

A Survey of QoS Architectures

Cristina Aurrecochea, Andrew T. Campbell and Linda Hauw
Center for Telecommunication Research
Columbia University, New York, NY 10027, USA
<http://www.ctr.columbia.edu/comet/members.html>
e-mail: [cris,campbell,linda]@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 been 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 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 evaluate a number of QoS architectures that have emerged in the literature.

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. Consider, for example, 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 Figure 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 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 [20]. 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) [20-28], end system [29-35] and network [36-66] support for quality of service. In end-systems, most of the

progress has been made in the areas of scheduling [11] [12] [31], flow synchronisation [18] [19] and transport support [36-45]. In networks, research has focused on providing suitable traffic models [2] and service disciplines [52], as well as appropriate admission control and resource reservation protocols [48] [51] [53]. Many current network architectures, however, address quality of service from a providers point of view and analyze 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 carried out in the context of the ODP [27].

The current state of QoS provision in architectural frameworks can be summarized as follows [20]:

- 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 an overall framework*: it is necessary to develop an overall architectural framework to build upon and reconcile the existing notion of quality of service at different system levels and among different network architectures.

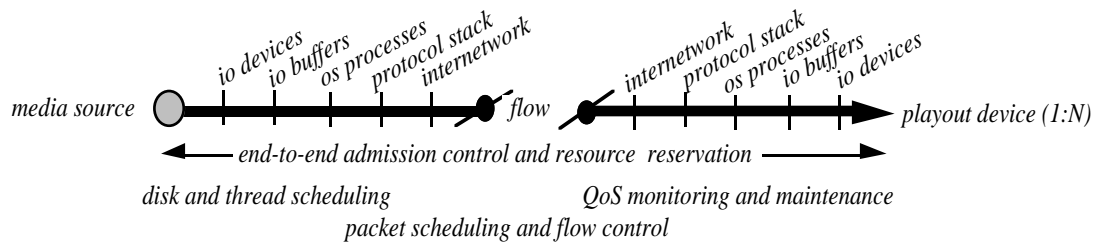


Figure 1. End-to-End QoS Scenario

In recognition of the above limitations, a number of research teams have proposed a systems architectural approach to QoS provision. In this paper these are referred to as *QoS architectures* [67-90]. 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.

In this paper we present, in section 2, a *generalized QoS framework* and terminology¹ for distributed multimedia applications operating over multimedia networks with QoS guarantees. The generalized QoS framework is based on a set of principles that govern the behavior of QoS architectures. Following this, we evaluate a number of QoS architectures found in the literature that have been developed by the telecommunications, computer communications and standards communities. We then present a short qualitative comparison and discussion in sections 4 and 5, respectively. Finally, in section 6 we offer some concluding remarks.

1. Where appropriate we have adopted the standard terminology of the ISO QoS Working Group [67].

2. Generalised QoS Framework

In what follows, a set of elements used in building QoS into distributed multimedia systems is described. This includes QoS principles which govern the construction of a generalized QoS framework, QoS specification which captures application-level QoS requirements, and QoS mechanisms which realize the desired application end-to-end QoS behaviour.

2.1 QoS Principles

A number of QoS principles motivate the design of a generalized QoS framework:

- *integration principle* states that QoS must be configurable, predictable and maintainable over all architectural layers to meet end-to-end QoS [2]. Flows¹ traverse resource modules (e.g., CPU, memory, multimedia devices, network, etc.) at each layer from source 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;
- *separation principle* states that media transfer, control and management are functionally distinct architectural activities [69]. The principle states that these tasks should be separated in architectural QoS frameworks. One aspect of this 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;
- *transparency principle* states that applications should be shielded from the complexity of underlying QoS specification and QoS management. An important aspect of transparency is the QoS-based API [74] [9] at which desired QoS levels are stated (see QoS management policy in section 2.2). The benefits of transparency are that it reduces the need to embed functionality in the application, hides the detail of underlying service specification from the application and it delegates the complexity of handling QoS management activities to the underlying framework;
- *multiple timescales principle* [69] 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 resource management activities (e.g., scheduling, flow control, routing, QoS management, etc.) in distributed communications environments; and
- *performance principle* subsumes a number of widely agreed rules for the implementation of QoS-driven communications systems which guide the division of functionality in structuring communication protocols for high performance in accordance with systems design principles [6], avoidance of multiplexing [7], recommendations for structuring communications protocols [8], and the use of hardware assists for efficient protocol processing [40] [55].

1. 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.

