layers) and *QoS adaptation*. The latter is a facility whereby users can pre-specify actions to be taken in the event of QoS degradations. For example, users can choose to take no action, to be informed via a degradation indication, or to transparently renegotiate.

11.5.3 Other QoS Research

Work carried out at the University of Pennsylvania [Nahrstedt,93] describes a *brokerage* model which incorporates QoS translation, and QoS negotiation and renegotiation. The notion of *eras* is introduced to describe variations in QoS parameters for complex, long lived applications. Negotiation and renegotiation provide a mechanism to signal variations in QOS performance parameters at the user-network interface. They are invoked at era boundaries, and can aid resource allocation. In the model application requirements and network resource allocation are expressed in fundamentally different terms and languages. A key part of the model, called a *broker* is responsible for the translation of QoS at the user-network interface.

In addition, several projects in the RACE programme are concerned with QoS for integrated broadband networks. A significant contribution has been made by the QOSMIC (R.1082) project which studied QoS concepts in broadband networks, focusing on the user-network interface in particular. The major goal of the project was the specification of a QoS model for service life-cycle management which maps the user communication requirements to network performance parameters in a methodical manner. The model takes into account the life cycle of service from both the user and network provider viewpoints, and QOS performance mapping between the viewpoints based on the decomposition of end-to-end services into elements. The service life-cycle covers conception, planning, provision and operation of multimedia services in an integrated broadband environment. In some related work Jung and Seret [Jung,93] propose a framework for the translation of the performance parameters between the ATM Adaptation Layer (AAL) and ATM layers. They extend the QOSMIC model to include QoS verification. In this case the user can verify whether the achieved bearer QoS provided by the ATM network meets the contracted requirements expressed in terms of performance parameters.

11.6 Conclusion

In this chapter we have aimed to indicate the importance of Quality of Service management in distributed systems, and to show how this could be built into the framework of the OSI communications standards.

We have summarised and evaluated key research in QoS support and have ended the chapter by describing work with the ambitious aim of integrating and extending some of layer specific research into broader architecture. The notion of a flow and a service contract were introduced as key concepts in capturing, requesting and negotiating end-to-end QoS. We also introduced the idea of flow management which provides for the monitoring and maintenance of the contracted QoS. These QoS concepts emerged from work carried out at Lancaster University and are motivated by the widely accepted communication needs of distributed multimedia applications.

The proposed QoS architecture promotes the idea of *integrated* QoS, spanning the end-systems and the network, and takes the support of QoS for a wide range of applications as its primary goal. Many researchers to date have concentrated on either the network or the end-system in isolation. In contrast, QoS concepts are coherently applied across all architectural layers, resulting in a complete framework for the specification and implementation of the multimedia flows in the local ATM environment.

The area of network support for flows remains an important aspect of the QoS architecture which we have not yet addressed. This future work will draw heavily from the recent literature on providing QoS guarantees in packet switched networks. In particular we plan to investigate suitable switch scheduling disciplines and resource management strategies for the QoS-A, given the types of service commitment we are advocating at the transport service interface.

Acknowledgement

The QoS-A project is funded as part of the UK SERC Specially Promoted Programme in Integrated Multiservice Communication Networks (GR/H77194) in co-operation with Netcomm Ltd. The authors would like to thank Frank Ball and Francisco García, as this paper builds on their earlier work in the area of communication services for distributed multimedia applications.

Abbreviations Used

QoS	Quality of Service
ATM	Asynchronous Transfer Mode
B-ISDN	Broadband Integrated Services Digital Network
MTTR	Mean Time to Repair
MTBF	Mean Time Between Failures
OSI-RM	Reference Model for Open Services Interconnection

REFERENCES

Accetta, M., Baron, R., Golub, D., Rashid, R., Tevanian, A., and M. Young, 1986, Mach: A New Kernel Foundation for UNIX Development, *Technical Report*, Department of Computer Science, Carnegie Mellon University.

Anagnostou, M.E. et al, 1991, Quality of Service Requirements in ATM-Based B-ISDNs, *Computer Networks and ISDN Systems*, Vol 14, No 4, pp 197-204.

Anderson, D.P., Osawa, Y., Govindan, R., 1992, A File System for Continuous Media, *ACM Transactions on Computer Systems*, Vol 10 No 4, pp 311-337.

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

ANSI, 1991, High Speed Transport Protocol (HSTP) Specification, *X3S3.3/91-264*, American National Standards Institute.

