Journal of Interconnection Networks
© World Scientific Publishing Company
Vol. 0, No. 0 (1999) 000–000

ANALYTICAL PERFORMANCE EVALUATION OF LOW-BITRATE REAL-TIME TRAFFIC MULTIPLEXING IN UMTS OVER IP NETWORKS

MICHAEL MENTH
Department of Distributed Systems, Institute of Computer Science,
University of Würzburg, Am Hubland, D-97074 Würzburg, Germany
E-mail: menth@informatik.uni-wuerzburg.de

Received 15 March 2000
Revised 22 January 2001

Voice and circuit switched data will be carried over IP networks in the wireline part of the future UMTS. The commonly used protocol suite for low-bitrate real-time traffic transmission in the Internet leads to large header overhead, i.e., to low utilization of the network resources by user data. Multiplexing packets of different flows into a single IP packet reduces this effect. We model packet tunneling and multiplexing with subsequent spacing in IP networks. We derive a discrete-time analysis based on a framework for solving Markov chains. The numerical results show the superiority of multiplexing schemes provided that parameters are set in an appropriate way since there are many performance tradeoffs that have substantial effect on the efficiency of the system.

Keywords: Discrete-Time Analysis, Transport Protocols, Multiplexing, IP Networks, Quality of Service, UMTS

1. Introduction

The 3rd Generation Partnership Project (3GPP) is the standardization organization of the future Universal Mobile Telecommunication System (UMTS). It has chosen the Internet Protocol (IP) to supersede Asynchronous Transfer Mode (ATM) [1] networks as the transport technology in the terrestrial network of UMTS.

UMTS offers wireless services for voice, circuit switched (CS), and packet switched (PS) data. Because of their real-time nature, the wireline part of the network must provide quality of service (QoS) guarantees in terms of packet loss and delay. In turn, these networks need traffic conditioners to protect them against overload. Traffic streams are transported with the booked QoS as long as they meet the negotiated traffic characteristics.

The commonly used protocol suite for real-time transport in the Internet yields a header which is 60 bytes long. Low-bitrate real-time traffic comes in small quantities of about 20 bytes to limit the packetization delay. Carrying every small-sized packet by itself yields eventually 300% overhead per packet and low bandwidth utilization by user data. When several real-time connections with the same destination share a common route, their IP packets have almost identical headers. Multiplexed transmission can be more bandwidth efficient than tunneling because redundant header information is suppressed. A single IP packet accommodates data from different streams with a small multiplex header. Then, the packetization delay is still short, the original payload items can be restored, and the overhead decreases.

In this work, we investigate transmission alternatives for UMTS traffic over IP networks. We consider tunneling and propose different multiplexing approaches. In all cases, a timer limits the multiplexing delay. A traffic conditioner delays or even drops the resulting IP packets if they do not comply with the negotiated traffic contract. This affects the QoS for the real-time packets which is measured in overall waiting time and loss probability. In such a transmission system, parameters like the timer value and the traffic descriptors for the IP packet stream must be set properly to maximize the number of supportable connections.

A problem of similar nature occurs in ATM networks and is solved by using the ATM Adaptation Layer Type 2 (AAL2) for multiplexing. This scenario is thoroughly investigated [14,13,12,20,25].

In Section 2, we describe the data flows in the user plane of UMTS. The structure of voice and CS data is discussed and we describe a simple QoS IP network with traffic conditioners. We present various techniques to overcome the overhead caused by tunneling. In Section 3, we establish a model for the proposed transmission alternatives and derive a discrete-time analysis. The numerical results in Section 4 compare the performance between tunneling and multiplexing and illustrate several tradeoffs for different traffic characteristics. Section 5 summarizes the work.

2. Transport of Low-Bitrate Real-Time Data

In this section, we give a sketch of the data flows in the user plane of UMTS. We explain their characteristics and motivate a model for the superposition of several streams. We assume a QoS enabled IP network that uses traffic conditioners at the edges and describe how a spacer works. We present several protocol suites for real-time transmission and outline the QoS criteria for UMTS networks.

2.1. *UMTS User Plane*

In UMTS, a mobile handset communicates over the air interface with a base station, called NodeB, which is driven by the radio network controller (RNC). The data travel over the RNC to the UMTS mobile switching center (UMSC) and are forwarded from there either into the UMTS core network or into the public switched telephone network (PSTN). If possible, the mobile transmits data over the wireless link to up to three NodeBs, simultaneously. This is called macro-diversity and helps to improve the transmission quality. The data arrive via the corresponding drift-RNCs (D-RNC) at the serving-RNC (S-RNC) where upstream signals are combined, i.e., the best packet is chosen, and downstream data are multicast (Fig. 1). The different legs of macro-diversity need to send data synchronously to the mobile terminal, therefore, even PS data have real-time constraints in the UMTS Terrestrial Radio Access Network (UTRAN). The bitrate of the traffic streams ranges from 7.6 kbps for voice up to 64 kbps for CS data. Hence, mostly low-bitrate real-time traffic is transported in the UTRAN, so it makes sense to employ techniques to improve the utilization of the resources in the wireline network.

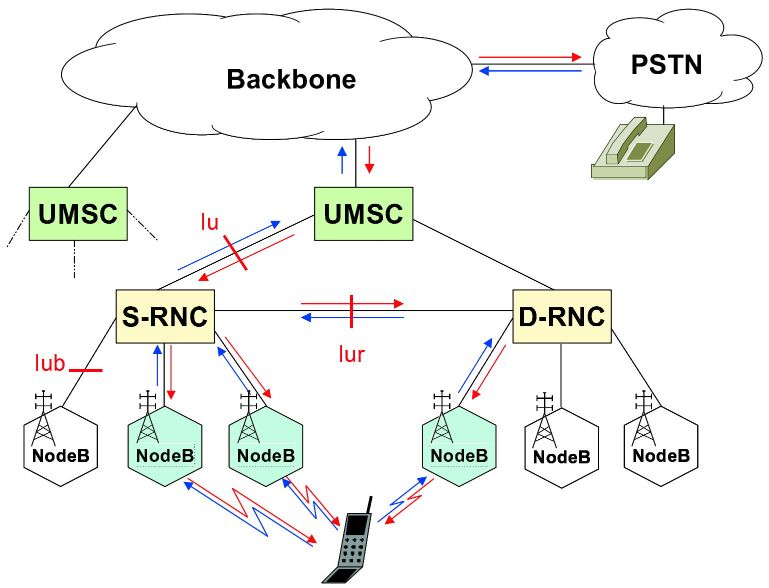

Fig. 1. Macro-diversity in UMTS.

2.2. *Services in UMTS*

So far, voice communication is the major use of wireless systems. PS data services cover WAP and WWW clients and are already realized in the Generalized Packet Radio System (GPRS) within the Global System for Mobile Communication (GSM) which is the currently used technology for cellular systems. Multimedia applications like video conference can not run on GPRS because they require more bandwidth and are as sensitive to transmission delay as voice traffic. To this end, UMTS offers CS data services in addition to voice and PS data. The maximum bitrate is currently 64 kbps.

2.2.1. *Voice*

For speech digitization, a sampling rate of at least 8000 Hz is required due to the sampling theorem, so, the PSTN transmits voice samples of 8 bits every 125 μsec. Subsequent samples of a conversation are highly correlated, especially during the on-phase or off-phase of a speaker. Voice compression techniques, like the adaptive multi-rate (AMR) vocoder, which is used in UMTS, take advantage of this fact and the compression rate increases with the number of collected samples. However, the interval for collecting can not be arbitrarily long because interactive voice requires little delay for its end-to-end transport. As a consequence, voice samples from a time frame of $F_v = 20$ msec are collected and packets of compressed information are issued every 20 msec.

To quantify the voice traffic, we compressed representative voice traces [3] using an AMR vocoder. We assumed the 12.2 kbps mode, i.e., an optimum air interface, to get an estimate of the maximum traffic coming from a wireless voice user. Table 1 shows the distribution of the compressed voice sample sizes. They include a 3 byte user plane protocol that serves for signaling purposes. The measured average rate amounts to 7.6 kbps.

Table 1. Packet size distribution for compressed voice.

Packet Size [bytes]	4	11	36
Probability	0.4754	0.0728	0.4518

2.2.2. *Circuit Switched Data*

The 64 kbps CS data service denotes also one byte per 125 μsec. The data are not compressed any further, so, the network needs to transport 64 kbps in contrast to 7.6 kbps for voice services. Carrying a single byte over a packet switched network is not efficient due to additional header overhead. Therefore, CS data are also collected during a time frame F_{CS} to form a a packet of size $B_{CS} = F_{CS} \cdot 64$ kbps.

2.2.3. *Superposition of Streams*

A voice user starts a connection at an arbitrary time instant and issues packets of size B_v with a period of F_v msec. Therefore, the superposition of several connections is characterized by the arrival process during single interval of length F_v. If we assume a mean call holding time of 90 sec, a connection takes 4500 time frames. This pattern is only slowly changing over time when new calls are accepted and others terminate.

In our investigation, the allowed delay budget for transmission is about 1 msec. So, the packets can not observe the periodic structure of the arrival process. Therefore, the arrival pattern influences the performance of the system more than the number of connections. The phase of call within the frame interval F_v is uniformly distributed. Neglecting the periodic behavior of the system leads to exponentially (continuous-time) or geometrically (discrete-time) distributed inter-arrival times between consecutive compressed voice packets from different streams. The mean arrivals within one period is constant in a periodic arrival process. We renounce on that because it is more important to randomize the arrival pattern. These considerations also hold for CS data with a different time frame F_{CS}.

