INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. 2007; **20**:65–81
Published online 16 May 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dac.814

Adaptive and reliable video transmission over UMTS for enhanced performance

Antonios Alexiou[1,2,‡], Dimitrios Antonellis[1,2,§] and Christos Bouras[1,2,*,†]

[1] *Research Academic Computer Technology Institute, Greece*
[2] *Computer Engineering and Informatics Department, University of Patras, Greece*

SUMMARY

This paper proposes a mechanism for the congestion control for video transmission over universal mobile telecommunications system (UMTS). Our scheme is applied when the mobile user experiences real-time multimedia content and adopts the theory of a widely accepted rate control method in wired networks, namely equation-based rate control. In this approach, the transmission rate of the multimedia data is determined as a function of the packet loss rate, the round trip time and the packet size and the server explicitly adjusts its sending rate as a function of these parameters. Furthermore, we examine the performance of the UMTS for real-time video transmission using real-time protocols. Through a number of experiments, we measure performance parameters such as end-to-end delay, delay in radio access network, delay jitter and throughput in the wireless link. Copyright © 2006 John Wiley & Sons, Ltd.

Received 1 July 2005; Revised 1 February 2006; Accepted 1 March 2006

KEY WORDS: TFRC; UMTS; W-CDMA cellular network; multimedia applications; video streaming; real-time protocols

1. INTRODUCTION

As communications technology is being developed, user's demand for multimedia services raises. Meanwhile, the Internet has enjoyed tremendous growth in recent years. Consequently, there is a great interest in using the IP-based networks to provide multimedia services. One of the most important areas in which the issues are being debated, is the development of standards for the universal mobile telecommunications system (UMTS) [1].

*Correspondence to: Christos Bouras, Research Academic Computer Technology Institute, N. Kazantzaki, GR-26500 Patras, Greece.
† E-mail: bouras@cti.gr
‡ E-mail: alexiua@cti.gr
§ E-mail: antonel@ceid.upatras.gr

UMTS constitutes the third generation of cellular wireless networks which aims to provide high-speed data access along with real-time voice calls. Wireless data is one of the major boosters of wireless communications and one of the main motivations of the next generation standards [1].

Bandwidth is a valuable and limited resource for UMTS and every wireless network, in general. Therefore, it is of extreme importance to exploit this resource in the most efficient way. It is essential for a wireless network to have an efficient bandwidth allocation algorithm in order the mobile user to experience both real-time video applications and Internet applications such as HTTP or SMTP. Consequently, when a user experiences a streaming video, there should be enough bandwidth available at any time for any other application that the mobile user might need. In addition, when two different applications run together, the network should guarantee that there is no possibility for any of the above-mentioned applications to prevail against the other by taking all the available channel bandwidth. Taking into consideration the fact that Internet applications adopt mainly TCP as the transport protocol, while streaming applications mainly use RTP, the network should guarantee that RTP does not prevail against the TCP traffic. Consequently, this means that there should be enough bandwidth available in the wireless channel for the Internet applications to run properly [2].

Rate control is an important issue in both wired and wireless streaming applications. A widely popular rate control scheme over wired networks is equation-based rate control [3,4], also known as TCP friendly rate control (TFRC). There are basically three main advantages for rate control using TFRC: first, it does not cause network instability, which means that congestion collapse is avoided. More specifically, TFRC mechanism monitors the status of the network and every time that congestion is detected, it adjusts the sending rates of the streaming applications. Second, it is fair to TCP flows, which are the dominant source of the traffic on the Internet. In case that a TCP flow co-exists in the network with a non-TCP flow, the mechanism adjusts the sending rate of the non-TCP flow so as to enhance the performance of the TCP flows. Third, the TFRC's rate fluctuation is lower than TCP, making it more appropriate for streaming applications which require constant video quality [5]. This means that the client does not deal with a great variety of rates of the streaming application and it is able to make more accurate estimations of the packets' arrival time.

In this work, we focus on solutions for streaming video over UMTS transport channels, which only require insignificant modifications in the streaming server and client, but provide a certain guaranteed quality of service (QoS). This could be achieved through the use of the TFRC mechanism. In our approach, we have modified the TFRC mechanism that is mainly used in wired networks, in order to support the specific characteristics and architecture of the UMTS network. With the aid of the TFRC mechanism, we monitor the network state of the UMTS and estimate the appropriate transmission rate of the video data. Our mechanism, handles fairly all the data flows sharing the available channel resources without causing network congestion. In parallel, we control the packet losses of the transmitted video sequence, improving the representation quality of the transmitted video in the terminal of the mobile client. Furthermore, the performance of the UMTS-dedicated channels (DCHs) for real-time video transmission is examined.

There are several proposed algorithms that are able to provide smooth transmission rate and slowly responsible congestion control. The main disadvantage of these methods is the fact that they require significant modifications in the functionality of the network and its components.

On the contrary, our mechanism provides simplicity and adaptability to unpredictable wireless channel conditions and it is advisable for real-time flows.

This paper is structured as follows. Section 2 is dedicated to the description of the related work. In Section 3, we provide an overview of video streaming protocols that could be used in UMTS, while Section 4 presents the TFRC protocol. In Section 5, the TFRC mechanism for the UMTS air interface is analysed. Following this, Section 6 reviews the operation of the TFRC mechanism and presents how the critical parameters of the TFRC mechanism are estimated. Section 7 is dedicated to the experiments' results. Finally, some concluding remarks and planned next steps are briefly described.

2. RELATED WORK

In Reference [5], a widely accepted rate control method in wired networks which is the equation-based rate control, also known as TFRC, is proposed. In this approach, the authors use multiple TFRC connections as an end-to-end rate control solution for wireless streaming video. As Cheng Peng Fu and Liew [6] presents, TCP Reno, treats the occurrence of packet loss as a manifestation of network congestion. This assumption may not apply to networks with wireless channels, in which packet loss is often induced by noise, link error, or reasons other than network congestion. Equivalently, TCP Vegas uses queuing delay as a measure of congestion [7]. Thus, the authors [6] propose an enhancement of the TCP Reno and TCP Vegas for the wireless networks, namely TCP Veno.

