POLITECNICO FEDERALE DI LOSANNA
SWISS FEDERAL INSTITUTE OF TECHNOLOGY LAUSANNE

Communication Systems Division (SSC)
EPFL CH-1015 Lausanne, Switzerland
http://sscwww.epfl.ch

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Circuit Emulation over IP Networks

Raffaele Noro, Maher Hamdi and Jean-Pierre Hubaux

February 1999

Technical Report SSC/1999/002

Abstract

The best-effort service offered by the Internet is not suitable for a large variety of multimedia applications, which require strict guarantees on losses and end-to-end delay.

We design functionalities and mechanisms for RTP in order to provide a transport service with the properties of circuit switched networks and with the possibility of transporting both constant and variable rate streams.

We consider a source that generates *structured* data units, which are collected in RTP packets. These packets are forwarded to an IP network and reach the receiver with a stochastic delay. An extra delay is computed (on a per-packet basis) and added at the receiver to have delay equalization. The data units in the packet are then delivered to the application at a rate specified in the RTP header. The total delay Δ is a session parameter and can be adjusted by exchanging RTCP messages.

The dimensioning of the receiver buffer is the first problem to be solved. The buffer space results in a function of the network delay bounds and the maximum RTP packet size. This information can be conveyed from the source to the receiver through RTCP messages.

The timing of the data delivery process and the source clock recovery are also required functionalities. They are both implemented by means of an extension to the RTP header. We developed a method for source clock recovery that removes the network induced jitter by processing the clock indications contained in some RTP header. The recovered clock is then used in combination with other information contained in the packets to control the delivery process.

Simulations prove that the circuit emulation service can be implemented on top of the RTP with satisfactory performance.

1 Introduction

One of the most active areas of research is that of protocols and services for the transport of multimedia data: they include, for example ATM, RTP and RSVP. We restrict our attention to the transport of delay-sensitive data in the Internet. It is well known that the native service model offered by the Internet is best-effort [1]. Attempts to enrich this model have been made and are detailed in the literature [2, 3, 4, 5]: the common goal is to add QoS, to media streams in particular.

Multimedia applications may or may not accept the best-effort approach. Those that can wait for data to arrive and tolerate losses and varying delays are called asynchronous or adaptive applications: examples are Vic, RealAudio and RealVideo (based on RTP) [6, 7]. Their existence can not be taken as proof of the usefulness of such adaptivity. In contrast, those applications that have strict requirements of delay, delay variation and losses are called synchronous: a popular example is MPEG-2 [8] audio-video distribution.

The MPEG standard specification, for example, makes the assumption that coded and packed data are transported over a constant delay channel[1]: this property, which is natively supported only in circuit switched networks, poses a serious challenge to the current packet switched network infrastructure such as the Internet.

Our work presents a solution to this problem and proposes a Circuit Emulation Service (CES) layer provided on top of an Internet protocol stack. A mechanism for the equalization of the end-to-end delay is developed, by which network induced jitter is reduced within application tolerance limits, without introducing significant losses.

The equalization process performed by the CES layer does not exceed the delay bounds provided by the network: thus, if the underlaying network is interactive, the CES layer is able to support interactive applications. If the network is not interactive, the CES simply stops the propagation of network jitter to the application level.

The document is organized as follows. In Section 2 we formalize the problem. In Section 3 we propose a solution based on the RTP protocol. In Section 4 the problem of source clock recovery is studied. In Section 5 we give some concluding remarks.

2 Statement of the problem

The system to be studied is as shown by Fig. 1

The problem to be solved can be stated as follows: Given a network delay that varies arbitrarily between d_{min} and d_{max}, how it is possible to

[1] In fact, the standard relaxes this assumption by specifying a tolerance to ± 4 ms of variable delay for receiving equipments

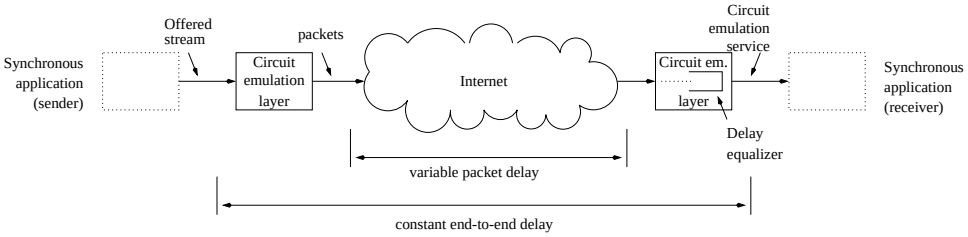


Figure 1: Scheme of the studied system

deliver data within an end-to-end delay $\Delta \pm \delta$ ($\delta < (d_{max} - d_{min})/2$ being a service tolerance), while still maintaining the discard rate below a given tolerance q? We decompose this problem in a set of sub-problems.