2.2 QoS Specification

QoS specification is concerned with capturing application level QoS requirements and management policies. QoS specification is generally different at each system layer and is used to configure and maintain QoS mechanisms resident in the end-system and network. For example, at the distributed system platform level QoS specification is primarily application-oriented rather than system-oriented. Lower-level considerations such as tightness of synchronisation of multiple related audio and video flows, or the rate and burst size of flows, or the details of thread scheduling in the end-system should all be hidden at this level. QoS specification is therefore declarative in nature: applications 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 characterizes the degree of synchronisation (i.e., tightness) between multiple related flows [18]. 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 synchronization in multimedia flows does not need to be absolutely precise [19] when the main information channel is auditory and video is only used to enhance the sense of presence;
- *flow performance specification*, which characterizes the user's flow performance requirements [5]. 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 QoS frameworks 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 [49], predictive [47] and best effort [8]). 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 refined in a qualitative way to allow a distinction to be made between hard and soft performance guarantees. Level of service expresses a degree of certainty that the QoS levels requested at the time of flow establishment or re-negotiation will be honored;
- *QoS management policy*, which captures the degree of QoS adaptation [74] that the flow can tolerate and scaling actions to be taken in the event of violations in the contracted QoS [86]. 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 application-level selection for QoS indications (aka QoS alerts [67]) in the case of violations in the requested quality of service, and periodic QoS availability notifications for bandwidth, delay, jitter and loss. QoS availability notification are suitable for adaptive applications [13]; and
- *cost of service*, which specifies the price the user is willing to incur for the level of service [10]. Cost of service is a 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, e.g., guaranteed service.

2.3 QoS Mechanisms

QoS mechanisms are selected and configured 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 QoS control and QoS management is characterized by the different timescales which they operate over: QoS control operates on a faster timescale than QoS management.

2.3.1 QoS Provision Mechanisms

QoS provision is comprised of the following components:

- *QoS mapping*, which 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. As illustrated in Figure 2, QoS mapping derives the scheduler QoS parameters (viz. period, quantum and deadline times of the threads) from the transport level QoS specification parameters [34];

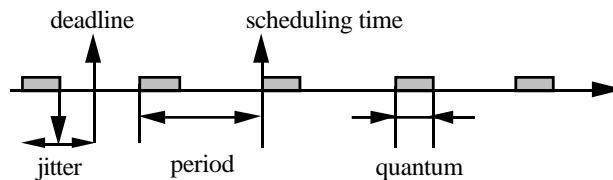


Figure 2. QoS Parameters derived during QoS Mapping

- *admission testing*, which is responsible for comparing the resource requirement arising from the requested QoS against the available resources in the system. The decision whether a new request can be accommodated generally depends on system-wide resource management policies and 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; and
- *resource reservation protocols*, which 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 mechanisms such as network-level cell/packet schedulers and end-system thread schedulers are configured accordingly.

2.3.2 QoS Control Mechanisms

QoS control mechanisms operate on timescales at or 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. The fundamental QoS control mechanisms include the following:

- *flow shaping*, which regulates flows based on user supplied flow performance specifications. Flow shaping can be based on a fixed rate throughput (i.e., peak rate) or some form of statistical representation (i.e., sustainable rate and burstiness) of the required bandwidth [49]. The benefit of shaping traffic is that it allows QoS frameworks 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 proven¹ that the combination of traffic shaping at the edge of the network and scheduling in the network can provide hard performance guarantees;
- *flow scheduling*, which manages the forwarding of flows (chunks of media based on application layer framing) in the end-system [30-35] and network (packets and/or cells) in an integrated manner [52]. 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 discipline [2] adopted;
- *flow policing*, which 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 [53]. Flow shaping schemes at the source allow the policing mechanism to detect misbehaving flows;
- *flow control*, which 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 [41] or network [64]. Applications using closed loop flow control based protocols must be able to adapt to fluctuations in the available resources. On the other hand, 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*, which is required to control the event ordering and precise timings of multimedia interactions. Lip-sync is the most commonly cited form of multimedia synchronisation (i.e., 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 [44].

1. Parekh [56] 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 [58], it is possible to achieve strong guarantees on delay.

2.3.3 QoS Management Mechanisms

To maintain agreed levels of QoS it is often not sufficient to just commit resources. Rather, 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 [15]. The fundamental QoS management mechanisms include the following:

- *QoS monitoring*, which allows each level of the system to track the ongoing QoS levels achieved by the lower layer. QoS monitoring often plays an integral part in a QoS maintenance feedback loop which maintains the QoS achieved by resource modules. Monitoring algorithms operate over different timescales. For example, they can run as part of a 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 admission control [51]. Alternatively, QoS monitoring can operate on an end-to-end basis as part of a transport level feedback mechanism [44] or as part of the application itself [13];
- *QoS maintenance*, which compares the monitored QoS against the expected performance and then exerts tuning operations (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 management, throughput via the flow regulation, and queuing delays and continuous media playout calculation via the flow scheduling [86]);
- *QoS degradation*, which 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-back[85] to a reduced level of service (i.e., end-to-end renegotiation);
- *QoS availability*, which allows the application to specify the interval over which one or more QoS parameters (e.g., delay, jitter, bandwidth, loss, synchronisation) can be monitored and the application informed of the delivered performance via a QoS signal [74]. Both single and multiple QoS signals can be selected depending whether the user requested QoS management policy (see section 2.2); and
- *QoS scalability*, which 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 QoS. Based on the user supplied QoS management policy, QoS adaptation in the end-systems can take remedial actions to scale flows appropriately. Resolving heterogeneous QoS issues is a particularly acute problem in the case of multicast flows. Here individual receivers may have differing QoS capabilities to consume audio-visual flows; QoS filtering helps to bridge this heterogeneity gap while simultaneously meeting individual receivers' QoS requirements [90].