In Reference [8], the authors present two algorithms that formulate resource allocation in wireless networks. These procedures constitute a preliminary step towards a systematic approach to jointly design TCP congestion control algorithms, not only to improve performance, but more importantly, to make interaction more transparent. Xu *et al.* [9] study the performance characteristics of TCP New Reno, TCP SACK, TCP Veno and TCP Westwood under the wireless network conditions and they propose a new TCP scheme, called TCP New Jersey, which is capable of distinguishing wireless packet losses from congestion.

Recent work [10] provides an overview of MPEG-4 video transmission over wireless networks. A critical issue is how we can ensure the QoS of video-based applications to be maintained at an acceptable level. Another point to consider is the unreliability of the network, especially of the wireless channels because we observe packet losses resulting in a reduction of the video quality. The results demonstrate that the video quality can be substantially improved by preserving the high-priority video data during the transmission.

Although these adaptive streaming techniques show noticeable benefits in streaming video over heterogeneous networks, they require significant modifications in the streaming client, the streaming server, or both.

3. VIDEO STREAMING PROTOCOLS FOR UMTS

A generic framework for a typical video streaming service consists of a content creation and a retrieval system. When a streaming service is provided, a media server opens a connection to the client terminal and begins streaming the media to the client at approximately the playout rate. During the media receiving, the client presents the media with some or no

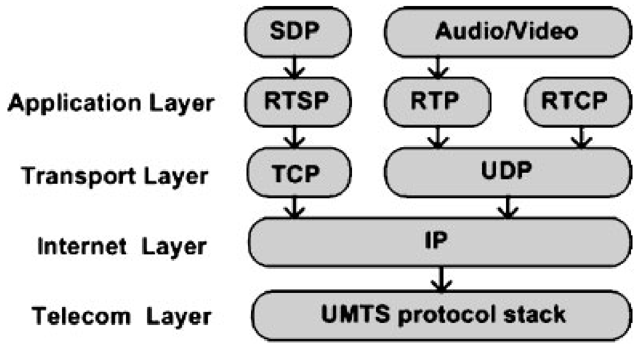

Figure 1. Protocol stack for signalling and media flows of streaming services in UMTS.

delay. This technique does not only free up limited terminal memory, but also it allows the clients to experience live streaming of the media. Additionally, the user needs a player, a special program that decompresses and sends video and audio data to the display and speakers, respectively. This client application must be able to control the streaming flows (control plane) and manage the media flows (user plane). Moreover, the client has to interface with the underlying transport network technology, its specific protocols and data bearers dedicated to the service [11].

The 3GPP PS multimedia streaming service is being standardized based on control and transport IETF protocols such as real-time streaming protocol (RTSP), real-time transport protocol (RTP) and session description protocol (SDP), as Figure 1 shows.

RTSP is an application level client–server protocol, which is used to control the delivery of real-time streaming data [12]. RTP transports media data flows over UDP, in the same way as its related control protocol that is called real-time transport control protocol (RTCP) [13]. The latter provides feedback to applications about the transmission quality and more specifically, it uses sender reports and receiver reports, which contain useful statistical information, like total transmitted packets, packet loss rate and delay jitter during the transmission of the data. This statistical information is very useful because it can be used for the implementation of congestion control mechanisms improving the performance streaming mechanism.

For the transport over packet-switched networks, the video traffic is typically packetized, which means that the video data is packaged into packets. In general, the packetization strategy can be selected from a large set of alternatives depending on the overall objective and set-up of a specific simulation. In order to illustrate the issues involved in the above procedure, we discuss it in accordance with the real-time protocol (RTP) [13]. An RTP packet consists of a 12-byte RTP header, an 8-byte UDP header and a 20-byte IPv4/40-byte IPv6 header. In Reference [14], it is stated that a given RTP packet carries data from only one video frame, which means that the loss of an RTP packet will affect only one video frame. The amount of video data in an RTP packet should be adjusted appropriately so that the total RTP packet (consisting of the video data plus the headers) does not exceed the maximum transfer unit (MTU) on the path through the network. Thus, we avoid fragmentation in the network, except for wireless links that may perform fragmentation of the RTP packet carried over the wired network. In case the video frames are small, it is permitted to carry multiple consecutive video frames in one RTP packet.

4. THE TFRC PROTOCOL

This section presents an overview of the TFRC protocol details. TFRC is not actually a fully specified end-to-end transmission protocol, but a congestion control mechanism that is designed to operate fairly along with TCP traffic. Generally, TFRC should be deployed with some existing transport protocol such as UDP or RTP in order to present its useful properties [3].

The main idea behind TFRC is to provide a smooth transmission rate for streaming applications. The other properties of TFRC include slow response to congestion and the opportunity not aggressively trying to make up with all available bandwidth. Consequently, in case of a single packet loss, TFRC does not halve its transmission rate like TCP, while, on the other hand it does not respond rapidly to the changes in available network bandwidth. TFRC has also been designed to behave fairly when competing for the available bandwidth with concurrent TCP flows, that comprise the majority of flows in today's networks.

A widely popular model for TFRC is described by the following equation [4]:

$$T = \frac{kS}{\mathrm{RTT}\sqrt{p}} \tag{1}$$

T represents the sending rate, S is the packet size, RTT is the end-to-end round trip time, p is the end-to-end packet loss rate, and k is a constant factor between 0.7 and 1.3 [15], depending on the particular derivation of Equation (1).

The equation describes TFRC's sending rate as a function of the measured packet loss rate, RTT and used packet size. More specifically, a potential congestion in the nodes of the path will cause an increment in the packet loss rate and in the RTT according to the current packet size. Given this fluctuation, it is easy to determine the new transmission rate so as to avoid congestion and packet losses. Generally, TFRC's congestion control consists of the following mechanisms:

1. The receiver measures the packet loss event rate and feeds this information back to the sender.
2. The sender uses these feedback messages to calculate the RTT of the packets.
3. The loss event rate and the RTT are then fed into the TRFC rate calculation equation (described later in more detail) in order to find out the correct data sending rate.

5. ANALYSIS OF THE TFRC MECHANISM FOR UMTS

The typical scenario for streaming video over UMTS is shown in Figure 2, where the server is denoted by Node1 and the receiver by UE1. The addressed scenario comprises a UMTS radio cell covered by a Node B connected to an RNC. The simulation model consists of a UE connected to DCH as it is shown in Figure 2. In this simulation, we use the DCH to transmit packet data. DCH is a bi-directional channel and is reserved only for a single user. The common channels are the forward access channel (FACH) in the downlink and the random access channel (RACH) in the uplink.

