[StWi94b] *Ralf Steinmetz, Hartmut Wittig;* **Challenges in Multimedia System Development;** Technical Report 43.9406, IBM European Networking Center Heidelberg, Germany, 1994.

Challenges in Multimedia System Development

Ralf Steinmetz, Hartmut Wittig

Technical Report 43.9406

IBM European Networking Center
Vangerowstraße 18 • 69115 Heidelberg • Germany

Phone: +49-6221-59-3000 • Fax: +49-6221-59-3400

{rst, wittig} @ dhdibmip.bitnet

Abstract:
This paper starts with presenting a survey on multimedia applications, which are ordered according to the respective function in the media processing chain: from the media capture process over media preparation, media integration to media interaction and media communication. The most specific stringent multimedia requirements for these applications are the handling and the storage of the huge data amount and throughput, taking into account the real-time demands and the restricted end-to-end delivery. I.e., data processing needs, specified as quality of service parameters, must obey rules which allow to provide time guarantees. This is done by an appropriate resource management which is comprised of the establishment as well as of the enforcement of these guarantees. Therefore, the system architecture of all involved resources comprises of real-time and non real-time domains. Following the application's survey, this paper outlines the required system capabilities for local as well as distributed systems.

Keywords:
Multimedia Applications, Interactive TV, Quality of Service, Resource Management, Media Scaling

1 Introduction

'Multimedia' comprises audio and video in computing and communication. A multimedia application makes use of continuous media as well as the traditional discrete media such as text and images. A multimedia system must be capable of coping with the demands arising from the integration of these media and to allow for such applications to run in the digital computer domain [Stei93]. At the IBM European Networking Center we have developed over the last 5 years multimedia operating system extensions, multimedia communication systems, multimedia programming toolkits, and multimedia applications. With respect to the challenges of these developments we encounter two questions, which we have been asked very frequently:

Which are the multimedia applications?
Is is certainly difficult to clearly distinguish between multimedia and non-multimedia applications. This can be approached by making use of a definition on what is said to be multimedia and what is not, taken from [Stei93] and [ICMCS94]: "Multimedia is characterized by the computer-controlled integrated generation, manipulation, presentation, storage and communication of independent digital information. This information is most often coded in a continuous time-dependent media (e.g., audio, video) as well as discrete time-independent media (e.g., text, graphics)." However, in the following we do not want to focus on this precise definition. We want to cover the whole set of applications which make use of digital audio and video. Therefore we developed a media flow and processing chain in which all applications are embedded: From the media capture process over media preparation, media integration to media interaction and media communication.
In contrast to other surveys we decided not to group applications according to the market segment or branches like education, entertainment, medicine or many others. The reason is twofolded: (1) the same applications are often used with different contents in the various market segments, and (2) many marketing oriented application surveys exist, which do not help to identify the key issues multimedia systems must provide.

What are the key issues of multimedia systems?
It is easy to understand that video requires higher throughput and storage capacity than other media; this would just makes the systems larger and faster. However, analysing the multimedia demands derived from the applications, we encounter that continuous media data must be processed with real-time constraints in the same way as digital telephone data is processed at digital switches today. Therefore for guaranteed quality all the required resources must be allocated at the call set-up phase. This resources provide the demanded guarantees, which are specified in terms of quality of service parameters. At the actual data transmission phase the guarantees must be enforced by appropriated data handling including real-time scheduling. It must be observed that the guarantees apply to the whole path from the source(s) to the destination(s). Even in a local environment we encounter the same paradigm of call establishment before the actual data move, copy or transmission takes place. However, the path of media transmission is often comprised of traditional computer networks, which do not allow for guaranteed reservations; hence best-effort quality of service is applied which is based on appropriate media scaling techniques.

In this paper we answer both questions; and we want to establish a clear understanding of what are the multimedia application and why all facts around resource management are the key issues. We subsume tools such as editors to be considered applications as well, this allows us to broaden the scope without running into discussions about the bordering line between tools and applications.
Certainly there are more issues which might be addressed including workstation architectures,

optical storages (like CD-ROM/XA), compression (like JPEG, MPEG and H.261), interactive document architecture and presentation standards (like MHEG, HyTime, ScriptX), content based database retrieval, object-oriented toolkits for programming, and synchronization. We decided to restrict our view to the roots, which must be available and all other issues rely on, because these allow us to provide the reader with a deeper understanding and not just with an annotated set of catchwords.
In Section 2 we survey multimedia applications and in Section 3 we summarize the multimedia application requirements. Section 4 introduces the notion of quality of service, resources and resource management. Section 5 presents the resource management applied to the CPU and the subsequent section show how to apply it to networks. In Section 7 we finally outline the resulting system architectures.

2 Multimedia Applications

It is very difficult to extract the specific multimedia requirements out of the set of applications and tools. However, in the following text we will (1) survey and categorize the whole set of known multimedia applications and (2) to derive from each category the key requirements. Hence, we ordered the applications according to the respective function in the media processing chain starting with the capture items and ending with the interaction and communication process.

Media Capture
Media Capture covers all applications which help to introduce media into the computer or to transform information coded in one media to another. "Media Capture is the process of transformation of analog environmental information to the corresponding digital representation in the computer." All types of frame grabber, audio analog to digital conversion and subsequent coding tools, image scanning, MIDI code generation tools, OCR (optical character recognition) software, and speech recognition applications are typical examples. Also all control applications of compression and decompression are subsumed into this category of applications.
Continuous media data gathered from a live source needs to be processed in real time, otherwise information is lost. Continuous media data captured from the analog storages, such as from an video player, can also be processed off-line without real-time requirements. The same applies to discrete media capture applications. Hence, the derived requirements depend more on the actual use of the media capture application than on the respective media itself. In applications where the result of media capturing is presented immediately, real-time constraints apply. Here we also need to keep the end-to-end delay within certain boundaries.

Media Preparation
Media Preparation denotes all applications which transform digitally coded media. By 'transformation' we mean to apply some perceivable change to the respective data. Typical media preparation applications for images are CorelDraw (for the purpose of creating and changing drawings) and Adobe PhotoShop (for modifying and assembling images). An audio mixer provides all features to be applied to audio streams and a video postprocessing application allows to edit and manipulate video streams. Here we can also make use of tools for automatic parsing of media streams for later indexing and easier retrieval from databases (like for videos as shown in, e.g., [ZKSm93]). Often we encounter applications derived from the television studio environment, which allow to manipulate audio and video together. In order to keep the scope

of our categories well defined, such an editing application belongs to the next category. The requirements for media preparation applications are the same as those for media capture.

Media Integration

Media Integration comprises of all applications which assemble data encoded in different media. It is assumed that the media itself (the single media) is processed by appropriate media preparation applications.

A audio-video editor is an example of a media integration application, which very often comprises of media preparation as well as of media capture functions. Any video editor applications can either be linear or non-linear. A linear editor allows us to edit audio video data in the sequential mode only, i.e., portions of the data are cut and later-on assembled together. The technique is most often based on the control of externally attached video recorder machines, where data is copied from one machine to another. On the other hand non-linear editing allows us to assemble data in a random access mode, like in the cut and paste paradigm of window systems. This is certainly the better approach but, it mostly requires extensive digital processing of the respective data in the computer.

In the area of discrete media we encounter a large amount of word processing and desk-top publishing tools which allow us to assemble and edit data coded in several discrete media.

The integration of both domains is done by several hypermedia systems and by presentation generation tools which allow us to include user interaction. The IBM Ultimedia Builder and the Asymmetrix ToolBook are products which provide such capabilities.

The interaction with media data demands real time if it operates in a WYSIWYG mode. And even here the response time (also known as end-to-end delay) has to be low, but it is not as crucial as in the case of telephone-like dialogue applications.

Media Interaction

Media Interaction is the focal point of most consumer application: Whereas today we have less interaction and more media distribution (a passive consumer behaviour) with multimedia computing we allow for more interaction. However, the passive mode will also survive; this transition to more interactivity will happen smoothly.

Presentations which were assembled by applying media integration can be read, traversed, played back or experienced using these media interaction applications. A typical example are kiosk systems, a computer based information system. They are located in public areas, with which, through an intuitive user interface information can be recalled or transactions can be triggered [HoHe94].

The core of kiosk systems are the stored content data and the structure of how this data is related; both are often stored in a database. Most of todays systems still do not distinguish between the different categories of data: (1) the structure of how the information items are related to each other, (2) the generic content data which is not allowed to be modified by a kiosk provider, and (3) the specific content which may be modified by the kiosk provider. As an example, let us assume we have an electronic banking kiosk which includes offers of a real estate agency. The local kiosk provider will not change the application and the respective background images but, he will often add or remove the actual offers of houses to be sold. This structure follows the model of different groups of people interacting with the kiosk: The user just plays back data and makes use of this system. The owner at the local agency or bank makes changes of content but does not need to have any knowledge on the design issues. The editorial provides the whole layout and the respective style guides. A kiosk production team, which closely interacts with the editorial, builds-up the structure of the respective kiosk application.

Inside the kiosk application the stored data is interpreted by an engine which closely interacts with the user interface agent. Here again we find that todays system have both function integrated into one block. A clear separation allows for (1) portability (as the user interface agent covers most of the system dependencies), (2) easier adaptation to different kiosk data formats (formats like MHEG are not yet widely used), and (3) ease of synchronization (in order to have, e.g., two remotely located systems in the same state it is just required to have one engine driving two user interface agents).