3. 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 points that end-systems

are attached to [81]. 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 a number of distinct approaches which have recently emerged in the literature [67-90].

3.1 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 [71]. The communications architecture includes a continuous media transport systems (HeiTS/TP) [42] which provides QoS mapping and media scaling [85] as illustrated in Figure 3. Underlying the transport is an internetworking layer based on ST-II [46] 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 *HeiRAT* (*resource administration technique*) [71]. HeiRAT comprises a comprehensive QoS management scheme which includes QoS negotiation, QoS calculation, admission control, QoS enforcement and resource scheduling. The HeiRAT operating system scheduling policy is a rate-monotonic scheme whereby the priority of a system thread performing protocol processing is proportional to the message rate requested.

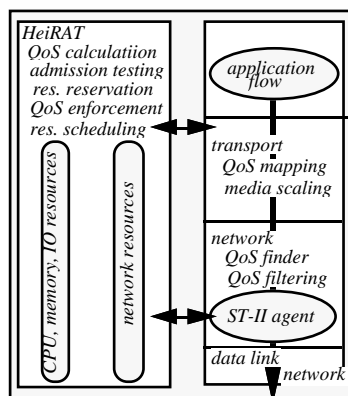


Figure 3. Heidelberg QoS Model

The Heidelberg QoS model has been designed to handle heterogeneous QoS demands from individual receivers in a multicast group and to support QoS adaptivity via flow filtering and media scaling techniques. Media scaling [85] and codec translation at the end systems, and flow filtering and resource sharing in the network are fundamental 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 flows traverse bridges, switches and routers.

3.2 XRM

The COMET group at Columbia University is developing an *Extended Integrated Reference Model* (XRM) [28] as a modeling framework for control and management of multimedia telecommunications networks (which comprise multimedia computing platforms and broadband networks). The

COMET group argues 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. The XRM is divided into five distinct planes [69] as illustrated in Figure 4:

- *management function*, which resides in the network management plane (N-plane) and 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, 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
- *teletbase*, which resides in the data abstraction and management plane (D-plane) and collectively represents the information, data abstractions existing in the network and end-systems. The teletbase implements data sharing among all other XRM planes.

The XRM is built on theoretical work of guaranteeing QoS requirements in ATM networks and end-systems populated with multimedia devices. General concepts for characterizing the capacity of network [82] and end-system [73] 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 characterized 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.

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 characterizing 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. The implementation of XRM including key resource abstractions (viz. schedulable and multimedia capacity region) is currently being realised as part of a *binding architecture* [28] for open signalling, control and management of multimedia networks.

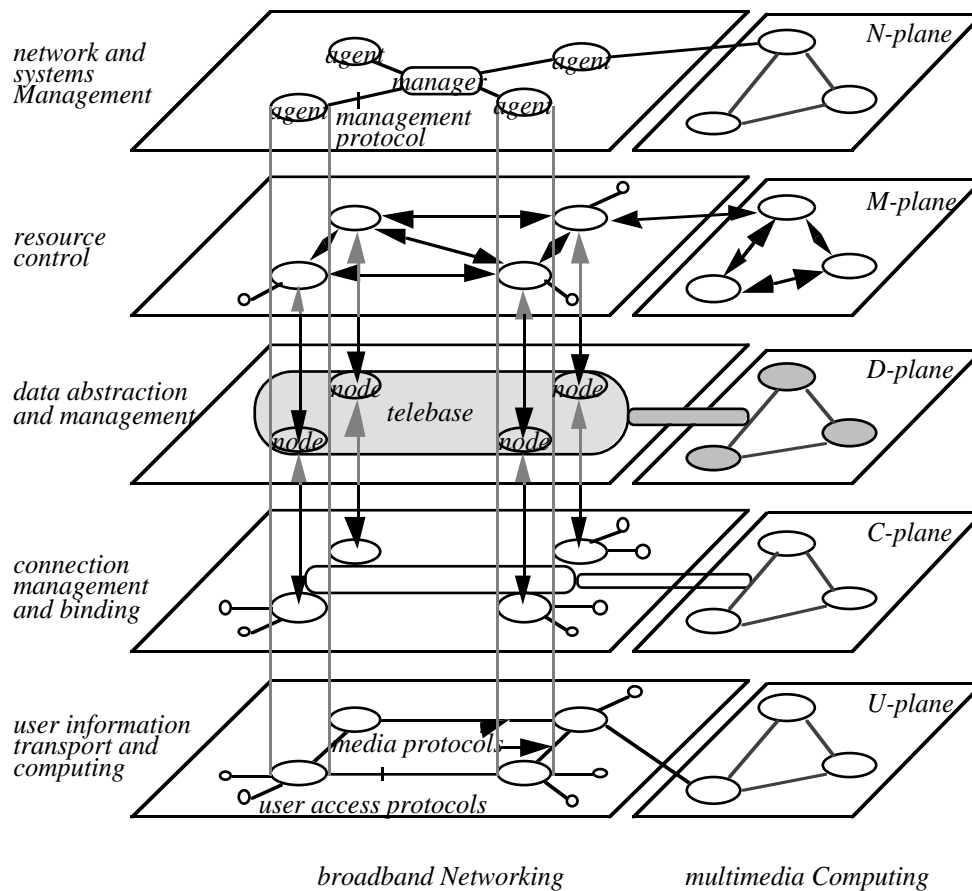


Figure 4. XRM

3.3 OMEGA

Over the past three years the University of Pennsylvania has been developing an end-point architecture called the OMEGA architecture [70]. 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 illustrated in Figure 5 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. Communication is preceded by a call setup phase where application requirements, expressed in terms of QoS parameters, are negotiated, and guarantees are made at several 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 has also developed a *QoS brokerage model* [88] which incorporates QoS translation, and QoS negotiation and renegotiation (see [89] for full details on similar work on QoS negotiation protocol at University of Montreal).

