STREAMING MULTIMEDIA DATA WITH ADAPTIVE QOS CHARACTERISTICS

Ch. Bouras[1] *A. Gkamas*[2]
[1]Computer Technology Institute, Greece - Computer Engineering and Informatics Dept., Univ. of Patras, Greece, Riga Ferraiou 61, GR 262 21 Patras, Greece - P.O. Box 1122, GR 261 10 Patras, Greece, e-mail: bouras@cti.gr
[2]Computer Technology Institute, Greece - Computer Engineering and Informatics Dept., Univ. of Patras, Greece, Riga Ferraiou 61, GR 262 21 Patras, Greece - P.O. Box 1122, GR 261 10 Patras, Greece, e-mail: gkamas@cti.gr

ABSTRACT

In this paper, we describe the design and the implementation of an adaptive streaming architecture, which is based on real time protocols. The adaptive streaming application, which we have developed, can be used for streaming multimedia data over heterogeneous networks like the Internet and has the capability to adapt the streaming of multimedia data to network changes. The adaptive streaming application uses a friendly to the network users congestion control mechanism to control the transmission of the media. We evaluate the adaptive streaming application through a number of experiments. During these experiments, we demonstrate the friendly behaviour of the adaptive streaming application to TCP and UDP data streams.

1 INTRODUCTION

Internet is a heterogeneous network environment and the network resources that are available to real time applications can be modified very quickly. Real time applications must have the capability to adapt their operation to network changes. In order to add adaptation characteristics to real time applications, we can use techniques both to the network and application layers.

Today, the underlying infrastructure of the Internet does not sufficiently support Quality of Service (QoS) guarantees. At the same time new technologies, which are used for the implementation of networks, like the Asynchronous Transfer Mode (ATM) will change the above conditions. As a result, in the future users may have the capability to request specific QoS characteristics even over the Internet.

Even if the Internet eventually supports reservation mechanisms or differentiated services, it is likely to be on per-class than per-flow basis. Thus, flows are still expected to perform congestion control within their own class.

Adaptive streaming applications have the capability to stream multimedia data over heterogeneous networks and adapt media transmission to network changes.

In order to implement an adaptive streaming application, someone must implement mechanisms to monitor the network conditions and mechanisms to adapt the transmission of the data to the network changes. In this paper, we concentrate on the implementation of a mechanism for monitoring the network condition and estimating the appropriate rate for the transmission of the multimedia data.

Someone can say that due to the use of new technologies, which offer QoS guarantees (like ATM) for the implementation of the networks, adaptive streaming applications will not be used in the future. We believe that this is not true and adaptive streaming application will be used in the future for the following reasons:

- Users may not always want to pay the extra cost for a service with specific QoS guarantees when they have the capability to access a service with good adaptive behaviour. For example if a user wants to watch a movie, perhaps he will prefer a service with specific QoS guarantees. However if a user is just browsing a video database, he may not want to pay

the extra cost for a service with specific QoS guarantees.

- Some networks may never be able to provide specific QoS guarantees to the users. For example mobile networks may not support QoS guarantees due to frequent changes to the network conditions.
- With the use of the differential services network model in the future, networks will support services with QoS guarantees together with best effort services and adaptive services. For example the ATM technology supports Constant Bit Rate (CBR) services, Available Bit Rate (ABR) services and Variable Bit Rate (VBR) services.

The rest of this paper is organised as follows: We review some of the related work in section 2. In section 3 we present various aspects of the implemented adaptive streaming application. Section 4 presents some implementation issues of the adaptive streaming application. Detailed description, of our experimental results is presented in section 5. Finally, section 6 concludes the paper and discusses some of our future work.

2 RELATED WORK

The subject of adaptive streaming of multimedia data over networks has engaged researchers all over the world. During the design and the implementation of an adaptive streaming application someone must pay special attention to the following critical modules:

- The module, which is responsible for the transmission of the multimedia data.
- The module, which is responsible for monitoring the network, conditions. This module determines the change to the network conditions and instructs the adaptive streaming application to adapt the transmission of the multimedia data to network changes.
- The module, which is responsible for the adaptation of the multimedia data to the network changes.
- The module, which is responsible of handling the transmission errors during the transmission of the multimedia data.

A common approach for the implementation of adaptive streaming applications is the use of UDP for the transmission of the multimedia data and the use of TCP for the transmission of control information ([1], [4]). Another approach for the transmission of the multimedia data is the use of RTP over UDP ([5], [8]).

It is important for adaptive streaming applications to have friendly behaviour to the dominant transport protocols of today's Internet (TCP and UDP). Paper [11] presents an end-to-end rate-based congestion control mechanism for real time streaming in the Internet, which follows the macroscopic behaviour of TCP.

For the implementation of the network monitoring module a common approach is the use of packet loss as an indication of congestion to the network ([5], [8]). One other approach for monitoring the network conditions is the use of utilisation of client buffer ([2], [3], [4]). An other important factor that can be used for monitoring the network conditions and especially for indication of network congestion is the use of delay jitter during the transmission of the multimedia data.

