# Resource Management in Networked Multimedia Systems

**Klara Nahrstedt**
***University of Pennsylvania***

**Ralf Steinmetz**
***IBM European Networking Center***

**User expectations for multimedia applications are imposing enormous demands on network delivery. Proper resource management is crucial to high throughput, fast rates, and time and space guarantees.**

Multimedia computing and communications impose new requirements on network system components. High-speed networks must not only provide fast data transfer but also guaranteed delivery. Continuous media, such as video and audio, must be delivered error-free to users within well-defined time constraints to attain the desired quality during end-system media presentation. Achieving guaranteed, end-to-end delivery in networked multimedia systems will require the solution of several multimedia-specific control-management problems at the host and underlying network levels. This article presents possible approaches and solutions for one of those problems, resource management.

## BASIC CONCEPTS

Networked multimedia systems (NMSs) are a new generation of systems consisting of Broadband Integrated Service Digital Networks, high-performance multimedia workstations, and personal computers (see Figure 1). Their primary goal is proper integration of all components—application, operating system, and network—that ensures portability to different platforms, accommodates dynamic resource changes, and guarantees delivery of the right information at the right time.

### Requirements

Distributed multimedia applications, such as multimedia mail, conferencing, screen sharing, and virtual desktops, are imposing new requirements on data transmission.

- *High data throughput.* The stream-like behavior of audio and video data demands high data throughput. Even when compressed, streams can range from less than 4 Mbits/second up to 100 Mbits/second.
- *Fast transfer rates.* Different applications coexisting in the same end system impose transmission requirements ranging from error-free to time-constrained.
- *Time and space guarantees.* Users will be judging the new applications against the high standards of quality set by radio, television, and telephone services.

All three requirements imply careful multidimensional—time, space, and frequency—resource management to meet quality standards and to prevent problems such as dropped or delayed packets.

### Constraints

Existing operating systems and communication protocols impose several constraints on multimedia transmission.



0018-9162/95/$4.00 © 1995 IEEE

Continuous media transmission is constrained by the end-system architecture. Typically, data is obtained from a source (microphone, camera, or video adapter) and forwarded to a sink (speaker, display, network adapter, or video adapter) as shown in Figure 2. One possibility for satisfying delivery requirements with current systems is to take the "shortest possible path" through the system and move the data from adapter to adapter. The application sets the correct switches for the data flow, but it never really touches the data as in traditional processing. This approach is fast, but it does not provide the necessary resource control and adaptability.

The layered architecture of communication systems implies considerable data movement in the protocols. Because of the expense of physically copying data, virtual copying mechanisms are used. One approach, integrated layer processing, is based on the concept of application-level framing.[1] It structures protocols into one or two integrated processing loops that operate over a single, common data unit instead of over different data units in the various layers of the communication protocol stack. Layering is also used to compress and store data; examples include the MPEG (Moving Pictures Experts Group) system's seven layers and data stored in a CD-ROM/XA (extended architecture) format.[2]

Different communication-system layers may also have different protocol data-unit sizes. If the upper layer wants to transmit a large data size, the data units must be broken into the size required by the underlying layer (for example, ATM cell-sized data units). This segmentation is performed at the sender, and the underlying layer's data units must be reassembled at the receiver.

Some protocols use a retransmission mechanism to achieve reliable data delivery, but this requires additional buffer space for queues and increases end-to-end delays.

## Quality-of-service and resource specifications

Because of the heterogeneity of requirements from different applications, multimedia services are parameterized. Parameterization allows for flexibility and customization of services, so that each application does not implement a new set of system services.

The International Standards Organization (ISO) uses quality of service (QoS), a concept for specifying how "good" networking services are, to define parameterization. Researchers have yet to determine the "best" set of QoS parameters for multimedia systems (or benchmarks to compare the different approaches), but our examples use QoS parameters that are common in the multimedia community (see Table 1).

**QOS SPECIFICATION.** An NMS consists of at least three abstraction layers below the human user, as shown in Figure 3: the application level; the system level, including