Delay equalization We call t_s^i the time when the i-th data unit enters the CES layer and t_r^i the time when it reaches the receiver's peer; we have the following relationship

$$t_r^i = t_s^i + d_i \tag{1}$$

where $d_i (> 0)$ counts both packing and transmission (varying) delays. Equalization consists of adding an extra delay at the receiver so that

$$t_{eq}^i = t_r^i + \Delta_i = t_s^i + \Delta \tag{2}$$

with Δ a constant. The timeline of this process is represented in Fig. 2

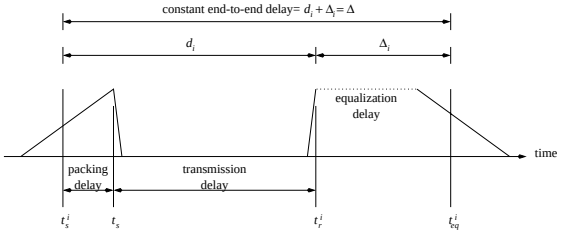


Figure 2: Representation of t_s^i, t_r^i, t_{eq}^i, d_i and Δ_i in the timeline

Buffer A buffer is provided to handle the variability of network delay. Its size must be large enough to backlog early arrivals but must be kept reasonably small to introduce the smallest possible total delay, as required in most synchronous applications.

As an example, consider a source of data units in the form of bytes emitted at a rate C bytes/s, $C_{min} < C < C_{max}$. If the network delay d (of a packet) is between d_{min} and d_{max} (in s), and packets are of fixed size P bytes, the following conditions hold:

maximum packing delay[2]

$$p_{max} = \frac{P}{C_{min}} \tag{3}$$

packing delay of the first data unit of the current packet

$$p = \frac{P}{C} \tag{4}$$

value of end-to-end delay after equalization

$$\Delta = d_{max} + p_{max} \tag{5}$$

value of equalization delay

$$\Delta_i = d_{max} + p_{max} - d_i \tag{6}$$

buffer space evaluated for the worst case of the arrival process

$$B = (d_{max} - d_{min}) \cdot C_{max} + p_{max} \cdot (C_{max} - C_{min}) + p_{max} \tag{7}$$

The assumption of bound jitter clearily simplifies the problem. In a general case, Δ and hence B would be chosen taking into account both losses and delay parameters of QoS.

Clock errors Eqs. 1 to 7 assume the existence of a global and accurate common clock. These conditions are rarely or never met in reality and this causes an additional problem. In the case where source and receiver use local and independent clocks t_s and t_r, we can assume the following relationship [9]

$$t_s = a \cdot t_r + b \tag{8}$$

where a is the frequency drift and b an initial phase offset.

Terms in the form t_r^2 are neglected. If $a = 1$ and $b \neq 0$, the Eq. 2 computes an error equivalent to b: this means that either the backlog is larger than B ($b > 0$, overflow), or data appear late and are discarded ($b < 0$, underflow). If $b = 0$ and $a \neq 1$, the consumption rate at the receiver differs from the emission rate at the sender by $C \cdot (a-1)$. In this case B data are consumed faster ($a > 1$, underflow) or slower ($a < 1$, overflow) within the time T

$$T = \frac{B}{C \cdot |a - 1|} \tag{9}$$

causing losses or stops to the application (for simplicity, we have assumed a constant rate C).

[2]assumption is made on the fact that emission rate is changed only at packet boundaries

In general, whereas b can be manually or automatically compensated, computer clocks tend to have drifts $(a-1)$ to the order of 10^{-3} (few minutes per week or so, [10]). With jitter to the order of 100 ms for the Internet and C to the order of 10 Mbit/s for ordinary multimedia applications, B can be relatively small (125 kbytes) but also T is to order of 100 s, which means that losses or stops occur after one or few minutes of service.