For the implementation of the adaptation module a common approach is the use of rate shaping ([5], [6]), the use of layered encoding ([3]), the use of frame dropping ([2], [4]) or a combination of the above techniques ([7]). The implementation of the adaptation module depends of the encoding that is used for the transmission of the multimedia data. For example if we want to use the frame dropping technique for the adaptation of a MPEG video stream, we have to use selective frame dropping, due to the fact that MPEG video uses inter-frame encoding and some frames contain information relative to other frames. Paper [15] gives a detailed survey of application level adaptation techniques.

The architecture that we propose for the implementation of the adaptive streaming application uses RTP/RTCP (Real time Transmission Protocol / Real time Control Transmission Protocol) for the transmission of the multimedia data. The RTP protocol seems to be the de facto standard for the transmission of multimedia data over the Internet and is used by mbone tools (vit, vat, etc) and ITU H.323 applications. In addition RTCP offers capabilities for monitoring the transmission quality of multimedia data. We will use these monitoring capabilities of RTCP for the implementation of the network monitoring module of the proposed architecture. In order to implement the media adaptation module, we use a rate shaping technique. In this work, we do not investigate the problem of error control during the transmission of the multimedia data. This is because during the transmission of multimedia data a small error rate is acceptable.

The most prominent feature of our architecture that makes it different among the above mentioned existing architectures is the network monitoring module which uses a combination of parameters in order to determine the network conditions. In addition, all the required modules for the implementation of the adaptive streaming mechanism are located on the server side only. This means, that any application, which is compatible with the transmission of multimedia data through RTP sessions (like mbone tools for example) can access

our service and benefit from its adaptive streaming characteristics.

3 THE ARCHITECTURE OF ADAPTIVE STREAMING APPLICATION

The architecture that we propose for the implementation of the adaptive streaming application is based on the client – server model. Figure 1 displays the architecture of the adaptive streaming application.

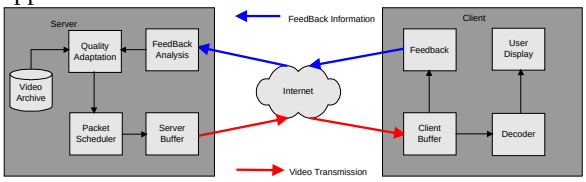

Figure 1. System Architecture

The server of the adaptive streaming architecture consists of the following modules:

♦ *Video archive:* Video archive consists of a set of hard disks in which the video files are stored. The adaptive streaming application supports the following video formats: MPEG, JPEG and H.263. It is possible for one video file to be stored in the video archive in more than one format in order to serve different target user groups. For example it is possible to store the same video in MPEG format in order to serve the users of the local area network (who have fast network connection with the server) and in H.263 format in order to serve distant users with slow network connection. In this paper, we do not investigate the problem of video storage in video archives in order to achieve the optimal performance of the server.

♦ *Feedback analysis:* This module is responsible for the analysis of feedback information from the network. The role of this module is to determine the network condition based on packet loss rate and jitter information, which are provided by RTCP receiver reports. After the examination of network condition, the feedback analysis module informs the quality adaptation module, in order to adapt the transmission of the video to current network conditions. We analyse the feedback analysis module in a following paragraph.

♦ *Quality adaptation:* It is responsible for the adaptation of the video transmission quality, in order to match with the current network conditions. This module, which is based on the rate shaping technique, is described in a following paragraph.

♦ *Packet scheduler / Server buffer:* This module is responsible for the encapsulation of multimedia information in the RTP packets. In addition, this module is responsible for the transmission of the RTP packets in the network. In order to smooth accidental problems to the transmission of the multimedia data from the server to the network, we use an output buffer on the server.

The client of the adaptive streaming architecture consists of the following modules:

♦ *Client buffer:* The use of the buffer on the client for the implementation of streaming applications is very important. The client application, it stores the incoming data to the buffer before starting present data to the user. The presentation of the multimedia data to the user starts only after the necessary amount of the data is stored in the buffer. The capacity of the client buffer depends to the delay jitter during the transmission of the multimedia data. In any case the capacity of the client buffer must be greater than the maximum delay jitter during the transmission of the data (we suppose that we measure the buffer capacity and the delay jitter in the same units, for example in seconds).

♦ *Feedback:* This module is responsible of monitoring the transmission quality of the data and informs the server. The monitoring of the transmission quality is based on RTCP receiver reports that the client sends to the server. RTCP receiver reports include information about the packet loss rate and the delay jitter during the transmission of the data. With the above information the feedback analysis module of the server determines the network's condition.

♦ *Decoder:* This module reads the data packets from the client buffer and decodes the encoded multimedia information. Depending on the packet losses and the delay during the transmission of the packets, the quality of the multimedia presentation can vary. The decoding and the presentation of the multimedia data can stop, if the appropriate amount of data does not exist in the buffer.

♦ *User Display:* It is responsible for the presentation of the multimedia data to the user.

In the following paragraphs we give a detailed description of the most important modules of the above described architecture.

3.1 Transmission of multimedia data

The transmission of the multimedia data is based on the protocols RTP/RTCP. We use the protocol RTP for the transmission of the multimedia data from the server to the client and the client uses the RTCP protocol, in order to inform the server of the transmission quality.