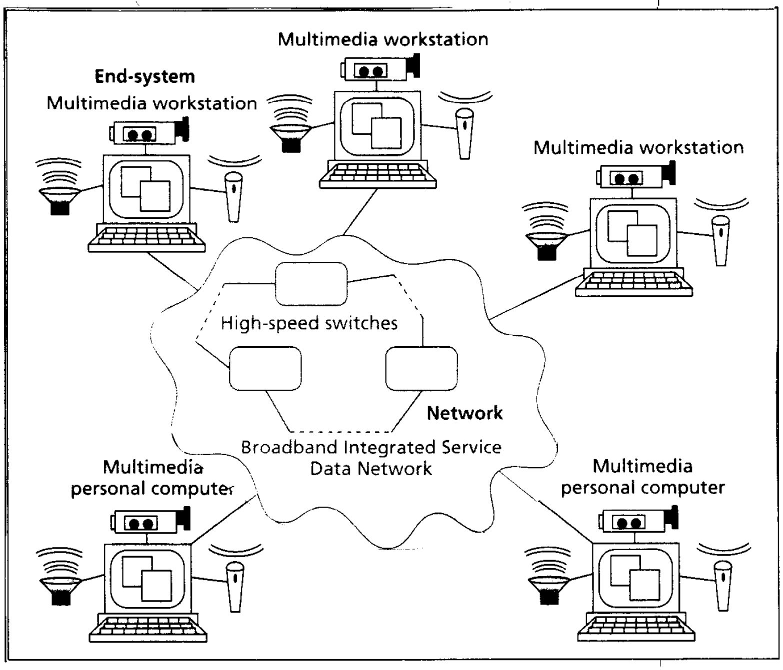


**Figure 1. Networked multimedia system.**

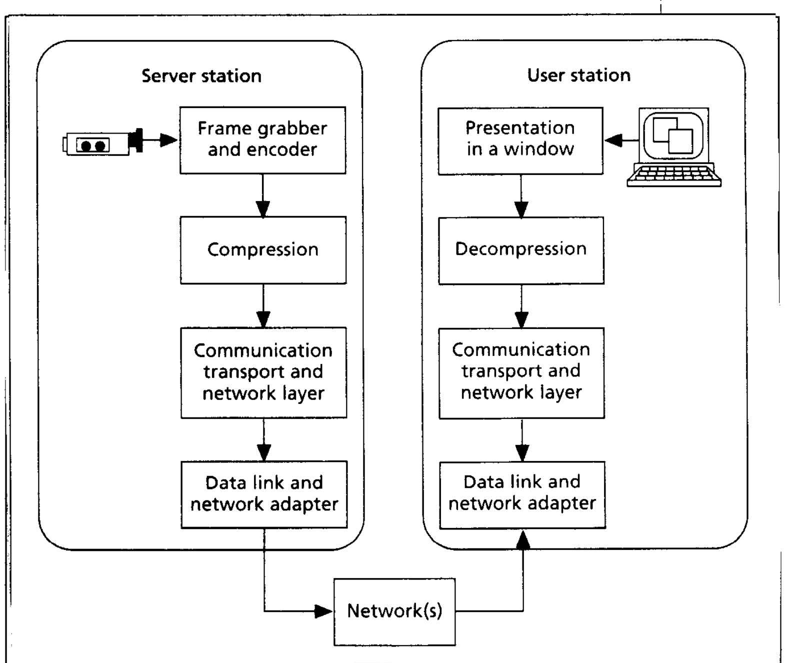


**Figure 2. System components for video data transmission.**

**Table 1. Examples of possible quality of service (QoS) parameters.**

| Medium Type | Quality of Service Parameter | Range | Characterization of Quality |
|---|---|---|---|
| Audio (Application QoS) | Sample size | 8-bit | Telephone voice quality |
| | Sample rate | 8 kHz | (Intermediate delay 125 μs) |
| | Sample size | 16-bit | CD audio |
| | Sample rate | 44.1 kHz | (Intermediate delay 22.7 μs) |
| | Playback point | ≈ 100 to 150 ms | Depends on network delay |
| Audio (Network QoS) | End-to-end delay | 0 to 150 ms | Acceptable for most user apps |
| | | 150 to 400 ms | May impact some apps |
| | | Above 400 ms | Unacceptable |
| | Round-trip delay | Up to 800 ms | Acceptable for conversations |
| | Packet loss | $\leq 10^{-2}$ | Telephone quality |
| | Bandwidth | 16 Kbps | Telephone speech |
| | | 32 Kbps | Audioconferencing speech |
| | | 64 Kbps | Near-CD-quality audio |
| | | 128 Kbps | CD-quality audio |
| Video (Application QoS) | Frame rate | 30 fps | NTSC format |
| | | 25 fps | PAL format |
| | | 60 fps | HDTV format |
| | Frame width | ≤ 720 pixels | Video signal MPEG coded |
| | Frame height | ≤ 576 pixels | Vertical size |
| | Color resolution | 8 bit/pixel | Gray-scale resolution of 256 colors |
| | | 16 bits/pixel | 65,536 possible colors |
| | Aspect ratio | 4:3 | NTSC, PAL TV format |
| | | 16:9 | HDTV format |
| | Compression ratio | 2:1 | Lossless compression of HDTV |
| | | 50:1 | Lossy compression of HDTV |
| Video (System QoS) | Decoder buffer | ≤ 376,832 bits | MPEG related parameters |
| Video (Network QoS) | Bandwidth | ≤ 1.86 Mbps | MPEG encoded video |
| | | 64 Kbps to 2 Mbps | H.261 encoded video |
| | | 1,544 Kbps to 2 Mbps | H.120 |
| | | 140 Mbps | TV, PCM coding |
| | | Over 1 Gbps | HDTV uncompressed quality |
| | | Around 500 Mbps | HDTV lossless compression |
| | | 20 Mbps | HDTV lossy compression |
| | Bit error rate | $\leq 10^{-6}$ | Long-term bit error rate |
| | Packet loss | $\leq 10^{-2}$ | Uncompressed video |
| | | $\leq 10^{-11}$ | Compressed video |
| | End-to-end delay | ≈ 250 ms | |
| Audio/video | Sync skew | +/– 80 ms | Lip synchronization |
| Audio/image | Sync skew | +/– 5 ms | Music with notes |
| Audio/pointer | Sync skew | + 750 ms | (+) audio-ahead pointer |
| | | – 500 ms | (–) pointer-ahead audio |
| Data (Network QoS) | Bandwidth | 0.2 to 10 Mbps | File transfer |
| | End-to-end delay | ≈ 1 sec. | |
| | Packet loss | $10^{-11}$ | |

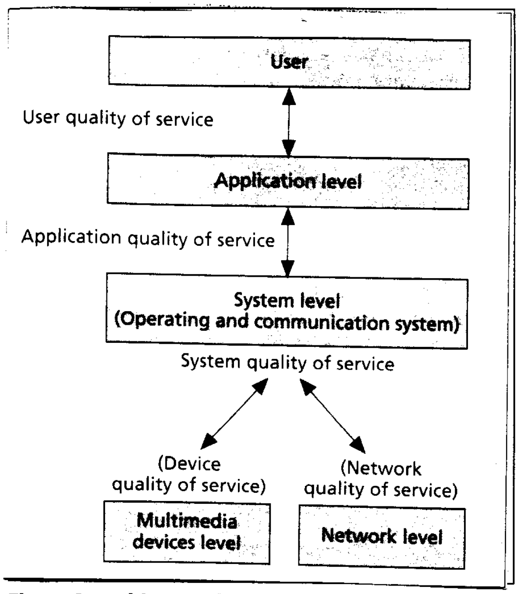


**Figure 3. Architectural view of a networked multimedia system.**

communication and operating system services; and the device level, including both network and multimedia devices. Since all three layers include services, a QoS parameterization is considered in all layers.

*Application QoS parameters*, describing requirements for the application services, are usually specified in terms of media quality and media relations. Media quality includes source/sink characteristics such as media data-unit rate and transmission characteristics such as end-to-end delay. Media relations specify relationships among media, such as media conversion and inter/intrastream synchronization.[3]

*System QoS parameters* describe communication and operating system requirements resulting from the application QoS. These parameters are specified in quantitative and qualitative terms. Quantitative criteria are those that can be evaluated in terms of concrete measures, such as bits per second, number of errors, task processing time, and data unit size. Qualitative criteria specify expected services, such as interstream synchronization, ordered delivery of data, error-recovery mechanism, and scheduling mechanism. Specific parameters can be connected with expected services. For example, interstream synchronization can be defined by an acceptable skew relative to another stream or virtual clock.[2]

*Network QoS parameters* can be specified in terms of network load and network performance. Network load refers to ongoing traffic requirements such as interarrival time. Network performance describes requirements that must be guaranteed, such as latency, bandwidth, and jitter (the variance in delay across many packets).[4] Note that network services depend on a traffic model (arrival of connection requests) and perform according to traffic parameters such as peak data rate or burst length. Hence, calculated traffic parameters are dependent on network QoS parameters and are specified in a traffic contract.

*Device QoS parameters* typically specify timing and throughput demands for media data units.

*Parameter values* determine the types of service. If QoS parameter values are specified with either deterministic or statistical bounds, they require guaranteed services.[4] If QoS parameter values are estimated from past behavior of a service, they require predictive services.[1] If no QoS parameters are specified, best-effort services are employed.