APM Ltd., 1989, *The ANSA Reference Manual Release 01.01*, Architecture Projects Management Ltd., Poseidon House, Castle Park, Cambridge CB3 0RD, UK.

APM Ltd., 1991, *ANSAware 3.0 Implementation Manual*, APM Ltd., Poseidon House, Castle Park, Cambridge CB3 0RD, UK.

APM Ltd., 1993, The ANSA Application Programmer’s Guide, Release 4.1, APM Cambridge Limited, Poseidon House, Castle Park, Cambridge CB3 0RD, UK.

APM Ltd., 1993, *The ANSA Application Programmer’s Guide, Release 4.1*, APM Ltd., Poseidon House, Castle Park, Cambridge CB3 0RD, UK.

Arnould, E.A., Bitz, F.J., Cooper, E.C., Kung, H.T., Sansom, R.D. and P.A. Steenkiste, 1989, The Design of Nectar: A Network Backplane for Heterogeneous Multicomputers, *Computer Architecture News*, Vol. 17, No. 2, pp 205-216.

Ball, F., D. Hutchison, A.C. Scott, and W.D. Shepherd, 1990, A Multimedia Network Interface, *3rd IEEE COMSOC International Multimedia Workshop (Multimedia '90)*, Bordeaux, France.

Birman, K.P., and T.A. Joseph, 1982, Exploiting Virtual Synchrony in Distributed Systems, *Operating System Review*, Vol. 21 No. 5, pp 123-138.

Birman, K.P., Schiper, A., Stephenson, P., 1991, Lightweight Causal and Atomic Group Multicast, *ACM Transactions on Computer Systems,* Vol. 9, No. 3, pp 272-314.

Boerjan, J., Campbell A., Coulson G., García F., Hutchison D., Leopold, H. and N. Singer, 1992, The OSI 95 Transport Service and the New Environment, *ISO/IEC JTC1/SC6/WG4 N824*, International Standards Organisation, UK, De, 1992, and Internal Report No. MPG-92-38 Department of Computing, Lancaster University, Lancaster LA1 4YR.

Bricker, A., Gien, M., Guillemont, M., Lipkis, J., Orr, D., and M. Rozier, 1991, Architectural Issues in Microkernel-based Operating Systems: the CHORUS Experience, *Computer Communications*, Vol 14, No 6, pp 347-357.

Bulterman, D.C. and R. van Liere, 1991, Multimedia Synchronisation and UNIX, *Proc. Second International Workshop on Network and Operating System Support for Digital Audio and Video*, Heidelberg, Springer Verlag.

Campbell, A., and D. Hutchison, 1992, Contribution to the new ISO Work Item: Key Issues in Distributed Multimedia Communications, *Draft BSI/IST6/-/2/738,* British Standards Institute, UK.

Campbell, A., Coulson G. and D. Hutchison, 1992, A Suggested QoS Architecture for Multimedia Communications, *ISO/IEC JTC1/SC21/WG1 N1201*, International Standards Organisation, UK, November, 1992, and Internal Report No. MPG-92-37 Department of Computing, Lancaster University, Lancaster LA1 4YR.

Campbell, A., Coulson G., Garcia F., and D. Hutchison, 1992, A Continuous Media Transport and Orchestration Service, *Proc. ACM SIGCOMM '92,* Baltimore, Maryland, USA.

Campbell, A., Coulson, G., García, F., Hutchison, D., and H. Leopold, 1993, Integrated Quality of Service for Multimedia Communications, *Proc. IEEE INFOCOM'93,* pp. 732-739, San Francisco, USA.

CCITT, 1990, Draft Recommendations I.*, CCITT, Geneva, Switzerland.

Cidon, I., Gopal, I., Gopal P.M., Janniello and M. Kaplan, 1992, The plaNET/ORBIT High Speed Network, *Internal Report No. 18270*, IBM T.J. Watson Research Center.

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

Cocchi, R., Estin, D, Shenker, S. and L. Zhang, 1991, A Study of Priority Pricing in Multiple Service Class Networks, *Proc. ACM SIGCOMM '91,* pp. 123-130.

Coulson G., Blair G.S., Stefani, J.B., Horn, F. and Hazard, L., 1993, Supporting the Real-time Requirements of Continuous Media in Open Distributed Processing, *Internal Report MPG-93-10*, Department of Computing, Lancaster University, Lancaster LA1 4YR, UK. .