A single trace of compressed voice packets is clearly correlated due to the bursty nature of talk-spurts of a speaker. However, successive packets of a superposition of many packet streams are hardly correlated because they stem from different connections. A well dimensioned queuing system can not even see the effects of the correlation after F_v time because a stable queuing system with periodic input is empty at least once in a period. Hence, we model successively arriving voice packet sizes by an independently and identically distributed (i.i.d.) random variable according to Table 1.

2.3. *Traffic Treatment in QoS Networks*

Voice and CS data are sensitive against loss and delay and require QoS guarantees for the transport from the network. In the recent years, the Integrated Services (IntServ) [30] and Differentiated Services (DiffServ) [4] approach have enhanced the IP technology with real-time capabilities. For both holds that the network can only provide QoS if it has enough resources for the served data streams. To that aim, the flows declare their traffic descriptors, so that the network can household bandwidth. If all the bandwidth is booked, further customers are rejected by connection admission control (CAC). Traffic conditioners at the network boundaries protect it against overload. They either delay or even drop non-conforming packets when they arrive.

A simple traffic description is a maximum peak rate C. It describes a virtual leased line (VLL) service in the expedited forwarding hop behavior (EF-PHB) [17] in DiffServ. The time between the arrival of a packet of size B and the subsequent one must be at least $\frac{B}{C}$. A spacer may be placed at the network ingress to ensure the conformance of the traffic flows with their traffic descriptors. It stores packets that are in advance and releases them only when they are in time. If the amount of stored data exceeds a threshold, further IP packets are dropped.

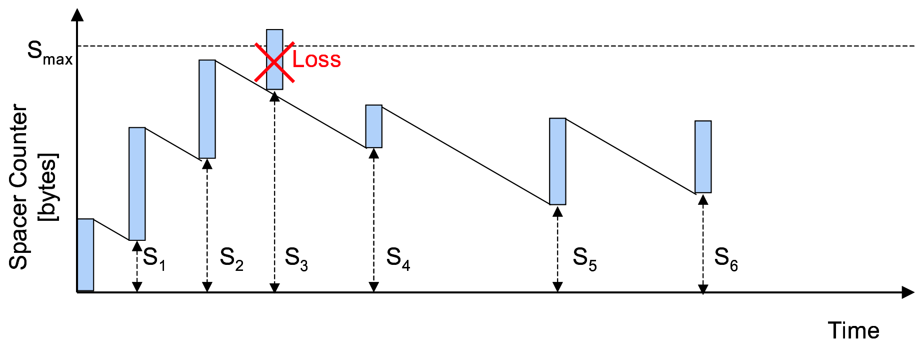

Fig. 2. State evolution of a spacer.

We model the spacer with a spacer counter S that shows the virtual occupancy of its queue. It is decreased linearly by the link rate C over time but does not fall short of zero. When an IP packet of size B arrives, it is accepted if the counter S plus the size of the new packet do not exceed the limit of the spacer buffer S_{max}. In this case, the counter is increased by B bytes and the packet is sent after $\frac{S}{C}$ time. Otherwise, the IP packet is discarded. Increasing the spacer buffer reduces the loss probability but the packet delay is an immanent consequence of the spacer and a property of the flow. We assume in the following that all packet losses and delays are caused by a spacer.

2.4. *Protocol Suites for Real-Time Transport*

In UMTS, real-time packets are transported between different network entities, e.g., from the RNC to the UMSC. The transport technology in UMTS is still ATM and AAL2 is employed to make the transmission more efficient. In IP networks, compression techniques can be used to overcome the large header overhead. Finally, we present various options for low-bitrate real-time traffic multiplexing and give an outlook on RTP switching.

2.4.1. *Real-Time Transport in ATM Networks*

ATM networks have fixed size transport units (cells) that consist of 48 bytes payload and 5 bytes header. When small real-time packets are tunneled, i.e., each of them is carried in a single cell, the transmission is inefficient due to wasted payload. If the packets have an average size of 20 bytes, the overall ATM overhead amounts to 165%. The ATM Adaptation Layer Type 2 (AAL2) [16] has been introduced to reduce that overhead. Fig. 3 illustrates AAL2. An ATM cell begins with a starter field (STF) that carries an offset field (OSF), a sequence number (SN), and a parity bit (P), which try to maintain consistency in case of cell losses and bit errors.

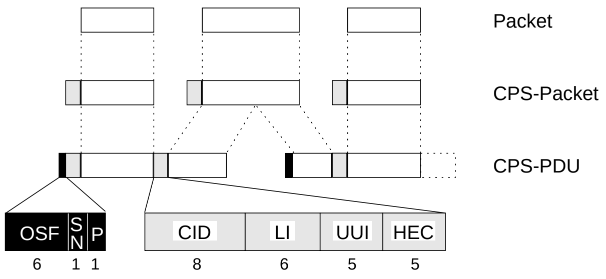

Fig. 3. Structure of the AAL2 protocol.

The 3 bytes common part sublayer (CPS) packet header carry mainly a connection identifier (CID) to distinguish the different flows (Fig. 4) and a length indicator (LI) to delineate the CPS packet boundaries. Note that a packet may be split over two cells. The first CPS packet within a cell needs to wait for cell completion by later arriving packets. To avoid extensive delays, the multiplexing process is controlled by a timer common usage. We denote the value for this timer by TCU. Under heavy load, the multiplexing process is likely to be stopped by cell completion while in a lightly loaded system, its end is mostly triggered by the timer.

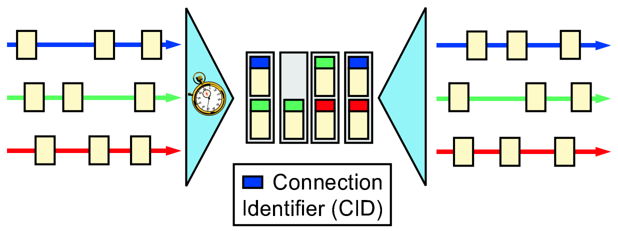

Fig. 4. Packet flow distinction using the connection identifier (CID).

2.4.2. *Real-Time Transport in IP Networks*

We illustrate several options for real-time transmission in IP networks. First, we present the conventional RTP tunneling approach. Header compression can be done on a link-by-link basis. We discuss various multiplexing techniques and propose RTP switching.

RTP Tunneling. For real-time transport in the Internet, the Real-Time Transport Protocol (RTP) is used. The RTP header comprises 12 bytes. It carries a synchronization source identifier (SSRC), a timestamp, a sequence number, and some flags. It provides information to resynchronize different streams within an application. The port numbers of the communicating applications at the sender and the receiver machine are qualified in the User Datagram Protocol (UDP). Its header is 8 bytes large. Moreover, it records the length of the UDP packet and protects it with a checksum against errors. The IP protocol header carries the addresses of the source and the destination machine. In the old IP version 4 (IPv4) [24], the header comprises 20 bytes, thereof 4 octets for each IP address. The address space of IPv4 is likely to run out in the future, especially as soon as an all IP architecture requires lots of end devices. Therefore, the new IP version 6 (IPv6) [9] spends 16 octets per IP address and has a header size of 40 bytes. We use this alternative in our investigation. The RTP/UDP/IP protocol suite amounts to 60 bytes while the average voice packet size is not even 20 bytes large. This results into a protocol overhead of more than 300%.

RTP/UDP/IP Header Compression. In UMTS, traffic is transported between different network entities that perform call processing. These flows are just different streams within the same application in the node (e.g., UMSC) and many of them share the same path. Therefore, their IP headers have almost the same entries. The timestamp and the sequence number steadily change, but the change is almost constant for the packets of the same RTP stream. Header compression associates a connection with every RTP stream. At the beginning, a full header is sent and for the following IP packets, just the first and second order differences of the header to the previous one are encoded [10,8,11]. RTP/UDP/IP header compression works only on a link-by-link basis because of the compressed header. Without an IP header, packets can not be routed through an arbitrary transit IP network.

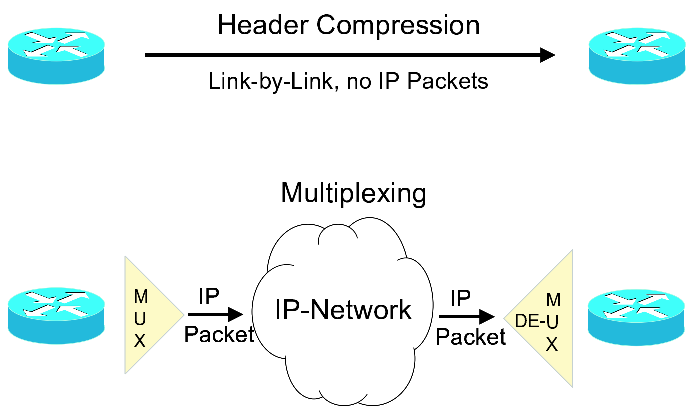

Fig. 5. Header compression versus multiplexing.

A means to overcome this handicap is to apply header compression and to transport several of the compressed packets in a single IP packet to the destination. This is called multiplexing. Fig. 5 shows the advantage of multiplexing techniques in IP over pure header compression. The resulting packet has an ordinary IP header and can be carried transparently through the Internet. There are different options for multiplexing. We classify them according to the aggregating protocol layer.