One attempt to solve this probelm in the Internet can be considered the Network Time Protocol (NTP, [11]): it is however of little use in the distribution of multimedia data, but is well designed to support and control distributed applications, like NFS, that have only loose synchronization requirements (ordering of events and so on). The problem with NTP is that its accuracy, dozens of ms, is unappropriate for most multimedia applications (e.g., MPEG-2) and also that its accuracy fluctuates in response to network overloads [12]. Moreover, there is no guarantee that a sender and a receiver are synchronized with the same synchronization source (i.e., $b \neq 0$ in Eq. 9).

The listed problems are studied in the remainder of this paper.

3 Circuit emulation based on RTP

The Real-Time Protocol (RTP, [13]) is generally implemented as part of the application and operates as the network adaptation layer. We design an enhancement layer for circuit emulation: it enhances the transport service of RTP and can be integrated in existing implementations (e.g., RFC2250, [14]) or used as an independent application of RTP.

The description of the fuctions and the necessary protocol is given in Sections 3.1 and 3.2

3.1 Functions

The functions to be performed depend on the requirements of the higher layer or application.

The CES should provide:

- structuring of the data units to be delivered (can be any group of bits and bytes, including a whole packet)
- delivery of data units at the specified rate (variable)
- constant end-to-end delay (of individual data units)
- indication of losses and errors (which is not addressed here)

The functions that support this service include: handling of packet delay variation, handling of data structures and source clock recovery and reconstruction at the receiver

Handling of packet delay variation A buffer in the receiver is used to handle variable delay. Its size can be computed by knowing the characteristics of the network jitter and the packetization process at the source: Eq. 7 gives the buffer space when the network delay and packing delay bounds are known. The actual information about network delay characteristics and packing delay can be provided to the receiver through RTCP messages.

Handling of data structures The data structure is uniquely identified during the RTP session and must be communicated by the source at the session setup, via an RTCP message. It can be dynamically changed during the session. Data structures can be any group of bits or bytes and even whole packets[3] that are exchanged between the CES layer and the application. The temporal space between consecutive data units τ is expressed in units of the source clock and is encoded in each RTP packet header.

The instant of delivery of a data unit i is computed from the timestamp value t_s. With reference to Fig. 2 and Eqs. 1 to 7, the delivery instant is

$$t_{eq}^{i} = t_s - p + \Delta + i \cdot \tau \tag{10}$$

that gives an end-to-end delay of exactly Δ for all data units in a session.

Source clock recovery This is probably the most crucial issue. A recovered source clock must have satisfactory jitter and drift performance. The source clock is derived from a timebase available at both sender and receiver. For interoperability with existing implementations, we consider that the common timebase has a frequency resolution f_n equivalent to the 4.3 GHz resolution of the NTP (at most). The source frequency is expressed as f_n/x, where x is a rational. Information about the source clock is periodically inserted by the source to allow the receiver to adjust its clock drift. The receiver has the choice of whether to ignore this information (and use only the local clock or an NTP synchronized clock) or to process the information. In the preferred implementation, neither source nor receiver use NTP, but a session clock, which is obtained by slaving the receiver clock to the source clock.

3.2 Protocol

The protocol stack consists of a CES layer between the RTP and the application layers (Fig. 3)

All the information is encoded in different fields of RTCP and RTP messages. RTCP packets carry information about the network delay variation

[3] in RFC2250 [14] a single Transport Stream packet can be encapsulated in an RTP packet (although it is not an efficient scheme). In this case it is the whole payload of the RTP packet that is delivered to the Transport Stream decoder

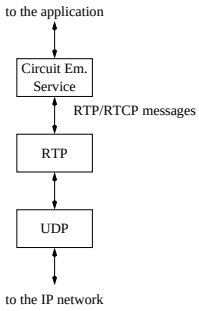
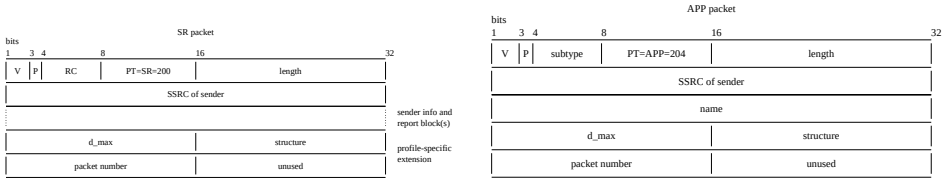


Figure 3: Protocol stack of the circuit emulation service

statistics (i.e., d_{max} and $E\{d\}$) and the structure of data units. A 32-bit field of a Sender Report (SR) or Application-defined (APP) RTCP message supports this function. It is partitioned in two 16-bit fields: the first field carries the value of d_{max} in units of the source clock, the second field indicates the length of data units in bits. Fig 4 illustrates both cases.