Media Interaction will also enter the consumer market and step by step it will be part of television and radio sets. This is often addressed in the context of **Interactive Television.** Today we have video distribution and the user can just select one out of about 30 channels. In the future the user may retrieve news at the time he wants and with the focus he prefers; he may also start a movie at any time, and he will be allowed to traverse the movie in a very interactive mode by, for example, being asked to participate to some extent and make choices at certain scenes. The way to full digital Interactive Television Systems consists of a sets of steps:
First, with digital video program distribution, we will just get more channels than today, perhaps 500 instead of 30 or 5 channels. Therefore, the user needs some kind of guidance as he will not be able to switch from channel to channel with the purpose of selecting a program. Also the traditional weekly TV program guide on paper will not help as it does today. An offline transmitted multimedia TV program, which is used by selection assistance applications, will guide the user to choose his favourite program by filtering the information flood with a personal user profile. Such an electronic TV program can also be transmitted on-line together with the TV signal in a multiplexed mode. This can be easily implemented if the whole transmission is done in the digital domain over packet-switched networks. It could also be included in the analog TV signal at the expense of an additional effort for user profile processing.
The user gets most of the channels for free and can get access to some others in a rudimentary pay TV mode. This is the **pay per channel** mode where we pay on a monthly basis regardless of the watched programs. In the next step a channel back connects the user to the sender. This communication line can be used to initiate some transactions for ordering goods. In such applications, mostly **home-shopping** type, the TV shows some articles to be bought together with the respective phone number to be dialled for ordering purposes. Also other interactive application which require very few interaction can run in this mode.
Video program on demand is a combination of these two steps mentioned above. The daily program may be retrieved and displayed, while the same computer which interprets and decodes the TV program can automatically send requests back to the sender via dedicated channels (e.g., modem dial).
In the **slotted video on demand** mode the user can select one movie among a few and drop into the program within the next 10 minutes. In this mode a part of the huge set of channels is used to play the same movie starting at 10 minute intervals. A rudimentary random access with a coarse granularity is possible; i.e., the recipient just switches from one channel to another. The higher the resolution is, the more channels are required.
The **video on demand** set-up allows a user to retrieve audio, video, and other media at the time the user wants it [CoMa94]. Here the potential offer of contents is substantially higher than in the slotted video on demand mode. Hence, it will not be sufficient to transmit this information multiplexed with the video signal. A second data connection operating in the duplex mode is required. Depending on the amount of content, narrow-band ISDN can be used. Over this connection the user gets the actual offerings; based on this he will make his choice. Subsequently the video data is transferred to the user and displayed.
Depending on the price of this 'pay per view' technique, it may replace all video cassette rental

stores and perhaps also the retailers. In a realistic metropolitan set-up it will be very difficult to allow each user to select any movie, start this movie immediately and control it like a VCR today. The reason is the large amount of data to be stored and transferred if we assume there will be 100.000 users attached to this service who might want to watch concurrently. By hierarchies of video servers and a tariff which enforces the user to book movie at least some minutes in advance this problem can be alleviated. Then it will be possible to transfer video data off-line to a server (which in this case operates as video cache) next to the user. From the feasibility point of view such a tariff could look like (1) to charge DM 15 for the immediate play back and full VCR control capabilities, (2) to charge DM 5 for an advanced booking of at least 15 minutes and full VCR capabilities, (3) to cost DM 1 for an advanced reservation of 24 hours or more and no VCR capabilities, (4) to cost nothing or very few if the users drops into a channel in the mode it is in today. In the third case the system can make use of multicasting the same movie to other potential users, therefore no one should have exclusive control over the movie. Another factor of the tariffs is to correlate the price with the amount of commercials. The same is possible for audio only, which in such a case would replace audio CD archives. It is very likely that video on demand as described above may first be commercially successful in restricted set-ups with 100 to 500 users, like for example in hotels.

By interactive television the user may directly influence the behaviour and the next step in the movie. This is done by allowing the viewer to choose between alternatives at different points in the movie. The easiest way is to just have the action being described from different viewpoints. In a more elaborated mode the choice of path leads to different stories. An early example is the movie 'I'm your man' developed by the Controlled Entropy Entertainment in New York [Bufo94]. This movie is stored on a videodisc and runs for a total length of 20 minutes whereas the total footage is 90 minutes. Every few minutes there is a branch and depending on this choice the movie is continued. The creation of this type of movie will cost considerably more than traditional movies with the same level of quality. It is said to be around 3 to 8 times the cost.

With the advent of interactive TV a movie may be either a traditional linear presentation or even a full interactive game. Multimedia capabilities are required for more realistic games and audio visual simulations. However the requirement for playing games are different to those of todays multimedia workstations: Media is always presented and never captured. Such a system demands high throughput in exactly one direction but not the reverse path into the computer.

Media Communication

Media Communication denotes all kind of applications, which involve more than one person and is related to interpersonal communication. Video telephony and video conferences are applications which belong to this category. At this point we can broaden the scope to cover not only audio and video but, also other media in the conferencing domain and encounter a large set of CSCW applications. We also have applications which include type of asymmetric communication like one teacher who interacts with a class in form of tele-tutoring. All multimedia mailing applications with, e.g., MIME, also belong to the media communication group. In general the classical CSCW classification of a two by two matrix between communication at the same time/at different time vs. communication at the same place/at different locations is suitable for the whole set media communication applications.

From the requirements point of view, applications with humans interacting simultaneously demand for real time and for a well defined end-to-end delay. If the involved parties communicate at different times then there are no such demands.

3 Aggregated Application Requirements

Having surveyed the above mentioned multimedia applications, they by themselves impose new requirements on data handling in computing and communications because they need (1) a substantial data throughput, (2) fast data forwarding, and (3) service guarantees.

Audio and video data occur as streams and demands, even in a compressed mode, for high data throughput. The MPEG-2 standards defines various compression schemes for video with associated audio. There are 3 audio layers specified with different implementations and quality requirements. Video is arranged in a set of profiles and layers which correspond to different image qualities and sizes. The resulting data streams range from up to 4 MBit/s to at most 100 Mbit/s. An excellent quality, compared with today's television, can already be achieved with about 4 Mbit/s. In a workstation or a network, several of those streams may exist concurrently demanding high throughput. Further, the data movement requirements on the local end-system, translate into terms of manipulation of large quantities of data in real-time where, for example, data copying can cause bottleneck in the system.
The fast data forwarding imposes a problem on the end-systems where these different applications can coexist in the same system, and they may have requirements on data movement ranging from normal error-free data transmission to new time-constraint types of traffic. In general, the faster the communication system can transfer a data packet, the fewer packets need to be buffered. This requirement leads to a careful spatial and temporal resource management in the end-systems and routers/switches. The application imposes constraints on the end-to-end delay. In a retrieval like application, such as the video on demand example, a delay of up to 1 sec may be easily tolerated. On the other hand, dialogue application such as a videophone or video-conference demand end-to-end delays lower than typically 200 ms in order to allow for a natural communication between the users.

The multimedia applications need service guarantees, or at least probabilistic commitments, otherwise they will not be accepted as these systems compete with, e.g., radio and television services. In order to achieve services guarantees, resource management must be used. Without resource management in end-systems, switches, and routers, multimedia systems cannot provide reliability to the users in case of resource bottlenecks transmission over unreserved resources lead to dropped or delayed packets.

4 Resource Management Systems

The multimedia applications need for specific qualities of services (QoS) imply two important requirements to the underlaying system: There must be an interface between the application-oriented and the system-oriented modules of multimedia applications, which provides specification and exchange of the applications QoS needs and system QoS.

A resource management system is required to handle the specified QoS parameter by comparing the application needs with the available resource capacities in the local system and networks. The resource management system also reserves the required resources and enforces the guaranteed QoS for the application. Management of local and network resources can also be done with best-effort QoS. There is only a probabilistic commitment for the set of QoS parameters, however the QoS provided to the application will only be degraded when a resource becomes a bottleneck. In case of resource overloads the application requirements are scaled down for a certain time. Best-effort QoS is well-suited if the underlaying resource does

not provide QoS guarantees (e.g., non-deterministic networks such as Ethernet) and if it is only required to reserve resources for the average case.

There are also combinations of both approaches: A minimum QoS can be reserved by using the guaranteed reservation mechanisms, the additional QoS requirements are served on a best-effort basis. This approach can help to save guaranteed bandwidth in case of variable bit-rate (VBR) multimedia streams. The following section explains the notion of QoS, resources, and describes both types of resource management.

4.1 Notion of QoS

Traditional QoS is provided by the network layer per connection. An enhancement of QoS is achieved through the introduction of QoS for transport services. For multimedia networked systems, the notion of QoS has to be extended because many other services contribute to the end-to-end service quality. To discuss further QoS concepts and resource management principles, we need a system model of the multimedia systems. We assume the following model for the multimedia system: the multimedia system consists of two domains: application and system.

The application domain contains modules for the programmer of a multimedia application, e.g., programming abstractions for audio/video handling, or the run-time system for the dynamic module management. The system domain is divided into two parts: communication services and operating system services.