**Resource specification.** NMS services need both *active* and *passive* resources for manipulating data. An active resource, such as the CPU or a network adapter for protocol processing, provides a service. A passive resource, such as the main memory (buffer space) or bandwidth (link throughput), denotes system capabilities required by active resources.

A resource can be used exclusively by one process or shared among processes. For example, a loudspeaker is an exclusive resource, whereas bandwidth is a shared resource.

A resource that exists only once in the system is known as a single resource; otherwise, it's called a multiple resource. In a multiprocessor system, the individual CPU is a multiple resource; in a traditional workstation, the CPU is a single resource.

From the above descriptions, we can conclude that QoS parameters specify requirements for resource capacities allocated to NMS services as well as for the service disciplines managing the resources in NMS. For example, the end-to-end delay QoS parameter determines the behavior of transmission services along the path between media source and sink with respect to packet scheduling (bandwidth allocation), queuing (buffer allocation), and task scheduling (CPU processing time allocation).

### Resource management architecture

Resources are managed by various components of a resource management subsystem in an NMS (Figure 4). The main goal of resource management—guaranteed delivery of multimedia data—implies three main actions:

(1) reserve and allocate resources (end-to-end) during multimedia call establishment so that traffic can flow according to the QoS specification,
(2) adhere to resource allocation during multimedia delivery using proper service disciplines, and
(3) adapt to resource changes during an ongoing multimedia session.

The resource management subsystem includes resource managers at the hosts as well as at the network nodes. Resource management protocols are used to exchange information about resources among the resource managers.

## ESTABLISHING AND CLOSING THE MULTIMEDIA CALL

Any multimedia user expects the application to provide a certain level of quality. Before any multimedia data is transmitted, these user-defined requirements must be communicated to the resource management entities of all involved system components. The QoS parameters are then negotiated and, where specifications differ, translated between layers. Finally, the required resources must be admitted, reserved, and allocated along the path between the sender(s) and receiver(s). These basic steps are performed during multimedia call establishment. The call close-down procedure, from the resource management point of view, concerns resource deallocation.

### QoS negotiation and translation

A general architecture for communicating QoS parameters comprises two services: negotiation and translation of QoS parameters.

**NEGOTIATION.** Two parties are always encountered in the generic QoS negotiation process (Figure 5). The negotiation can be peer-to-peer (for example, application-to-application) or layer-to-layer (for example, application-to-system or human user-to-application). In ISO terminology, the peer-to-peer negotiation is known as the caller-to-callee negotiation and the layer-to-layer negotiation is called the service user-to-service provider negotiation.

The purpose of the negotiation is (1) to establish common QoS parameter values among the service users and providers (that is, negotiate a contract) and (2) to capitalize scarce resource capacities by reserving only the real demand at any point in time.

The most significant variations of negotiation among the service users (caller/callee) and the service provider[5] are bilateral peer-to-peer and layer-to-layer negotiations and triangular negotiation for a contractual value.

*Bilateral peer-to-peer negotiation* takes place between the two service users (caller/callee). The service provider is not allowed to modify the value proposed by the service user (Figure 6).

*Bilateral layer-to-layer negotiation* takes place only between the service user and provider. This negotiation covers two possible communications: (1) between local service users and providers, for example, between application requirements and OS service guarantees, and (2) between host and network, for example, when the sender wants to establish multicast connections for multimedia streams.

*Triangular negotiation for a contractual value* takes place between both service users (caller/callee) and the provider. QoS parameters are specified through a minimal requested value and an upper bound. The caller's goal in this negotiation is to obtain a minimal contractual value.

## Definitions

**Admission control**—a mechanism to decide if a new request to use a shared resource will be admitted, modified, or rejected.

**Best-effort service**—a service that performs its functions as well as possible, uses as much of a shared resource as available, but doesn't provide any guarantees about its performance or resource availability.

**Burst length**—the number of messages that might arrive ahead of their deadline (and before they can be scheduled) at a resource.

**Connection**—a virtual link between a sender and a receiver.

**Data segmentation and reassembly**—a process where a large data unit is broken up into smaller data units (for example, ATM cell-sized pieces). Reassembly is a reverse process to segmentation where small data units are reassembled into the original large data unit.

**End-to-end delay**—the time span between the generation of a data unit at its origin (source, sender) and its presentation at the destination (sink, receiver). The time span starts when an application data unit is created and ends when it is presented (displayed).

**End system**—a system (for example, a workstation or personal computer) that resides at the beginning or the end of a transmission path.

**Guaranteed service**—a service that performs its function according to negotiated QoS requirements and uses shared resources according to an admitted contract.

**Interarrival time**—the time $\Delta$ between two consecutive data units (packets $p_{k+1}$ and $p_k$) when they arrive at a resource; that is, $\Delta = t(p_{k+1}) - t(p_k)$

**Jitter**—a time variation a packet may get through a service. Let $d$ be the minimal service processing time, and $D$ be the maximal service processing time. Then jitter $x$ varies $0 \le x \le (D-d)$.

**Maximal end-to-end delay**—the largest end-to-end delay that may occur.

**Network management system (NMS)**—a management system that monitors and controls availability of network resources.

**Peak data rate**—the maximal data rate of a data stream.

**Quality of service (QoS)**—is a characterization of a service that specifies how well this service should perform.

**Resource**—a system entity required by tasks to perform their duties. Examples of resources are processor, memory, and bandwidth.

**Resource adaptation**—a mechanism to adapt the capabilities of one resource when changes occur to the allocation of another system resource.

**Resource allocation**—a mechanism to allocate resource according to a promised resource reservation.

**Resource monitoring protocol**—a protocol between network and end systems to report on resource availability during the transmission phase.

**Resource reservation/allocation protocol**—a mechanism between sender(s) and receiver(s) to propagate requests and information about reserving/allocating resources.

**Time-constrained media traffic**—a traffic carrying continuous media. The respective traffic parameters, such as end-to-end delay, jitter, intermediate delay, and bandwidth are bounded.

**Time-division multiplexing**—a packet-switching mechanism where time slots are assigned to specific packets and only these time slots can be used to transmit the respective packets.