Coulson, G., and G.S. Blair, 1993, Micro-kernel Support for Continuous Media in Distributed Systems, *To appear in Computer Networks and ISDN Systems, Special Issue on Multimedia*; also available as Internal Report MPG-93-04, Computing Department, Lancaster University, Bailrigg, Lancaster, U.K.

Coulson, G., Blair, G.S., Davies, N. and N. Williams, 1992, Extensions to ANSA for Multimedia Computing, *Computer Networks and ISDN Systems*, 25, pp 305-323.

Crosby, S., 1993, MSNL Connection Management, *ATM Document Collection 2, Technical Notes*, pp. 12-1, 12-11, Systems Research Group, Computer Laboratory, University of Cambridge, UK.

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

Davies, N., G. Coulson, N. Williams, and G.S. Blair, 1991, Experiences of Handling Multimedia in Distributed Open Systems, *Proc.SEDMS '92, Usenix Symposium on Distributed and Multiprocessor Systems,* Newport Beach, CA, USA.

Davies, N.A. and Nicol, J.R., 1991, A Technological Perspective on Multimedia Computing, *Computer Communications*, Vol 14, No 5.

DEC, 1991, *Digital Network Architecture, Naming Service Functional Specification,* Version V2.0.0 Order No. EK-DDNANS-FS-002.

DePrycker, M., 1991, *Asynchronous Transfer Mode: Solution for Broadband ISDN*, ISBN 0-13-053513-3, Ellis Horwood.

Feldmeier, D., 1992, Architectural Concepts for High Speed Communications Systems, *Internal Report*, Bellcore, Morristown, NJ, USA.

Ferrari, D., and D.C. Verma, 1990, A Scheme for Real-Time Channel Establishment in Wide-Area Networks, *IEEE JSAC*, Vol 8, No 3, pp 368-377.

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

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

Forin, A., Golub, D. and Bershad, B., 1990, An I/O System for Mach 3.0, *Internal Report*, Carnegie Mellon University, Pittsburg, USA.

García, F., 1993, A Continuous Media Transport and Orchestration Service, *Ph.D Thesis*, Department of Computing, Lancaster University, Lancaster LA1 4YR, UK.

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

Hanko, J.G., Keurner, E.M., Northcutt, J.D. and G.A. Wall, 1991, Workstation Support for Time Critical Applications, *Proc. Second International Workshop on Network and Operating System Support for Digital Audio and Video*, Heidelberg, Springer Verlag.

Hehmann, D.B., M.G. Salmony, and H.J. Stüttgen, 1990, Transport Services for Multimedia Applications on Broadband Networks, *Computer Communications*, Vol. 13, No. 4, pp 197-203.

Hehmann, D.B., R.G. Herrtwich, W. Schulz, T. Schuett, and R. Steinmetz, 1991, 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.

Henshall, J., and S. Shaw, 1988, *OSI Explained: End-to-end Computer Communication Standards*, ISBN 07458-0253-2, Ellis Horwood.

Hutchison, D., Campbell, A. and H. Leopold, 1992, “Key Issues in Multimedia Communications”, *ISO/IEC JTC1/SC6/WG4 SD/14*, International Standards Organisation, UK, November, 1992, and Internal Report No. MPG-92-39 Department of Computing, Lancaster University, Lancaster LA1 4YR.

ISO, 1992, Draft Guidelines for Enhanced Communication Function and Facilities for the Lower Layers, ISO/IEC JTC1/SC6/WG4 N7309 International Standards Organisation, UK.

ISO, 1992, Draft Recommendations X.903: Basic Reference Model of Open Distributed Processing”, ISo/IEC JTC1/SC21/WG7, International Standards Organisation, UK.

ISO, 1992, Quality of Service Framework - Outline, *ISO/IEC JTC1/SC21/WG1 N1145*, International Standards Organisation.

ISO, 1992, User Requirements for Quality of Service, *ISO/IEC JTC1/SC21/WG1 N1146*, International Standards Organisation.

Leopold, H., Blair, G., Campbell, A., Coulson, G., Dark, P., Garcia, F., Hutchison, D., Singer, N., and N. Williams, 1992, Distributed Multimedia Communications System Requirements, *OSI95/Deliverable ELIN-1/C/V3*, Alcatel ELIN Research, A-1210 Vienna, Ruthnergasse 1-7, Austria.