The wireless link is assumed to have available bandwidth B_W, and packet loss rate p_W, caused by wireless channel error. This implies that the maximum throughput that could be achieved in the wireless link is $B_W(1-p_W)$. There could also be packet loss caused by congestion at wired

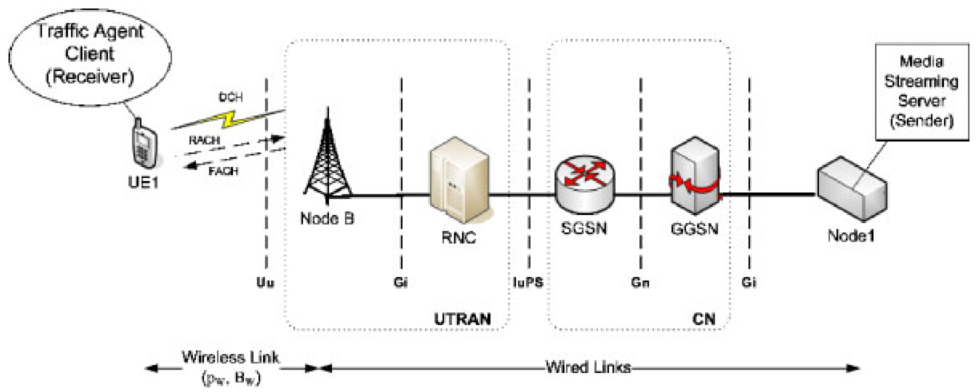
Traffic Agent Client (Receiver)
DCH
RACH
FACH
UE1
Node B
RNC
SGSN
GGSN
Node1
Media Streaming Server (Sender)
Uu
Gi
IuPS
Gn
Gi
UTRAN
CN
Wireless Link (p_W, B_W)
Wired Links

Figure 2. Typical scenario for streaming video over UMTS.

nodes denoted by $p_{\text{node name}}$ (node name: GGSN, SGSN, RNC, Node B). The end-to-end packet loss rate observed by the receiver is denoted as p. The streaming rate is denoted by T. This means that the streaming throughput is $T(1-p)$. Under the above assumptions, we characterize the wireless channel as underutilized if $T(1-p) < B_W\,(1-p_W)$.

Given the above-described scenario, we assume the following:

1. The wireless link is assumed to be the long-term bottleneck. This means that there is no congestion due to streaming traffic to the nodes GGSN, SGSN and RNC.
2. There is no congestion at Node B due to the streaming application, if and only if the wireless bandwidth is underutilized, i.e. $T(1-p) < B_W\,(1-p_W)$. This also implies that no queuing delay caused at Node B, which means that the RTT for a given route has the minimum value, i.e. $\text{RTT}_{\min}$. Thus, this assumption can be restated as follows: for a given route, $\text{RTT} = \text{RTT}_{\min}$ if and only if $T(1-\text{p}) \leqslant B_W(1-p_W)$. This in turn implies that if $T(1-p) > B_W(1-p_W)$ then $\text{RTT} \geqslant \text{RTT}_{\min}$.
3. The packet loss rate caused by wireless channel error (p_W) is random and varies from 0 to 0.16.
4. The backward route is error-free and congestion-free.

We use the TFRC model described in Equation (1) to analyse the problem. As it has already been mentioned, the average throughput measured at the receiver is $T(1-p)$, when the streaming rate is T and the overall packet loss rate is p. According to Reference [5], the end-to-end packet loss rate p is a combination of p_W and $p_{\text{node name}}$ (node name: GGSN, SGSN, RNC, NODEB) and is computed as follows:

$$\begin{aligned} p = {} & p_{\text{GGSN}} + (1 - p_{\text{GGSN}})p_{\text{SGSN}} + (1 - p_{\text{GGSN}})(1 - p_{\text{SGSN}})p_{\text{RNC}} \\ & + (1 - p_{\text{GGSN}})(1 - p_{\text{SGSN}})(1 - p_{\text{RNC}})p_{\text{NODEB}} \\ & + (1 - p_{\text{GGSN}})(1 - p_{\text{SGSN}})(1 - p_{\text{RNC}})(1 - p_{\text{NODEB}})p_W \end{aligned} \tag{2}$$

According to Equation (2), the total packet loss rate is the sum of the packet loss rate in each node of the network (Figure 2). More specifically the packet loss rate in each node is the product of the packet loss rate in the specific node and the offered rate of the previous node. Furthermore, we use $p^{(1)}_{\text{node name}}$ and $p^{(2)}_{\text{node name}}$ to represent the packet loss rate at the specific node caused by streaming traffic itself, i.e. self-congestion, and by other traffic flows, i.e. cross congestion, respectively. Thus, $p_{\text{node name}} = p^{(1)}_{\text{node name}} + p^{(2)}_{\text{node name}}$. Given the fact that there is no self-congestion at the wired part of the network since the wireless link is assumed to be the long-term bottleneck (Assumption 1), p can be re-written as

$$\begin{aligned} p =\, & p^{(2)}_{\text{GGSN}} + (1 - p^{(2)}_{\text{GGSN}})p^{(2)}_{\text{SGSN}} + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})p^{(2)}_{\text{RNC}} \\ & + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})(p^{(1)}_{\text{NODEB}} + p^{(2)}_{\text{NODEB}}) \\ & + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})(1 - p^{(1)}_{\text{NODEB}} - p^{(2)}_{\text{NODEB}})p_W \Leftrightarrow \\ p =\, & p^{(2)}_{\text{GGSN}} + (1 - p^{(2)}_{\text{GGSN}})p^{(2)}_{\text{SGSN}} + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})p^{(2)}_{\text{RNC}} \\ & + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})p^{(2)}_{\text{NODEB}} \\ & + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})(1 - p^{(2)}_{\text{NODEB}})p_W \\ & + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})p^{(1)}_{\text{NODEB}} \\ & - (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})p^{(1)}_{\text{NODEB}}p_W \Leftrightarrow p = p_1 + p_2 \end{aligned} \tag{3}$$

where

$$\begin{aligned} p_1 =\, & p^{(2)}_{\text{GGSN}} + (1 - p^{(2)}_{\text{GGSN}})p^{(2)}_{\text{SGSN}} + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})p^{(2)}_{\text{RNC}} \\ & + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})p^{(2)}_{\text{NODEB}} \\ & + (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})(1 - p^{(2)}_{\text{NODEB}})p_W \end{aligned} \tag{4}$$