IP Multiplexing. The Internet Engineering Task Force (IETF) has proposed the Transport Multiplexing Protocol (TMux) [6] to allow multiplexing of several transport protocol units into one IP packet. It is marked with the protocol number 18 to interpret it as an agglomeration of several multiplexed items that we call mini packets in the following. Every mini packet is equipped with a 4 bytes TMux header that contains the length of the following data, the protocol ID, and a checksum that protects this mini header. This approach allows to share a 40 bytes IP header among mini packets that have an TMux/UDP/RTP header of 24 bytes in the UMTS context.

UDP Multiplexing. The same scheme can be performed on application level. This means, that several RTP packets are put into one UDP packet to get rid of the 4 bytes for port numbers. To that aim, 2 bytes data length and 2 bytes data checksum are prepended in front of every RTP header. So, one UDP packet of 48 bytes contains mini packets with 16 bytes muliplex header and RTP header overhead. Note that this is not a standardized solution.

RTP Multiplexing. The present Internet draft for RTP multiplexing [27] is based on an enhanced RTP/UDP/IP header compression algorithm [8,7] and a layer independent multiplexing protocol [23,22]. A further compression step reduces the size of the multiplexing layer [29]. There have been several earlier and simpler application specific proposals in the IETF that define a RTP multiplexing scheme [26,15,5].

Our intention is not protocol verification but we want to show general performance issues for multiplexing of low-bitrate real-time traffic. To that aim, we propose a simple method on which we base the investigation. The entries of the RTP headers are usually different, however, the same information can be coded in fewer bytes. The timestamp is about the same for all mini packets, so it can be shared. The SSRC is 4 bytes and the sequence is 2 bytes long which is more than abundant. We suggest the following mini header. There are 7 bits to indicate the packet length, i.e., mini packets up to 128 bytes can be transported this way. A one bit flag allows the extension of this field for larger packet sizes. 4 bits are spent for sequence numbering and 12 bits record a connection identifier (CID) like in AAL2. Both the sender and the receiver of the mini packets need a lookup table to realize a mapping between the original SSRC and the CID. Therefore, RTP multiplexing is based on a connection concept. After all, the fixed header portion is an RTP/UDP/IP header of 60 bytes and the mini header consists of 3 bytes. Fig. 6 shows an overview of the presented possibilities. All proposed schemes require a timer to limit the multiplexing delay of a mini packet. Unlike in ATM, there is no notion of a full IP packet. So, the multiplexing process is always stopped by the timer. Therefore, the setting for the TCU is crucial to yield good performance while meeting a given delay bound.

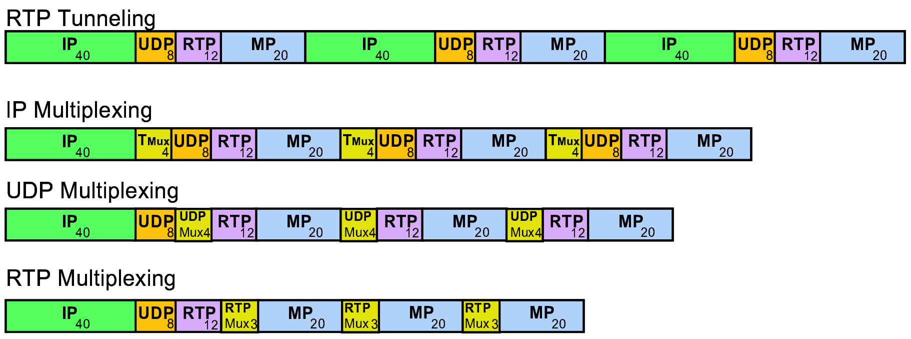

Fig. 6. Transport alternatives for three mini packets.

RTP Trunking and Switching. IP and UDP multiplexing work only for traffic with identical source and destination. RTP multiplexing introduces the concept of a connection for RTP streams. The source and destination do not change rapidly even for mobile communication, therefore, both can be stored in the connection table. The information (source / destination IP number, source / destination UDP port, RTP SSRC) maps to a CID for the multiplexing process and can be restored at the demultiplexing router. Bundled streams do not have to share the whole path like in IP or UDP multiplexing. This kind of trunking allows that more traffic can be multiplexed which leads to a better overhead reduction. The connection concept facilitates also RTP switching, i.e., routers perform demultiplexing and multiplexing. Then, even more streams can be multiplexed on a link. This sounds unreal for implementation costs but, in fact, AAL2 switching [21] is performed in UMTS.

2.5. *Quality of Service Conditions in UMTS Multiplexing Nodes*

The 3GPP considers the maximum transmission delay for real-time data in a forwarding hop [2]. They discuss values between 0.5 msec and 5 msec. In this investigation, we assume a delay budget of 1 msec for multiplexing and spacing. Meeting strictly that delay budget is very hard, therefore, we formulate a new QoS criterion: The probability that the waiting time exceeds the delay budget (DB) must by be smaller than 10^{-4}, i.e., the 99.99% quantile must be smaller than the delay budget. This is illustrated in Fig. 7. W assume that a loss probability of at most 10^{-6} can be tolerated.

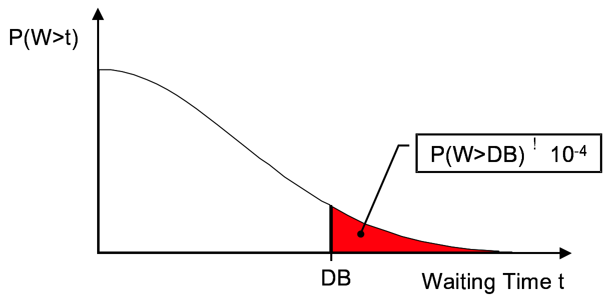

Fig. 7. Quantile of the waiting time.

3. Protocol Analysis

In this section, we derive an analysis for real-time transmission over a QoS enabled IP network that uses a peak rate spacer [18]. We set up a discrete-time Markov model for the tunneling and multiplexing approach and compute its stationary state distribution. This allows for calculating the loss probability, the waiting time distribution, as well as the average protocol overhead for the mini packets. For the sake of simplicity we introduce in Table 2 some notations regarding an arbitrary random variable X.

Table 2. Notation regarding a random variable X.

Notation	Meaning
$\mathcal{X}$	Value space of X,
X_{min}, X_{max}	Minimum and maximum value of X,
$x[i]$	Probability $\Pr(X = i)$,
x	Probability distribution of X
$X(Y)$	Random variable X conditioned on random variable Y,
$x(Y)$	Distribution conditioned on random variable Y,
$\overline{X} = \sum_{i \in \mathcal{X}} x[i] \cdot i$	Mean of X,
$p = \sum_{i \in \mathcal{X}} p(X = i) \cdot x[i]$	Unconditioning probability $p(X = i)$ by X

3.1. *Stationary State Distribution of the Transmission Model*

First, we summarize a method that we employ for computing the stationary state distribution of a discrete-time finite Markov chain. Then, we derive a discrete-time model for the transmission scenario that we have discussed in the previous section. Finally, we determine the required input distributions so that we can calculate the stationary state distribution.

3.1.1. *Computation of the Stationary State Distribution of Discrete and Finite Markov Chains*

We summarize briefly a numerical framework for solving discrete and finite Markov models [20] that we use to find the stationary state distribution of the transmission model. It is a generalized formalization of discrete-time analysis [28].

We describe a finite Markov model by a state variable $X(t)$. The Markov model exhibits the memoryless property at the transition points t_n, $n \in \mathbb{N}_0$ and we abbreviate $X(t_n)$ by X_n. A state transition depends on the present state X_n and some factor $Y(X_n)$. The factor $Y(X_n)$ may depend on the state X_n. The distribution of X_n depends on the transition step n, whereas $Y(i)$, $i \in \mathcal{X}$ is an i.i.d. random variable. The transition behavior is expressed by the state transition function $X_{n+1} = f(X_n, Y(X_n))$. The consecutive state distributions x_n can be computed by Alg. 1. The limit $x = \lim_{n\to\infty} \frac{1}{n+1} \sum_{i=0}^{n} x_i$ yields eventually the stationary state distribution of X. For aperiodic Markov chains, the expression $x = \lim_{n\to\infty} x_n$ leads to the same result but it converges faster.

Alg. 1. Iteration step for the next state distribution.

```
**Input:** state distribution $x_n$ and factor distribution $y(i)$, $i \in \mathcal{X}$
  Initialize $x_{n+1}$ with zeros
  **for all** $i \in \mathcal{X}$ **do**
    **for all** $j \in \mathcal{Y}(i)$ **do**
      $x_{n+1}[f(i,j)] := x_{n+1}[f(i,j)] + x_n[i] \cdot y(i)[j]$
    **end for**
  **end for**
**Output:** $x_{n+1}$
```

We decompose the state transition function into $f = f^{d-1} \circ \cdots \circ f^0$ and introduce X^k and Y^k for $0 \le k < d$, accordingly. Analogously to Alg. 1, x_n^k are calculated. Decomposition helps to decrease the complexity of Alg. 1 to achieve a computation speedup per iteration step. We have developed a compiler that turns the description

$$\mathcal{D} = \Big\{ \{\mathcal{X}^k,\, \{y^k(i),\, i \in \mathcal{X}^k\},\, f^k\},\, 0 \le k < d \Big\} \tag{3.1}$$

syntactically into a numerical program. This program prepares the start vector x_0^0

by a short Markov chain simulation which saves again many iteration steps until convergence.

Simulations are very time consuming when small probabilities are computed. In contrast, the numerical calculation of the presented analysis takes about two minutes on a 500 MHz Pentium III processor. This illustrates the advantage of the analytical approach over simulation.

