



Audio Engineering Society Convention Paper

Presented at the 120th Convention
2006 May 20–23 Paris, France

This convention paper has been reproduced from the author's advance manuscript, without editing, corrections, or consideration by the Review Board. The AES takes no responsibility for the contents. Additional papers may be obtained by sending request and remittance to Audio Engineering Society, 60 East 42nd Street, New York, New York 10165-2520, USA; also see www.aes.org. All rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

Network Music Performance (NMP) in narrow band networks

Alexander Carôt¹, Ulrich Krämer², and Gerald Schuller²

¹ISNM - International School of New Media, Willy-Brandt-Allee 31c, 23554 Lübeck, Germany

²Fraunhofer Institute for Digital Media Technology IDMT, Ehrenbergstr. 29, 98693 Ilmenau, Germany

Correspondence should be addressed to Alexander Carôt (Carot@isnm.de)

ABSTRACT

Playing live music on the Internet is one of the hardest disciplines in terms of low delay audio capture and transmission, time synchronization and bandwidth requirements. This has already been successfully evaluated with the Soundjack software which can be described as a low latency UDP streaming application. In combination with the new Fraunhofer ULD Codec this technology could now be used in narrow band DSL networks without a significant increase of latency. This paper first describes the essential basics of network music performances in terms of soundcard and network issues and finally reviews the context under DSL narrow band network restrictions and the usage of the ULD Codec.

1. GOAL

The term NMP (Network Music Performance) typically describes an online music event in which musicians can perform as if in the same room though separated by potentially thousands of kilometers [1][8]. In order to provide a typical rehearsing atmosphere, the audio quality should be 48kHz/16Bit and the one-way delay has to be less than the EPT (ensemble performance threshold) of 25ms (ensemble performance threshold) [3][5]. Due to these restrictions, NMP was feasible in broadband LANs only in the case that their physical distance did not exceed a one-way ping of 15ms and the connec-

tion was stable in terms of network jitter. Assuming these conditions the soundcard typically works with 48kHz/16Bit and a frame size of 256 Samples/frame resulting in a blocking delay of 5.3ms. With low latency kernel patches and better hardware or drivers 128 or even 64 samples/frame can be achieved. Though ADSL connections only provide a much lower bandwidth and the low frame sizes generate a larger overhead on the ADSL protocol stack, the NMP principle can be applied to DSL networks.

2. PROBLEM

In terms of bandwidth and delay, NMP in narrow

band networks (i.e. DSL) used to be declared as impossible. First of all, even a simple ICMP (internet control message protocol) echo response time from a remote peer often reached values beyond the 50ms R-EPT (roundtrip ensemble performance threshold) and secondly the upstream of typically 128 to 512 kBit/s prevented the signal to be sent out over these links ($48\text{kHz} * 16\text{Bit} * 1 \text{ channel} = 768 \text{ kBit/s}$).

When using audio compression techniques to reduce the required bit-rate, conventional audio coders introduce too much encoding/decoding delay for NMP purposes. For instance, standard coders like MP3 or AAC have a delay of about 100ms or more. Even AAC-Low Delay still features about 20ms at 48kHz sampling rate [6].

Moreover, conventional Quality of Service (QoS) approaches for reducing network jitter don't work with NMP since they cause too much additional delay by large jitter buffers and possible packet retransmission. Thus NMP requires a different QoS approach than VoIP or video-conferencing.

3. BASICS

The total latency with uncompressed audio is caused by the soundcard delay and the network delay. Though they interdepend in some aspects, as a first step they can be observed separately.

3.1. Network propagation delay

Assuming a direct path between two peers, the propagation delay for fiber optic lines is caused by a limited transmission speed of $\sim 0.7 * c$ (c : speed of light), which results in an average latency of 5ms each 1000km. Taking a look at the European backbone structure, straight-line paths do not exist. Thus packets always have to pass central routers and switches depending on the actual routing, which increases the physical distance between two peers. For example, the distance between Lübeck (northern Germany) and Copenhagen (Denmark) is about 400km, resulting in a propagation delay of 2ms. Watching the actual route, packets first travel to Frankfurt, Stockholm and finally to Copenhagen, which corresponds to a physical distance of 2300km and a theoretical propagation delay of 11.5ms. The real propagation delay is even higher due to facts we will describe later in this paper.