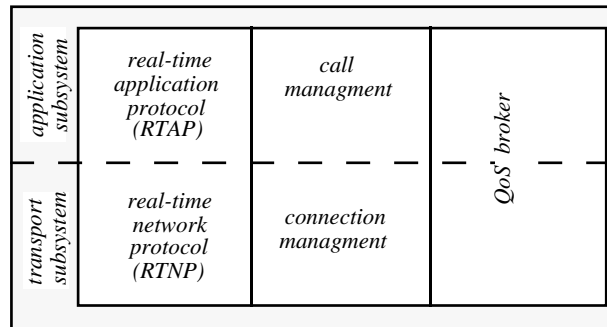


Figure 5. OMEGA

3.4 int-serv Architecture

The work by the Integrated Services (int-serv) Group [62] of the Internet Engineering Task Force (IETF) is a significant contribution to providing controlled QoS for multimedia applications over an integrated services Internetwork. The group has defined a comprehensive int-serv architecture [62] and a QoS framework [79] used to specify the functionality of internetwork system elements (known as elements) which make multiple, dynamically selectable QoS available to applications. The behaviour of elements, which constitute routers, subnetworks and end-point operating systems, is captured as a set of services, of which some or all are offered by each element. Each element is QoS-aware and supports interfaces required by the service definition [62]. The concatenation of service elements along an end-to-end data path provides an overall statement of end-to-end QoS. The following int-serv services are offered in addition to best effort: (i) *controlled delay*, which attempts to provide several levels of delay which the application can choose from; (ii) *predicated delay*, which provides a statistical delay bound similar to Tenet Group's statistical service [49] and the COMET Group's guaranteed service [61]; and (iii) *guaranteed delay*, which provides an absolute guaranteed delay bound.

Flows in an int-serv architecture are characterized by two specifications: a *traffic specification*, which is a specification of the traffic pattern which a flow expects to exhibit; and a *service request specification*, which is a specification of the QoS a flow desires from a service elements. The int-serv architecture, which is restricted to the network but applicable in the end-system too, is comprised of four components [62]:

- a *packet scheduler*, which forwards packets streams using a set of queues and timers;
- a *classifier*, which maps each incoming packet into a set of QoS classes;
- an *admission controller*, which implements the admission control algorithm to determine whether a new flow can be admitted or denied; and
- a *reservation setup protocol* (e.g., RSVP [48]), which is necessary to create and maintain flow-specific state in the routers along the path of the flow.

In [80] Clark introduces some early work on a *Quality of Service Manager (QM)* as part of the end-system int-serv architecture. The QM as illustrated in Figure 6 presents an abstract management layer designed to isolate applications from underlying details of specific services provided in a QoS-driven Internet [62]. 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

described above; in this case, the QM provides a degree of transparency whereby applications express desired levels of QoS in application-oriented language rather than using communication QoS specifics. The QM is responsible for determining what QoS management capabilities are available on the application's communication path, and chooses the path best suited to the application.

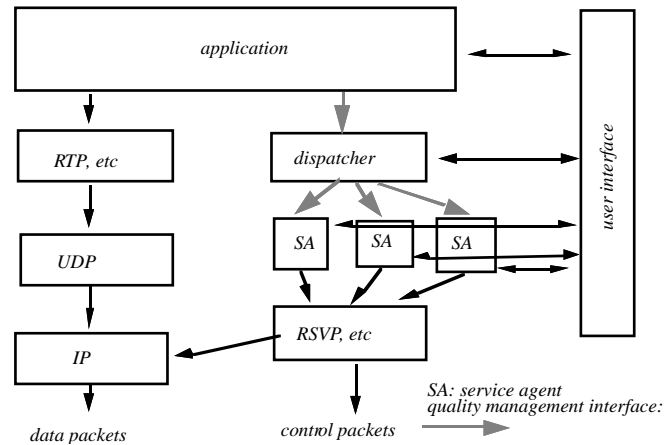


Figure 6. int-serv Architecture QoS Manager

3.5 QoS-A

The *Quality of Service Architecture (QoS-A)* [68] is a layered architecture of services and mechanisms for quality of service management and control of continuous media flows in multiservice networks. The architecture incorporates the following key notions: *flows*, which characterize 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 realization of the flow concept demands active QoS management and tight integration between device management, end-system thread scheduling, communications protocols and networks.

In functional terms, the QoS-A (as illustrated Figure 7) 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 [24]. Below the platform level is an orchestration layer which provides jitter correction and multimedia synchronization services across multiple related application flows [44]. Supporting this is a transport layer which contains a range of QoS configurable services and mechanisms [34]. Below this, an internetworking layer and lower layers form the basis for end-to-end QoS support.