3.1.2. *Discrete-Time Model for Real-Time Transmission*

We set up a discrete-time Markov model for the transmission scenario that we have discussed in the previous section. We choose the above specified description format $\mathcal{D}$ to characterize it. The framework requires discrete system states as well as discrete factors. The spacer counter S and packet sizes are measured in bytes while time is measured in discrete time units.

We consider the following scenario. During a multiplex interval of length TCU, a multiplexer puts mini packets into an IP packet that is possibly queued in the spacer. We realize that a multiplex interval alternates with an inter-multiplex time. So, we set up two state transition functions f^0 and f^1 that cover these two intervals. The entire model is described by $f = f^1 \circ f^0$ and illustrated in Fig. 8.

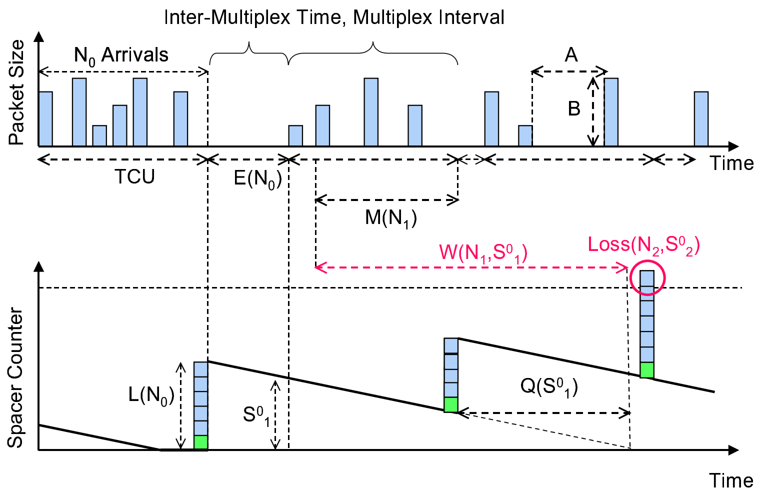

Fig. 8. State transitions in a Markov model for a multiplexer with a subsequent spacer.

The multiplexing process starts at the arrival instant of a first mini packet and is limited by the timer after TCU time units. The multiplexer collects N mini packets and forms an IP packet of size $L(N)$. The spacer counter is denoted by is S^0 at the beginning of the multiplex interval and it is decreased during that time by $TCU \cdot C$ bytes, so that we get $S' = max(S^0 - TCU \cdot C, 0)$. The spacer accepts the new IP packet if the updated spacer occupancy S' plus the size of the new IP packet $L(N)$ do not exceed the spacer size S_{max}. In this case, the spacer counter is increased. Otherwise, the mini packets in the IP packet are lost. The spacer counter and the number of mini packets in the new IP packet compose the state (S^1, N^1) at the end of the multiplex interval. These actions are formalized in Alg. 2.

Alg. 2. Function f^0 covers the multiplex interval

```
**Input:**  state $(S^0)$, number of multiplexed mini packets $N$, and resulting
            IP packet size $L(N)$
  $S' := \max(S^0 - TCU \cdot C, 0)$
  **if** $(S' + L(N) \le S_{max})$ **then**   {IP packet sent}
    $S^1 := S' + L(N)$
  **else**   {IP packet lost}
    $S^1 := S'$
  **end if**
  $N^1 := N$
**Output:**  state $(S^1, N^1)$
```

The inter-multiplex time $E(N)$ depends on how many samples have been collected for the last IP packet. Within that time, the spacer counter S^1 is diminished by $E(N) \cdot C$ resulting S^0. It represent the state of the model at the end of the inter-multiplex time. Alg. 3 abstracts this process.

Alg. 3. Function f^1 covers the inter-multiplex time.

```
**Input:**  state $(S^1, N^1)$ and inter-multiplex time $E(N^1)$
  $S^0 := \max(S^1 - E(N^1) \cdot C, 0)$
**Output:**  state $(S^0)$
```

We are almost done with the formal description $\mathcal{D}$ of the Markov model. The state spaces $\mathcal{X}^0$ and $\mathcal{X}^1$ are given by $\{0 \le i \le S_{max}\}$ and $\{0 \le i \le S_{max}\} \times \{0 < j \le A_{max}\}$. The state transition functions f^0 and f^1 are specified in Alg. 2 and Alg. 3. Finally, we need to derive the distributions for N with $\mathcal{N} = \{0 < i \le \lfloor \frac{TCU}{A_{min}} \rfloor + 1\}$ as well as $l(i)$ and $e(i)$ for $i \in \mathcal{N}$. Then, the stationary state distribution s^0 can be computed using the presented framework.

3.1.3. *Specification of Input Distributions*

We review the source model to specify the distribution for the inter-arrival time A and the packet size B. Furthermore, we need the distributions for N, $L(N)$, and $E(N)$ as input for the analysis to get numerical results for the stationary state distribution s^0. We specify these distributions in the case of tunneling and multiplexing.

Source Model. The source model is given by a geometrically distributed inter-arrival time A of consecutively arriving mini packets. However, we also wish an i.i.d. inter-arrival time with variable coefficient of variation to conduct sensitivity studies about the source model. In case of compressed voice traffic, the mini packets size B is characterized by the histogram shown in Table 1. CS data have constant packet size, so, the distribution is trivial. When we consider a mix of voice and CS data, we interpolate their distributions according to the traffic mix.

The required bandwidth of a voice (CS data) user is $C_v = \frac{\overline{B_v}}{F_v}$ $\left(C_{CS} = \frac{\overline{B_v}}{F_v}\right)$. We denote the bandwidth utilization by user data with ρ^* and refer to it as the offered load. The percentage of bandwidth that is used by voice (CS data) users is c_v $\left(c_{CS} = 1 - c_v\right)$. The overall bandwidth C leads to the number of voice (CS data) users $n_v = \frac{\rho^* \cdot c_v \cdot C}{C_v}$ $\left(n_{CS} = \frac{\rho^* \cdot c_{CS} \cdot C}{C_{CS}}\right)$. The mean inter-arrival time between consecutive voice (CS data) packets is then $\overline{A_v} = \frac{F_v}{n_v}$ $\left(\overline{A_{CS}} = \frac{F_{CS}}{n_{CS}}\right)$. That yields an overall arrival rate of $\frac{1}{\overline{A}} = \frac{1}{\overline{A_v}} + \frac{1}{\overline{A_{CS}}}$. Consequently, the probability that an arbitrary packet contains voice (CS data) is $p_v = \frac{\overline{A}}{\overline{A_v}}$ $\left(p_{CS} = \frac{\overline{A}}{\overline{A_{CS}}}\right)$. We find the overall packet distribution $b = p_v \cdot b_v + p_{CS} \cdot b_{CS}$ by weighting the packet size distributions b_v and b_{CS} with these probabilities.

Tunneling. The conventional way to carry real-time packets over the Internet is RTP tunneling. It can also be investigated by the presented analysis by setting the distribution parameters in an appropriate way. Every mini packet is equipped with an RTP/UDP/IP header and carried separately over the network, i.e., N is always one. We have $H^{fix} = H_{IP} + H_{UDP} + H_{RTP} = 60$ bytes and $H^{mux} = 0$, so that the packet length equals $L = H^{fix} + B$. The multiplex interval is zero time units long, the multiplexing delay M is zero, and the distribution of the inter-multiplex time E equals the distribution of the inter-arrival time A.

Multiplexing. For multiplexing, the required distributions of N, $L(N)$, and $E(N)$ are not so easy to determine. We derive them from the mini packet inter-arrival time distribution a and the mini packet size distribution b. We start with the distribution of the number of mini packets in an IP packet. Exactly i mini packet arrivals happen during the multiplex interval if the sum of $i-1$ inter-arrival times is shorter than the multiplex interval and if the sum of i inter-arrival times exceeds it. Hence, the probability $n[i]$ is defined by the condition

$$\sum_{0 \leq j < i-1} A_j \leq TCU < \sum_{0 \leq j < i} A_j. \tag{3.2}$$

The IP packet size $L(N)$ is the sum of a fixed header H^{fix}, the sizes of the contained mini packets B_j $(0 \leq j < N)$, and the corresponding multiplexing headers H^{mux}. The values for H^{fix} and H^{mux} depend on the chosen protocol suite and are given in Table 3. The IP packet size is finally given by the conditional random variable

$$L(N) = H^{fix} + \sum_{0 \leq i < N} (H^{mux} + B_j). \tag{3.3}$$

The inter-multiplex time is the duration from a timeout until the next mini packet arrives. In other words, it is the remainder of the last inter-arrival time after the

Table 3. Header overhead for different multiplexing techniques in [byte].

Option	H^{fix}	H^{mux}
IP_Mux	$H_{IP} = 40$	$H_{IP}^{mux} + H_{UDP} + H_{RTP} = 24$
UDP_Mux	$H_{IP} + H_{UDP} = 48$	$H_{UDP}^{mux} + H_{RTP} = 16$
RTP_Mux	$H_{IP} + H_{UDP} + H_{RTP} = 60$	$H_{RTP}^{mux} = 3$

termination of the previous multiplexing process. Given that $N = j$ packets have arrived in the last multiplex interval, the distribution of $E(j)$ is determined by the condition

$$\Big(E(j) = \sum_{0 \le i < j-1} A_i - TCU\Big) \wedge \Big(\sum_{0 \le i < j-1} A_i \le TCU < \sum_{0 \le i < j} A_i\Big). \tag{3.4}$$

The inter-multiplex time equals the inter-arrival time if it is geometrically distributed.