and

$$p_2 = (1 - p^{(2)}_{\text{GGSN}})(1 - p^{(2)}_{\text{SGSN}})(1 - p^{(2)}_{\text{RNC}})(1 - p_W)p^{(1)}_{\text{NODEB}} \tag{5}$$

p_1 is independent of packet loss caused by streaming traffic itself, and hence also independent of streaming rate T. Furthermore, p_1 combines congestion due to nonstreaming flows and wireless channel error in one quantity. Therefore, it can be interpreted as equivalent wireless channel packet loss rate with no congestion due to other traffic flows on the wired part of the UMTS network. On the other hand, p_2 depends on packet loss due to self-congestion, i.e. $p^{(1)}_{\text{NODEB}}$ and thus may vary according to the streaming rate. Equation (3) shows that p_1 is a lower bound for p and that the bound is reached if and only if there is no congestion due to streaming traffic, i.e. $p^{(1)}_{\text{NODEB}} = 0$ and hence, $p_2 = 0$. By combining Equations (1) and (3), an upper bound T_b, on the streaming rate of one TFRC connection can be derived as follows:

$$T \leqslant \frac{kS}{\text{RTT}_{\min}\sqrt{p_1}} \equiv T_b \tag{6}$$

If there is no congestion due to streaming traffic, i.e. $p^{(1)}_{\text{NODEB}} = 0$ and hence no queuing delay caused by the streaming traffic, we get $\text{RTT} = \text{RTT}_{\min}$, $p_2 = 0$, $p = p_1$ and therefore $T = T_b$ in Equation (6). In this case, the throughput is $T_b(1-p_1)$, which is the upper bound of throughput given one TFRC connection for the scenario shown in Figure 2. In Reference [5], the authors define the wireless link to be under-utilized if the overall end-to-end throughput is less than

$B_W(1-p_W)$. Consequently, the sufficient and necessary condition for one TFRC connection to under-utilize the wireless link is

$$T_b(1-p_1) < B_W(1-p_W) \tag{7}$$

6. TFRC MECHANISM PARAMETERS ESTIMATION

6.1. The operation of the mechanism

The communication between the sender and the receiver is based on RTP/RTCP sessions and the sender, denoted by Node 1 (Figure 2), uses the RTP protocol to transmit the video stream, whereas the client, denoted by UE1 (Figure 2), uses the RTCP protocol in order to exchange control messages. In the following paragraphs, details about the different aspects of mechanism are given.

The mobile user (client) in recurrent time space sends RTCP reports to the media server. These reports contain information about the current conditions of the wireless link during the transmission of the multimedia data between the server and the mobile user. The feedback information contains the following parameters:

- *Packet loss rate*: The receiver calculates the packet loss rate during the reception of sender data, based on RTP packets sequence numbers.
- *Timestamp of every packet arrived at the mobile user*: This parameter is used by the server for the RTT calculation of every packet.

The media server extracts the feedback information from the RTCP report and passes it through an appropriate filter. The use of filter is essential for the operation of the mechanism in order to avoid wrong estimations of the network conditions.

In the sender side, the media server using the feedback information estimates the appropriate rate of the streaming video so as to avoid network congestion. The appropriate transmission rate of the video sequence is calculated from Equation (6) and the media server is responsible for adjusting the sending rate with the calculated value. Obviously, the media server does not have the opportunity to transmit the video in all the calculated sending rates. However, it provides a small variety of them and has to approximate the calculated value choosing the sending rate from the provided transmission rates.

This extends the functionality of the whole congestion control mechanism. More specifically, the sender does not have to change the transmission rate every time it calculates a new one with a slight difference from the previous value. Consequently, it changes the transmission rate of the multimedia data to one of the available sending rates of the media server as has already been mentioned. In this approach, the number of the changes in the sending rate is restricted and the mobile user does not deal with a continually different transmission rate.

In order to implement the above assumptions, it is essential to keep a history of the previous calculated values for the transmission rate. Having this information, the media server can estimate the smoothed transmission rate, using the m most recent values of the calculated sending rate from Equation (8).

$$T^{\text{Smoothed}} = \frac{\sum_{i=1}^{m} w_i \cdot T_{m+1-i}^{\text{Smoothed}}}{\sum_{i=1}^{m} w_i} \tag{8}$$

The value m used in calculating transmission rate determines TFRC's speed in responding to changes in the level of congestion [16]. The weights w_i are appropriately chosen so that the most recent calculated sending rates receive the same high weights, while the weights gradually decrease to 0 for older calculated values. In our simulations we use $m = 8$ and the following values for the weights w_i:{1, 1, 1, 1, 0.8, 0.6, 0.4, 0.2}. Thus, we have chosen to keep track of eight values according to Reference [16].

6.2. Packet loss rate estimation

The mobile user (client) measures the packet loss rate p_1 based on RTP packets sequence numbers. The packet loss rate that is computed in the client is in accordance with Equation (3). This information is sent to the media server via an RTCP report. In order to prevent a single spurious packet loss having an excessive effect on the packet loss estimation, the server smoothes the values of packet loss rate using the filter of Equation (9), which computes the weighted average of the m most recent loss rate values [17].

$$p_1^{\text{Smoothed}} = \frac{\sum_{i=1}^{m} w_i \cdot p_{1,m+1-i}^{\text{Smoothed}}}{\sum_{i=1}^{m} w_i} \tag{9}$$

The value of p_1^{Smoothed} is then used by Equation (6) for the estimation of the transmission rate of the multimedia data. The weights w_i are chosen as in the transmission rate estimation, which means that keeping the eight most recent instances, they take the following values {1, 1, 1, 1, 0.8, 0.6, 0.4, 0.2} [16].