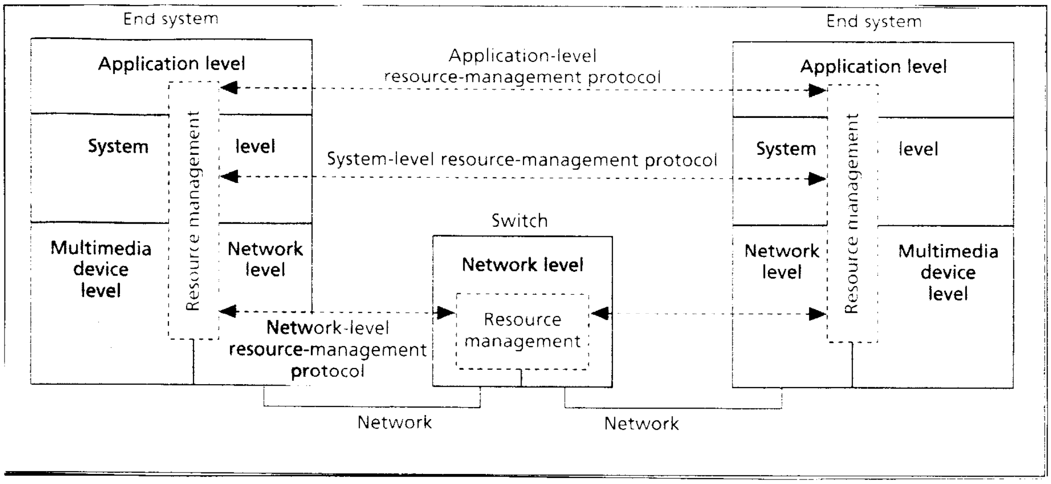


**Figure 4. Typical setup of resource management in networked multimedia systems.**

The service provider can modify the minimal value toward the upper bound value. The callee makes the final decision and reports with a response/confirm primitive to the caller. The final contractual value can also be a maximal QoS parameter or an average value.

**Translation.** Since different NMS layers operate on different objects to provide or use services, they may require different QoS parameters. For example, the mean loss rate of packet networks has no meaning for a video capture device. Likewise, frame quality, the number of pixels in both axes used to initialize frame capture buffers, is of little use to network layer services.

Coordinating services in different NMS layers requires translating QoS parameters between layers. The translation often requires additional knowledge stored with the specific component. For example, the end-to-end delay of audio data depends on the specific application. For a retrieval application such as playback of audio files from a remote server, this value is less stringent than for a dialogue application such as audio workstation telephony between two people. Hence, translation is an additional service for layer-to-layer communication during the call establishment phase.

The NMS architecture (Figure 3) requires translation between layers. The *human interface/application QoS translation* might be provided by a tuning service. This service provides a user with an interface for input of application QoS as well as for output of the negotiated application QoS. The translation can be represented by video and audio clips in the case of audio-

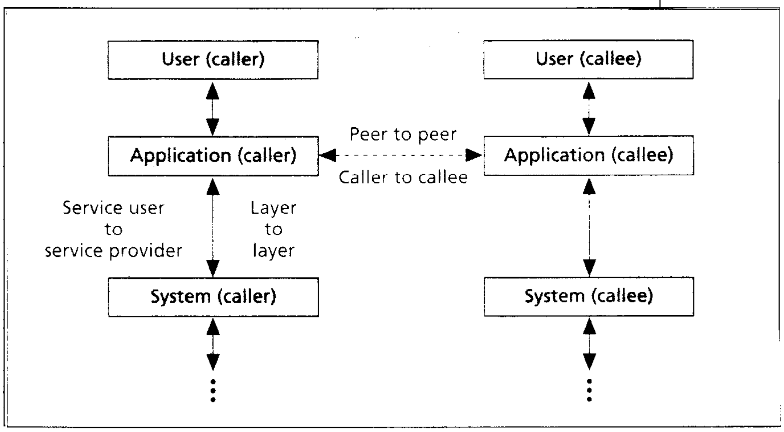


**Figure 5. Resource management instance involved in the QoS negotiation process.**

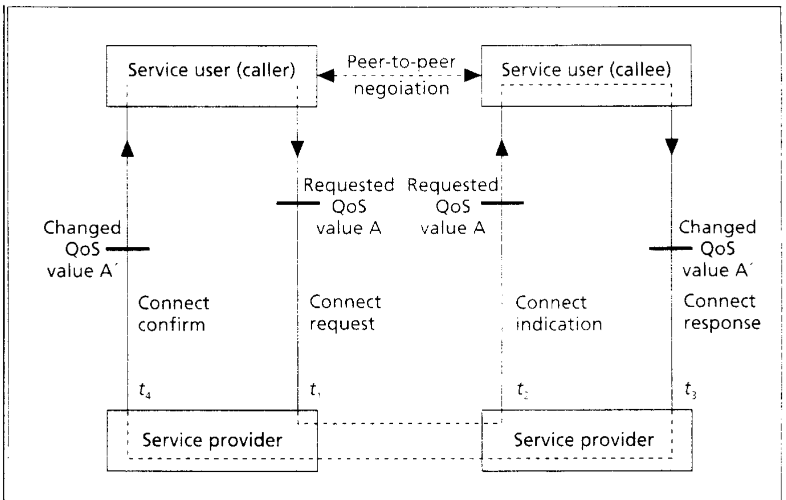


**Figure 6. Example of a specific type of negotiation: bilateral peer-to-peer negotiation.**

visual media, for example, that will run at the negotiated quality corresponding to the video frame resolution supported by the end system and the network.

The *application QoS/system QoS translation* maps the application requirements onto the system QoS parameters. That may lead to such things as translating two "in sync" media streams to a skew parameter defined as a temporal value measured in milliseconds, or translating video frame size to transport packet size with segmentation and reassembly functions.

The *system QoS/network QoS translation* maps the system QoS (for example, transport packet end-to-end delay) into the underlying network QoS parameters (for example, in ATM, end-to-end delay of cells) and vice versa.

The translation service must be bidirectional,[3] which can cause problems if the translation mappings are ambiguous. For example, application parameters for video rate and video frame size must be taken together to specify network throughput value. If the throughput bound must be relaxed, the new value may lower either the image quality or the video frame rate. A bidirectional translation is possible, however, using the components' additional knowledge. This knowledge might result in either reducing the frame size (for example, maintaining the same horizontal and vertical ratio until there are 112 pixels in the horizontal direction) or reducing the frame rate (for example, until it is 1 frame per second) and indicating when no further reduction is possible and the connection must be closed.

### Resource admission

The next step, after every layer inquires or gets its own QoS specification through negotiation and translation, is resource admission. The admission process is based on the QoS specification and uses a service embedded in a resource manager. The admission service checks every node on the path between the source (sender) and sink (receiver). To control resource availability, the admission service uses admission tests. There are at least three types:

- *schedulability test* for sharing resources, for example, a CPU schedulability test and a packet schedulability test at the network admittance point and at each network node for delay, jitter, throughput, and reliability;
- *spatial test* for buffer allocation for delay and reliability guarantees; and
- *link bandwidth test* at the host bus and at the network for throughput guarantees.