The RTP/RTCP protocols have been designed for the transmission of real time data like video and audio. Initially, the RTP/RTCP protocols are designed for multicast transmission but were also used for unicast transmissions. RTP/RTCP can be used for one-way communication like video on demand or for two-way communication like videoconference. RTP/RTCP

offer a common platform for the representation of synchronisation information that real time applications needs. The RTCP protocol is the control protocol of RTP. The RTP protocol has been designed to operate in cooperation with the RTCP protocol, which provides information about the transmission quality.
RTP is a protocol that offers end to end transport services with real time characteristics over packet switching networks like IP networks. RTP packet headers include information about the payload type of the data, numbering of the packets and timestamping information.
RTCP offers the following services to applications:

- *QoS monitoring:* This is one of the primary services of RTCP. RTCP provides feedback to applications about the transmission quality. RTCP 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.
- *Source identification:* RTCP source description packets can be used for identification of the participants in a RTP session. In addition, source description packets provide general information about the participants in a RTP session. This service of RTCP is useful for multicast conferences with many members.
- *Inter-media synchronisation:* In real time applications it is common to transmit audio and video in different data streams. RTCP provides services like timestamping, which can be used for inter-media synchronisation of different data streams (for example synchronisation of audio and video streams).

More information about RTP/RTCP can be found in RFC 1889 ([16]).

3.2 Feedback from the network

The presentation quality of real time data depends on the packet loss rate and the delay jitter during the transmission over the network. In addition, packet losses or rapid increases of delay jitter may they considered as an indication of problems during the transmission of the data over the network. In such a case the adaptive streaming application must adapt the transmission of the data in order to avoid phenomenon like network congestion. Real time applications have upper bounds to the packet loss rate and to the delay jitter. If packet loss rate or jitter get to be over these upper bounds, the transmission of real time data can not be continued.
Packet loss rate is defined as the fraction of the total transmitted packets that did not arrive at the receiver. In today's network the main reason of packet losses is congestion.
It is difficult to define delay jitter. Some researchers define delay jitter as the difference between the maximum and the minimum delay during the transmission of the packets for a period of time. Some other researchers define delay jitter as the maximum difference between the delay of the transmission of two sequential packets for a period of time. In this paper, in order to define the delay jitter, we use the definition that is used in RFC 1889 ([16]). According to RFC 1889 ([16]) delay jitter is defined to be the mean deviation (smoothed absolute value) of the difference D in packet spacing at the receiver compared to the sender for a pair of packets. This is equivalent to the difference in the "relative transit time" for the two packets. The relative transit time is the difference between a packet's timestamp and the receiver's clock at the time of arrival. If S_i is the timestamp from packet i and R_i is the time of arrival for this packet, then for two packets i and j, D is defined as:

$$D(i, j) = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i)$$

The delay jitter is calculated continuously as each packet i arrives, using the difference D for that packet and the previous packet, according to the following formula:

$$J_i = J_{i-1} + (|D(i-1, j)| - J_{i-1})/16$$

The above formula defines that the new value of delay jitter depends on the previous value of the delay jitter and on a gain parameter, which gives good noise reduction.
Delay jitter occurs when sequential packets encounter different delays in the queue of the network devices. The different delays are related to the serve model of each queue and the cross traffics in the transmission path.
May delay jitter occur during the transmission of real time data, which does not originate from the network but is originated from the transmission host (host included delay jitter). This is because during the encoding of the real time data, the encoder places a timestamp in each packet, which gives information about the time that the packet's information, must be presented to the receiver. In addition this timestamp is used for the calculation of the delay jitter during the transmission of the real time data. If a notable time passes from the encoding of the packet and transmission of the packet in the network (because the CPU of the transmitter host is busy) the calculation of the delay jitter is not valid. Host included delay jitter can lead to erroneous estimation for the network conditions.
As a conclusion we can say that delay jitter can not lead to reliable estimation of network condition by itself. Someone must use delay jitter in combination

with other parameters, like packet loss rate, in order to make reliable estimations of the network conditions. During our experiments, we ascertain that the combination of packet loss rate and delay jitter can be used for reliable indication of network congestion.

3.3 Feedback analysis and estimation of transmission rate

The feedback analysis module is responsible to analyse the feedback information that the client sends to the server (with the use of RTCP receiver reports), concerning the transmission quality of the multimedia data. After the analysis of the feedback information, the feedback analysis module informs the quality adaptation module to increase, decrease or keep the current transmission rate of the data. Figure 2 displays the components of the feedback analysis module.

Every time the server receives a RTCP receiver report from the client, the feedback analysis module runs the procedure, which is described, in this section in order to estimate the new transmission rate, which will match the new network conditions.

Client in repeated time space send to the server one RTCP receiver report, which contains information about the packet loss rate and the delay jitter during the transmission of the multimedia data between the server and the client. The feedback analysis module extracts the packet loss rate and the delay jitter of the RTCP receiver report and passes them through the appropriate filters (packet loss rate filter and delay jitter filter respectively).

The use of the filters is essential in order to avoid a solely phenomenon to affect the behaviour of the adaptive streaming application and lead to wrong estimations about the network conditions.