6.3. RTT estimation

When the client receives a RTP packet from the server, it uses the following algorithm in order to estimate the RTT between the client and the server. If we assume that the media server and the mobile client have synchronized clocks, the client can use the timestamp of the RTP packet ($t_{\text{timestamp}}$) and the local time that the packet is received (t_{client}) in order to estimate the one-way delay (t_{oneway}) between the server and the client:

$$t_{\text{oneway}} = t_{\text{client}} - t_{\text{timestamp}} \tag{10}$$

If the path between the sender and the receiver was symmetric and it had the same delay into both directions, the RTT between the sender and the receiver would be twice the t_{oneway}:

$$\text{RTT} = 2 \cdot t_{\text{oneway}} \tag{11}$$

Until now, we have made two assumptions: (1) the sender and the receiver have synchronized clocks (2) the path between the sender and the receiver is symmetric. The above assumptions are not true for UMTS. Consequently, in order to get accurate RTT estimations we have to take the above assumptions into account. For this reason, we use a parameter α and we can write Equation (11) as

$$\hat{\text{RTT}} = (1 + \alpha) t_{\text{oneway}} \tag{12}$$

The parameter α is used in order to smooth the estimation of the RTT due to the potential unsynchronized clocks between the server and the client and due to the potential asymmetry of the path between the sender and the receiver.

In order to estimate the value of parameter α, the client needs an effective estimation of RTT, which can be acquired with the use of RTCP reports as follows: the server sends a RTCP report to the client containing information about the timestamp of the RTCP packet generated at the server. When the client receives the RTCP packet from the server, it sends an RTCP report, indicating the timestamp of the last received sender report (t_{LSR}) and the delay between receiving the last sender report and sending the receiver report (t_{DLSR}). As a result of the above, the server can make an effective RTT measurement for the path between it and the client by using the following equation (where A is the time which the server receives the client report).

$$\mathrm{R\hat{T}T} = A - t_{\mathrm{DLSR}} - t_{\mathrm{LSR}} \tag{13}$$

The media server estimates an appropriate value for the RTT every time it receives a receiver report from the client and includes this RTT measurement in the next RTP packet. Since, the client has received an effective RTT measurement from the server, it estimates an appropriate value for the parameter α using the following equation:

$$\alpha = \frac{\mathrm{R\hat{T}T}}{t_{\mathrm{oneway}}} - 1 \tag{14}$$

7. EXPERIMENTS

7.1. Simulation topology and input parameters

To validate the above-presented analysis, we carry out experiments using the NS-2 simulator [18]. NS-2 is a discrete event simulator targeted at networking research. NS-2 provides substantial support for simulation of transport, routing, and multicast protocols over wired and wireless networks. For our simulation model, some modification on the existing code of NS-2 is essential to provide the functionality of the TFRC mechanism. The topology for the NS-2 simulations is the one shown in Figure 2. The input parameters of the simulation can be clearly seen in Table I.

Table I. Input parameters.

Input parameter	Value
UMTS transport channel type	DCH
Downlink bit rate B_W (kbps)	384
Uplink bit rate (kbps)	128
Downlink TTI (ms)	10
Uplink TTI (ms)	20
Average delay Node1 > GGSN (ms)	15
Average delay GGSN > SGSN (ms)	10
Average delay SGSN > RNC (ms)	1
Average delay RNC > Node B (ms)	15
RTCP report rate	Every 1 s
Packet size S (bytes)	800
p_W	0–0.16
Parameter k [4]	$1.5\sqrt{2/3}$

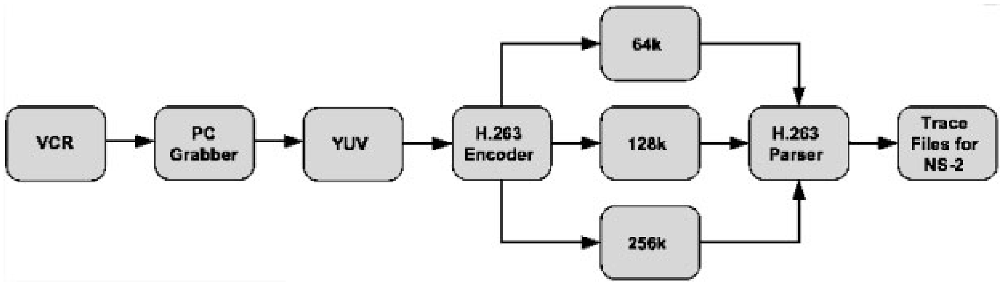

Figure 3. The H.263 encoding procedure.

For the transmission of the video data we use RTP. The feedback information is sent via RTCP. RTCP uses sender and receiver reports, which contain useful statistical information like total transmitted packets, packet loss rate and delay jitter during the transmission of the data. In addition, we use three TCP flows that have variable sending rate so as to observe the response of our mechanism. More specifically, the overall sending rate of the TCP flows has initially a low value (10 kbps). Then, it increases gradually until it reaches the maximum value of 100 kbps in the middle of the simulation. Finally, we adjust the sending rate of the TCP flows and the overall sending rate so as to decrease gradually until the value of 10 kbps. The four flows (the three TCP flows and the video) coexist in the same transport channel and the duration of the experiments is 200 s.

As far as the video sequence is concerned, it is encoded to ITU-H.263. H.263 is a low bit rate video-coding standard aimed at providing interactive video for Internet and mobile connections [19]. The video traces we use are taken from Reference [20]. Following this, we give a brief overview of encoding of the digital video. Let us start with an analog video signal generated by an analog video camera. The analog video signal consists of a sequence of video frames. The video frames are generated at a fixed frame rate (25 frames per second in the PAL format). To obtain a digital video signal, the analog video signal is passed to a digitizer. The digitizer samples and quantizes the analog video signal. Each sample corresponds to a picture element (pel). The digital frame format that we use is the quarter CIF (QCIF). Each video frame is divided into three components. These are the luminance component (Y) and the two chrominance components: hue (U) and intensity (V). Then, it follows the H.263 video source coding algorithm that uses inter picture prediction to reduce the temporal redundancy and discrete cosine transform (DCT) coding to reduce the spatial redundancy. Afterwards, the video is encoded into three different bit rates—64, 128 and 256 kbps. The H.263 parser processes the encoded video files into the three different bit rates and creates the trace files that will be used for the simulation model in the NS-2 (Figure 3) [20].

7.2. Results

In our experiments, we consider that the media server initially transmits the video with 256 kbps bit rate. During the simulation, as we described above, we change the transmission rate of the TCP flows. When the overall sending rate is increased, we observe increased packet losses due to congestion. Measuring this packet loss rate, we can estimate the congestion and adjust the transmission rate of the video.