The requirements of multimedia applications and data streams have to be served by the single components of a multimedia system. The resource management maps these requirements onto the respective capacity. The transmission and processing requirements of local and distributed multimedia applications can be specified according to the following characteristics:

1. The throughput is determined by the data rate a connection needs in order to satisfy the application requirements. It also depends on the size of the data units.
2. We can distinguish between local and global delay:
   a. The local delay at the resource is the maximum time span of a certain task to wait for serving plus processing at this resource.
   b. The global end-to-end delay is the total delay for a data unit being transmitted from the source to its destination. The end-to-end delay is the accumulated local delay of all resources which are involved in the transmission line from data source to data sink. For example, the source of a video telephone is the camera. The destination is the video window on the screen of the partner.
3. The jitter (or delay jitter) determines the maximum allowed variance in the arrival of data at the destination.
4. The reliability parameter defines error detection and correction mechanisms used for the transmission and processing of multimedia tasks. Errors can be ignored, indicated and/or corrected. It is important to notice that error correction through re-transmission is rarely appropriate for time-critical data because the re-transmitted data will usually arrive late. Forward error correction mechanism are more useful.

In accordance with communication systems these requirements are known as the quality of service parameters.

4.2 Notion of Resources

A resource is a system entity required by tasks for manipulating data. Each resource has a set of distinguished characteristics described in the following model:

- There are active and passive resources. An active resource is the CPU or a network adapter for protocol processing, it provides a service. A passive resource is the main memory, bandwidth, or a file system; it denotes some system capability required by active resources.
- A resource can either be used exclusively by one process at the time or shared between various processes. Active resources are often exclusive, passive resources can usually be shared among processes.
- A resource that exists only once in the system is known as a single resource, otherwise it is a multiple resource. In a transputer based multiprocessor system the individual CPU is a multiple resource.

Each resource has a capacity which results from the ability to perform tasks on the resource in a given time-span. In this context capacity refers to CPU capacity, frequency range or, for example, to the amount of storage. For real-time scheduling only the temporal division of resource capacity among real-time processes is of interest.

For example, process management belongs to the category of active, shared, and most often to single resources. A file system on an optical disc with CD-ROM XA format is a passive, shared, single resource.

A possible realization of resource allocation and management is based on the interaction between clients and the respective resource managers. The client selects the resource and requests a resource allocation by specifying its requirements through a QoS specification. This is equivalent to a workload request. First the resource manager checks its own resource utilization, and decides if the reservation request can be served or not. All existing reservations are stored, and their share in terms of the respective resource capacity is guaranteed. Moreover, this component negotiates the reservation request with other resource managers if necessary.

The following example of a distributed multimedia system illustrates this generic scheme: During the connection establishment phase the QoS parameters are usually negotiated between the requester (part of the multimedia application) and the addressed resource manager. The negotiation starts in the simplest case with specification of the QoS parameters by the application. The resource manager checks whether these requests can be guaranteed or not. A more elaborate method is to optimize single parameters. In this case two parameters are determined by the application (e.g. throughput and reliability), the resource manager then calculates the best achievable value for the third parameter (e.g. delay). To negotiate the parameters for end-to-end connections over one or more computer networks, protocols like ST-II [Topo90] and RSVP [ZBEH94] are employed. Here, the resource managers of the single components of the distributed system allocate the necessary resources.

In the following case shown in Figure 1 two computers are connected over a LAN. The transmission of video data between a camera connected to a computer Server and the screen of the computer User involves, for all depicted components, a resource manager.

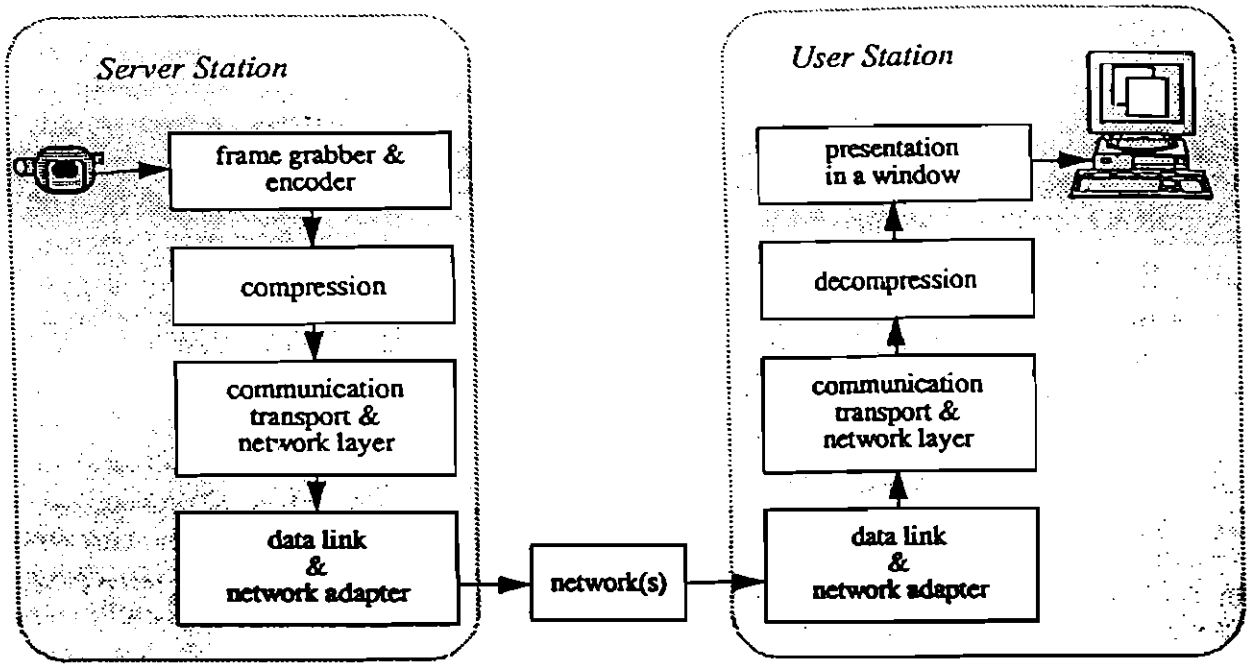


Figure 1: Components grouped for the purpose of video data transmission

This example illustrates that in addition to the individual resource managers there must exist a protocol to coordinate these services, like the one in ST-II.

4.3 Resource Reservation and Management

The tasks of a resource manager covers different phases of the allocation and management process:

1. **Schedulability Test:** The resource manager checks with the given QoS parameters (e.g. throughput and reliability) if there is enough remaining resource capacity available to handle this additional request.
2. **Quality of Service Calculation:** After the schedulability test the resource manager calculates the best possible performance (e.g. delay) the resource can guarantee for the new request.
3. **Resource Reservation:** The resource manager allocates the required capacity in order to meet the QoS guarantees for each request.
4. **Resource Scheduling:** Incoming messages from connections are scheduled according to the given QoS guarantees. At the process management the allocation of the resource is done by the scheduler at the moment the data for processing arrives.

With respect to the last phase for each resource a scheduling algorithm is defined. The schedulability test, QoS calculation and resource reservation depend upon this algorithm used by the scheduler.

5 Resource Management Applied to the CPU

This section describes the specification, reservation, and the enforcement of guaranteed QoS parameters with the CPU as resource. The CPU, being the principal local resource to be managed for all multimedia applications, requires functions especially for schedulability test, resource reservation, and the enforcement of guaranteed QoS parameters, to enable the processing of real-time multimedia data. In general, application generated CPU workloads can be divided into the following service classes:

- non real-time processes
- control processes handling real-time and non-real-time tasks
- real-time processes with soft deadlines
- real-time processes with hard deadlines

The first class contains processes without tight deadlines. Examples for this type of applications are programs for most *Media Capture, Media Preparation*, and parts of *Media Interaction* applications. Because these applications have no real-time requirements they may run with the lowest priority among the above classes. In most CPU reservation and scheduling schemes at least a small amount of CPU capacity will be reserved for such applications to avoid starvation. This part of the CPU capacity can be managed using classical strategies such as round robin or multi-level feedback.

Control processes regulate the access to certain resources. For example, an important control process for the CPU is the scheduler. A control process can also be used to reserve network resources in the setup phase of *Media Communication* applications. On the other hand the same control process can be involved in the transmission of real-time messages, e.g., acknowledgment messages to trigger retransmissions of lost or corrupted data units in small networks [DHHH93]. Control processes should be handled in a deterministic manner with high priority.

Real-time processes with hard deadlines (e.g., manufacturing control systems) are most often processed by dedicated systems. For this class of applications there exists a great variety of scheduling and reservation mechanisms [CSRa88]. Though it is difficult to integrate hard real-time systems with multimedia applications, it can be done [SCZh93].

Most multimedia processes, especially applications for *Media Interaction* and *Media Communication*, can be considered as soft-real-time processes. If a data unit is processed too late or lost, this is not necessarily noticed by the human viewer. To avoid disruptions a characteristic quality of the media presentation is required. To describe these processing requirements of multimedia applications, a characteristic set of QoS parameters must be defined. For example, the playback of a video stream requires a picture loss rate below a certain percentage, and that each logical data unit (e.g. video frame) must be displayed at a certain point in time softened by a certain delay jitter. In a multimedia system resource reservation mechanisms and QoS enforcement strategies are used to guarantee QoS requirements of multimedia applications. The CPU resource management part of this paper focuses on multimedia applications which process continuous media with soft deadlines.