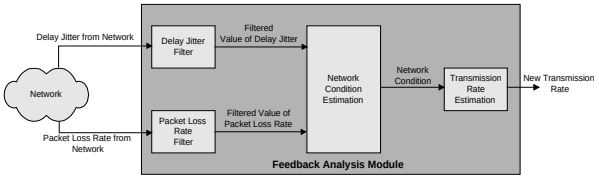

Figure 2. Feedback Analysis Module

More particularly the value of the packet loss rate passes the following filter:

$$LR_{new} = a * LR_{old} + (1-a) * LR_{net} \quad (1)$$

Where:

- LR_{new}: The new filtered value of packet loss rate.
- LR_{old}: The previous filtered value of packet loss rate. When the transmission of the data starts $LR_{old} = 0$.
- LR_{net}: The value of the packet loss rate from the RTCP receiver report that the client sent.
- a: This parameter specifies how aggressive the feedback analysis module will be to the value of the packet loss rate, which receives from the RTCP receiver report. For the parameter a stands $0 \le a \le 1$. It is obvious that for $a = 0$, the feedback analysis module uses only the values of packet loss rate, which it receives from the RTCP receiver reports in order to estimate the network condition. For $a = 1$, the feedback analysis module ignores the values of packet loss rate, which it receives from the RTCP receiver reports.

The value of the delay jitter passes the following filter:

$$J_{new} = \beta * J_{old} + (1-\beta) * J_{net} \quad (2)$$

Where:

- J_{new}: The new filtered value of delay jitter.
- J_{old}: The previous filtered value of delay jitter. When the transmission of the data starts $J_{old} = 0$.
- J_{net}: The value of the delay jitter from the RTCP receiver report that the client sent.
- β: This parameter specifies how aggressive the feedback analysis module will be to the value of the delay jitter, which it receives from the RTCP receiver report. For the parameter β stands $0 \le \beta \le 1$. It is obvious that for $\beta = 0$, the feedback analysis module uses only the values of delay jitter, which it receives from the RTCP receiver reports in order to estimate the network condition. For $\beta = 1$, the feedback analysis module ignores the values of delay jitter, which it receives from the RTCP receiver reports.

With the appropriate selection of a and β parameters values, we can designate the operation of the feedback analysis module.

The network conditions estimation component of the network estimation module (see figure 2) uses the filtered values of packet loss rate and delay jitter in order to characterise the network conditions. The network condition estimation component characterises the network on the following conditions:

- *Condition congestion:* When the network is in congestion condition, the packet loss rate is high and the transmission quality of the data is low. In this case the server must reduce the transmission rate of the multimedia data in order release some of the bandwidth that it is using and help the network to come back to normal operation.
- *Condition load:* When the network is in load condition the transmission quality is good. The packet loss rate is in affordable value, which does not cause problems to the presentation of the multimedia data. In this case, the server must keep the transmission rate of the multimedia data constant.
- *Condition unload:* When the network is in unload condition either packet losses does not exist or the

packet loss rate is very small. In this case the server can increase the transmission rate of the multimedia data and bind the available bandwidth.

The changes among the network conditions are based on the filtered values of the packet loss rate and delay jitter. More particularly for the packet loss rate, we define two values LR_c (congestion packet loss rate) and LR_u (unload packet loss rate), which control the changes among the network conditions based on the following algorithm:

$$\begin{aligned} &if\,(LR_{new} \geq LR_c) \rightarrow congestion \\ &if\,(LR_u < LR_{new} < LR_c) \rightarrow load \\ &if\,(LR_{new} \leq LR_u) \rightarrow unload \end{aligned} \quad (3)$$

For the filtered value of delay jitter no bounds exist, which designate the change among the network conditions because the value of delay jitter depends on the transmission path and the cross traffics in this transmission path. As a result no absolute values which control the behaviour of delay jitter exist. The analysis of the filter delay jitter by the network condition estimation component is based on the fact that abrupt increase of delay jitter may denote that the buffers of the queues on the transmission path had been overloaded and this may cause congestion to the network during the next moments. This approach ignores the host included delay jitter and we assume that the server either does not include any delay jitter to the transmission of the multimedia data or it includes a constant delay jitter during all the transmission period (in order to avoid the host included delay jitter during our experiments, we use high speed workstations as server). Network condition estimation component apprehends the abrupt increase of delay jitter as a precursor of network congestion and set the network condition to congestion. More particularly the network condition estimation component uses the following algorithm for the analysis of filtered delay jitter:

$$if\,(J_{new} > \gamma * J_{old}) \rightarrow congestion \quad (4)$$

Where γ is a parameter, which specifies how aggressive the network condition estimation component will be to the increase of delay jitter. In other words γ specifies quantitatively the expression "abrupt increase of delay jitter".

Network condition estimation component gives its estimation about the network condition to the transmission rate estimation component, which estimates the new transmission rate based on the network conditions. The transmission rate estimation component uses an Additive Increase, Multiplicative Decrease (AIMD) algorithm in order to estimate the new transmission rate. This algorithm is similar to the algorithm that the TCP rate control uses. Our algorithm tries to exploit the advantages of the TCP rate control algorithm and eliminates its disadvantages.

We chose an algorithm similar to TCP's rate control algorithm for fairness reasons to the allocation of network resources (like bandwidth) usage especially during network congestion periods. Applications with "bad" behaviour during congestion periods make the congestion problem worst and lead to unfair allocation of network resources among the network users. When a network is in congestion condition all the applications must reduce the transmission of the data and share with fairness the network resources.