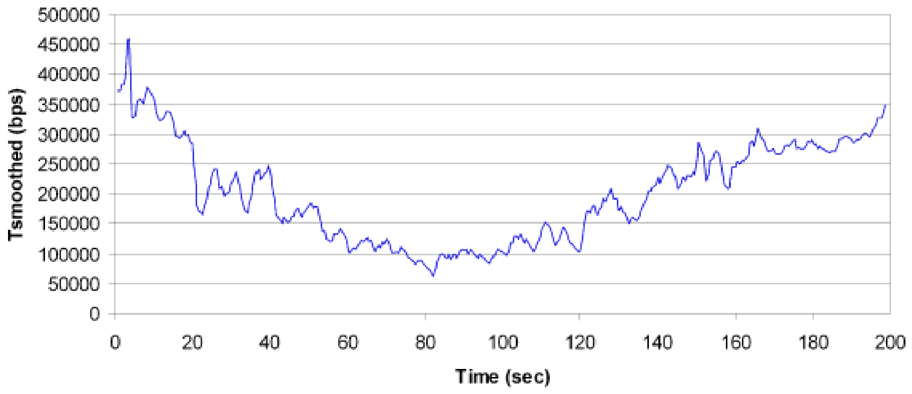

Figure 4. Calculated video transmission rate (T^{Smoothed}).

Figure 4 depicts the estimated transmission rate of the video sequence and it is computed according to Equations (6) and (8). The y-axis presents the estimated transmission rate, while the x-axis shows the duration of the simulation. According to the TFRC mechanism, the media server estimates the new transmission rates every time that the path profile changes so as to overcome the variations in the path loss rate, as well as to serve efficiently the TCP flows. This means that when the media server transmits the video with the greater bit rate and observes an increase in the packet loss rate, it has to decrease the sending rate of the video sequence in order to (1) avoid network collapse, (2) decrease the packet loss rate and (3) continue serving the TCP flows. This explains the initial decrement in the smoothed calculated transmission rate of the video that is depicted in Figure 4.

In approximately half of the simulation time, we observe the maximum packet loss rate due to the increased overall sending rate of the TCP flows, which results to the minimum estimated transmission rate (Figure 4). Additionally, 100 s after the beginning of the simulation, the smoothed transmission rate of the video increases. This is explained by the fact that the current period the sending rate of the TCP flows decreases with a respective decrement in the packet loss rate. Overall, we understand that the calculated transmission rate of the video does not perform weird transitions, but it changes smoothly and according to the parameters of the network.

The throughput of the video in the wireless link is depicted in Figure 5. The y-axis presents the throughput in bps while the x-axis represents the duration of the simulation. As it is shown, the media server initially uses the video with bit rate 256 kbps and 20 s after the beginning of the simulation, the TFRC mechanism calculates the smoothed transmission rate to be under the value of 256 kbps. This means that the media server has to change the bit rate of the video to 128 kbps in order to avoid congestion problems and maintain a TCP-friendly behaviour. Consequently, the throughput of the video in the wireless link the specific interval is around the value of 128 kbps. The rest transitions among the provided bit rates of video are obvious and result from the corresponding values of the smoothed transmission rate. As we observe in Figure 5, there are multiple transitions between the transmission rate of the video in the regions A and B. This occurs because the calculated smoothed sending rate is differentiated very little from the two specific values and the media server every time refreshes the appropriate transmission rate.

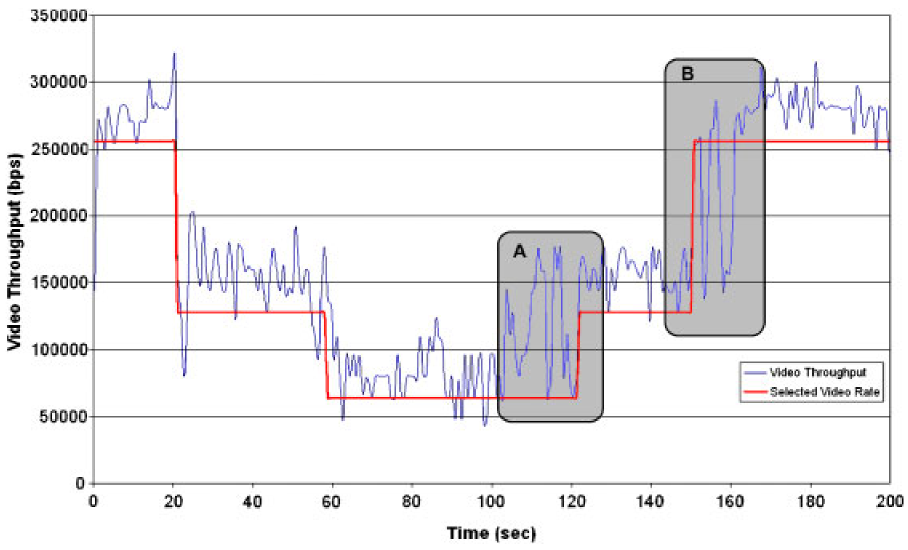

Figure 5. Video throughput in wireless link.

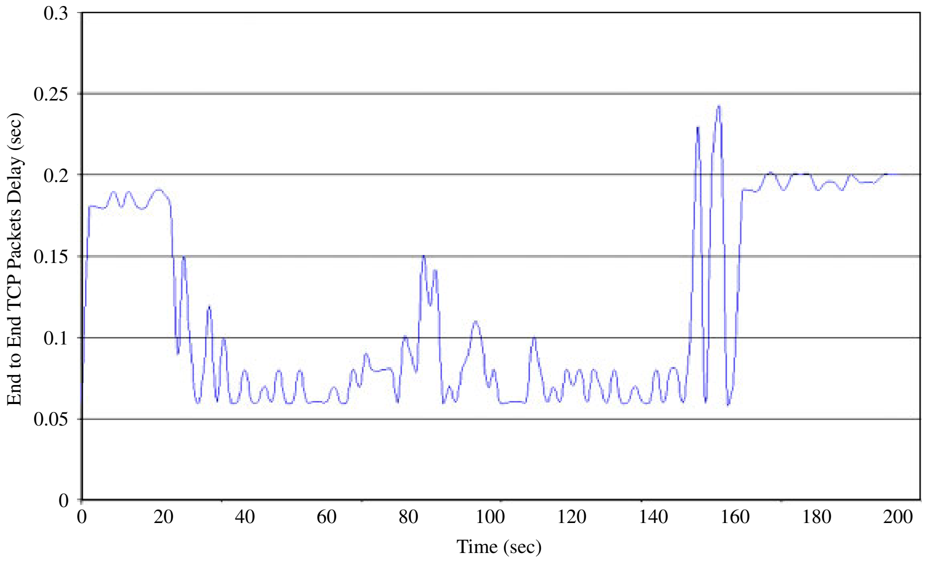

Figure 6. End-to-end delay of the TCP packets.