5.1 CPU Scheduling Algorithms

There are two prominent algorithms for CPU scheduling: Earliest Deadline First (EDF) and Rate Monotonic (RM), which are used in multimedia systems.

Earliest Deadline First

EDF is one of the best known algorithms for real-time processing. At every new ready state, the scheduler selects among the tasks that are ready and not fully processed the one with the earliest deadline. The requested resource is assigned to the selected task. At any arrival of a new task, EDF must be computed immediately heading to a new order - i.e. the running task is preempted and the new task is scheduled according to its deadline. The new task is processed immediately if its deadline is earlier than the deadline of the interrupted task. The processing of the interrupted task is continued according to the EDF algorithm later on. EDF is not only an algorithm for periodic tasks but also for tasks with arbitrary requests, deadlines and service execution times [Dert74]. In this case, no guarantee about the processing of any task can be given.
EDF is an optimal, dynamic algorithm: I.e., it produces a valid schedule whenever one exist. A dynamic algorithm schedules every incoming task according to its specific demands. Tasks of periodic processes have to be scheduled in each period. With n tasks, which have arbitrary ready-times and deadlines, the complexity is $\Theta(n^2)$.

EDF is used by different models as basic algorithm. An extension to EDF is the time-driven scheduler. Tasks are scheduled according to their deadline. Furthermore, the time-driven scheduler is able to handle overload situations. If an overload situation occurs the scheduler aborts tasks which can not meet their deadlines any more. If there still is an overload situation the scheduler removes tasks which have a low "value density". The value density corresponds to the importance of a task.
Applying EDF to the scheduling of continuous media data on a single processor machine with priority scheduling, process priorities are likely to be rearranged quite often. A priority is assigned to each task ready for processing according to its deadline. Common systems usually provide only a restricted number of priorities. If the computed priority of a new process is not available, the priorities of other processes have to be rearranged until the required priority is free. In the worst case, the priorities of all processes have to be rearranged. This can cause a considerable overhead. The EDF scheduling algorithm itself makes no use of the previously known occurrence of periodic tasks.

Rate Monotonic

The Rate Monotonic scheduling principle was introduced by Liu and Layland in 1973 [LiLa73]. It is an optimal, static, priority-driven algorithm for preemptive, periodic jobs. Optimal in this context means that there is no other *static* algorithm that is able to schedule a task set which can not be scheduled by the rate monotonic algorithm. A process is scheduled by a static algorithm at the beginning of the processing. Subsequently, each task is processed with the priority calculated at the beginning. No further scheduling is required. The following five assumptions are necessary prerequisites in applying the rate-monotonic algorithm:

1. The requests for all tasks with deadlines are periodic. I.e., with constant intervals between consecutive requests.
2. The processing of a single task has to be finished before the next task of the same data stream becomes ready for execution. Deadlines consist of run-ability constrains only. I.e., each task must be completed before the next request occurs.
3. The request of tasks is independent. I.e., the requests for a certain task do not depend on the initiation or completion of requests for any other task.
4. Run-time for each request of a task is constant. Run-time denotes the maximum time

which is required by a processor to execute the task without interruption.

5. Any non-periodic task in the system has no required deadline. Typically they initiate periodic task or they re tasks for failure recovery. They usually displace periodic tasks.

Further work has shown that not all of these assumptions are mandatory to employ the rate-monotonic algorithm [LSST91, SKGo91]. Static priorities are assigned to tasks at connection set up phase, according to their rates. The task with the shortest period gets the lowest priority.

5.2 Schedulability Test

To test whether a new task can be scheduled using RM or EDF scheduling, the following condition must be met:

$$\sum_{all\ tasks\ i} R_i \times P_i \leq B$$

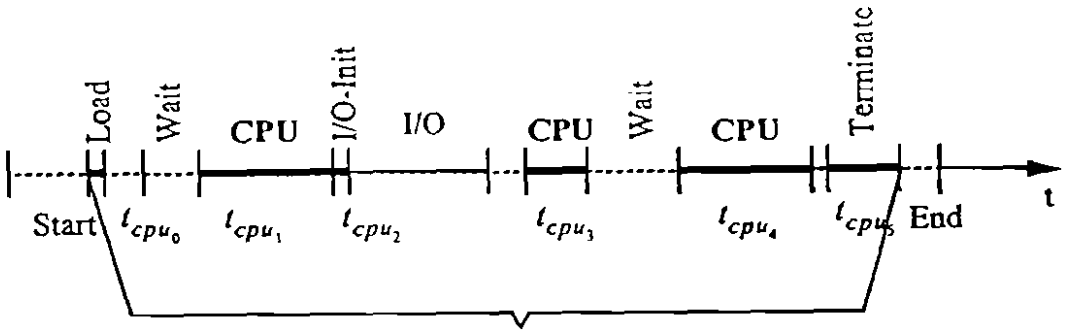
Index i identifies all multimedia tasks, R_i denotes the maximum processing rate of task i, P_i is the processing time per-period, and B is the schedulability bound. For RM scheduling the schedulability bound is determined by $B=\ln(2)=0.69$. The schedulability bound of EDF scheduling is $B=1$.

Each multimedia application has to specify the workload it will generate. This workload QoS specification consists of the processing rate R and execution time per period P. The resource management system performs the schedulability test to decide whether this new application can be accepted. If enough CPU capacity is available to execute the new application without disturbing existing applications the schedulability test returns successfully. As seen from the above formula, the processing time is an important parameter for the schedulability test, however, the determination of these values is still a largely unresolved issue in the area of CPU scheduling.

5.3 Specification and Measurement of Processing Time

The processing time is a key parameter for the schedulability test. Reservation and scheduling of multimedia processes can be only performed if the processing time is previously well-known. Developers of multimedia applications must therefore be supported by the resource management system to determine the consumption of processing time for the CPU resources. We define the processing time of an application process as follows: "The processing time of an application process is the overall duration in which the CPU is occupied in order to perform this application task."
In Figure 2 it can be seen that the processing time consists of two different parts. In the first place, it includes the pure application code execution time on the CPU. Additionally, there are operating system activities to make the execution of multimedia applications possible, e.g., context switches, initialization and termination of I/O operations. An example is a multimedia application which plays an MPEG video stream coming from the hard disk. Most parts of the CPU are utilized for the software decompression algorithm, yet the read operation also needs CPU time, at least to set up an asynchronous disk I/O operation to read the multimedia data.

To automate the measurement of these processing times the Heidelberg Predictor of Execution Times Tool (HeiPOET) has been developed. HeiPOET provides a computer-based prediction of CPU processing times and a refinement of measured values during the execution time of multimedia applications which are based on a stream-handler model. HeiPOET offers such functionality as:



$$t_{cpu} = \sum_{i=0}^{n} t_{cpu_i}$$

t_{cpu_i}...Duration of i-th processing cycle

t_{cpu}... Overall CPU utilization of the multimedia module

Figure 2: Definition of Processing Time

- Specification language for the definitions of parameters of stream handlers
- High precision of measurement results by eliminating operating system interruptions
- Computation of statistical measurements to check the reliability

Based on these measurement results the schedulability test and CPU reservations can be done. For further detail of the HeiPOET architecture and use see [WWVo94].

5.4 CPU Resource Reservation

When a schedulability test is successful, then the reservation parameters are stored in the reservation database. The reservation database is used by the resource management system for book-keeping operations. Each accepted QoS request is added to the reservation table, and must be kept until the multimedia application terminates or the QoS requirements change. Enhancements to existing reservations are only allowed if the additionally required resources are available. An example reservation table is shown in Table 1.

Table 1: Example Reservation Table

Index	Processing Time per Period (ms)	Rate (s^{-1})	CPU Utilization (%)	Priority (RM)
0	10.9	30	32.7	1
1	0.025	8000	20.0	0
..	..	..	..	..
..	..	..	..	..
5	200	1	20.0	2

The most important information in the reservation table is the CPU utilization of the specific multimedia processes (the product of processing time and rate). This information is used in the schedulability test. Additionally, the processing time of an application and its related parame-

ters are kept. If the rate-monotonic scheduling algorithm is used, the fixed scheduling priority of the application is computed and stored. Typical parameters of a Motion-JPEG software decoder are shown in the first row of Table 1. The application takes 10.9 ms to decode an incoming M-JPEG frame, and the processing rate is 30 frames per second. This process takes 32.7 percent of the whole CPU capacity. Because the process has the second highest period in this process system the process is assigned priority 1.

5.5 QoS Enforcement

The enforcement of the QoS guarantee is done by a two-staged architecture: scheduler and dispatcher. Based on the scheduling algorithm the scheduler assigns each process of the system its corresponding priority. The dispatcher ensures that the process with the highest priority is running on the CPU. This means the dispatcher is responsible to switch the processor context if a process is running and a process with a higher priority needs to be served.