QoS management is realized 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 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 [44], 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 QoS adaptation for coarse grained QoS maintenance control).

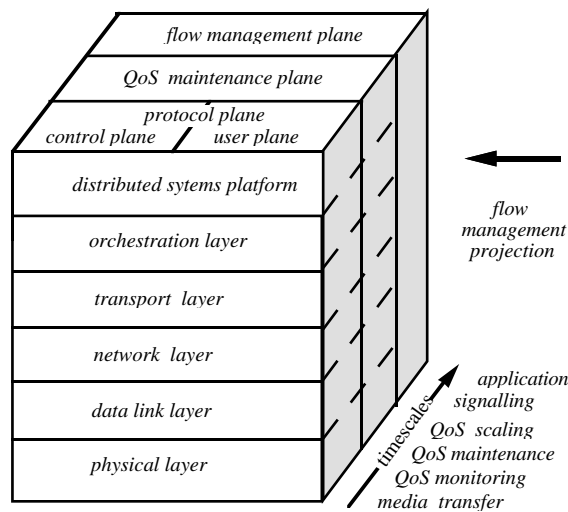


Figure 7. QoS-A

3.6 OSI QoS Framework

One early contribution to the field of QoS-driven architecture is the *OSI QoS Framework* [67] which concentrates primarily on quality of service support for OSI communications. 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 OSI 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 (as illustrated in Figure 8) is made up of two types of management entities (*viz. layer specific* and *system-wide entities*) that attempt to meet the QoS requirements by monitoring, maintaining and controlling end-to-end QoS. The task of the policy control function is to determine the policy which applies at a specific layer of an open system. The policy control func-

tion models any priority actions that must be performed to control the operation of 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. 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.

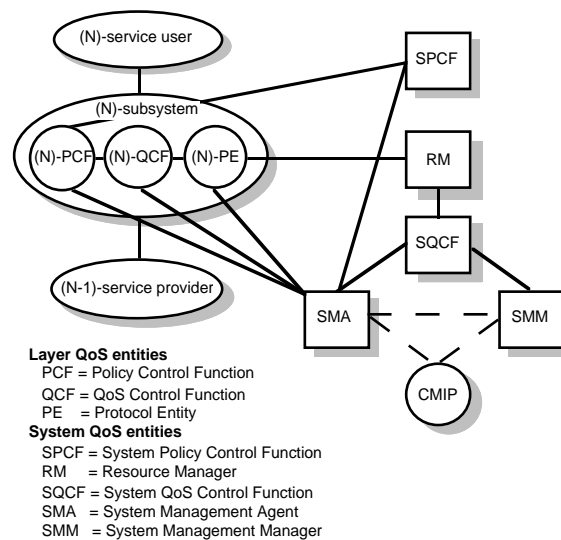


Figure 8. OSI QoS Framework

3.7 Tenet Architecture

The Tenet Group at the University of California at Berkeley has developed a family of protocols [37] [49] which run over an experimental wide area ATM network. As illustrated in Figure 9 the *Tenet Architecture* [84] includes a Real Time Channel Administration Protocol (RCAP) [51] in addition to Real Time Internet Protocol (RTIP), Continuous Media Transport Protocol (CMTP) [37]. 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 bounds. The Tenet Group [50] makes 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.

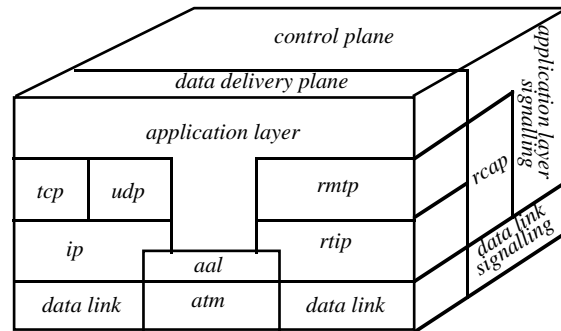


Figure 9. Tenet Architecture

3.8 TINA QoS Framework

The TINA QoS Framework [76] describes a framework for specifying QoS aspects of distributed telecommunications within the context of the Computing Architecture [76]. 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)*; 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 [27] and CNET Framework [26]). 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 declaratively, applications are relieved of the burden of coping with complex resource management mechanisms needed for ensuring QoS guarantees.

3.9 MASI End-to-End Model

The CESAME Project [77] at Laboratoire MASI, Université Pierre et Marie Curie, is developing an architecture for multimedia communications which takes end-to-end QoS support as its primary objective. As with the Lancaster QoS-A, the MASI architecture (shown in Figure 10) 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 spans the host operating system, host communication subsystem 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 [23]; and *iii)* the need to provide suitable communication protocol support for multimedia services being developed at Université Pierre et Marie Curie.