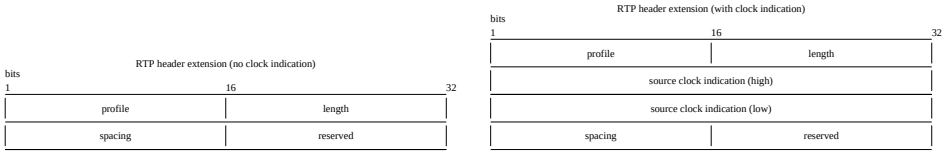


V: version of RTP
P: padding, set to one if padding octets are included at the end of packet
RC: number of report blocks in the packet
PT: payload type
length: length of packet in 32-bit words minus one
SSRC: synchronization source originating the packet
subtype: to allow subtypes or application-dependent data
name: 4 octets to identify the name of the APP

d_max: maximum packet delay in source clock units (zero if undefined)
structure: length of data units (in bits, zero means one bit)
packet number: first RTP data packet subject to d_max and structure

Figure 4: Usage of RTCP packets

RTP packet header extensions contain the temporal spacing (τ), the source clock information and the timestamp. The RTP packet payloads contain the application data. In the header extension, a first 16-bit field is used to encode temporal spacing in units of the source clock, a second 64-bit field (if present) carries the source clock indication. The overhead introduced by the last field must not exceed 0.1 % of the session traffic (i.e., one source clock indication field each 64 ms for a 1 Mbit/s session). This is illustrated by Fig. 5



profile: defines the CES
length: length of header extension in 32-bit words minus one (either 1 or 3)

spacing: temporal distance between data units in source clock units
source clock indication: absolute value of source clock in units of 4.3 GHz
reserved: for future use

Figure 5: Usage of RTP packets

The functions and information flow at the CES receiver are represented in Fig. 6. The buffer space in the middle is controlled by the value of d_{max}. The data delivery depends on the timestamp, the data structure and the temporal spacing. Processing of source clock indications provides clock synchronization.

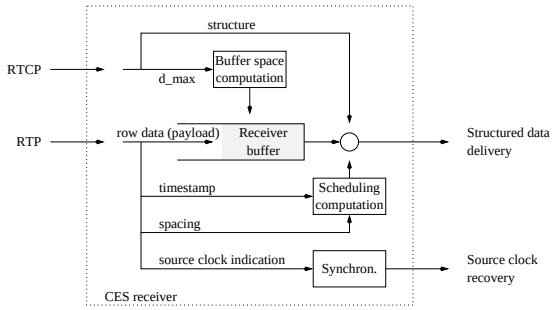


Figure 6: Scheme of the CES receiver

4 Source clock recovery

One of the fundamental issues in circuit emulation is the recovery and reconstruction of the source clock. The performance of a source clock recovery method is evaluated through its accuracy and its stability: accuracy is the variance σ_o^2 of the clock error between sender and receiver, whereas stability is the relative frequency error. The method discussed here is based on a Least square Linear Regressive (LLR) estimation. An equivalent method has been proposed by the authors for the system clock recovery of MPEG-2 decoders [15].