The AIMD algorithm, which we propose is similar to the rate control algorithm of many application of today's Internet. When the application notices available bandwidth to the network, it increases the transmission rate by adding a factor to the transmission rate (probing). In the case of network congestion the application decreases the transmission rate by multiplying the transmission rate with a factor less that 1 (back of). More particularly the transmission estimation module uses following algorithm:

$$\begin{aligned} &if\,(network = unload) \rightarrow R_{new} = R_{old} + R_{increase} \\ &if\,(network = load) \rightarrow R_{new} = R_{old} \\ &if\,(network = congestion) \rightarrow R_{new} = R_{old} * R_{decrease} \end{aligned} \quad (4)$$

Where:

- R_{new}: The new value of the transmission rate.
- R_{old}: The old value of the transmission rate.
- $R_{increase}$: The factor with which the server increases the transmission rate in the case of available bandwidth.
- $R_{decrease}$: The factor with which the server decreases the transmission rate in the case of network congestion. For this factor stands: $0 < R_{decrease} < 1$.

The new value R_{new} of the transmission rate is used by the quality adaptation module in order to adapt the quality of the transmitted video to the new transmission rate.

The operation and the behaviour of the feedback analysis module are influenced by the parameters, which are used $(\alpha, \beta, \gamma, LR_c, LR_u, R_{increase}, R_{decrease})$. The choice of the above parameters depends on the network and the kind of the dominant traffic in it. The appropriate parameters for each network can be defined through a series of experiments. In paper [5] (where a similar algorithm is used which is based only to packet loss rate) Busse et al suggest specific values for some of the above parameters. From our experiments, we found some values that tune the behaviour of the adaptive streaming application:

- $\alpha = 0.5$: With this value, the filtered value of packet loss rate is based with the same weight to the

previous filtered value and to the packet loss rate from the network.

- $\beta = 0.8$: We select this value due to the fact that abrupt increase of delay jitter indicates network congestion as a result the delay jitter form the network must influence heavily the filter value of delay jitter.
- $\gamma = 2$: With this value, we consider as abrupt increase of delay jitter the doubling of filtered delay jitter between two calculations.
- LR_c: The value of this parameter depends on the network and the video encoding. For our experiments we use $LR_c = 0.05$.
- LR_u: The value of this parameter depends on the network and the video encoding. For our experiments we use $LR_u = 0.02$.
- $R_{increase}$: The value of this parameter depends on the network and the video encoding. We can say that the higher value of this parameter result the faster allocation of available bandwidth and the opposite.
- $R_{descrease}$: The value of this parameter is very important because determines the behaviour of the adaptive streaming application during network congestion. We can say that high value of this parameter results in "bad" behaviour to other network users during network congestion and the opposite. In addition we can say that low values of this parameter results in the phenomenon of "slow start" and "bad" behaviour to network congestion with small duration (this is one of the major drawbacks of TCP rate control). In addition the value of this parameter depends to the dominant traffic type the network as we ascertain to our experiments.

A major result of our experiments is that the choice of the above parameters is a trade off. When we changed one parameter in order to improve the behaviour of the adaptive streaming application to one point, the behaviour of the adaptive streaming application worsened to other points. The choice of the above parameters depends on the network and especially on the dominant traffic in the network. To our experiment, we show that the adaptive streaming application, which we have implemented, can have friendly behaviour to the dominant traffics (TCP traffic and UDP traffic) of today's Internet, with the use of the appropriate parameters.

3.4 Quality adaptation

Quality adaptation module is based on rate shaping technique. According to rate shaping technique, if we change some parameters of the encoding procedure, we can control the amount of the data that the video encoder produces (either increase or decrease the amount of the data) and as a result we can control the transmission rate of the multimedia data.

The implementation of rate shaping techniques depends on the video encoding. Rate shaping techniques change one or more of the following parameters:

- *Frame rate:* Frame rate is the rate of the frames, which are encoded by video encoder. Decreasing the frame rate can reduce the amount of the data that video encoder produce but will reduce quality.
- *Quantizer:* Quantizer specifies the number of DCT coefficients that are encoded. Increasing the quantizer decreases the number of encoded coefficients and the image is coarser.
- *Movement detection threshold:* This is used for inter-frame coding, where the DCT is applied to signal differences. The movement detection threshold limits the number of blocks which are detected to be "sufficiently different" from the previous frames. Increasing this threshold decreases the output rate of the encoder.

In this paper, we use the rate shaping capabilities of the implementation platform (JAVA-JMF) in order to implement the quality adaptation module of the adaptive streaming application.

4 IMPLEMENTATION

The implementation of the above described adaptive streaming application is based on JAVA technology and more particular to Java Media Framework (JMF) Application Programmable Interface (API). JMF provides capabilities for importing in JAVA applications and applets time-based media like video and audio. JMF supports the most common audio and video formats like AIFF, AU, AVI, GSM, MIDI, MPEG, QuickTime, RMF and WAV. In addition JMF supports transmission of real time data (like multimedia) with the use of RTP/RTCP. More over JMF gives capabilities for elaboration of audio and video data during the transmission over the network with the use of RTP/RTCP. All the above characteristics of JMF, make JMF an attractive platform for the implementation of adaptive streaming applications. More information about RTP/RTCP can be found in [18].