Consider an audio and a video stream scheduled according to EDF and RM algorithm: Let the audio stream have a rate of 8000 samples/s and the video stream a rate of 30 frames/s. Using RM scheduling the priority assigned to the audio stream is then higher than the priority assigned to the video stream. The arrival of messages from the audio stream will interrupt the processing of the video stream. If it is possible to complete the processing of a video message that requests processing at the critical instant before its deadline, then the processing of all video messages to their deadlines is ensured. A feasible schedule exists. The priority of a task

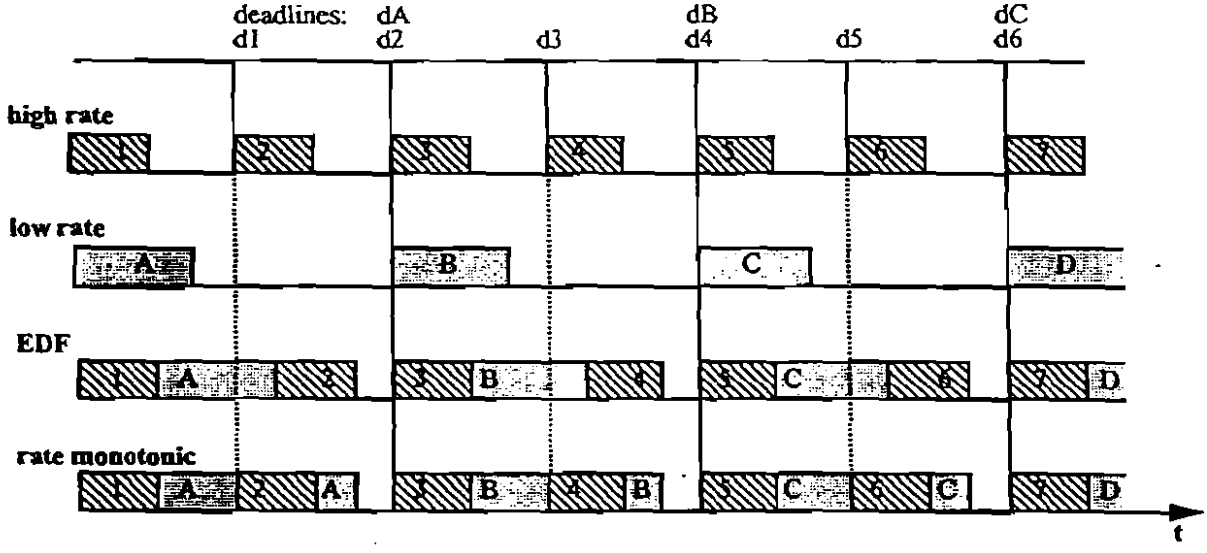


Figure 3: QoS Enforcement: RM versus EDF

scheduled by using EDF depends on its task specific deadline. Figure 3 shows example schedules for both EDF and RM algorithms. For further details on CPU resource management see, e.g., [Stei94].

6 Resource Management Applied to Networks

Today we encounter a great variety of dedicated networks for multimedia communication. Analog telephone networks are mostly used for audio communication, terrestrial and satellite networks for TV broadcasting, and digital computer networks for data communication. The

future of communications will be a service-integrated network which provides the end-to-end transport of all kind of digitized media using integrated QoS management mechanisms. There is research into translation from existing networks to networks of the future.

A classification of the most prominent networks is shown in Figure 4. QoS management in circuit-switched networks is based on obvious binary decisions, e.g., if the telephone network is capable to handle an additional call then the connection is accepted. Once accepted such a connection, the QoS for speech transmission would only be degraded in case of severe failures in the telephone system. Because of the a-priori channel reservation broadband networks do not need to have QoS management.

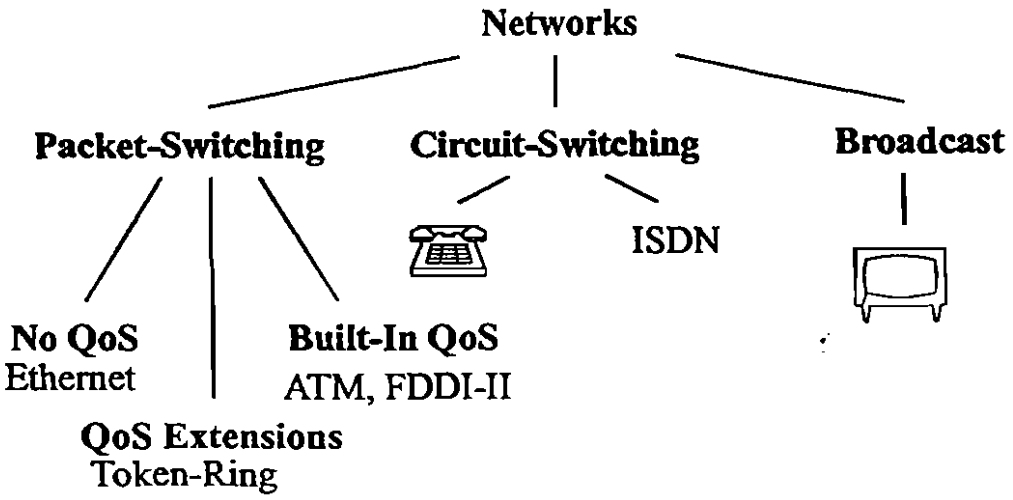


Figure 4: Classification of Networks

Therefore our work focuses on QoS management for packet-switched networks. One key issue is the integration of the different types of packet-switched networks into a common integrated service network with high bandwidth and QoS support. In this section we describe a scheme which enables best-effort QoS in packet-switched networks with appropriate mechanisms for scaling of multimedia streams. There are many reason to use best-effort QoS as an alternative to the guaranteed QoS for network resource management:

- Different multimedia applications have different requirements on the communication system. As an example, while video-on-demand or audio-on-demand applications (in the next generation of present HiFi equipment) will be very sensitive to quality reductions, it is quite easy to accept small transmission errors in video conferencing scenarios (in the next generation of present phones). According to the QoS requirements, different service classes can offer different service costs.
- A pre-requisite for resource reservation is a resouree management facility. Different classes of packet-switched networks are to be differentiated: those that already contain resource management mechanisms (e.g., ATM, 100BASE-VG-Ethernet), those that can be extended by appropriate mechanisms (e.g., Token Ring, FDDI), and those that do not provide any resource management at all (e.g., Ethernet, Fast Ethernet). According to the local buffering or processing schemes, reservation mechanisms for the local resources can be supplied by routers and communication endpoints. Often the requests for a certain QoS must be handled

by a network consisting of a broad mix of resources. If there is one resource in a guaranteed connection which cannot give guaranteed processing bounds, the whole service is not guaranteed.
Using congestion control, it is not possible to give guarantees, but an integrated transmission scheme resolves access conflicts to overloaded resources by a selective decrease of the overall transferred data. Multimedia streams belonging to a high QoS class can be protected by using the prioritization scheme.

- Guaranteed QoS for variable bitrate (VBR) data transmission is inefficient. In most reservation protocols a constant bandwidth is required. Therefore VBR transmissions often are considered as constant bandwidth streams with the average overall bandwidth as a parameter. However, in the worst case all multimedia streams reach their peek bandwidth at the same moment and the resources may not be sufficient. In this case an uncontrolled loss of packets will occur. There is also a compromise between guaranteed transmission for the worst case and no guarantee at all. A certain base part of the VBR stream is guaranteed using a resource management system. The variable part of the bandwidth requirements is handled with a best-effort service using congestion control mechanisms, e.g., network layer scaling. Resource monitoring detects arising overloads. A controlled reduction of bandwidth requirements down to the guaranteed base parts are used to handle the congestion.

We describe the overall architecture proposed for scaling in the network layer, monitoring mechanisms to detect congestions, and the dynamic adjustment of filters in the network (for details see [WWSa94]).

6.1 Scaling

The definition of scaling is derived from [CSZ92], [TTCM92]: "Scaling is the dynamic adjustment of traffic load to the currently available resource capacity."

An example clarifies the principles of network layer scaling: Figure 5 shows a simple network topology with two streams of a multiparty connection branching in Router B to a target via Router C and to Target D, and a point-to-point connection from Sender A to a target via Router C. The monitor functions of Router C detect an overload situation and report this to the scaler of Router C. The scaling device (*scaler*) decides that the congestion can be solved by reducing the traffic in Sender A, Router B and Router C. Router C sends a SCALE_DOWN message to Sender A and Router B. Both use a priority scheme to decide which packets should be dropped to meet the new, reduced data rate. The amount to be dropped is defined relative to the maximum transmission rate negotiated in the setup phase of multimedia streams.
The link from Router B to the Target D is not overloaded. So Router B forwards all incoming packets of the stream to Target D (multiparty connection). This results in different QoS in the two branches of the multicast tree.
The reporting of resource bottlenecks to the upper layers of the end systems is the key issue of an integrated scaling scheme. Thereby upper layer scaling mechanisms at the senders and scaling mechanisms in the network layer can cooperate.

6.2 Monitoring Mechanisms

A key resource for distributed multimedia applications is the network. If a network is unable to handle the application requirements, the network adapter buffer will be filled. This is a typical situation for bottlenecks in distributed multimedia communication scenarios: A mid-term capacity problem of the resource "network" results in a capacity problem of another resource, e.g., the previous buffer. In case of a long-term overload there is an overflow of sender buffers.