Leopold, H., Campbell, A., Hutchison, D. and N. Singer, 1992, Towards an Integrated Quality of Service Architecture (QoS-A) for Distributed Multimedia Communications, *Proc. 4th IFIP Conference on High Performance Networking*, University of Liege, Liege, Belgium.

Little, T.D.C., and A. Ghafoor, 1990, Synchronisation Properties and Storage Models for Multimedia Objects, *IEEE Journal On Selected Areas in Communications*, Vol 8, No 3, pp 413-427.

Liu, C.L. and Layland, J.W., 1973, Scheduling Algorithms for Multiprogramming in a Hard Real-time Environment, Journal of the Association for Computing Machinery, pp 46-61.

Lougher, P., Shepherd, D., 1993, The Design of a Storage Server for Continuous Media, *The Computer Journal*, Vol 36 No 1, pp 32-42.

Mullender, S. (ed), 1993, “Distributed Systems, 2nd Edition, Addison-Wesley.

Nicolaou C., 1991, *A Distributed Architecture for Multimedia Communication Systems*, Ph.D Thesis, University of Cambridge, Computer Laboratory, UK.

Nicolaou, C., 1990, An Architecture for Real-Time Multimedia Communication Systems, *IEEE Journal on Selected Areas in Communications,* Vol 8, No 3, pp 391-400.

Parekh, A., 1993, 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.

Partridge, C., 1990, A Proposed Flow Specification; RFC-1363, *Internet Request for Comments*, no. 1363, Network Information Center, SRI International, Menlo Park, CA, USA.

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

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

Shepherd, W.D., D. Hutchison, F. Garcia and G. Coulson, 1991, Protocol Support for Distributed Multimedia Applications, *Second International Workshop on Network and Operating System Support for Digital Audio and Video,* IBM ENC, Heidelberg, Germany.

Sluman, C., 1991, Quality of Service in Distributed Systems, *BSI/IST21/-/1/5:33*, British Standards Institute, UK.

Stankovic, J.A. and K. Ramamritham, 1988, Tutorial on Hard Real-Time Systems, *IEEE Computer Society Order Number 819,* ISBN 0-8186-0819-6.

Stefani, J-B., 1993, Some Computational Aspects of QoS in ANSA Technical Report available from: Centre National d'Etudes des Télécommunications (CNET), 38-40 rue du Général Leclerc, 92131 Issy-les-Moulineaux, France.

Stefani, J.B., 1992, Sur les Proprietes Temporelles d'une Machine d'Execution ESTEREL, *Internal Report,* Centre National d'Etudes Telecommunications, 38-40 rue du General Leclerc, 92131 Issy-les-Moulineaux, Paris, France.

Tanenbaum, A.S., van Renesse, R., van Staveren, H. and S.J. Mullender, 1988, A Retrospective and Evaluation of the Amoeba

Distributed Operating System, *Technical Report*, Vrije Universiteit, CWI, Amsterdam.

Tennenhouse, D.L., 1990, Layered Multiplexing Considered Harmful, Protocols for High-Speed Networks, Elsevier Science Publishers B.V. (North-Holland).

The Open Software Foundation, 1992, Distributed Computing Environment, 11 Cambridge Center, Cambridge, MA 02142, USA.

Tokuda, H., Kitayama, T., 1993, 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.

Topolcic, C, 1990, Experimental Internet Stream Protocol, Version 2 (ST-II), *Internet Request for Comments No. 1190* , RFC-1190.

Vin, H.M., Rangan, P.V., 1993, Designing a Multi-User HSTV Storage Server, *IEEE Journal on Selected Areas in Communications*, Vol 11, No 1, pp 153-164.

Williams, N., G.S. Blair, 1991, Distributed Multimedia Application Survey, *Internal Report N. MPG-91-11.* Computing Department, Lancaster University, Bailrigg, Lancaster LA1 4YR, UK.

Wolfinger, B., and M. Moran, 1991, 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.

Zhang, H., and S. Keshav, 1991, Comparison of Rate-Based Service Disciplines, *CACM*, pp 113-121.

Zhang, L. and D. Ferrari, 1993, Rate-Controlled Static-Priority Queueing, *Proc. IEEE INFOCOM'93*, San Francisco, USA, pp. 227-237.

Zhang, L., Deering, S., Estin, D, Shenker S. and D. Zappala, 1993, RSVP: A New Resource ReSerVation Protocol", Draft available via anonymous ftp from parcftp.xerox.com:/transient/ rsvp.ps.Z.