The admission tests depend on the implementation of control mechanisms at every NMS layer. Also, any QoS negotiation and resource admission must be closely related to a cost function. For example, assume you run a video-on-demand service. You can save resources by moving a video clip to a server near the client, which can be done more easily with some advance notice. Thus, you may decide to charge a user who reserves in advance less than another user who demands immediate access. If the client is not forced to pay extra, he or she will probably demand the best available QoS, which could reduce the QoS available to other clients or even deprive them of the service. Clearly, with the introduction of appropriate accounting, the QoS negotiation becomes a real negotiation.

### Resource reservation/allocation

Resource reservation/allocation is based on the results of the admission tests. In most systems, reservation/allocation is simplex; that is, resources are reserved in only one direction along the sender/receiver path. Resource reservation/allocation requires a set of reservation/allocation functions embedded in the resource manager and reservation protocols to communicate the information among resource managers.

The managers use reservation/allocation tables and functions to detect and solve conflicts during the reservation/allocation process, which can take a pessimistic or optimistic approach. The *pessimistic approach* avoids resource conflicts by reserving for the worst case, that is, for the longest CPU processing time, highest bandwidth, and so on. This approach can lead to resource underutilization, but it avoids conflicts and guarantees quality.

The *optimistic approach*, which reserves resources according to an average workload, meets QoS parameters as far as possible and uses resources fully. However, since an overload situation may cause failure, a monitor function should be implemented with this approach. The monitor function detects the overload and solves the problem by preempting processes according to their importance.

Both approaches represent points on a continuum because the process requires a resource in a stochastic fashion. Assignment is possible at any value between an average and a peak value. The closer the assignment is to the peak value, the lower the probability that the process will be denied the use of the resource at a certain time. Hence, the assignment represents a trade-off between peak rate multiplexing (pessimistic approach) and statistical multiplexing (optimistic approach).

Resource reservation/allocation protocols do not actually reserve or allocate resources; they only transfer information about requirements and their QoS values. In general, resource reservation/allocation protocols work as follows (see Figure 7):

- The initiator of the call establishment sends QoS specifications in a reservation message (connection request).
- At each router, switch, or other entity along the path, the reservation protocol passes a new resource reservation request to the respective resource manager.
- After the admission decision, the resource manager reserves the resources and updates the particular service information for QoS provision.
- At the end of the path, the last entity sends an allocation message (connection confirmation) back to the initiator with an accept/modify/reject answer and final QoS values.
- The allocation message travels the path back to the initiator. Each resource manager in turn allocates, relaxes, or releases reserved resources according to the message's instructions to accept, modify, or reject the final QoS parameters.

Reservation protocols are further characterized by *direction* and *style*. With respect to direction, they can be sender-initiated or receiver-initiated. In a sender-initiated reservation, the sender describes its sending resource requirements in a QoS specification and sends it to the receivers in a reservation message; resources are reserved along the path from the sender to the receiver. The receiver returns an allocation message, and resources are allocated along the path from the receiver to the sender. In receiver-initiated reservation, the receiver describes its receiving resource requirements and sends it to the sender; resources are reserved from the receiver to the sender and allocated from the sender to the receiver. It is assumed that the sender has already sent a path control message, providing information about outgoing data.[6]

The reservation style refines the reservation protocol with respect to communication scenarios and the reservation request's timing. Possible communication scenarios are single sender/single receiver, single sender/multiple receivers, or multiple senders/multiple receivers. The style for a sender-initiated reservation may be for the sender to create a single reservation along the link to the receiver or a multicast reservation to several targets.

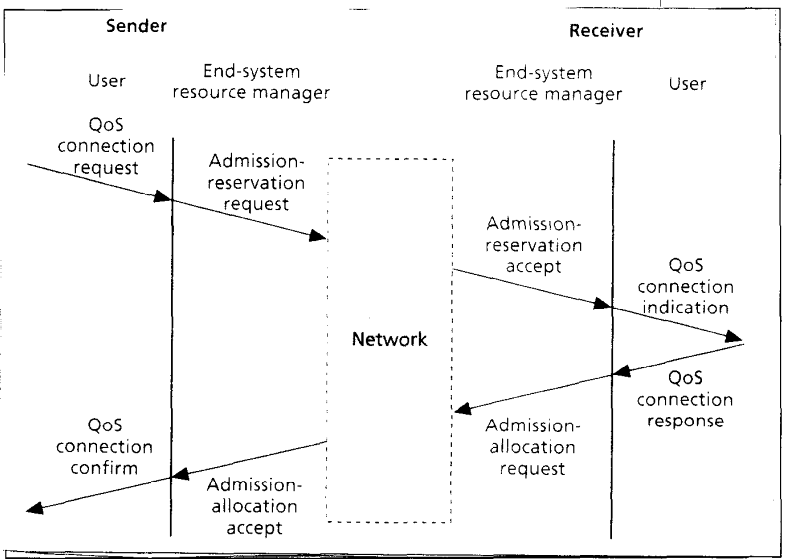

**Figure 7. Resource reservation/allocation protocol with "accept" response.**

There are three types of receiver-initiated reservation styles. The first is the *wildcard-filter* style, in which a receiver creates a single reservation, or resource pipe, along each link, shared among all senders for the given session. In the second style, *fixed filter*, each receiver selects the particular sender whose data packets it wants to receive. In the *dynamic filter* style, each receiver creates *N* distinct reservations to carry multimedia data from up to *N* different senders.[6]

The reservation style can also be divided according to resource allocation time as an immediate or advance reservation. The advance reservation service, essential in multiparty multimedia applications, has two possible approaches: centralized, where an advance reservation server exists, and distributed, where each node on the channel's path "remembers" the reservations.[7]

### Resource deallocation

After the transmission, resources are deallocated. The CPU, network bandwidth, and buffer space are freed, and the media flow connections are closed down. The close-down process must be accomplished without disrupting other network flows and implies resource availability updating by the resource manager. Resource deallocation can be initiated in three ways:

- *Sender requests closing of the multimedia call.* This implies that the resources for all connections corresponding to the multimedia call along the path between sender and receiver(s) have to be deallocated and the resource availability has to be updated at every node.
- *Receiver requests closing of the multimedia call.* This request is sent to the sender and resources are deallocated as the request traverses the path.
- *Resource management system initiates closure.* It can happen that, although transmission has terminated, a proper resource deallocation never took place. Hence, monitoring components working with timers should oversee connections and check whether participating parties are still active. In such a case, the deallocation is initiated by the resource management system itself.

## MANAGING RESOURCES DURING TRANSMISSION

Let's assume successful negotiation of QoS and resource allocation requirements—that is, the contract has been negotiated and signed. Resource management must now sustain resource accessibility during the multimedia transmission—that is, fulfill the contract. The job of satisfying time, space, device, frequency, and reliability requirements belongs to various management components, such as process management, buffer management, and rate and error control components.