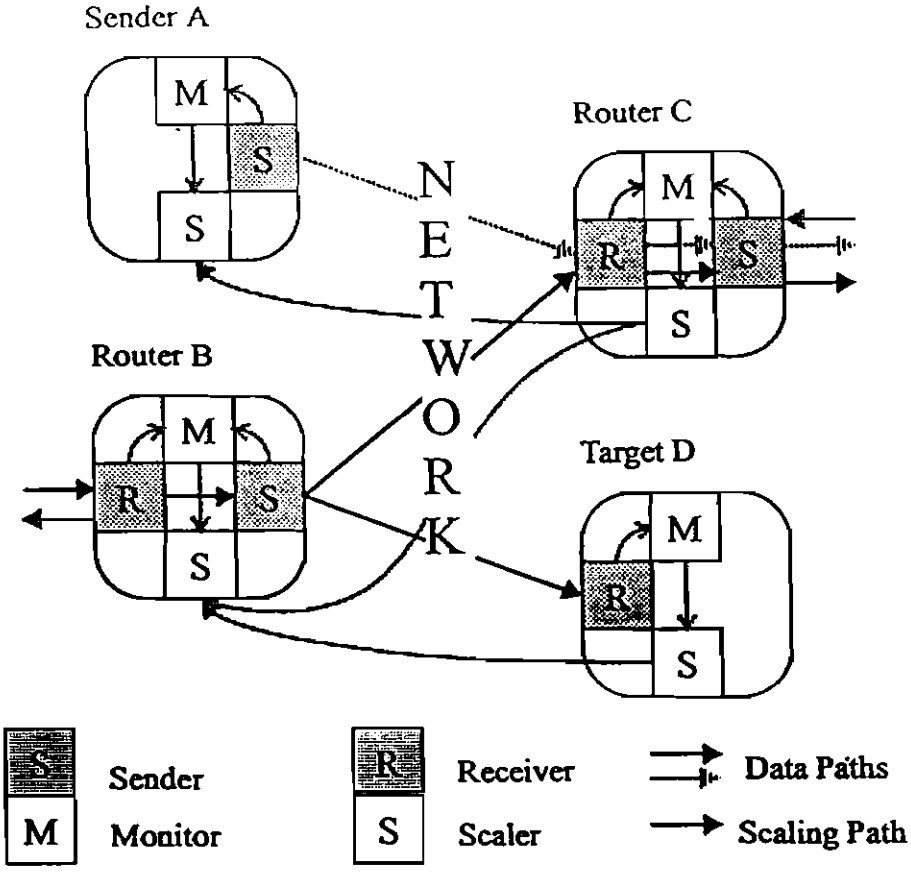


Figure 5: Network Layer Scaling Mechanism

The detection of overflowing buffers on the sender side is a direct indication of the bottleneck network. Here different monitoring mechanisms are surveyed: monitoring of delay, throughput, utilization, and loss rate.

Delay Monitoring

Delay monitors observe the processing times of packets. Delay monitoring is suggested for detection of CPU and network bottlenecks. There are different ways of delay monitoring:

In the transport layer, each logical data unit is represented by a transport service data unit (TSDU). Each TSDU has an expected arrival time on the next network node. A TSDU arriving later than expected indicates congestion.
There are several ways to define the expected arrival time of a TSDU. One could, for example, define its value simply as the actual arrival time of the previous TSDU plus the period of the message stream (the reciprocal value of its rate). Additionally, the arrival time of the first TSDU of the stream (or any other earlier packet) rather than the arrival time of the previous packet could be used as the basis for the calculation. This helps to avoid false indications of congestions in cases where the previous TSDU happened to arrive early and the current TSDU has a normal delay. The expected arrival time is calculated as the "logical arrival time" of the previous packet plus the stream period.
But common routers work on the network layer and do not know TSDUs. This algorithm fails when based on network service data units (NSDUs), because it is often impossible to assume

the sequence of network service data units (NSDUs) being a periodic stream. There are bursts consisting of a various number of NSDUs which are caused by the segmentation algorithm in the transport layer. The above algorithm can work with NSDU only if there is a regulator in the network layer which smooths the bursts, and if all TSDUs have the same size (i.e., the same number of NSDUs). Then the NSDUs will also have a periodic behavior. However, in most networking stacks these assumptions are not met.
Filling each NSDU with a timestamp at the sender and in each router can also solve the above problem, because the delay measurement is based on the links. This mechanism is expected to be too expensive for the routers and the network layer protocol. Therefore it is often appropriate to monitor delays in the network nodes within routers. Assuming there would be a mechanism to mark each TSDU beginning the NSDU header at the sender, the monitoring of TSDU delays is possible. To solve this problem a "start bit" in each NSDU header is used to mark the first packet of a TSDU (detailed description see [WWSa94]).

Throughput Monitoring

Throughput monitoring is used to detect CPU and network overloads by watching the packets leaving this resource. The monitoring mechanisms that are mostly used for throughput are leaky bucket and moving window mechanisms.

Leaky bucket throughput monitoring mechanisms are based on simple packet counters which are incremented by 1 every time a packet arrives and decremented by 1 in fixed time intervals. Another technique is the jumping window mechanism, in which the throughput is measured within a fixed time interval. The new time interval starts immediately at the end of the preceding time interval.

In the moving window throughput monitoring mechanism the throughput (*TPT*) of a stream in a given time interval τ_{TPT} is determined by the sum of the length of packets being transmitted for this stream in this interval, devided by the time interval. For example, the interval τ_{TPT} can be defined as the last second $[t-1, t]$, where t is the current time.

All types of throughput monitoring are suitable for constant bit-rate stream, whereupon case the expected throughput is a fixed value. Throughput monitoring can also be used for multimedia streams with variable bit rates. When the monitored throughput falls below a minimum threshold, this indicates problems of previous resources.

Utilization Monitoring

Utilization monitoring can be used for CPU, network, and buffer resources. The utilization of resources can be computed by dividing the currently used parts of a resource by the overall capacity of the resource.
For example, the CPU utilization can be computed by measuring the time spent in the IDLE state. Usually, the CPU is loaded with time-critical and non-time-critical processes. The time critical processes should have a reservation for a certain percentage of the processing time and will also have a high priority for the scheduling scheme. The rest will be spent for non time critical processes. So an overall CPU load near 100% does not necessarily mean that the time critical processes cannot get enough time. It is recommended to separately measure the utilization for multimedia real-time communication. Additionally, per-process monitoring of CPU consumption allows a more sensitive scaling by noting the different loads of multimedia applications.

Loss Rate Monitoring

A problem arising when only delay is monitored is the definition of a threshold value above which congestion is assumed. Another direct indicator to detect congestion is the rate of packet lost. To adjust the value of the threshold value for the delay, the monitor continuously compares the mean delay to the corresponding value of the mean loss rate.
An assumption needed to enable the detection of lost packets is that all NSDUs of a stream are numbered in sequence. The numbering is done by the sender. The effort is very low.
The transmission algorithm must keep the sequence of packets through the network. Unlike the transmission of packets with IP, the sequence of packets in most protocols for real-time transport of multimedia data will be retained, because the route of a stream through the network will not be changed and there is no parallel processing of packets of the same stream in a router. If there are gaps in the order of packet numbers, the packets that have these numbers are considered as lost packets.

The detection of corrupted packets is an indication of network failures, but not a good indication of resource bottlenecks. Additionally, the detection of corrupted packets requires a very time-consuming computation of error-check sequences. Corrupted NSDUs will therefore not be considered.
The threshold for loss-rate monitoring must be variable. Assuming a network problem has been detected packets are dropped. In this phase the threshold for loss rate monitoring must be increased to prevent more bottleneck detections from the loss-rate monitor.

6.3 Filter Mechanisms

Filters can be changed in the stream setup phase as well as dynamically at run time. Short term congestion of resources can be managed using filters before the resource. Mid-term and long-term changes in filtering scheme should be propagated through the network nodes in direction to the sender; this affects the work of filters on this route.
If a resource bottleneck is detected, the monitor sends a scaling message to the previous filter, which often consists of three values (see Figure 6).

RELAXATION	DURATION	STREAM-ID

Figure 6: Scale-Down Protocol Data Unit

The first parameter describes the relaxation of the data throughput in relation to a maximum throughput value negotiated between sender and receiver. The relaxation is not described in relation to the actual traffic to prevent unjusted scaling of multimedia streams. For example, when the scale down message is related to the actual data rate, a renewed scaling message would affect multimedia streams being currently downscaled much more than unscaled multimedia streams. The relaxation parameter is accompanied by a time value which specifies the validity of the scaling message: the duration parameter. When the scaling message times out, the sender can increase the data rate again. If the congestion still continues, the receiver can send further scaling messages to increase the value of reduction, or the time value, or both. Later scaling messages substitute older ones. If the time value matches the constant PERMANENT, a permanent filter is installed and the scaling message results in a permanent degradation of the data rate. This mechanism is used to handle different receiver requirements.
To gain more flexibility in scaling policies the scaling message shall hold a list of multimedia stream identifiers. If this value is set to ALL, all streams flowing through the overloaded

resource (e.g. the succeeding network) are affected by the scaling activities. Otherwise only the streams addressed by the STREAM-ID field will be scaled. As an alternative to the reduction value, the scaling message can contain a priority mask which specifies the priorities of the packets to be dropped. This allows upper networking layers to describe and propagate static and dynamic filters. In case of long- and mid-term overload, scaling messages are also forwarded by filters to previous filters.