3.2. *Performance Measures*

We derive the loss probability for the mini packets p_{loss}^{mini}, their waiting time distribution w, and their protocol overhead o using the stationary state distribution s^0 of the spacer counter. We compute these performance measures in the case of tunneling and multiplexing.

3.2.1. *Tunneling*

We start the derivation for RTP tunneling since the equations are easier to derive.

Loss Probability. A mini packet is lost in the tunneling scenario if the resulting IP packet size L exceeds the spacer counter, i.e., if $S_{max} < S^0 + L$ is true . Therefore, we can calculate the loss probability by

$$p_{loss}^{mini}(S^0) \quad = \sum_{k+S^0 > S_{max}} l[k]. \tag{3.5}$$

Unconditioning by S^0 yields p_{loss}^{mini}.

Protocol Overhead. The header overhead $o = \overline{H_{sent}}/\overline{G_{sent}}$ is the quotient of the mean sent protocol header size $\overline{H_{sent}}$ and the mean sent payload size $\overline{G_{sent}}$ of an IP packet. The overhead is constantly $H = H^{fix}$, so the mean sent overhead is $\overline{H} = (1 - p_{loss}^{mini}) \cdot H^{fix}$ while $\overline{G_{sent}}(S^0)$ depends on S^0 because of the loss probability.

$$\overline{G_{sent}}(S^0) \quad = \quad (1 - p_{loss}^{mini}(S^0)) \cdot \sum_{k+S^0 \le S_{max}} l(S^0)[k] \cdot (k - H^{fix}) \tag{3.6}$$

We get $\overline{G_{sent}}$ if we uncondition by S^0.

Waiting Time. The waiting time of a tunneled mini packet is the queuing time in the spacer

$$Q(S^0) \quad = \quad \frac{S^0}{C}. \tag{3.7}$$

Waiting applies only to sent packets. Therefore, we compute its probability distribution by

$$w[r] = \frac{1}{1-p_{loss}^{mini}} \cdot \sum_{i \in \mathcal{S}^0} q(i)[r] \cdot (1-p_{loss}^{mini}(i)) \cdot s^0[i]. \tag{3.8}$$

3.2.2. *Multiplexing*

The equations for p_{loss}^{mini}, o, and w are more complex for the multiplexing scenario. In principle, they even subsume the tunneling case.

Loss Probability. We consider the multiplexing start. The next IP packet is ready for spacing after TCU time. In the meantime, the spacer counter is diminished to $S' = \max(S^0 - TCU \cdot C, 0)$ bytes. If the updated counter value S' plus the freshly arrived IP packet with $L(N)$ bytes exceed the spacer capacity S_{max}, the IP packet is discarded. Hence, the IP packet loss probability $p_{loss}^{IP}(S^0, N)$ of an IP packet depends on the spacer counter S^0 at multiplexing start and on the number N of contained mini packets.

$$p_{loss}^{IP}(S^0, N) = \sum_{k+\max(S^0-TCU\cdot C,0)>S_{max}} l(N)[k] \tag{3.9}$$

The loss probability $p_{loss}^{mini} = \overline{N_{lost}}/\overline{N}$ of a mini packet is the average number $\overline{N_{lost}}$ of contained mini packets in a lost IP packet divided by the average number $\overline{N}$ of mini packets that are in a sent or lost IP packet.

$$\overline{N_{lost}}(S^0, N) = p_{loss}^{IP}(S^0, N) \cdot N \tag{3.10}$$

$$\overline{N}(N) = N \tag{3.11}$$

Unconditioning them by N and S^0 yields $\overline{N_{lost}}$ and $\overline{N}$.

Protocol Overhead. The overhead $o = \overline{H_{sent}}/\overline{G_{sent}}$ is defined in the same way as in case of tunneling. The protocol header size $H(N)$ of an IP packet is the sum of the fixed header size H^{fix} and the multiplex header sizes H^{mux}. Its overall payload size $G(N)$ is the IP packet size $L(N)$ without the header $H(N)$.

$$H(N) = H^{fix} + H^{mux} \cdot N \tag{3.12}$$

$$G(N) = L(N) - H(N) \tag{3.13}$$

The number of mini packets in an IP packet influence the loss probability. Therefore, $H_{sent}(S^0, N)$ and $G_{sent}(S^0, N)$ depend on S^0 and N. Again, unconditioning by N and S^0 yields $\overline{H_{sent}}$ and $\overline{G_{sent}}$.

$$\overline{H_{sent}}(S^0, N) = (1 - p_{loss}^{IP}(S^0, N)) \cdot H(N) \tag{3.14}$$

$$\overline{G_{sent}}(S^0, N) = \sum_{k+\max(S^0-TCU\cdot C,0)\leq S_{max}} l(N)[k] \cdot G(N). \tag{3.15}$$

Waiting Time. The mini packet waiting time $W(S^0, N) = M(N) + Q(S^0)$ consists of the multiplexing delay $M(N)$ and the queuing delay $Q(S^0)$ in the spacer. It can only be computed for packets that are not lost. The multiplexing delay M that a mini packet encounters is the time from its arrival instant until the end of the multiplex interval. It depends on the number of multiplexed mini packets in the IP packet: If there is only one multiplexed mini packet, it is clear that the multiplexing delay is TCU; if there are more than one, the average multiplexing delay is shorter. The distribution for $M(i)$ is determined by the condition

$$\Big(M(i) \;=\; TCU - \sum_{0<k<i} \sum_{j=1}^{k} A_j\Big) \wedge \Big(\sum_{j=1}^{i-1} A_j \leq TCU < \sum_{j=1}^{i} A_j\Big). \tag{3.16}$$

Next, we compute the queuing delay. The spacer counter is reduced to $S' = \max(S^0 - TCU \cdot C, 0)$ at the end of the multiplex interval. At this time, the IP packet gets possibly accepted for transmission and encounters a conditional spacing time of

$$Q(S^0) \;=\; \frac{\max(S^0 - TCU \cdot C, 0)}{C}. \tag{3.17}$$

The probability that a mini packet waits $W = r$ time units is the quotient

$$w[r] = \frac{\overline{N_W}(r)}{\overline{N_{sent}}}. \tag{3.18}$$

$\overline{N_W}(r)$ is the average number of mini packets in an IP packet that have waited exactly r time units. It depends on N and S^0, and pertains only to sent packets

$$\begin{aligned} \overline{N_W}(r, N = i, S^0, N) \;&=\; (1 - p_{loss}^{IP}(S^0, N)) \cdot N \cdot \\ &\qquad \sum_{\{(i,j)|r=k+j\}} m(N)[i] \cdot q(S^0)[j]. \end{aligned} \tag{3.19}$$

Unconditioning by N and S^0 yields $\overline{N_W}(r)$. The numerical program takes advantage of the fact that the distribution of $Q(S^0)$ takes only the values 0 and 1. The average number of sent mini packets in an IP packet may be calculated by

$$\overline{N_{sent}} = \overline{N} - \overline{N_{lost}}. \tag{3.20}$$

4. Results

In UMTS, many low-bitrate real-time streams share a common path. Therefore, multiplexing techniques can be applied in IP networks to avoid the extensive protocol overhead due to the RTP/UDP/IP header. The transmission requires QoS guarantees but if the booked bandwidth C is too tight, loss and delay occur due to peak rate spacing. In this section, we study the performance of different transmission alternatives. We investigate traffic with different characteristics and study the influence of various parameters [19]. We will first consider the transmission of voice, then CS data, and, finally, we will investigate traffic mixes of voice and CS data.

4.1. *Voice Transmission*

First, we explain the performance measure "critical load" and illustrate the the composition of the waiting time. We compare several protocol options regarding their protocol overhead and critical load. We show the existence of an optimum length for the multiplex interval and perform a sensitivity studies regarding the traffic characteristics. Finally, we investigate the influence of the delay budget on the protocol performance.

4.1.1. *QoS Behavior*

We consider RTP multiplexing ($TCU = 0.5$ msec) over an 8 Mbps link that carries voice traffic. We vary the offered net user load ρ^*. We can always set the spacer buffer so large that the maximum mini packet loss probability of 10^{-6} is met as long as the gross traffic rate does not exceed the bandwidth. Therefore, only the delay criterion is crucial for the maximum protocol performance in terms of transported net user traffic. Fig. 9 shows that the 99.99% quantile hits the delay

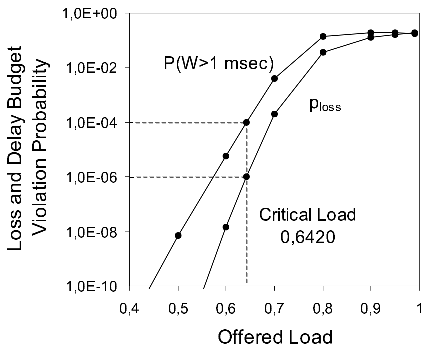

Fig. 9. The critical load.