5 EXPERIMENTS

In this section, we present a number of experiments that we made in order to analyse the behaviour of the adaptive streaming application. Primary aim of the experiments was the study of adaptive streaming application behaviour regarding the dominant traffic model of today's Internet.

Experiments realisation to the Internet is a difficult procedure due to the fact that Internet is a very

changeable environment and it is difficult to achieve the same fixed conditions in every experiment. In the Internet, it is possible accidental events affect the results of the experiments.

For the above reason and in order to realise our experiments to a stable network environment, we used the following environment as testbed: With the use of ATM infrastructure, we established a virtual circuit (VC), with CBR characteristics, between the server and the client of the adaptive streaming application. The VC had capacity 300 kbps and at the same time we transmitted video with the use of adaptive streaming application and TCP or UDP traffic in order to evaluate the behaviour of the adaptive streaming application.

The video file, which was transmitted through the VC, was encoded to ITU-H.263 encoding format and the encoder initially was producing data with a rate of 50 kbps. The video file had 5 minutes duration and dimension 177*144 pixels (CIF format).

Although, the implemented adaptive streaming application has the capability to transmit high quality video (for example MPEG video at 2-3 Mbps), we chose for the realisation of our experiments to use transmission of low quality video in order to ensure that the delay jitter which was appearing during the experiments was originated by the network and not by the server (host included delay jitter). The transmission of high quality video is a very resource consuming procedure and if the CPU of the server is not enough powerful, it is possible to result host included delay jitter.

We realise two experiments:

- Transmission of adaptive video and UDP traffic at the same time: During this experiment, we investigate the behaviour of adaptive streaming application against UDP traffic. UDP traffic is important because as the transmission of multimedia data over the Internet increases at the same time increases the UDP traffic to the Internet.
- Transmission of adaptive video and TCP traffic at the same time: During this experiment, we investigate the behaviour of adaptive streaming application against TCP traffic. TCP traffic is important because it is the traffic that produce popular applications like the web (HTTP) and the FTP.

5.1 Experiment one: Transmission of adaptive video and UDP traffic

In this experiment, we transmit at the same time video with the use of the adaptive streaming application and UDP traffic through the VC. During this experiment, we investigate the behaviour of adaptive streaming application against to UDP traffic.

During this experiment, we follow the following scenario: Initially, we transmit only video to the VC with the use of adaptive streaming application. Two minutes after the beginning of video transmission, we transmit for the following two minutes, together with the adaptive video, UDP traffic through the VC. After the end of UDP traffic, the transmission of adaptive video continues for one minute until the video files ends.

Figures 3 and 4 displays the video transmission rate, the packet loss rate and the delay jitter during the experiment one.

In all charts, in this section the video transmission rate refers only to video data (payload) and does not include headers data. The header overhead (RTP header, UDP header, IP header and ATM cell header) during our experiments is significant. For this reason, the video transmission rate never reaches all the capacity of the VC (300kbps).

During this experiment the parameters which control the behaviour of the adaptive streaming application had the following values $\alpha = 0.5$, $\beta = 0.8$, $\gamma = 2$, $LR_c = 0.05$, $LR_u = 0.02$, $R_{increase} = 20.000$ (bps), $R_{decrease} = 0.85$.

Initially, the adaptive streaming application starts to transmit video with transmission rate 50 kbps and gradually reserves all the available bandwidth in the VC.

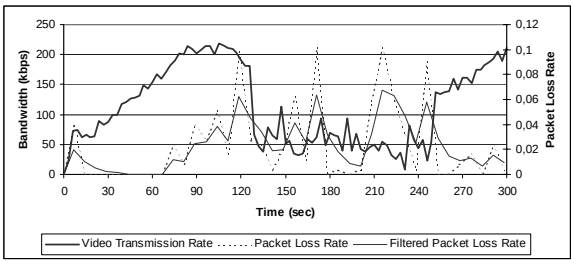

Figure 3. Video transmission rate and packet loss rate during experiment one

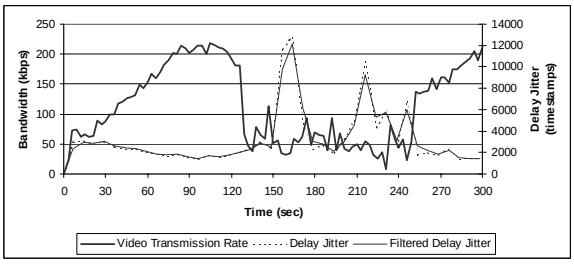

Figure 4. Video transmission rate and delay jitter during experiment one

In minute two, when the transmission of UDP traffic starts, congestion occurs to the network and the adaptive streaming application perceives the congestion condition through the increase of packet loss rate and delay jitter. Due to congestion condition, adaptive streaming application reduces its transmission rate near to 50% and keeps this transmission rate for the next two minutes, during

which the transmission of UDP traffic takes place. When the transmission of UDP traffic stops (minute 4) the adaptive streaming application gradually reserves again all the available bandwidth.
From the figure 3 and 4, it is obvious that the adaptive streaming application has "friendly" behaviour to UDP traffic. It is important for all network users to have "friendly" behaviour to other users for the proper operation of a network. The fact that the transmission rate of the adaptive streaming application was dropped beyond 50% of the total available bandwidth during this experiment, is because UDP traffic (that we use during our experiment) did not have any congestion control policy.
In addition, figure 4 shows, how important is the use of delay jitter as network congestion indication. During, this experiments, some times the packet loss rate was indicating load network (0.02<packet loss rate<0.05) but the rapid increase of delay jitter was indicating correctly network congestion.