The next interesting parameter that presents the functionality of the TFRC mechanism is the end-to-end delay of the TCP packets depicted in Figure 6. More specifically, we observe that when the media server transmits the video with 256 kbps bit rate, the end-to-end delay of the TCP packets is increased and our mechanism has to restrict this. Thus, as we described above, the media server decreases the sending rate of the video to 128 kbps and the end-to-end delay of the TCP packets reduces, respectively. According to Figure 6, the end-to-end delay of the TCP packets has a low value in the interval between 20 and 150 s, except the period that the transmission rate of the TCP flows is increased and we observe congestion in the path. This occurs because a significant packet loss rate results to a greater number of retransmitted TCP packets which in turn entails an increased traffic in the network. Additionally, when the media server uses the 64 kbps bit rate to transmit the video to the mobile user and we observe that the packet loss rate decreases, our mechanism decides to increase the transmission rate of the video so as to increase the total throughput.

The above functionality of the TFRC mechanism is also proved by the calculation of the delay jitter of the TCP packets depicted in Figure 7. In other words, from the time that the media server reduces the sending rate of the video until it decides to increase it to the value of

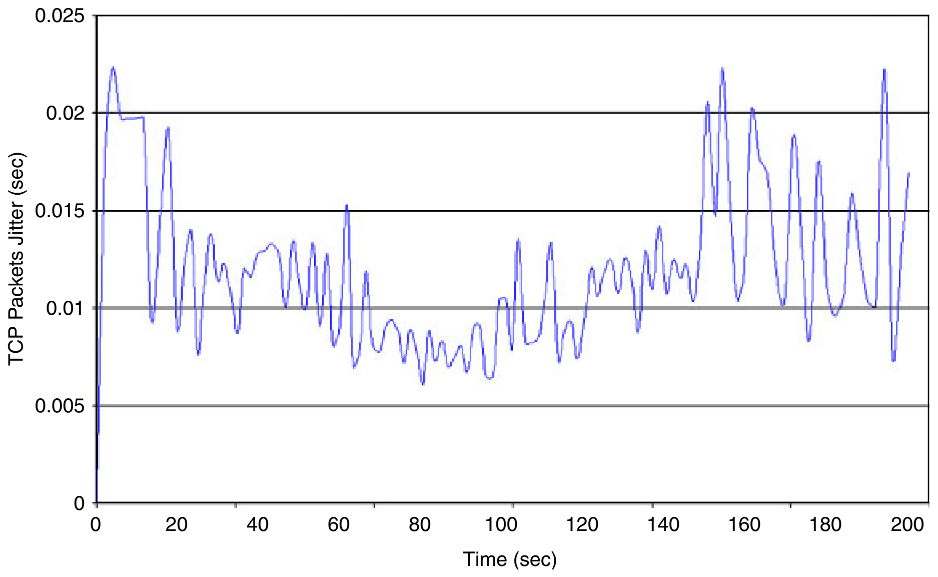

Figure 7. Delay jitter of the TCP packets.

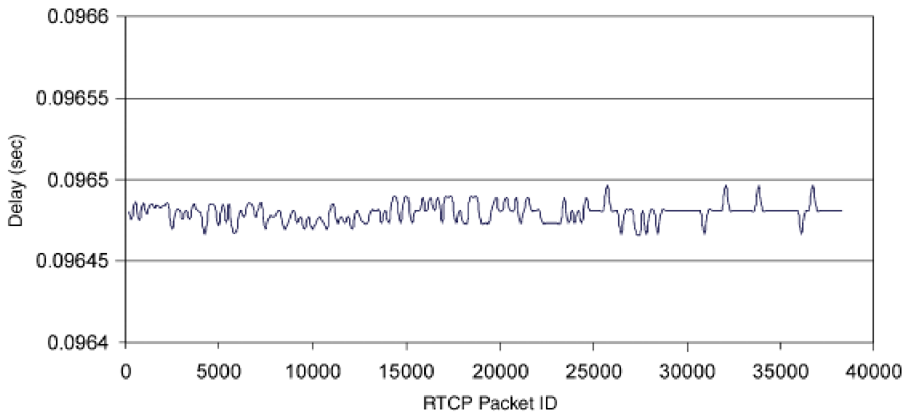

Figure 8. Delay of the RTCP receiver report.

256 kbps, the delay of the TCP packets is smooth (Figure 7) and restricted (Figure 6). This means that the mobile user does not deal with different delays in the TCP flows and it is able to make more accurate estimations of the TCP packets' arrival time.

Figure 8 presents the packet delay of the RTCP reports that the mobile client (UE1) sends to the media server (Node 1) (Figure 2). The x-axis presents the RTCP packet ID while the y-axis shows the delay in seconds. In our experiments, this delay is approximately 61 ms. The delay of the RTCP packets is of extreme importance for any rate control mechanism due to the fact that a lost or a delayed RTCP report could result to wrong estimations of the current conditions in the UMTS network.

8. FUTURE WORK

The step that follows this work is to evaluate the performance of adaptive video transmission over high-speed downlink packet access transmissions (HSDPA). HSDPA supports the introduction of high bit rate data services and will increase network capacity, while minimizing operators' investment. The introduction of shared channels for different users will guarantee that channel resources are used efficiently in the packet domain and will be less expensive for users than the use of dedicated ones.

Furthermore, shared channels give the opportunity to perform multicast in UMTS, which could result in bandwidth savings. In UMTS, bandwidth is a limited resource and the available radio resources can support only a handful high data-rate user simultaneously over a single cell. UMTS is therefore likely to have an interface to multicast.

9. CONCLUSIONS

In this paper, we have presented an analysis of the TFRC mechanism for UMTS. With the aid of the TFRC mechanism, we have monitored the network state of the UMTS air interface and estimated the appropriate transmission rate of the video data. In parallel, packet losses of the transmitted video sequence were controlled, improving the representation quality of the transmitted video in the terminal of the mobile client.