7 System Architecture

The employment of continuous media in multimedia systems imposes additional, new requirements to the system architecture. A typical multimedia application does not require processing of audio and video to be performed by the application itself. Usually data is obtained from a source (e.g. microphone, camera, disk, network) and is forwarded to a sink (e.g. speaker, display, network). In such a case the requirements of continuous media data are satisfied best if it takes "the shortest possible path" through the system, i.e. to copy data directly from adapter to adapter. The program then merely sets the correct switches for the data flow by connecting sources to sinks. Hence the application itself never really touches the data as is the case in traditional processing. A problem with copying from adapter to adapter is the control and the change of quality of service parameters. In multimedia systems such an adapter to adapter connection is defined by the capabilities of the two involved adapters and the bus performance. In todays systems this connection is static. This architecture of low-level data streaming corresponds with proposals for using additional new busses for audio and video transfer within a computer. It also enables a switch-based rather then a bus-based data transfer architecture. Note, in practice we encounter header and trailers surrounding any kind of continuous media data coming from devices and being delivered to the devices. In the case of compressed video data, e.g. the MPEG-2 program stream, it contains several layers of headers compared with the actual group of pictures to be displayed.

Most of todays multimedia systems have to coexist with conventional data processing. They share hardware and software components. For instance, the traditional way of protocol processing is slow and complicated. In high speed networks protocol processing is the bottleneck because it can not provide the necessary throughput. Protocols like VMTP, NETBLT and XTP try to overcome this drawback but research in this area has shown that throughput in most communication systems is not bounded by protocol mechanisms but by the way they are implemented. Time intensive operations are, for example, physical buffer copying. Since the memory on the adapter is not very large and it may not store a few related and compressed images, data has to be copied at least once from adapter into main memory. Further copying should be avoided. An appropriate buffer management allows operations on data without performing any physical copy. In operating systems like UNIX the buffer management must be available in both, the user and the kernel space. The data need to be stored in shared memory to avoid copying between user and kernel space. For further performance improvement, protocol processing should be done in threads with upcalls, i.e. the protocol processing for an incoming message is done by a single thread. The cost of context switches of such threads are low. Developments to support such a protocol process management, are for example, STREAMS, Berkeley UNIX, and x-Kernel.

The architecture of the protocol processing system is just one issue to be considered in the system architecture of multimedia supporting operating systems. Multimedia data should be delivered from the input device (e.g. CD-ROM) to an output device (e.g. a video decompression board, or a video card) across the fastest possible path. The paradigm of streaming from

source to sink is an appropriate way of doing this. Hence the multimedia application opens devices, establishes a connection between them, starts the data flow, and returns to other duties.

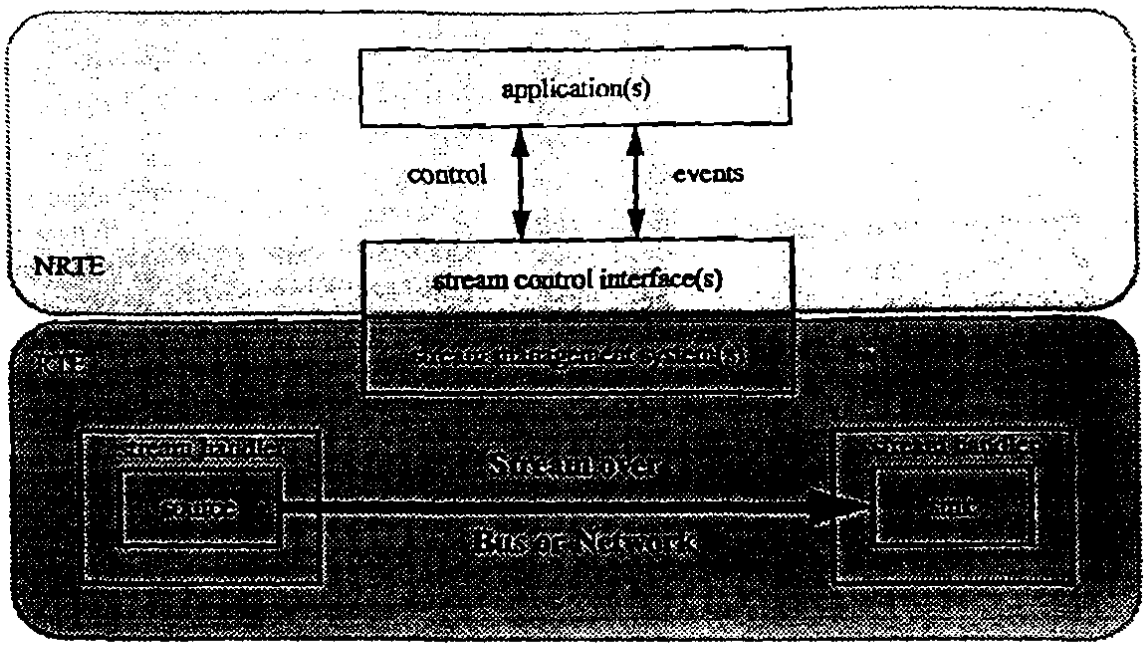


Figure 7: Real-time and non real-time environments

The most dominant characteristics of multimedia applications is to preserve the temporal requirement at the presentation time. Therefore multimedia data are handled in a ***real-time environment*** (RTE), i.e., its processing are scheduled according to inherent timing requirements of multimedia data. On a multimedia computer the RTE will usually coexist with a ***non-real-time environment*** (NRTE). The NRTE deals with all data that have no timing requirements. Figure 7 shows the approached architecture. Multimedia I/O devices are in general accessed from both environments. Data such as a video frame, for example, is passed from the RTE to the X-Window server. The RTE is controlled by related functions in the NRTE. The establishment of communication connections at the start of a stream must not obey timing requirements, but the data processing for established connections has to. All control functions are performed in the NRTE. The application usually only interfaces these control functions and is not involved in the active continuous media data handling. Therefore the multimedia application itself typically runs in the NRTE and is shielded from the RTE. In some scenarios, users may want applications to process continuous media data in an application specific way. In our model such an application comprises a module running as stream handler in the RTE. The rest of the applications run in the NRTE. Both use the available stream control interfaces. System programs such as communication protocol processing of database data transfer programs, make use of this programming in the RTE, whereas applications like media communication and media interaction are relieved from the burden of programming in the RTE. They just interface and control the RTE services. Applications determine processing paths which are needed for their data processing and the control devices and paths.

To reduce data copying buffer management functions are employed in the RTE as implementation means for data transfer. This buffer management is located between the stream handles. Stream handlers are any entities in the RTE which are in charge of multimedia data. Typical stream handlers are filter and mixing functions, but also parts of the communication subsystem

described above can be treated in the same way. Each stream handler has endpoints for input and output through which data units flow. The stream handler consumes data units from one or more input endpoints and generates data units through one or more output endpoints.

Digital multimedia data usually "enters" the computer through an input device, the source, and "leaves" it through an output device, the sink. Sources and sinks are implemented by a device driver. Applications access stream handlers by establishing sessions with them. A session constitutes a virtual stream handler for exclusive use by the application which has created it. Depending on the required QoS of a session, an underlying resource management subsystem multiplexes the capacity of the underlying physical resources among the sessions. To manage the RTE data flow through the stream handlers of a multimedia system, control operations are used which belong to the NRTE. These functions make up the **stream management systems** in the multimedia architecture. There are operations provided by all stream handlers (e.g., operations to establish sessions and to connect their endpoints) and operations specific to individual stream handler (they usually determine the content of a multimedia stream and apply to particular I/O devices).

The connected session endpoints have to generate and consume data streams of corresponding media types. Digital audio and video are encoded as sequences of bytes. Such sequences are commonly referred to as ropes.

Applications are placed in the NRTE. Some applications have the need to correlate discrete data such as text and graphics with continuous streams or to post-process multimedia data (e.g. to display the time stamps of a video stream like a VCR). These applications need to obtain segments of multimedia at the stream handler interface. With a grab function the segments are copied to the application as if stream duplication took place. Due to this operation the data units loose their temporal properties because they enter the NRTE. Applications that have to generate or transform multimedia data keeping the real-time characteristics must provide or use a stream handler included in the RTE, which performs the required processing.

8 Conclusion

This paper presents a comprehensive survey on what we assume and encounter to be the multimedia applications. After an analysis of these applications we derived the key demands, which lead to the necessity of having resource management for CPU and networking. We always encounter the schedulability test, QoS calculation, resource reservation and enforcement to be performed. We showed how real-time EDF and rate monotonic scheduling is applied for multimedia data processing as the key enforcement mechanisms of resource management. The same mechanisms can be applied to networking. There we described how traffic over non-deterministic networks is managed; key issues for network scaling mechanisms are monitoring and filtering. Finally all of today's multimedia systems comprise of a non real-time and of a real-time environment.

Having multimedia prototypes and products we still encounter a set of challenging issues with work in progress which we have not explicitly discussed above; among them the most important are:

(1) It remains difficult to estimate the processing requirements of the programs which handle multimedia data with time constraints. However, this is needed for reserving the involved resources correctly.