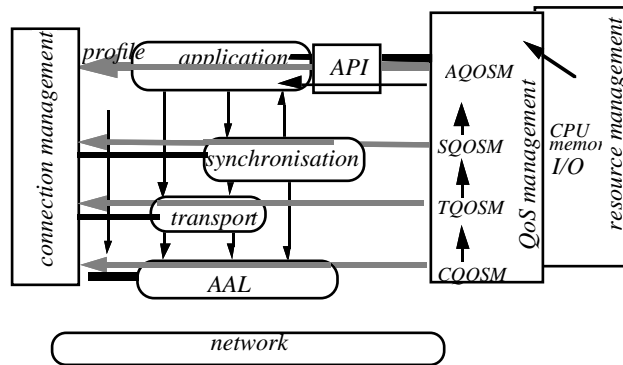


Figure 10. MASI Schematic

3.10 End System QoS Framework

At Washington University, Gopal and Purulkar [72] have 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 as illustrated in Figure 11: QoS specification, QoS mapping, QoS enforcement and protocol implementation. QoS specification is at a high level and uses 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 application session. Important system resources considered in [72] include the CPU, memory and network. 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 [81] has been developed for structuring protocols. RTUs are scheduled using a rate monotonic policy [12] 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.

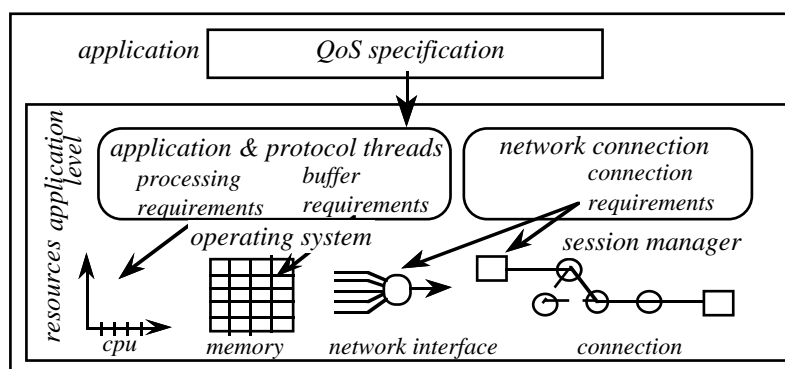


Figure 11. End System QoS Framework

4. Comparison

In this section we present a simple qualitative comparison of QoS architectures survey in section 3. We use the elements of the generalised QoS framework (described in section 2) as a basis for the comparison summarized in Table 1.

QoS Model [Ref]	QoS Provision			QoS Control					QoS Management	
	QoS Mapping	Adm. Control / Resource allocation	E2E ^a Coordination	Flow Scheduling	Flow Shaping	Flow Control	QoS Filtering	Flow Synchronization	Monitoring / Alerts	QoS Maintenance
XRM [28]	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 A D	E N R S
ISO [67]	(E) (N)	E N	E N	-	-	-	-	-	E N	E N
Heidelberg [71]	(E) (N)	E N	E N	E (N)	(E)	(N)	N	-	E D	E R S
TINA [76]	(E)	(N)	N	-	-	-	-	(N)	(N)	-
IETF [62]	E N	-	E	-	-	-	-	-	E N	E N R
Tenet [84]	E N	N	N	N	N	(E)	N	-	E D	E R S
MASI [77]	E (N)	E (N)	E	E	-	-	-	E	E	E
OMEGA [70]	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 Architectures

a. 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 (e.g., RSVP [48]), connection setup protocol (e.g., RCAP [51]) or signalling protocol (e.g., UNI 4.0 [53]).

The legend for the comparison table is as follows:

- “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 detail”
- D “QoS degradation addressed in detail”
- (D) “QoS degradation mentioned only”
- A “QoS availability in detail”

5. Discussion

All QoS architectures surveyed in section 3 consider QoS specification (e.g., services contracts, flow specs, and service and traffic classes, etc.) to be fundamental in capturing application-level QoS requirements. Although there is a broad consensus on the need for a *flow spec* which captures quantitative performance requirements, there exist two schools of thought on what it should be. On the one hand, XRM and ATM [53] solutions are based on a flow spec that is made up of one or two QoS parameters that identify a traffic class and an average bandwidth. On the other hand, the Tenet, QoS-A and OMEGA architectures adopt a multi-valued flow spec (cf. RFC1633, ST-II, RSVP, Hi-eTS). Although both of these proposals seem similar philosophically they are rather different in practice. The COMET group [28] argues that by limiting a flow spec to a set of well defined services in the end-system and traffic classes in the network, complexity in the end-system and network is more 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 the desired application-level QoS requirements.

Level of service (section 2.2) expresses the degree of certainty that the QoS levels specified in a flow spec will be honored. Each architecture offers a different set of services to applications. For example, the Washington University QoS Framework supports three application classes to which it maps applications level flows. These include: i) *an isochronous class*, which is suitable for continuous media flows; ii) *a burst class*, which is appropriate 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 application classes. 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 (IETF), statistical service (Tenet, XRM and Heidelberg) and the available bit rate service (ATM Forum).

With the exception of the IETF work (which uses RSVP maintained state) all architectures advocate connection oriented or 'hard state' solutions to network level QoS provision; that is, hard-state couples path establishment and resource reservation. Work in the IETF on an Integrated Services Architecture (using RSVP and IPv6 flows) described in section 3.4 assumes that network level QoS guarantees can be built 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. In [66] Turner suggests a hybrid approach called *ATM-soft* which benefits from 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.