Through a number of experiments we concluded that the TFRC mechanism performs efficiently in mixed traffic conditions. The three goals of our rate control could be stated as follows. First, the streaming rate should not cause any network instability, i.e. congestion collapse. Second, TFRC is assumed to be TCP friendly. Any application that transmits data over a network should present a friendly behaviour towards the other flows that coexist in the network and especially towards the TCP flows that comprise the majority of flows in today's networks. Third, it leads to the optimal performance, i.e. it results to highest possible throughput and lowest possible packet loss rate.

Furthermore, we have presented an overview of video transmission over UMTS using real-time protocols such as RTP/RTCP. We have evaluated the performance of UMTS for such an approach, measuring parameters such as end-to-end packet delay, delay jitter, delay of the RTCP reports and throughput in wireless link.

REFERENCES

1. Holma H, Toskala A. *WCDMA for UMTS: Radio Access for Third Generation Mobile Communications*. Wiley: New York, 2003.
2. Wong D, Varma V. Supporting real-time IP multimedia services in UMTS. *IEEE Communications Magazine* 2003; **41**(11):148–155.
3. Floyd S, Handley M, Padhye J, Widmer J. Equation-based congestion control for unicast applications. *Proceedings of the ACM SIGCOMM 2000*, August 2000; 43–56.
4. Floyd S, Fall K. Promoting the use of end-to-end congestion control in the internet. *IEEE/ACM Transactions on Networking* 1999; **7**(4):458–472.
5. Chen M, Zachor A. Rate control for streaming video over wireless. *Proceedings of the IEEE INFOCOM 2004*, 2004.
6. Fu CP, Liew SC. TCP Veno: TCP enhancement for transmission over wireless access networks. *IEEE Journal on Selected Areas in Communications* 2003; **21**(2):216–228.
7. Choe H, Low SH. Stabilized Vegas. *IEEE INFOCOM—The Conference on Computer Communications*, 2003; **22**(1):2290–2300.
8. Chen L, Low SH, Doyle JC. Joint congestion control and media access control design for ad hoc wireless networks. *IEEE INFOCOM*, March 2005.
9. Xu K, Tian Y, Ansari N. Improving TCP performance in integrated wireless communications networks. *Computer Networks, Science Direct* 2005; **47**(2):219–237.
10. Zhao J, Kok C, Ahmad I. MPEG-4 video transmission over wireless networks: a link level performance study. *Wireless Networks*. Kluwer Academic Publishers: Dordrecht, 2004; 133–146.
11. Curcio I, Leon D. Application rate adaptation for mobile streaming. *IEEE WoWMoM 2005*, 2005.
12. Schulzrinne H, Rao A, Lanphier R. Real time streaming protocol (RTSP). *IETF RFC 2326*, 1998.
13. Schulzrinne H, Casner S, Frederick R, Jacobson V. RTP: a transport protocol for real-time applications. *IETF RFC 1889*, 1996.
14. Kikuchi Y *et al*. RTP payload format for MPEG-4 audio/visual streams. *RFC3016*, 2000.
15. Mahdavi J, Floyd S. TCP-friendly unicast rate-based flow control. Available at http://www.psc.edu/networking/papers/tcp_friendly.html, January 1997.
16. Handley M, Floyd S, Padhye J, Widmer J. TCP friendly rate control (TFRC). *RFC 3448*, January 2003.
17. Vicisiano L, Rizzo L, Crowcroft J. TCP-like congestion control for layered multicast data transfer. *IEEE INFOCOM*, March 1998; 996–1003.
18. The NS-2 simulator. Available at http://www.isi.edu/nsnam/ns
19. Iskander C, Mathiopoulos P. Online Smoothing of VBR H.263 video for the CDMA200 and IS-95B uplinks. *IEEE Transactions on Multimedia* 2004; **6**(4):647–658.
20. Fitzek F, Reisslein M. MPEG-4 and H.263 video traces for network performance evaluation. *IEEE Network* 2001; **15**(6):40–54.

AUTHORS' BIOGRAPHIES

Antonios Alexiou obtained his Diploma from the Department of Electrical and Computer Engineering of the Aristotle University of Thessaloniki, Greece and his Master Degree from the Computer Engineering and Informatics Department of Patras University. He is currently a PhD Candidate of the Department of Computer Engineer and Informatics of Patras University. Furthermore, he is working as R&D Computer Engineer at the Research Unit 6 of the Research Academic Computer Technology Institute in Patras, Greece. His research interests include Mobile Telecommunications Networks, Multicast Routing, User Mobility in Cellular Networks, Congestion Control and Quality of Service, Mobile and Wireless *ad hoc* Networks. He has published two papers in journals and 12 papers in well-known refereed conferences.

Dimitrios Antonellis is an undergraduate student in the Computer Engineering and Informatics Department of Patras University, Greece. Furthermore, he is working as R&D Computer Engineer at the Research Unit 6 of the Research Academic Computer Technology Institute in Patras, Greece. His research interests include Data Networks, Third Generation Mobile Telecommunications Networks, Multicast Routing and Group Management, User Mobility in Cellular Networks, Congestion Control and Quality of Service, Mobile and Wireless *ad hoc* Networks. He has published three papers in international conferences.

Christos Bouras obtained his Diploma and PhD from the Computer Science and Engineering Department of Patras University, Greece. He is currently an Associate Professor in the above department. Also he is a scientific advisor of Research Unit 6 in Research Academic Computer Technology Institute (CTI), Patras, Greece. His research interests include Analysis of Performance of Networking and Computer Systems, Computer Networks and Protocols, Telematics and New Services, QoS and Pricing for Networks and Services, e-learning, Networked Virtual Environments and WWW Issues. He has extended professional experience in Design and Analysis of Networks, Protocols, Telematics and New Services. He has published 200 papers in various well-known refereed conferences and journals. He is a co-author of seven books in Greek. He has been a PC member and referee in various international journals and conferences. He has participated in R&D projects such as RACE, ESPRIT, TELEMATICS, EDUCATIONAL MULTIMEDIA, ISPO, EMPLOYMENT, ADAPT, STRIDE, EUROFORM, IST, GROWTH and others. Also he is member of, experts in the Greek Research and Technology Network (GRNET), Advisory Committee Member to the World Wide Web Consortium (W3C), IEEE Learning Technology Task Force, IEEE Technical Community for Services Computing WG 3.3 Research on Education Applications of Information Technologies and W 6.4 Internet Applications Engineering of IFIP, Task Force for Broadband Access in Greece, ACM, IEEE, EDEN, AACE and New York Academy of Sciences.