LLR method The time model that relates the source and receiver timebase is given in Eq. 8 and recalled here

$$t_s = a \cdot t_r + b \tag{11}$$

a and b represent frequency drift and clock offset if the clocks were free-running. To slave the receiver clock it is necessary to provide an estimate $(\hat{a}, \hat{b})$ of both terms. They are obtained through the processing of the arrival instants (in t_r units) of the source clock indication field of RTP messages (in t_s units). By minimization of the Mean Square Error (MSE), the values of $\hat{a}$ and $\hat{b}$ become

$$\hat{a} = \frac{m \sum t_{s_i}^2 - (\sum t_{s_i})^2}{m \sum (t_{s_i} t_{r_i}) - (\sum t_{r_i} \sum t_{s_i})} \tag{12}$$

$$\hat{b} = \frac{\sum t_{s_i} \sum (t_{r_i} t_{s_i}) - (\sum t_{r_i} \sum t_{s_i}^2)}{m \sum (t_{s_i} t_{r_i}) - (\sum t_{r_i} \sum t_{s_i})} \tag{13}$$

The computation is performed on the m last received source clock indications and uses recursion: this is known as the Durbin algorithm and is based on the assumption of stationary distribution of network jitter [16]. Similar methods for non-stationary distributions exist, but are not treated here.

The stability, in the steady state, depends on the duration of the locking phase (transient): the longer the transient, the better the stability. In fact, the frequency stability (maximum frequency error) is proportional to $1/m^2$, while transient duration is m samples at most. Accuracy is proportional to $1/m^2$ and to the variance σ_i^2 of the network jitter. It is a good practice to choose values of m larger than 100, as network jitter is measured in hundreds of ms and applicatons are tolerant of residual jitter to the order of few ms.

Second order Phaselock Loops (PLLs) can be used, however their performance is generally limited, because of the high amount of jitter. In fact, considering the stability ε, the accuracy σ_o^2 and the transient duration D, the following relationship holds [17]

$$D \cdot \sigma_o^2 = -log(\varepsilon) \cdot \sigma_i^2 \tag{14}$$

Eq. 14 can be interpreted as follows: if network jitter is large and accuracy and stability are maintained (close to zero), the D must be increased up to several tens of thousands, giving transient durations of minutes or so. Buffers can thus underflow/overflow before they reach the steady state and cause severe data loss.

Performance The overall performance mainly depends on the jitter absorption capability of the source recovery method. In our experiment we

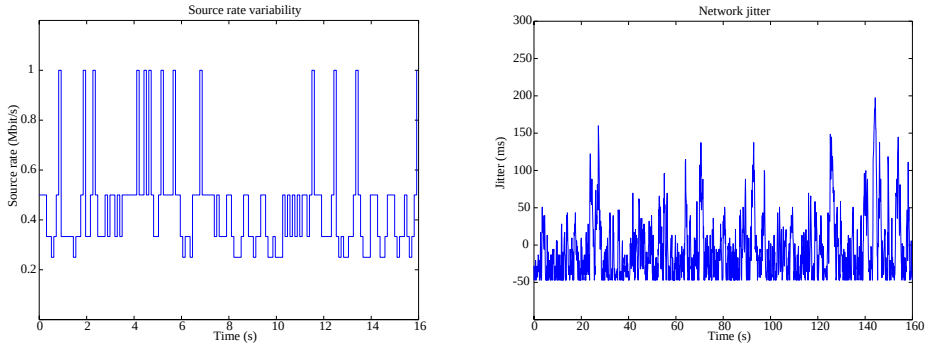
simulate a source, a network and a receiver with the the following characteristics:

- Source emission rate varying between 1 Mbit/s and 250 kbit/s, as shown by Fig. 7 a) for an interval of 16 s;
- Size of data units of one byte;
- Size of RTP packet payload of 188 bytes;
- Geometric distribution of the network jitter with 100 ms of minimum delay, 780 ms of maximum delay and 154 ms of average delay;
- Window length of the source recovery method $m = 1000$;
- Insertion of one source clock indication each 100 ms;
- Simulated duration of 160 s

The network jitter model is consistent with the results presented in [18]. The interarrival process of packets emitted at regular intervals δ fits with a geometric distribution

$$\text{Prob}\{(t_r^{i+1} - t_r^i) = k \cdot \delta, k \geq 0\} = p \cdot (1-p)^k \tag{15}$$

where p is a decreasing function of δ. In our experiment we have taken $\delta = 3.4$ ms and $p = 0.52$ [4]. We also assumed lossless transmission.



a) b)

Figure 7: Source rate in a) and jitter of clock indications in b) for the experiment

We first study the performance of the source clock recovery method. The source clock indications are subject to the variable delay shown in Fig. 7 b). Fig. 8 shows the recovered clock accuracy during the steady state