### Process management

Multimedia transmission involves many separate tasks for data movement, control, synchronization, and so on. Since all these tasks share the same resource processor, its use must be scheduled. This is done by the process manager, which is part of the resource manager. The process manager's scheduler maps tasks onto the resource processor according to a specified scheduling policy so that all tasks meet their requirements.

The choice of a scheduling policy depends on the characteristics of the data to be processed. Continuous media data require real-time processing in exactly predetermined, usually periodic, intervals. As an example, let's consider periodic tasks without precedence constraints where task processing is independent and must be guaranteed for the entire runtime. Apart from the demand of meeting guaranteed deadlines, a non-real-time task should not suffer from starvation because real-time tasks are executed. Multimedia applications rely on discrete media as much as on continuous media. Therefore, not all resources should be occupied by real-time tasks and their management.

Several scheduling algorithms are suitable for multimedia tasks.[2] The two most relevant are the *earliest deadline first* (EDF) and the *rate-monotonic* algorithms. In EDF, the scheduler selects among tasks with the earliest deadline that are (1) ready and (2) not fully processed. EDF produces a valid schedule whenever one exists. A priority-driven system scheduler assigns each task a priority according to its deadline. The highest priority goes to the task with the earliest deadline, the lowest to the one with the latest, and so on. Since priorities are reevaluated with every new task, EDF scheduling is apt to require frequent rearrangement.

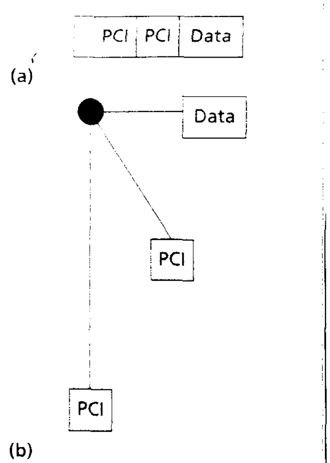


**Figure 8. Buffer management techniques: (a) offset management; (b) scatter/gather system.**

The rate-monotonic algorithm computes a task priority schedule at the beginning of processing. Priorities are assigned to tasks once during call establishment according to their request rate, and no further computation during scheduling is required. There are five prerequisites to application of the rate-monotonic algorithm:

- Requests for all tasks with deadlines must be periodic.
- The processing of one task must be complete before the next task of the same data stream is ready for execution.
- Task requests must not depend on the initiation or completion of a request for any other task.
- The maximum processing time required for uninterrupted execution of each task request must be constant.
- Only periodic tasks in the system can have deadlines.

In general, the cost of scheduling every task should be minimized, especially the overhead of context switching in the operating system. If more than one stream is processed concurrently, more context switches are likely with a scheduler using the rate-monotonic algorithm, so EDF is more suitable for scheduling multimedia streams.

## Buffer management

The limited memory bandwidth of host systems causes a severe problem for multimedia applications, which require efficient movement of large amounts of data. This goal cannot be accomplished by traditional data copying. Likewise, the efficient paging and swapping techniques current operating systems use to enlarge memory are unsuitable, since multimedia data must be pinned (locked) into the real main memory to satisfy timing requirements.

Multimedia applications require other buffer management techniques, such as offset management or a scatter/gather system (see Figure 8). Below, we describe these mechanisms in the context of the communication protocol stack, but they can be applied to other layers.

In an *offset management* approach, a buffer is placed in one memory segment. The buffer must be large enough to contain all data to be transmitted, as well as the protocol control information (PCI). For transmission, an application places its data into the buffer and leaves enough space for the future PCIs of the protocol stack. As the buffer is passed through the communication system, each protocol entity updates the offset in the buffer, which means it writes its own PCI in the buffer. The offset technique is easy to implement with little processing overhead for each buffer and prevents data copying. The problem is that the application has to allocate a maximum-sized buffer to ensure enough space for all protocol information. This implies that the application knows the sizes of all PCIs. It also means that this technique can only be used between layers where the data unit is not segmented/reassembled.

In a *scatter/gather system*, each request for buffer space is satisfied by linking in a new memory segment. A control structure keeps track of all memory segments for a buffer. The scatter/gather system prevents data copying, and also allocates memory space only as it is needed, not causing memory over-allocation. This technique efficiently implements protocol requirements such as dynamic expansion of buffers, segmentation and reassembly, and concatenation and separation of data units. However, it incurs additional overhead for maintaining control structures.

## Rate control

Because multimedia data transmits and uses network resources at certain negotiated rates, NMS communication protocols must include rate-based flow control. This kind of flow control employs a rate-based service discipline that provides users with a minimum service rate, independent of other users' traffic characteristics. Such a discipline, performed by a service provider at a switch, schedules and manages bandwidth, service time (priority), and buffer space. Combined with a proper admission policy, this scheduling discipline provides throughput, delay, delay jitter, and loss rate guarantees.

Of the rate-based scheduling disciplines, the most relevant for multimedia are Virtual Clock, Delay Earliest Due Date (Delay EDD), Jitter Earliest Due Date (Jitter EDD), and the weighted version of the Fair Queuing algorithm (WFQ).

- *Virtual Clock* emulates time division multiplexing. It allocates each packet a virtual transmission time; this is the time at which the packet would have been transmitted if the service provider were actually doing time division multiplexing.[8]
- *Delay EDD*[4] is an extension of earliest-deadline-first scheduling. The service provider works according to a service contract negotiated with each source. The contract states that if a source obeys a peak and average sending rate, the service will provide bounded delay. The service provider sets a packet's deadline to the time it should be sent according to the contract. This actually is the expected arrival time added to the delay bound at the switch. By reserving bandwidth at the peak

rate, Delay EDD can assure each connection a guaranteed delay bound.

- *Jitter EDD*[4] extends Delay EDD to provide jitter bounds. After a packet has been served by each switch, it is stamped with the difference between its deadline and actual finishing time. A regulator at the entrance of the next switch holds the packet for this period before making it eligible for scheduling. This provides minimum and maximum delay guarantees.
- *WFQ*[1] gives each packet a time stamp as it arrives and transmits packets in increasing time-stamp order.

To bound delays in the rate-based disciplines, traffic shaping schemes can be applied at the sending host. In Leaky Bucket and its variations (for example, Token Bucket),[1] the sending host places the data into the bucket, and data drains out the bottom of the bucket as it is sent on the network at a certain rate. The rate is enforced by a regulator at the bottom of the bucket. The bucket's size limits the data that can build up waiting for the network.