Commonalities exist between QoS control and management strategies found in the end-system and network: e.g, admission control, resource management, scheduling mechanisms. The extent to which network level QoS mechanisms are applicable in the end-systems (or vice versa) remains an

open issue. End-system and network devices can be modelled in a similar way: the only real difference is the overall goal that end-system or network devices are set to achieve. For example, the XRM models the end-system as a virtual switch [28] and a set of configurable multimedia devices based on a DAN architecture [16]. It is evident that commonalities exist 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 seems encouraging in the first instance. A counter argument, however, 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/packets. This means that in the end-system application execution times (i.e., quantum [16] of work in Figure 2) can vary widely (e.g., uncompressing a video flow is computationally more intensive than displaying video to a screen). In contrast, switch and router schedulers are generally moving packets/cells from queues to ports or vice versa and are optimized for that task. Therefore, techniques resident in switches (such as HRR [22]) may be inappropriate in host operating systems.

6. 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, multiple timescales and performance. Elements of our generalised framework include QoS specification and static and dynamic QoS management. We have summarized and evaluated key research in QoS architecture for distributed systems presenting a discussion of some of the open issues that emerged during a comparison of the existing QoS architectures. While the area of QoS research in multimedia networking is mature, work on QoS architecture remains in its early stages of development with no substantial performance results having been published to validate 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 end-to-end QoS into distributed systems.

7. References

GENERAL QoS BIBLIOGRAPHY

- [1] Keshav, S., "Report on the Workshop on Quality of Service Issues in High Speed Networks", ACM Computer Communications Review, Vol. 22, No. 1, pop 6-15, January, 1993.
- [2] 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, pop 6-15, January 1993.
- [3] Vogel, A.,Bochmann, G. v., Disallow, R., Geckos, J. and B. Kerherv, "Distributed Multimedia Applications and Quality of Service - A Survey", IEEE Multimedia, 1994.
- [4] Miloucheva, I. "Quality of Service Research for Distributed Multimedia Applications", ACM Pacific Workshop on Distributed Multimedia Systems, 1995.
- [5] Partridge, C., "A Proposed Flow Specification", Internet Request for Comments, no. 1363, Network Information Center, SRI International, Menlo Park, CA, September 1990.
- [6] Saltzer, J., Reed, D., and D. Clark, "End-to-end Arguments in Systems Design", ACM Transactions on Computer Systems,

Vol. 2., No. 4., 1984.

- [7] Tennenhouse, D.L., "Layered Multiplexing Considered Harmful", *Protocols for High-Speed Networks*, Elsevier Science Publishers (North-Holland), 1990.
- [8] Clark, D., and D.L. Tennenhouse, "Architectural Consideration for a New Generation of Protocols", *Proc. ACM SIGCOMM '90*, Philadelphia, 1984.
- [9] 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.
- [10] 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.
- [11] C. Liu, J. Layland, "Scheduling Algorithms for Multiprogramming in Hard Real Time Environment", *JACM*, 1973.
- [12] Stankovic et al., "Implications of Classical Scheduling Results for Real-Time Systems", *IEEE Computer*, Special Issue on Scheduling and Real-Time Systems, June 1995.
- [13] Jacobson, V., (1994) "VAT: Visual Audio Tool", vat manual pages, Feb 1993.
- [14] 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.
- [15] 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.
- [16] Hayter, M. and D. McAurely, "The Desk Area Network", *ACM Operating Systems Review*, October 1991.
- [17] 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 Oriented Approach to the Design of Media Processing Environments," *Proc. IEEE International Conference on Multimedia Computing and Systems*, Boston, 1994.
- [18] 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.
- [19] Escobar, J., Deutsch, D. and C. Partridge, "Flow Synchronisation Protocol", *IEEE GLOBECOM'92*, Orlando, FL., December 1992.

DISTRIBUTED SYSTEMS AND OPERATING SYSTEMS QoS BIBLIOGRAPHY

- [20] 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.
- [21] APM Ltd , "ANSAware 3.0 Implementation Manual", APM Ltd, Poseidon House, Castle Park, Cambridge CB3 0RD, UK, 1991.
- [22] Open Software Foundation, "Distributed Computing Environment", 11 Cambridge Center, Cambridge, MA 02142, USA, 1992.
- [23] ODP, "Draft recommendations X.903: basic reference model of open distributed processing", ISO/IEC JTC1/SC21/WG7, International Standards Organization, 1992.
- [24] 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.
- [25] Nicolaou, C., "An Architecture for Real-Time Multimedia Communication Systems", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, April 1990.
- [26] Hazard, L., Horn, F., and J. B. Stefani, "Towards the Integration of Real-Time and QoS Handling in ANSA", *CNET Report CNET.RC.ARCAD.E.01*, June 1993.
- [27] Guangxing, "An Model of Real-Time QoS for ANSA" , Technical Report APM.1151.00.04, APM Ltd, Cambridge, UK,

March 1994.