[4]This means that 52 % of packets are assumed to reach the receiver in bursts, as consequence of network overload. δ is the average of the packet emission interval

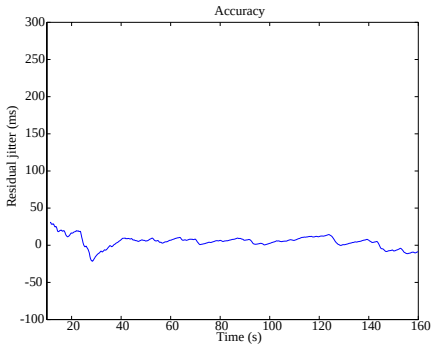
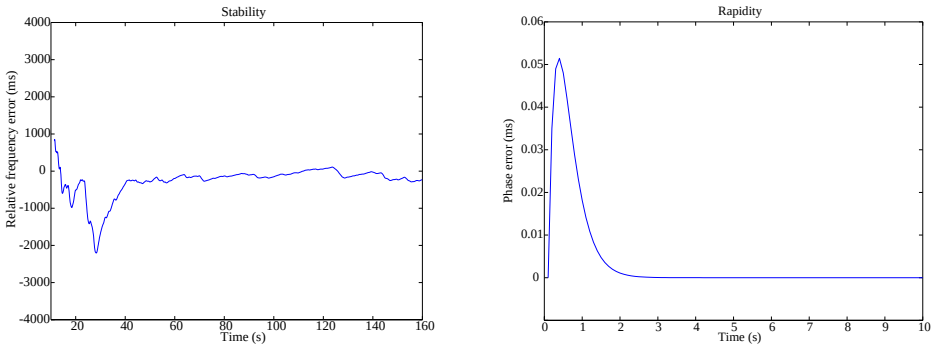


Figure 8: Accuracy of the recovered clock

With this value of m, the residual jitter is reduced by a factor of about 10. The frequency stability is to the order of 10^{-4} in the steady state region, as shown by Fig. 9 a). The rapidity is expressed by the time needed to reach the steady state. The Fig 9 b), obtained when the network jitter is muted, shows that the steady state is reached in 3 s when the receiver has an initial frequency drift of 10^{-3}. However, for optimal performance in the presence of jitter, a value ten times larger should be considered.



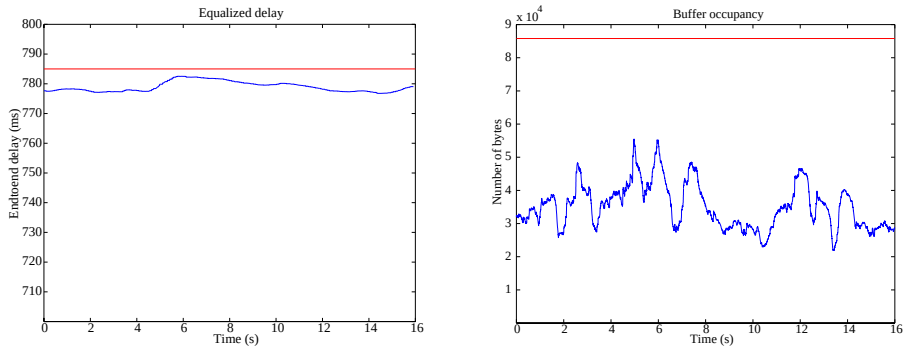
a) b)

Figure 9: Stability in a) and rapidity in b)

The end-to-end equalized delay of the data delivery process is directly affected by the residual jitter of the clock. Fig 10 a) shows the delay experienced by each delivered data unit during 16 s of the experiment

The receiver buffer occupancy reflects the fact that packets are generated and consumed at different speeds. As shown by Fig. 10 b), occupancy increases during bursts and tends to be lower during silences

In this experiment, we also measured a discard rate q of about 0.2 %, as a consequence of overflows/underflows of the receiver buffer. They are essentially due to the clock recovery error at the beginning of the session.



a) b)

Figure 10: Effect of residual jitter in a) and buffer occupancy in b)

5 Conclusion

Today data at medium-high bitrate can be transported on the Internet, but only loose guarantees on losses and end-to-end delay are provided.