(2) At run time we need a supervisor function which checks the allocated resource capacities against the currently used capacity. This task will become easier in the future if we will

have integrated multimedia capabilities into the operating systems themselves.

(3) EDF and rate monotonic scheduling assume the different processes to be independent. However, very often some processes run concurrently as they have a strong relationship between themselves in terms of, e.g., lip sychronization requirements.

(4) The resource management shall take into account all resources between source and sink. The most crucial resources are CPU, memory, network, file system, and the exclusively used resources (i.e. the components such microphones, speakers, compression hardware). With the advent of more multimedia application running concurrently, we will need to look more carefully at other resources like the local bus and the file system on CD-ROMs (as shared resources).

(5) In distributed multimedia systems we have the strong need for compatibility in terms of data formats and protocols for, e.g., resource management. Today in the resource management protocol domain, there is competition between proprietary solutions, RSVP and ST-II.

9 References

[Bufo94] John F. Buford; *Distributed Multimedia Systems*; proceedings of the tutorial at IEEE International Conference on Multimedia Computing and Systems, May 14-19, 1994, Boston, MA.

[CSRa88] C.S. Cheng, J.A. Stankovic, K. Ramamritham; *Scheduling Algorithms for Hard Real-Time Systems: A Brief Survey;* IEEE Tutorial on Hard Real-Time Systems, Washington D.C., Computer Society Press of the IEEE, S. 150-174, 1988.

[CoMa94] *Video on Demand*; Special Issue of IEEE Communications Magazine, May 1994, Vol. 32, No. 5, May 1994.

[Dert74] M. L. Dertouzos: Control Robotics; *The Procedural Control of Physical Processing*; Information Processing 74 - North Holland Publishing Company, 1974.

[DHHH93] L. Delgrossi, C. Halstrick, D. Hehmann, R.G. Herrtwich, O. Krone, J. Sandvoss, C. Vogt: *Media Scaling in a Multimedia Communication System*; Proceedings of the First ACM Multimedia Conference, 1993.

[HoHe94] Wieland Holfelder, Dietmar Hehmann; *A Networked Multimedia Retrieval Management System for Distributed Kiosk Applications*; Proceedings of the IEEE International Conference on Multimedia Computing and Systems, May 14-19, 1994, Boston, MA, pp. 342-351.

[ICMCS94] *Program Chairs' Message*; Proceedings of the IEEE International Conference on Multimedia Computing and Systems, May 14-19, 1994, Boston, MA.

[LiLa73] C. L. Liu, J. W. Layland; *Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment*; Journal of the Association for Computing Machinery, Vol.20, No.1, Jan. 1973, pp. 46-61.

[LSST91] J. P. Lehoczky, L. Sha, J.K. Strosnider, H. Tokuda; *Fixed Priority Scheduling Theory for Hard Real-Time Systems*; in Foundations of Real-Time Computing, Scheduling and Resource Management; Kluwer Academic Publisher; Norwell, 1991; pp. 1-30.

[NaSt94] Klara Nahrstedt, Ralf Steinmetz; *The Fundamentals in Multimedia Systems*; Prentice Hall, to appear in December 1994.

[SCZh93] S. Shenker, D.D. Clark, L. Zhang; *A Service Model for an Integrated Services Internet*; Internet-Draft, October 1993.

[SKGo91] L. Sha, M. H. Klein, J. B. Goodenough; *Rate Monotonic Analysis for Real-Time Systems*; in Foundations of Real-Time Computing, Scheduling and Resource Management; Kluwer Academic Publisher; Norwell, 1991; pp. 129-156.

[Stei93] Ralf Steinmetz; *Multimedia -Technologie*; Springer-Verlag, 1993, Berlin.

[Stei94] Ralf Steinmetz; *Multimedia Operating Systems: Resource Reservation, Scheduling, File Systems and Architectures*; IBM-ENC Technical Report no. 43.9402, 1994.

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

[Topo90] C. Topolocic (Ed.); *Experimental Internet Stream Protocol*, Version 2 (ST II); Internet Network Working Group, RFC 1190, October 1990.

[WWSa94] Hartmut Wittig, Jörg Winckler, Jochen Sandvoss; *Network Layer Media Scaling*; submitted to the COST237 Conference 1994, Vienna, also available as IBM-ENC Technical Report no. 43.9401.

[WWVo94] Hartmut Wittig, Lars Wolf, Carsten Vogt; *CPU Utilization of Multimedia Processes: The Heidelberg Predictor of Execution Times Measurement Tool*; Proceedings of the 2nd International Workshop on Advanced Teleservices and High-Speed Communication Architectures, September 26-28, 1994, Heidelberg.

[ZBEH94] L. Zhang, B. Braden, D. Estrin, S. Herzog, S. Jamin; *Resource Reservation Protocol (RSVP)*; Internet Draft, May 1994.

[ZKSm93] Hong Jiang Zhang, A. Kankanhalli, Stephen W. Smoliar; *Automatic Parsing of Video*; Multimedia Systems, Vol. 1, No. 1, pp.10-28.

List of European Networking Center Technical Reports

TR 43.9214	Andreas Mauthe Werner Schulz Ralf Steinmetz	Inside the Heidelberg Multimedia Operating System Support: Real-Time Processing of Continuous Media in OS/2
TR 43.9215	Emanuel Farber Thomas Schütt	The Heidelberg High Speed Transport System: First Performance Results
TR 43.9301	R. Oechsle M. Graf	The Internet Protocol Family over ATM
TR 43.9302	Luca Delgrossi Frank O. Hoffmann	A Detailed Tour of ST-II for the Heidelberg Transport System
TR 43.9303	Luca Delgrossi Ralf G. Herrtwich Frank O. Hoffmann	An Implementation of ST-II for the Heidelberg Transport System
TR 43.9304	Derick Jordaan Martin Paterok Carsten Vogt	Layered Quality of Service Management in Heterogeneous Networks
TR 43.9305	L. Delgrossi; Ch. Halstrick D. Hehmann; R. Herrtwich O. Krone; J. Sandvoss; C. Vogt	Media Scaling for Audiovisual Communication with the Heidelberg Transport System
TR 43.9306	Barbara Twachtmann Ralf G. Herrtwich	Multicast in the Heidelberg Transport System
TR 43.9307	R. Steinmetz	Compression Techniques in Multimedia Systems: A Survey
TR 43.9308	Martin Bever Ulrich Schäffer Claus Schottmüller	ISO OSI FTAM and High Speed File Transfer: No Contradiction
TR 43.9309	Ralf Steinmetz	Videodatenkompression für verteilte Multimedia-Anwendungen
TR 43.9310	Ralf Steinmetz Clemens Engler	Human Perception of Media Synchronization
TR 43.9311	Luca Delgrossi; Chr. Halstrick Ralf Herrtwich; F. Hoffmann Jochen Sandvoss; B.Twachtmann	Reliability Issues in Multimedia Transport
TR 43.9312	Brian Craig McKellar Jan Roos	Buffer Management in Communication Systems

TR 43.9313	Robert Erfle Johannes Rückert Ingo Barth; Gabriel Dermler Franz Fabian; Kurt Rothermel Frank Sembach	Multimedia Document Handling - A Survey of Concepts and Methods
TR 43.9314	Martin Paterok Stefan Kaetker Carsten Vogt Luca Delgrossi, Hartmut Wittig	An SNMP MIB for the ST-II Protocol and the Heidelberg Resource Administration Technique
TR 43.9315	Luca Delgrossi Ralf Herrtwich Carsten Vogt, Lars Wolf	Reservation Protocols for Internetworks: A Comparison of ST-II and RSVP
TR 43.9316	Thomas Käppner Lars Wolf	Architecture of HeiPhone: A Testbed for Audio/Video Teleconferencing
TR 43.9317	Ralf Herrtwich Luca Delgrossi Frank Hoffmann, Sibylle Schaller	Receiver-Initiated Communication with ST-II
TR 43.9318	Michael Altenhofen Juergen Dittrich Rainer Hammerschmidt, Ralf Herrtwich, Thomas Kaeppner Carsten Kruschel, Ansgar Kueckes, Florin Spanachi Thomas Steinig, Kathrin Werner, Joerg Winckler	Implementing the BERKOM Multimedia Collaboration Service
TR 43.9319	Lars Wolf Ralf Herrtwich	The System Architecture of the Heidelberg Transport System
TR 43.9401	Jochen Sandvoss Jörg Winckler Hartmut Wittig	Network Layer Scaling: Congestion Control in Multimedia Communication with Heterogenous Netoworks
TR 43.9402	Ralf Steinmetz	Multimedia Operating Systems: Resource Reservation, Scheduling, File Systems, and Architectures
TR 43.9403	Carsten Vogt	Quality-of-Service Calculation for Multimedia Streams with Variable Bit Rate
TR 43.9404	Heinrich J. Stüttgen	Network Evolution and Multimedia Communication
TR 43.9405	Marcus Wieland Ralf Steinmetz Peter Sander	Remote Camera Control: Requirements, Concepts and Implementation
TR 43.9406	Ralf Steinmetz Hartmut Wittig	Challenges in Multimedia System Development
TR 43.9407	Lars C. Wolf Wolfgang Burke Carsten Vogt	CPU Scheduling in Multimedia Systems