5.2 Experiment two: Transmission of adaptive video and TCP traffic

In this experiment, we transmit at the same time video with the use of the adaptive streaming application and TCP traffic through the VC. During this experiment, we investigate the behaviour of adaptive streaming application against to TCP traffic.
During this experiment, we follow the following scenario: Initially, we transmit only video to the VC with the use of adaptive streaming application. Two minutes after the beginning of video transmission, we transmit, together with the adaptive video, TCP traffic through the VC. After the end of TCP traffic the transmission of adaptive video continues for one minute until the video files ends. When the video transmission starts, the adaptive streaming application has a transmission rate of 50kbps.
Initially, we use the same values with the first experiment for the parameters that control the behaviour of adaptive streaming application ($\alpha = 0.5$, $\beta = 0.8$, $\gamma = 2$, $LR_c = 0.05$, $LR_u = 0.02$, $R_{increase} = 20.000$ (bps), $R_{decrease} = 0.85$). Figures 5 and 6 displays the video transmission rate, the packet loss rate and the delay jitter during this experiment.

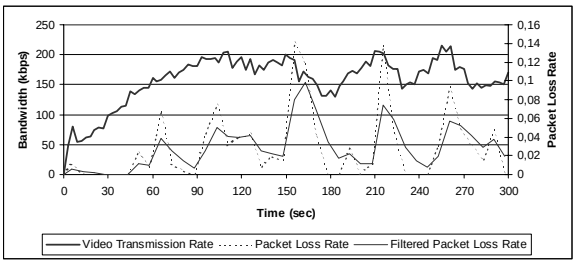

Figure 5. Video transmission rate and packet loss rate during experiment two with $R_{decrease} = 0.85$

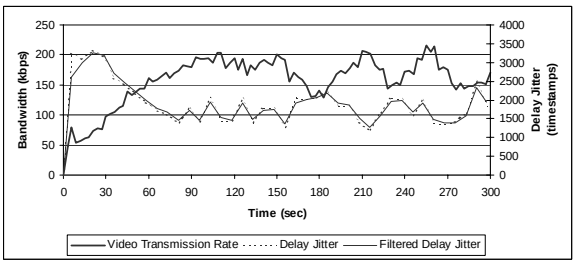

Figure 6. Video transmission rate and delay jitter during experiment two with $R_{decrease} = 0.85$

As someone can see in figures 5 and 6, with the above parameters the adaptive streaming application does not have "friendly" behaviour to TCP traffic. When the transmission of TCP traffic starts, the adaptive streaming application releases only small amount of bandwidth, which is re-allocated in few seconds. The reason for the above behaviour is the different congestion control policies of TCP traffic and adaptive streaming application. In the case of network congestion, the adaptive streaming application decreases its transmission rate to 85% ($R_{descrease} = 0.85$) and the TCP traffic decreases its transmission rate to 50%. The more aggressive behaviour of adaptive streaming application in congestion control has as result the TCP traffic not to be able to allocate enough bandwidth. This behaviour of adaptive streaming application is not desirable because TCP traffic is one of the dominant traffics of today's Internet. In order to adjust the behaviour of adaptive streaming application to be "friendly" with TCP, we repeat the above experiment and we set the parameter $R_{decrease} = 0.5$. With this change, the congestion control policy of the adaptive streaming application becomes "friendly" to TCP congestion control policy. Figures 7 and 8 display the video transmission rate, the packet loss rate and the delay jitter during the repeat of the experiment.

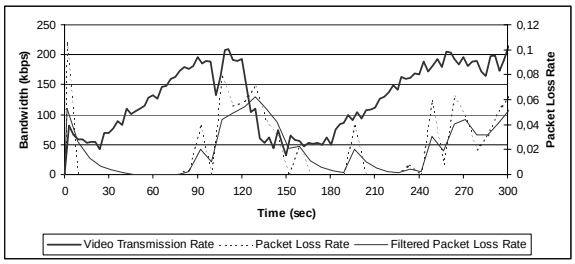

Figure 7. Video transmission rate and packet loss rate during experiment two with $R_{decrease} = 0.5$

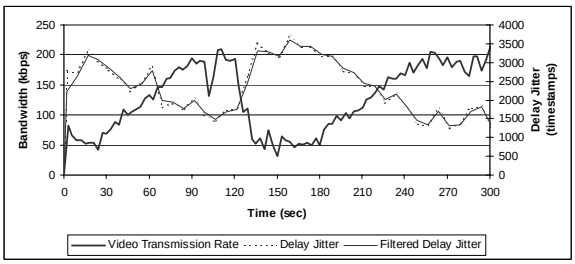

Figure 8. Video transmission rate and delay jitter during experiment two with $R_{decrease} = 0.5$

As someone can see in figures 7 and 8, with the change of parameter $R_{decrease}$ the behaviour of adaptive streaming application to TCP traffic becomes better. When we start to transmit TCP traffic and network congestion occurs, adaptive streaming application releases bandwidth in order to be used by the TCP traffic. Consecutively, the adaptive streaming application keeps steady its transmission rate until the transmission of TCP stops. Then the adaptive streaming application gradually reserves again all the available bandwidth.