Rate-based disciplines are divided according to whether they adopt a *work-conserving* or *nonwork-conserving* policy. The work-conserving discipline, which includes Delay EDD and Virtual Clock, serves packets at the higher rate so long as that does not affect the performance guarantees of other connections. This policy means a service provider is never idle when there is a packet to be sent. The nonwork-conserving discipline, which includes Jitter EDD, does not serve packets at a higher rate under any circumstance. This also means that each packet is assigned, explicitly or implicitly, an eligibility time. If no packets are eligible, none will be transmitted—even when the service provider is idle.

Rate-based disciplines allocate resources per service user. Hence, the service users need to specify traffic and guarantees, using the network load and performance specifications in network QoS parameters. All rate-based services provide throughput guarantees. Delay guarantees are provided by Delay-EDD, all nonwork-conserving services, and WFQ. Jitter guarantees are provided by nonwork-conserving disciplines.

### End-to-end error control

NMSs demand a substantial degree of reliability, so their components need end-to-end error detection and error correction mechanisms.

**Error detection.** Error detection mechanisms should be embedded in the application. For example, some errors in the AC parameters of MPEG-2 compressed B blocks may not matter in predictive encoded video data (they appear for only a fraction of a second and are hardly visible to the human eye). But if the frame boundaries are destroyed, the error cannot be recovered. This means that structural information within a data stream needs to be protected. This also implies that existing error detection mechanisms, such as checksumming and data unit sequencing, should be extended to convey further information. These existing mechanisms allow detection of data corruption, loss, duplication, and misordering at the lower levels (for example, packets in the network layer), but on the application level, where the decision should be made, error detection is omitted.

Error detection mechanisms for temporal misbehavior must be embedded in the system and network layers, too. Such detection mechanisms may open resources on the sender-receiver path to other connections (if work-conserving scheduling disciplines are used) and may indicate congestion points along the path. To identify late data, it is necessary to determine the lifetime of data units and compare actual arrival time with latest-expected arrival time. The latest-expected arrival time can be derived from the traffic model (throughput and rate) associated with a connection. Therefore, only the first data unit must carry a time stamp. This is not an ideal solution because it forces error detection to start with the first data unit and no interruption of the service is possible. With a time stamp in every data unit, error detection can start at any point during the media transmission. This mechanism requires a synchronized system clock at the sender and receiver to accurately determine end-to-end delay. A possible protocol for this kind of synchronization is Mill's Network Time Protocol.

**Error correction.** The traditional error correction strategies are not suitable for multimedia communications. *Preventive* error correction schemes,[9] such as *forward error correction* (FEC) and *priority channel coding*, are more appropriate.

In FEC, the sender adds additional (redundant) information so that the receiver can locate and correct bits or bit sequences. To use FEC, the NMS needs (1) the error probability of the connection between the sender and receiver and (2) the reliability required by the application. FEC results in a low end-to-end delay, and it does not require exclusive buffering of data before playout or a control connection from the receiver to the sender. The disadvantage of FEC is that it works only for detection and correction within a packet, not for complete packet loss. FEC cannot guarantee recovery of corrupted or lost packets. Also, its redundancy significantly increases throughput demands.

Priority channel coding refers to a class of approaches that separate a medium into multiple data streams with different priorities. During periods of congestion, the network discards low-priority packets carrying information that is less important for reconstructing the original media stream. Channel coding requires network control of packet loss during congestion through a priority mechanism. Further, the use of different streams for different priorities requires synchronization at a per-packet granularity to reconstruct the medium stream.

### Resource monitoring

Resources are monitored at end points as well as in networks during multimedia transmission. The monitoring function continuously observes whether processed QoS parameters are exceeding their negotiated values. This information on available and allocated capacity is stored in a management information base as a QoS specification.

Monitoring must be flexible so that the overhead it adds to transmission does not cause violation of QoS guarantees. This means keeping most monitoring variables optional and retaining the ability to turn monitoring on and off. Resource monitoring can operate in two possible modes. End-user mode requests a status report about the resources; network mode regularly reports the resource status on different nodes along the path.

Obviously, the design and implementation of such a monitoring function is a nontrivial task, and a clearly agreed upon notion of the QoS is a prerequisite.

**Resource adaptation**

Continuous media communication should support a framework for dynamically changing the allocated resource capacity. Achieving this goal requires (1) notification of a change and renegotiation of QoS parameters and (2) adaptation of resources to accommodate changes in the end points, the network, or both.

**NOTIFICATION AND QOS RENEGOTIATION.** If the monitoring function signals a change of QoS parameters during transmission, renegotiation of QoS parameters must begin. Hence, renegotiation is part of QoS negotiation when a call is already set up. The renegotiation request can come from the user, from the host system (due to workstation overload), or from the network (due to overload and congestion). The request is sent to the local resource manager, which propagates the request using resource management protocols (Figure 9).

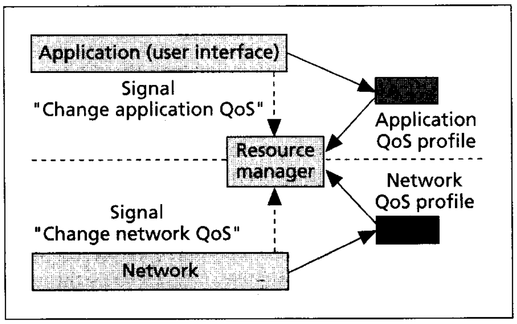


**Figure 9. Signaling of QoS change coming from application or from network.**

A user request for a QoS change can be initiated from a sender or a receiver. If the user-sender requires a change of QoS, this may imply adaptation of multimedia sources, sinks, host resources, and network resources. The changed QoS is passed from the sender to the receiver, and each resource manager checks resource availability for the changed QoS value. Changes are implemented by the admission/reservation/allocation mechanisms and reservation protocol described earlier. If the user-receiver requires a QoS change for the receiving media, the resource manager checks the local resource, reserves it, and notifies the sender. The admission procedure is the same as for a user-sender request. At the end, the receiver has to be notified to allocate local resources.

A host system request for a QoS change may come from the operating system when several users are admitted and some users violate their admitted application requirements (that is, system performance degradation is observed). A notification about the degradation of QoS performance and a renegotiation request are issued. The result of the renegotiation is an adaptation of the host system to the overload, either accommodating new quality requirements or negotiating lower contractual values. If host QoS changes degrade performance, the host resource manager may invoke the network resource management to lower the QoS parameters in the network between the sender and the receiver.

In an optimistic resource allocation approach, network overload at some nodes can cause a network request for a QoS change. This request comes as a notification, reporting the need for a resource allocation change, from the network resource management to the host. Below, we describe two approaches to resource adaptation caused by network overload. (Several adaptation schemes already exist, since this is an area of active research.) These adaptation mechanisms implicitly offer partial solutions for renegotiation requests originating from the user or the host system.