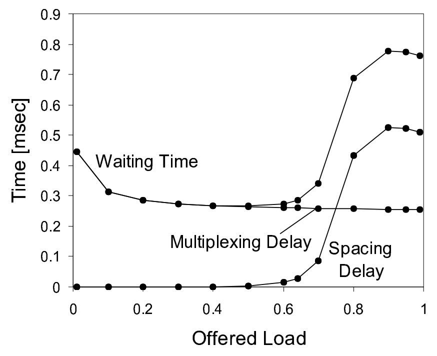

Fig. 10. The contribution of multiplexing and spacing delay to the waiting time.

budget at an offered load of 64.2%. We call this the critical load, no more traffic can be supported. The probability that the delay criterion is violated increases exponentially with the offered load ρ^*. In contrast, the mean waiting time of the mini packets starts at 0.5 msec in Fig. 10. First, it decreases and it increases only when ρ^* approaches the critical load. The curve for the mean waiting time is typical for a batch server with timeout. If very little traffic is carried, the probability $\Pr(A > TCU)$ is large. It is likely that there is only a single mini packet in the multiplexed IP packet. This packet faces the entire multiplex interval as waiting time. When the arrival rate is high, the mini packets are almost equally distributed over the multiplex interval and the mean multiplexing delay is $\frac{TCU}{2}$. The mean queuing time in the spacer is low for a weakly loaded system and increases strongly as soon as the critical load is exceeded. The addition of the multiplexing and spacing delay yields the overall mini packet waiting time.

The AAL2 protocol in ATM networks provokes a different waiting time behavior. ATM cells are small so that the end of the multiplexing process is mostly triggered by cell completion for a reasonable timer setting. The timer value has hardly influence on the multiplexing delay in a heavily loaded system. So, the queuing behavior of an AAL2 multiplexing system is different from the one of a batch server that is only triggered by a timeout.

4.1.2. *Protocol Performance of Different Tunneling and Multiplexing Techniques*

Tunneling techniques like the RTP/UDP/IP suite have constant overhead. In this case, it is so large that the network resources are not efficiently used. Multiplexing techniques are applied to reduce the protocol overhead. The multiplexed mini packets share the common header H^{fix} and the protocol performance improves with the number of multiplexed packets. Therefore, we vary the bandwidth in the next experiment to support more connections that contribute to the overhead reduction. The multiplexing timer is again set to 0.5 msec.

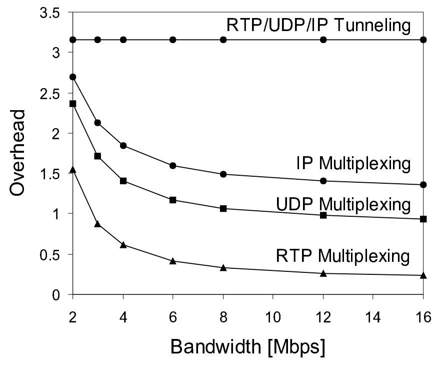

Fig. 11. Protocol overhead reduction due to multiplexing.

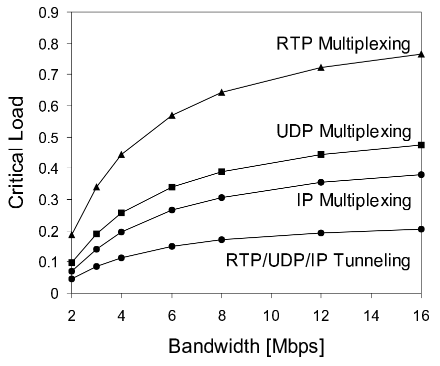

Fig. 12. Increased link utilization by user data due to multiplexing.

Fig. 11 shows that the multiplexing techniques clearly reduce the protocol overhead. The simple Tmux protocol reduces the overhead to less than half. Multiplexing based on the UDP layer is also simple and reduces it to a third. RTP multiplexing is more complex but it reduces the overhead even to 7.5% compared to RTP tunneling. It is crucial for the design of a multiplexing protocol that the multiplex header size H^{mux} is small even at the expense of the common header size H^{fix}. Due to economy of scale, the network resources can be better utilized at higher bandwidth without compromising the spacing time. So, the amount of overall transported traffic with tolerable IP packet waiting times is similar for all protocols. But the proportion of user data ($\frac{1}{1+overhead}$) in an IP packet is different. Therefore, almost four times more user traffic can be carried with RTP multiplexing than with tunneling. According to Fig. 13 , only 6 Mbps are needed to support 450 voice connections with RTP multiplexing instead of 16 Mbps with RTP tunneling. Therefore, we will use RTP multiplexing for the rest of this investigation.

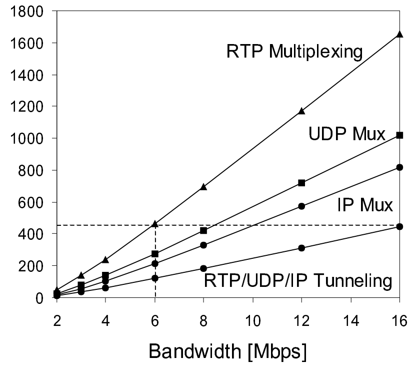

Fig. 13. Capacity differences due to protocol design.

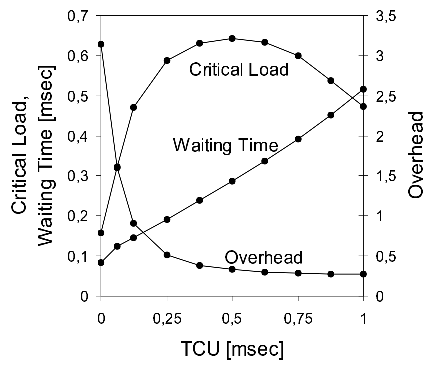

Fig. 14. The optimum timer value for delay budget of 1 msec.

4.1.3. *Optimum Timer Value*

It is clear that the multiplexing timer value TCU may be set at most as large as the delay budget. We conduct a case study on an 8 Mbps link and vary the timer value. Fig. 14 shows that the protocol overhead decreases with increasing TCU because the number of multiplexed mini packets in an IP packet is almost proportional to the length of the multiplex interval. This has a good influence on the critical load because the proportion of net user data increases.

On the other side, a larger timer value means longer multiplexing delay and larger IP packets. The large IP packets impair the burstiness of the resulting IP packet stream and deteriorate the spacing time. In addition, the allowed spacing time for the first multiplexed mini packet is $Q = DB - TCU$, the delay budget minus the duration of the multiplex interval. Both the increased burstiness and the shorter spacing time have negative effect on the delay criterion and on the critical load.

Hence, the TCU value trades good queuing characteristics versus overhead reduction so that we can observe a maximum critical load for a timer value of 0.5 msec. The strong influence of the timer value on the protocol performance is a fundamental difference between multiplexing systems in IP and ATM. In ATM networks, the timer limits the AAL-2 multiplexing time only rarely under heavy load because the small ATM cells are filled before a timeout occurs [14,25]. It has hardly any effect provided that it is set sufficiently large. Once a cell is filled, the overhead can not be further reduced and, therefore, the above tradeoff does not exist.

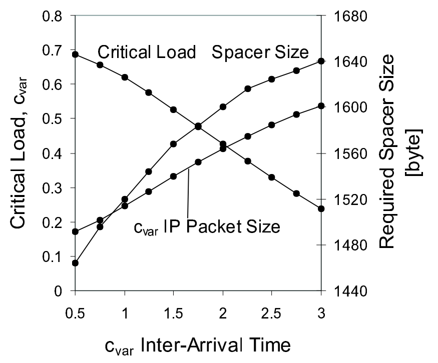

Fig. 15. The sensitivity to the source model.

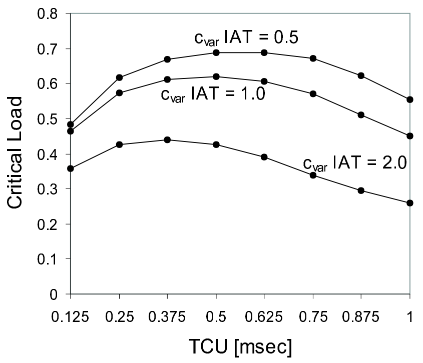

Fig. 16. The influence of the source model on the optimum timer value.

4.1.4. *Sensitivity to the Arrival Process*

So far, we have assumed that the inter-arrival time between successively arriving voice samples of an aggregate stream is geometrically distributed. We interchange this assumption with a distribution for which we control the mean and the coefficient of variation. We perform the following experiments on an 8 Mbps link. Fig. 15 shows that the variability in the arrival process influences the outcome of the multiplexing process. The variability in the inter-arrival time translates into variability of IP packet size. The burstiness of the resulting IP traffic increases with the inter-arrival time variability. This deteriorates the spacing properties, i.e., more spacer buffer is required to avoid losses. There is also more variability in the spacing time, so, it is harder to meet the delay criterion for QoS, and consequently, the critical load falls. This result shows that source modeling is crucial for the accurate estimation of delay quantiles, e.g., for the construction of a CAC.

Fig. 16 shows the impact of the coefficient of variation on the optimum timer value. We tested inter-arrival times with a coefficient of variation of 0.5, 1.0, and 2.0. The maximum number of connections can be admitted for a timer value of 0.625, 0.5 and 0.375 msec. These differences are clearly to observe but a timer value of 0.5 msec yields in all cases an acceptable system performance. The optimum timer value depends on the variability of the inter-arrival time. We already realized that the variability of the inter-arrival time propagates into variability of spacing time. The probability that the spacing time exceeds $DB - TCU$ rises with this variability. If the spacing delay takes sometimes longer, the multiplexing delay needs to be shorter to meet the delay budget. Hence, more variable IP packet sizes require a shorter multiplex interval. The variability of the inter-arrival time influences the spacing time which affects the optimum timer value.

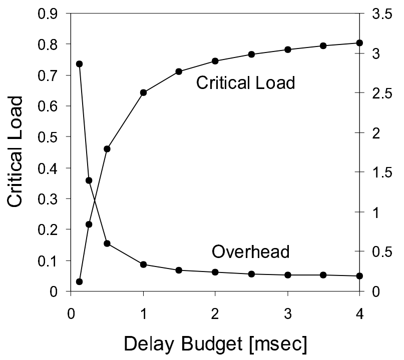

Fig. 17. Influence of the delay budget on the critical load.