Our intention is to enhance the basic service offered by the Internet in order to meet the requirements of multimedia applications that have strict QoS needs: the solution proposed here consists in having a Circuit Emulation Service (CES) on top of the RTP.

The CES delivers structured data to a receiver and guarantees constant end-to-end delay for each transported data unit, as well as sequencing.

The constant delay is obtained by buffering data at the receiver in order to equalize the delay of early arrivals. The amount of equalization delay, as well as the needed buffer space, depend on the delay variability (of the network) and rate variability (of the source). This is detailed in the first part of the paper.

The implementation in RTP consists of the definition of an RTP header extension (that carry information about source clock and source rate) and of an RTCP message type (that carry information useful to compute buffer space at the receiver). The implementation is discussed in Section 3.2.

In addition, we developed a source clock recovery method that is suitable to operate under the jitter conditions of the Internet. It is based on a Least-square Linear Regressive (LLR) method and is able to lock the receiver clock with the source clock, in order to efficiently control the data delivery process.

We have shown that it is possible to provide acceptable loss and end-to-end delay performance in the presence of the jitter introduced by the Internet.

The study presented here applies to the best-effort Internet. It is however worth to notice that the perspective of having resource-reservation can possibly alleviate but not remove the problem of network jitter and does

not change the nature of the study (that would result similar to the case of ATM type 1 AAL [19]).

Future work will include the implementation of a real CES terminal connected to the Internet and the optimization of the source clock recovery method to counteract the effects of different network jitter profiles.

References

[1] R Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", RFC1633, June 1994.

[2] L. Zhang, R. Braden, D. Estrin, S. Herzog, and S. Jamin, "Resource ReServation Protocol (RSVP) - Version 1 Functional Specification", draft-ietf-rsvp-spec-16.txt, June 1997.

[3] S. Shenker, C. Partridge, and R. Guérin, "Specification of guaranteed quality of service", draft-ietf-intserv-guaranteed-svc-06.txt, Aug. 1996.

[4] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services", draft-ietf-diffserv-arch-02.txt, Oct. 1998.

[5] D. Ferrari, "Multimedia network protocols: where we are?", *IEEE Multimedia Systems*, vol. 4, pp. 299–304, Dec. 1996.

[6] S. McCanne and V. Jacobson, "vic: A Flexible Framework for Packet Video", *in ACM Multimedia*, San Francisco, CA, USA, Nov. 95.

[7] "RealNetworks", http://www.real.com/.

[8] ISO/IEC, *Information technology - Generic coding of moving pictures and associated audio information - Part 1: Systems*, Oct. 1994, DIS 13818-1.

[9] F. Cristian, "Probabilistic clock synchronization", *Distributed Computing*, vol. 3, pp. 146–158, 1989.

[10] R. Wu, "clock drift in s/Linux?", Nov. 1997, http://www.geog.ubc.ca/s_linux/sparclinux-1997/msg01173.html.

[11] D.L. Mills, "Internet Time Synchronization: The Network Time Protocol", *IEEE Transactions on Communications*, vol. 39, pp. 1482–1493, Oct. 1991.

[12] B. Siegell, "Effect of clock drift on delay measurements", May 1998, http://felix.bellcore.com/~jjd/bss/clockdrift.html.

[13] H. Shulzrinne, S. L. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC1889, Jan. 1996.

[14] D. Hoffman, G. Fernando, V. Goyal, and M. Civanlar, "RTP Payload Format for MPEG1/MPEG2 Video", RFC2250, Jan. 1998.

[15] R. Noro and J.-P. Hubaux, "Clock Synchronization of MPEG-2 Services over Packet Networks", to appear in Telecomm. Systems Journal, Baltzer Ed.

[16] A. A. Giordano and F. M. Hsu, *Least Square Estimation with Applications to Digital Signal Processing*, John Wiley & Sons, Inc., New York, USA, 1985.

[17] F. M. Gardner, *Phaselock Techniques*, John Wiley & Sons, Inc., New York, USA, 1979.

[18] J.-C. Bolot, "Characterizing End-to-End Packet Delay and Loss in the Internet", *Journal of High-Speed Networks*, vol. 2, pp. 305–323, Dec. 1993.

[19] ITU-T Study Group 13, "B-ISDN ATM Adaptation Layer Specification, Type 1 AAL", ITU-T Recommendation I.363.1, Aug. 1996.