- [28] 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.
- [29] 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.
- [30] 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.
- [31] 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.
- [32] 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.
- [33] 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.
- [34] 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.
- [35] Jeffay, K and D. Bennett, "A Rate-Based Execution Abstraction For Multimedia Computing," Proc. Fifth International Workshop on Network and Operating Systems Support for Digital Audio and Video," Durham, NH, April 1995.

COMMUNICATIONS QoS BIBLIOGRAPHY

- [36] Danthine, A., Bagueette 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.
- [37] 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.
- [38] Feldmeier, D., "A Framework of Architectural Concepts for High Speed Communication Systems", Computer Communication Research Group, Bellcore, Morristown, May 1993.
- [39] 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.
- [40] Chesson, G., "XTP/PE Overview", Proc. 13th Conference on Local Computer Networks, Pladisson Plaza Hotel, Minneapolis, Minnesota, 1988.
- [41] Clark, D.D., Lambert, M.L., and L. Zhang, "NETBLT: A High Throughput Transport Protocol", Computer Communications Review, Vol. 17, No. 5, 1987.
- [42] 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.
- [43] 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.
- [44] 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.
- [45] 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.
- [46] Topolcic, C., "Experimental Internet Stream Protocol, Version 2 (ST-II)", Internet Request for Comments No. 1190 RFC-

1190, October 1990.

- [47] 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.
- [48] Zhang, L., et. al., "RSVP Functional Specification", Working Draft, draft-ietf-rsvp-spec-10.ps, 1996.
- [49] Ferrari, D. and Verma D. C., "A Scheme for Real-Time Channel Establishment in Wide-Area Networks", IEEE JSAC, 8(3), 368-77, 1990.
- [50] 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.
- [51] 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.
- [52] H. Zhang, S. Keshav, "Comparison of Rate-Based Service Disciplines", ACM SIGCOMM, 1991.
- [53] ATM Forum, ATM User-Network Interface Specifications, Version 4.0, ATM Forum, 1996.
- [54] 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.
- [55] Zitterbart, M., Stiller, B., and A Tantawy, "A Model for Flexible High-Performance Communication Subsystems", IEEE JSAC, May 1992.
- [56] 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.
- [57] Golestani, S.J., "A Stop and Go Queueing Framework for Congestion Management", Proc. ACM SIGCOMM'90, San Francisco, June 1990.
- [58] Keshav, S., "On the Efficient Implementation of Fair Queueing", Internetworking: Research and Experiences, Vol. 2, pp 157-173, 1991.
- [59] Guerin, 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.
- [60] Cruz, R., "A Calculus for Network Delay: Part I: Network Elements in Isolation", IEEE Transactions on Info. Theory, Vol. 37. No. 1, Jan. 1991.
- [61] 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.
- [62] Braden R., Clark, D., and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", Request for Comments, RFC-1633, 1994.
- [63] 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.
- [64] Jain, R., "Congestion Control and Traffic Management in ATM Networks: Recent Advances and a Survey", Computer Networks and ISDN Systems (to appear), 1996.
- [65] Dabbous, W. and C. Diot, "High Performance Protocol Architectures", Technical Report, INRIA, Sophia Antipolis, France, 1995
- [66] Turner, J. "ATM-Soft: A Mini-Proposal for ATM Network Control Using Soft State", Technical Note, Washington University, 1995.

QoS ARCHITECTURES BIBLIOGRAPHY

- [67] ISO, "Quality of Service Framework", ISO/IEC JTC1/SC21/WG1 N9680, International Standards Organization, UK, 1995.
- [68] 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.

- [69] 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.
- [70] 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).
- [71] Volg, C., Wolf, L., Herrtwich, R. and H. Wittig, "HeiRAT - Quality of Service Management for Distributed Multimedia Systems", Multimedia Systems Journal, 1996.
- [72] 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.
- [73] Lazar, A. A., "Challenges in Multimedia Networking", Proc. International Hi-Tech Forum, Osaka, Japan, February 1994.
- [74] Campbell, A., Coulson, G. and D. Hutchison, "A Quality of Service Architecture", ACM Computer Communications Review, April 1994.
- [75] OMG, (1993), "The Common Object Request Broker: Architecture & Specification", Rev 1.3., December 1993.
- [76] TINA-C, "The QoS Framework", Internal Technical Report, 1995.
- [77] 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.
- [78] Sluman, C., "Quality of Service in Distributed Systems", BSI/IST21/-/1/5:33, British Standards Institution, UK, October 1991.
- [79] Shenker, S., and J. Wroclawski, "Network Element Specification Template", Internet Draft, June 1995.
- [80] Slides from IETF meeting 31, Integrated Service Working Group, <ftp://mercury.lcs.mit.edu/pub/intserv>, 1995.
- [81] 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.
- [82] 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.
- [83] 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.\
- [84] Ferrari, D., "The Tenet Experience and the Design of Protocols for Integrated Services Internetworks", Multimedia Systems Journal, 1996.
- [85] 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.
- [86] Campbell, A., Coulson G. and D. Hutchison, "Supporting Adaptive Flows in a Quality of Service Architecture", Multimedia Systems Journal, 1996.
- [87] 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.
- [88] Nahrstedt K. and J. Smith, "The QoS Broker", IEEE Multimedia, Spring 1995.
- [89] 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.
- [90] 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