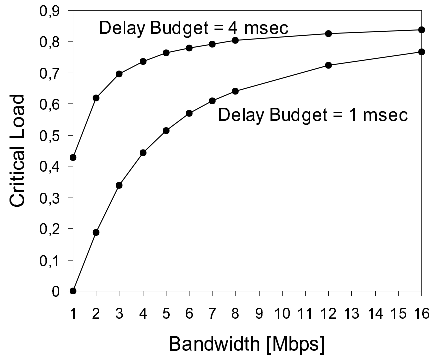

Fig. 18. The influence of the delay budget depends on the bandwidth.

4.1.5. *Influence of the Delay Budget*

As shown above, the delay budget is the criterion that limits the critical load and is by definition a crucial parameter for the protocol performance. We assume an 8 Mbps link and set the TCU to half the delay budget. With this setting, the delay budget controls the number of multiplexed mini packets. Fig. 17 illustrates how the overhead decreases for larger delay budgets. The net user data proportion on the link leads to a higher critical load. In addition, a longer delay budget allows also for longer spacing times which also benefits the performance.

We would like to know the influence of the delay budget on the optimum timer value. Numerical results show that a TCU of 2 msec as the optimum timer value for a delay budget of 4 msec. A system with a large delay budget seems to be very robust with respect to the multiplexing timer. Values between 1 and 3 msec do not change the critical load.

Moreover, we would like to know the impact of the delay budget depending on the bandwidth. To that aim we choose a delay budget of 1 and 4 msec and set in both cases the optimum TCU. Fig. 18 depicts the critical load depending on the bandwidth. It is trivial that the larger delay budget allows for a higher critical load. Furthermore, we observe that the difference is stronger for small bandwidth than for large bandwidth. This proposes to afford a large delay budget on slow access links and a smaller one for core networks with higher link speeds.

4.2. *Transmission of Circuit Switched Data*

In this section, we consider only the transmission of CS data traffic. As mentioned above, a 64 kbps data stream must be partitioned into packets. The data frame length $F_{CS} = 125 \frac{\mu sec}{byte} \cdot B$ is directly proportional to the packet length B. The packet arrival rate of a single CS connection is $\lambda = \frac{1}{F_{CS}}$. The superposition of n_{CS} CS connections has a packet arrival rate of $\lambda = \frac{n_{CS}}{F_{CS}}$ and a mean inter-arrival time of

$\overline{A} = \frac{F_{CS}}{n_{CS}}$. Consecutive packet arrivals are assumed to be geometrically distributed inter-arrival time in the model. Its variability of is calculated by

$$c_{var}(A) = \sqrt{\frac{\overline{A}-1}{\overline{A}}} = \sqrt{1 - \frac{n_{CS}}{F_{CS}}}. \tag{4.21}$$

If the data frame length decreases, the coefficient of variation shrinks. Also a growing number of active connections reduces the variability of the traffic. Hence, the packet length is a parameter that influences on the variability of the source model. As we have seen above, this is a crucial factor for the performance behavior of the system, and so, we are interested in its impact on the critical load.

4.2.1. *Optimum Packet Size*

We set the TCU to 0.5 msec and vary the packet length from 4 to 160 bytes on an 8 Mbps link. Fig. 19 illustrates the existence of an optimum data packet length. A length of 16 bytes is the best but packet lengths between 4 and 60 bytes yield still a reasonable bandwidth utilization by user data. Larger data packet size cause more variability in the IP packet size. This effect is similar to the one in Fig. 15 for which we have argued that the variability of the traffic impairs the queuing behavior in the spacer, thus, the critical load. However, larger CS data packet size means automatically less protocol overhead because the payload size is larger compared to the constant header size. An increased net user data proportion in the carried traffic influences the protocol performance positively. Good queuing properties are traded for overhead reduction. This tradeoff is not specific to multiplexing but for CS data.

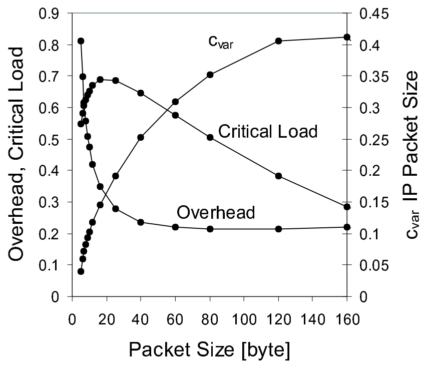

Fig. 19. The optimum CS data packet size for multiplexing.

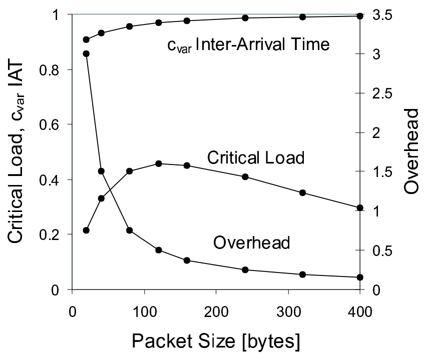

Fig. 20. The optimum CS data packet size for tunneling.

For tunneling, we observe in Fig. 19 an optimum packet length of 120 bytes and the critical load was in a tolerable range for packet sizes between 80 and 250 bytes. Note that for a CS packet size around 150 bytes, tunneling is clearly better than multiplexing, at least for this parameter setting.

We conclude from these results that an efficient CAC algorithm needs to take both the transmission rate and the packet size of CS data streams into account to guarantee a maximum delay. Moreover, for large CS data packets there is no performance gain by applying multiplexing.

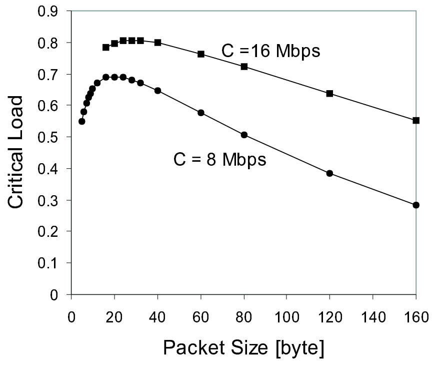

Fig. 21. Dependency of the optimum packet length on the bandwidth.

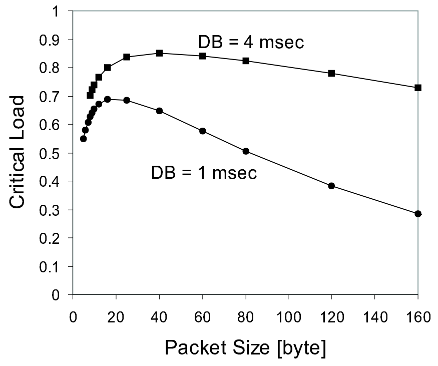

Fig. 22. Dependency of the optimum packet length on the delay budget.

When we double the bandwidth, it is interesting to observe that the optimum packet length increases. It is still around 28 bytes and the performance degradation with longer CS data packets is notable in Fig. 21. We trace this back to the fact that there is more economy of scale at larger bandwidth. So, the spacing delay shrinks and the effect of overhead reduction by longer CS data packets becomes more important.

The delay budget influences the optimum CS data packet length, too. Fig. 22 exhibits that for a delay budget of 4 msec, the maximum critical load is obtained for a packet length of 40 bytes but 16 bytes yields also good results. Furthermore, we observe that the influence of the packet length reduces for a larger delay budget but it is still noticeable.

RNCs are usually connected by relatively expensive leased lines with a bandwidth in the order of several Mbps. They determine the length of the CS data packets and should be aware of the fact that smaller CS packets can be transported more efficiently over the network than bigger ones. Therefore, they should adapt the CS packet length rather to 40 than to 160 bytes.

4.2.2. *Optimum Timer Value*

We argued that the data packet size influences the variability of the resulting IP packets. We already know that the variability in the arrival process affects the optimum timer value TCU. We study this on an 8 Mbps link with packet sizes of 8, 32, and 80 bytes. The results in Fig. 23 show that the optimum timer for CS data packets of 40 bytes is about 0.5 msec, which is the same as for voice traffic. For smaller packet sizes, the timer should be larger, and for larger packet sizes vice versa. Although the maxima are clearly to identify, a timer value of 0.5 msec is a good compromise for all packet sizes.

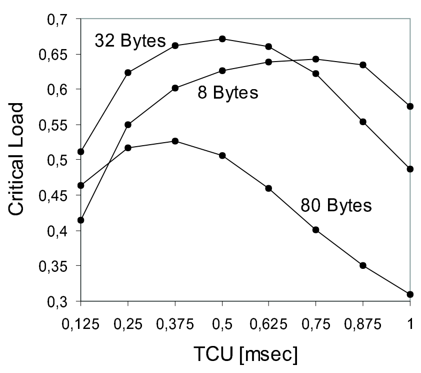

Fig. 23. The influence of the CS data packet length on the optimum timer.

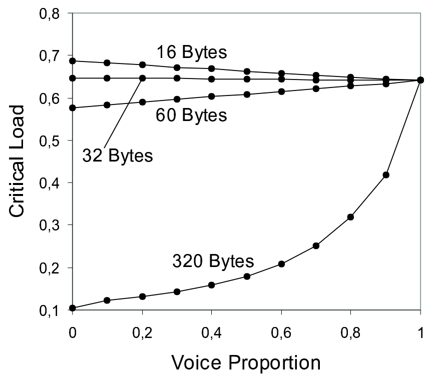

Fig. 24. The critical load for traffic mixes.