The better behaviour of adaptive streaming application is because adaptive streaming application and TCP traffic follow the same congestion control policy: when congestion occurs they decrease the transmission rate to 50%. From the figure 7 and 8, it is obvious that both packet loss rate and delay jitter indicates network congestion.

6 CONCLUSION - FUTURE WORK

In this paper we present the architecture and the implementation of an adaptive streaming application. During the design of this application we are concentrating to the implementation of a mechanism for monitoring the network condition and estimating the appropriate rate for the transmission of the multimedia data. The implementation of the adaptive streaming application is based on JAVA technology and uses RTP/RTCP protocols for the transmission of multimedia data, in order to achieve interoperability with all ready implemented tools like mbone tools.

Through a number of experiments, we draw the following conclusions:

- The adaptive streaming application, with the appropriate selection of parameters, can have "friendly" behaviour to the dominant traffic of today's Internet.
- Except packet loss rate, delay jitter can be used for network congestion indication.

Our future work includes experiments in an Internet environment, in order to evaluate the behaviour of the adaptive streaming application to the combination of traffics types in a changeable network environment. In addition, we plan to implement a quality adaptation mechanism based on frame dropping technique. Furthermore, we plan to design and implement a selective retransmission technique in order to handle with the packet losses during the transmission of multimedia data.

REFERENCES

[1] C. Cowan, S. Cen, J. Walpole, C. Pu. "Adaptive Methods for Distributed Video Presentation", ACM Computing Surveys, 27(4), pp 580-583, December 1995. Symposium on Multimedia.

[2] C. K. Hess, Master's Thesis, "Media Streaming Protocol: An Adaptive Protocol for the Delivery of Audio and Video Over the Internet", University of Illinois at Urbana-Champaign, 1998.

[3] R. Rejaie, D. Estrin, and M. Handley, "Quality Adaptation for Congestion Controlled Video Playback over the Internet" in Proc. of ACM SIGCOMM '99, Cambridge, Sept. 1999.

[4] J. Walpole, R. Koster, S. Cen, C. Cowan, D. Maier, D. McNamee, C. Pu, D. Steere, and L. Yu, "A player for adaptive mpeg video streaming over the internet," in Proceedings of the 26th Applied Imagery Pattern Recognition Workshop AIPR-97, SPIE, (Washington DC), Oct. 1997.

[5] I. Busse, B. Deffner, and H. Schulzrinne, "Dynamic QoS control of multimedia applications based on RTP," Computer Communications, Jan. 1996.

[6] S. Jacobs, A. Eleftheriadis, "Adaptive Video Applications for Non-QoS Networks", Proc. 5 th International Workshop on Quality of Service (IWQoS'97), Columbia University, New York, USA, Pages 161-165.

[7] R. Ramanujan, J. Newhouse, M. Kaddoura, A. Ahamad, E. Chartier, K. Thurber, "Adaptive Streaming of MPEG Video over IP Networks", Proceedings of the 22nd IEEE Conference on Computer Networks (LCN'97), November 1997.

[8] S. Jacobs, A. Eleftheriadis, "Real-Time Video on the Web using Dynamic Rate Shaping", Department of Electrical Engineering, Columbia University New York, NY 10027.

[9] V. Tsaoussidis, S. Wei, "Reliability / Throughput /Jitter Tradeoffs For Real-time Transport Protocols", Department of Computer Science.
[10] R. Rejaie, M. Handley, and D. Estrin. "Architectural considerations for playback of quality adaptive video over the Internet", Technical Report 98-686, USC-CS, November 1998.
[11] R. Rejaie, M. Handley, and D. Estrin. "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet", Proc. IEEE Infocom, March 1999.
[12] P. Goyal, H. Vin, C. Shen, P. Shenoy, "A Reliable Adaptive Network Protocol for Video Transport", Technical Report UTEXAS CS//CSTR -95-18, The University of Texas at Austin, Department of Computer Sciences, July 1995.
[13] P. Mundur, A. Sood, R. Simon, "Network Delay Jitter and Client Buffer Requirements in Distributed Video-on-Demand Systems", Department of Computer Science George Mason University Fairfax, VA 22030.
[14] S. Cen, C. Pu,, J. Walpole, "Flow and Congestion Control for Internet Media Streaming Applications", In Proceedings of Multimedia Computing and Networking, 1998.
[15] B. Vandalore, W. Feng, R. Jain, S. Fahmy, "A Survey of Application Layer Techniques for Adaptive Streaming of Multimedia", Journal of Real Time Systems (Special issue on Adaptive Multimedia), April 99.
[16] Shculzrinne, Casner, Frederick, Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, IETF, January 1996.
[17] Shculzrinne, Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", RFC 1890, IETF, January 1996.
[18] Java Media Framework: http://java.sun.com/products/java-media/jmf/index.html