**NETWORK ADAPTATION.** The first approach—a proper load-balancing policy—provides a solution where the network can adapt to the overload. Such a network adaptation policy may combine the following mechanisms: routing, resource monitoring (detecting load changes), load balancing control (deciding to reroute a connection), and dynamic rerouting (changing the route). The routing mechanism implements the routing algorithm, which selects a route in adherence to certain routing constraints. The resource monitoring mechanism monitors the appropriate network performance and reports it to the load-balancing control. The load-balancing control mechanism receives information from the resource-monitoring mechanism and determines whether load balancing can be attempted. If load balancing can be attempted, the routing mechanism provides an alternate route. The transition from the primary route to the alternate route uses the dynamic rerouting mechanism.[7]

**SOURCE ADAPTATION.** An alternative approach is to adapt the source rate according to the currently available network resources. This approach requires feedback from the network to the source, which causes graceful degradation in the media quality during periods of congestion. The feedback information is sent by the monitoring function, which at each switch monitors the buffer occupancy and the service rate per connection.

There are two ways to feed messages back to the source. In the first method, the per-connection state information is periodically appended to a data packet for the corresponding connection. At the destination, this information is extracted and sent back to the source. A switch updates the information fields in a packet only if the local service rate is lower than that reported by a previous switch along the path. In the second method, the feedback message is sent in a separate control packet, which is sent back along the connection path toward the source.[10]

RESOURCE MANAGEMENT MUST BE EMBEDDED in the multimedia operating system and communication architecture. This means that proper services and protocols in the end points and the underlying network architectures have to be provided.

Many NMS functions, mechanisms, and protocols are still research issues. However, examples of architectural choices, where QoS and resource management have been designed and implemented, do exist:

- The Open Systems Interconnection (OSI) architecture provides QoS in the network layer and some enhancements in the transport layer. The OSI 95 project considers integrated QoS specification and negotiation in the transport protocols.[5]
- Lancaster's QoS-Architecture offers a framework to specify and implement the required performance properties of multimedia applications over high-performance ATM-based networks.[11]
- The Heidelberg Transport System, based on the ST-II network protocol, provides continuous-media exchange with QoS guarantees, resource management, and real-time mechanisms.[2]
- UC Berkeley's Tenet Protocol Suite provides network QoS negotiation, reservation, and resource administration through the Real-Time Channel Administration Protocol.[7]
- An Internet protocol stack with the Resource Reservation Protocol will provide resource reservation.[6]
- The University of Pennsylvania's communication architecture—Omega—provides end-point resource guarantees using the distributed QoS Broker entity. QoS Broker relies on the underlying network resource management.[3]
- Application protocols for audio VAT and for video NV on top of Internet protocol suite RTP/UDP/IP[2] are now widely used for networked multimedia applications.

Resource management, based on QoS requirements, has become an important part of multimedia communication systems across all system components because of requests for resource guarantees. These requests, in turn, come from the increased variety of applications and new media—for example, tactile information—now being transmitted over high-speed networks. ▮

## Acknowledgments

Klara Nahrstedt's work was supported by the National Science Foundation, by the Advanced Research Projects Agency under Cooperative Agreement NCR-8919038 with the Corporation for National Research Initiatives, by Bell Communications Research under Project DAWN, by an IBM Faculty Development Award to Jonathan M. Smith, and by the Hewlett-Packard Company. The authors gratefully acknowledge many valuable ideas and suggestions from Arturo A. Rodriguez, Lawrence Rowe, Jonathan M. Smith, Lars Wolf, Jean McManus, and the anonymous reviewers. Thank you!

## References

1. D.D. Clark, S. Shenker, and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanics," *Proc. SIGComm 92*, ACM, New York, 1992, pp. 14-26.
2. R. Steinmetz and K. Nahrstedt, *Multimedia: Computing, Communications and Applications*, Prentice-Hall, Englewood Cliffs, N.J., 1995.
3. K. Nahrstedt and J.M. Smith, "The QoS Broker," *IEEE MultiMedia*, Vol. 2, No. 1, Spring 1995, pp. 53-67.
4. D. Ferrari and D.C. Verma, "A Scheme for Real-Time Channel Establishment in Wide Area Networks," *IEEE JSAC*, Vol. 8, No. 3, April 1990, pp. 368-379.
5. A. Danthine et al., "QoS Enhancements and the New Transport Services," *Local Networks Interconnection*, R.O. Onvural and A.A. Nilsson, eds., Plenum Press, New York, 1993, pp. 1-22.
6. L. Zhang et al., "RSVP: A New Resource Reservation Protocol," *IEEE Network*, Vol. 7, No. 5, Sept. 1993, pp. 8-19.
7. C. Parris, H. Zhang, and D. Ferrari, "Dynamic Management of Guaranteed Performance Connections," *Proc. Infocom 94*, Vol. 2, IEEE CS Press, Los Alamitos, Calif., 1994, pp. 698-707.
8. H. Zhang and S. Keshav, "Comparison of Rate-Based Service Disciplines," *Proc. SIGComm 91*, ACM, New York, 1991, pp. 113-122.
9. E.W. Biersack, "Performance Evaluation of Forward Error Correction in an ATM Environment," *IEEE JSAC*, Vol. 11, No. 4, May 1993, pp. 631-640.
10. H. Kanakia, P.P. Mishra, and A. Reibman, "An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport," *Proc. SIGComm 93*, ACM, New York, 1993, pp. 20-31.
11. A. Campbell, G. Coulson, and D. Hutchison, "A Quality of Service Architecture," *Computer Communication Review*, Vol. 1, No. 2, Apr. 1994, pp. 6-27.

***Klara Nahrstedt*** *is a research assistant in the Distributed Systems Laboratory and a PhD candidate in the Computer Science Department at the University of Pennsylvania. Her research is directed toward multimedia services, protocols, and end-point architectures, as well as quality-of-service provision and resource management in multimedia distributed systems. Previously, she was a research scientist at the Institute for Informatik in Berlin. She received her BA in mathematics and her MSc in numerical analysis from Humbold University, Berlin. Nahrstedt is a member of the IEEE Computer Society.*

***Ralf Steinmetz*** *manages IBM's Multimedia Center in Heidelberg, where in 1988 he initiated the first activities on networked multimedia. At the same time, he has supervised theses and lectured at the University of Mannheim, Frankfurt. He studied electrical engineering with a communications focus at the University of Salford, England, and at the Technical University of Darmstadt, Germany, where he received his MSc and PhD degrees. He is a member of ACM, Gesellschaft fur Informatic, Informations Techmische Gesellschaft, and a senior member of the IEEE.*

*Readers may contact Nahrstedt at the Distributed System Laboratory, University of Pennsylvania, 200 South 33rd St., Philadelphia, PA 19104. Her e-mail address is klara@ aurora.cis.upenn.edu.*