4.3. *Transmission of Traffic Mixes*

So far, we have considered the transmission of either only voice or only CS data traffic. In UMTS, however, there will be traffic mixes of voice and CS data. A voice proportion of p_v means that p_v of the net user traffic belongs to voice connections. Again, we experiment on an 8 Mbps link and set the TCU to 0.5 msec. Fig. 24 exhibits that packet sizes of 16, 32, and 60 bytes yield a reasonable performance for pure CS data. Their critical load of the traffic mixes interpolates linearly with the critical load of pure voice. But for a CS packet size of 320 bytes, the resource utilization degrades more than linearly with increasing CS data proportion.

If CS data streams with large packet sizes can tolerate a larger delay budget, we can improve the performance of the system. We introduce service differentiation: CS data and voice streams are multiplexed into different IP packet flows. The CS IP packet stream is carried with lower priority than the voice IP packet stream. In this way, CS data hardly affect the transmission of voice. This is a means to alleviate the performance degradation due to CS data with large packet sizes. A similar idea for AAL2 is called path selection.

5. Summary

Low-bitrate real-time traffic can not be transported efficiently in IP networks with the conventional RTP/UDP/IP tunneling approach. The protocol overhead is very large compared to the payload size of the sent mini packets. This leads to a bad bandwidth utilization by user data. We presented several alternatives to overcome that problem, e.g., header compression and various multiplexing techniques. For our investigation, we considered both tunneled and multiplexed transmission. The sent IP packets pass a spacer that delays or even drops them if they are not conform to the declared traffic descriptors. A source traffic model was developed for voice and CS data in UMTS. We introduced a framework for the computation of the stationary state distribution of a discrete and finite Markov chain and established a Markov model for the transmission scenario. We derived a discrete-time analysis to compute numerical results. The limiting criterion for the system load is the mini packet waiting time due to multiplexing and spacing delay. It must not exceed a delay budget of 1 msec with a probability of more that 10^{-4}. The maximum load in terms of net user data is the performance measure. We call it the critical load and use it compare transmission alternatives.

We showed that multiplexing low-bitrate real-time data over IP networks yields a multiple of transmission capacity compared to the tunneling alternative. RTP multiplexing turned out to be most efficient. To realize the multiplexing gain, many flows are necessary. Therefore, this approach is interesting for scenarios like in UMTS where a large amount of low-bitrate real-time traffic is transported. The delay budget limits the bandwidth exploitation and has most impact on links with little bandwidth. We illustrated the existence of an optimum length for the multiplex interval that maximizes the critical load. It is sensitive to the variability of the traffic depends on the length of the CS packet size. If the timer value is not appropriately set, the tunneled transmission can even outperform the multiplexing scheme for large packet sizes. There are also optimum CS packet sizes for the tunneling and for the multiplexing scenario that maximize the performance.

Finally, we found that multiplexing low-bitrate real-time traffic can make the transmission up to four times more efficient. However, as soon as CS data come into play, parameters must be set very carefully to optimize the critical load. Multiplexing makes the data forwarding more complex and special purpose router are more expensive. In ATM networks, however, AAL2 is state of the art and it is even mandatory in the present specification of UMTS. The future will show whether multiplexing low-bitrate real-time traffic will also be adopted in IP networks or not.

References

1. 3GPP. 3G TR23.922 version 1.0.0: Architecture for an All IP Network, Oct. 1999.
2. 3GPP. TSGR3#7(99)c05: Study Item (ARC/3) Overall Delay Budget within the Access Stratum. Status Report, Sep. 1999.
3. Bavarian Archive for Speech Signals (BAS). Verbmobil 6.1. http://www.phonetik.uni-muenchnen.de, 1996.
4. S. Blake, D. L. Black, M. A. Carlson, E. Davies, Z. Wang, and W. Weiss. RFC2475: An Architecture for Differentiated Services. ftp://ftp.isi.edu/in-notes/rfc2475.txt, Dec. 1998.
5. G. v. Bochman, G. Luo, and M. K. El-Khatib. Multiplexing Scheme for RTP Flows between Access Routers. http://www.ietf.org/internet-drafts/draft-ietf-avt-multiplexing-rtp-00.txt, June 1999.
6. P. Cameron, D. Crocker, D. Cohen, and J. Postel. RFC1692: Transport Multiplexing

Protocol (TMux). http://www.ietf.org/rfc/rfc1692.txt, Aug. 1994.

7. S. Casner, V. Jacobson, T. Koren, B. Thompson, D. Wing, P. Ruddy, A. Tweedly, and J. Geevarghese. Compressing IP/UDP/RTP Headers for Low-Speed Serial Links. http://www.ietf.org/internet-drafts/draft-ietf-avt-crtp-enhance-01.txt, Nov. 2000.
8. S. L. Casner and V. Jacobson. RFC2508: Compressing IP/UDP/RTP Headers for Low-Speed Serial Links. ftp://ftp.isi.edu/in-notes/rfc2508.txt, Feb. 1999.
9. S. Deering and R. Hinden. RFC2460: Internet Protocol Version 6 (IPv6) Specification. ftp://ftp.isi.edu/in-notes/rfc2460.txt, Dec. 1998.
10. M. Degermark, B. Norgren, and S. Pink. RFC2507: IP Header Compression. ftp://ftp.isi.edu/in-notes/rfc2507.txt, Feb. 1999.
11. M. Engan, S. L. Casner, and C. Bormann. RFC2509: IP Header Compression over PPP. ftp://ftp.isi.edu/in-notes/rfc2509.txt, Feb. 1999.
12. N. Gerlich. *Transporting Wireless Network Traffic on Wired Networks - A Performance Study*. PhD thesis, University of Würzburg, Institute of Computer Science, Am Hubland, Apr. 1999.
13. N. Gerlich and M. Menth. The Performance of AAL-2 Carrying CDMA Voice Traffic. In *11th ITC Specialist Seminar*, Yokohama, Japan, Oct. 1998.
14. N. Gerlich and M. Ritter. Carrying CDMA traffic over ATM using AAL–2: A Performance Study. Technical Report, No. 188, University of Würzburg, Institute of Computer Science, Sep. 1997.
15. M. Handly. GeRM: Generic RTP Multiplexing. http://www.ietf.org/internet-drafts/draft-ietf-avt-mux-rtp-00.txt, Nov. 1998.
16. ITU. *ITU-T Recommendation I.363.2. B-ISDN ATM Adaptation Layer Type 2 Specification*, Feb. 1997.
17. V. Jacobson, K. Nichols, and K. Poduri. RFC2598: An Expedited Forwarding PHB. ftp://ftp.isi.edu/in-notes/rfc2598.txt, June 1999.
18. M. Menth. Carrying Wireless Traffic in UMTS over IP Using Realtime Transfer Protocol Multiplexing. In *12th ITC Specialist Seminar*, pages 13 – 25, Lillehammer, Norway, March 2000.
19. M. Menth. The Performance of Multiplexing Voice and Circuit Switched Data in UMTS over IP Networks. In *Protocols for Multimedia Systems (PROMS2000)*, pages 312 – 321, Cracow, Poland, Oct. 2000.
20. M. Menth and N. Gerlich. A Numerical Framework for Solving Discrete Finite Markov Models Applied to the AAL-2 Protocol. In *MMB '99, 10th GI/ITG Special Interest Conference*, pages 0163–0172, Trier, Sep. 1999.
21. M. Menth and S. Schneeberger. A Scalable Scheduling Mechanism with Application to AAL2/ATM Multiplexing. In *12th ITC Specialist Seminar*, Barcelona, Spain, April 2001.
22. G. S. Pall, B. Palter, A. Rubens, W. M. Townsley, A. J. Valencia, and G. Zorn. RFC2661: Layer Two Tunneling Protocol L2TP. ftp://ftp.isi.edu/in-notes/rfc2661.txt, Aug. 1999.
23. R. Pazhyannur, I. Ali, and C. Fox. PPP Multiplexing. http://www.ietf.org/internet-drafts/draft-ietf-pppext-pppmux-01.txt, Oct. 2000.
24. J. Postel. RFC791: Internet Protocol. http://www.ietf.org/rfc/rfc0791.txt, Aug. 1981.
25. B. Subbiah, S. Dixit, and N. R. Center. Low-Bit-Rate Voice and Telephony over ATM in Cellular/Mobile Networks. *IEEE Personal Communications*, pages 37–43, Dec 1999.
26. B. Subbiah and S. Sengodan. User Multiplexing in RTP Payload between IP Telephony Gateways. http://www.ietf.org/internet-drafts/draft-ietf-avt-mux-rtp-00.txt, Aug. 1998.
27. B. Thompson, T. Koren, and D. Wing. Tunneling Multiplexed Compressed RTP

(TCRTP). http://www.ietf.org/internet-drafts/draft-ietf-avt-tcrtp-02.txt, Nov. 2000.

28. P. Tran-Gia. Discrete-Time Analysis Technique and Application to Usage Parameter Control Modeling in ATM Systems. In *8th Australian Teletraffic Research Seminar*, Melbourne, Dec. 1993.
29. A. J. Valencia. L2TP Header Compression (L2TPHC). http://www.ietf.org/internet-drafts/draft-ietf-l2tpext-l2tphc-03.txt, Nov. 2000.
30. J. Wroclawski. RFC2210: The Use of RSVP with IETF Integrated Services. ftp://ftp.isi.edu/in-notes/rfc2210.txt, Sep. 1997.