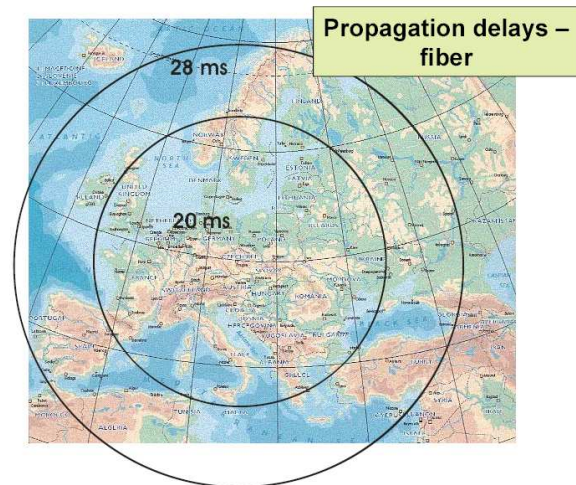


Fig. 1: Average propagation delay diameter [12]

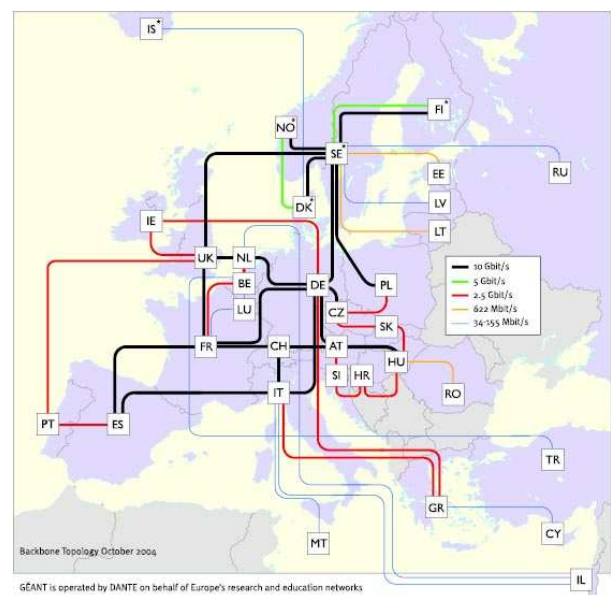


Fig. 2: European GEANT network backbone [7]

3.1.1. Network router and switch delay

An additional delay is caused by the number of passed switches and routers on the way to the destination. With each switch the packets are delayed by a small amount of microseconds but in terms of router delay the case is more complex. Depending

on the actual network traffic, routers queue the incoming packets more or less. In an ideal case scenario the packets are forwarded instantly and remain unqueued, but in case of a bandwidth overload the packets have to stay in the queue for an uncertain amount of time. This unpredictable effect typically causes the network jitter since it prevents a delay-constant packet propagation.

3.2. Interleaving in DSL networks

In DSL networks interleaving (IL) is used which increases the error tolerance when transmitting data between the local DSL peer and the DSL provider. If this line causes any (burst) disturbances, they can often be corrected via Forward Error Correction (FEC). But since data from a couple of consecutive blocks gets scrambled, this leads to a latency increase of about 30ms just for the first hop. With Fastpath (FP), which is the marketing name for a DSL line with switched off interleaving, this error correction doesn't work as well, as complete blocks of data get lost. Since the FEC overhead for IL as well as FP is put out of band, the channel capacity is not diminished by the additional FEC data.

Due to the higher latency, for NMP the interleaving has to be disabled.

3.3. Soundcard delay

The audio capture process happens in frames of x audio samples. The capture delay for one such block depends on the chosen sampling rate:

$$\text{blocking delay} = \text{frame size} / \text{sampling rate} \quad (1)$$

Since 48kHz has become a standard audio sampling rate, Table 1 shall show the corresponding blocking delay.

With standard components, frames of 256 samples can be reached easily, while frames of 128 samples often require optimized OS patches, hardware or drivers.

3.3.1. Total audio latency

The soundcard's audio processing typically works as a block processing system using one input double buffer (input buffer A and B) and one output double buffer (output buffer A and B):

frame size / samples	Blocking / ms
2048	42.7
1024	21.3
512	10.7
256	5.3
128	2.7
64	1.3

Table 1: Soundcard blocking delay at $f_s = 48\text{kHz}$

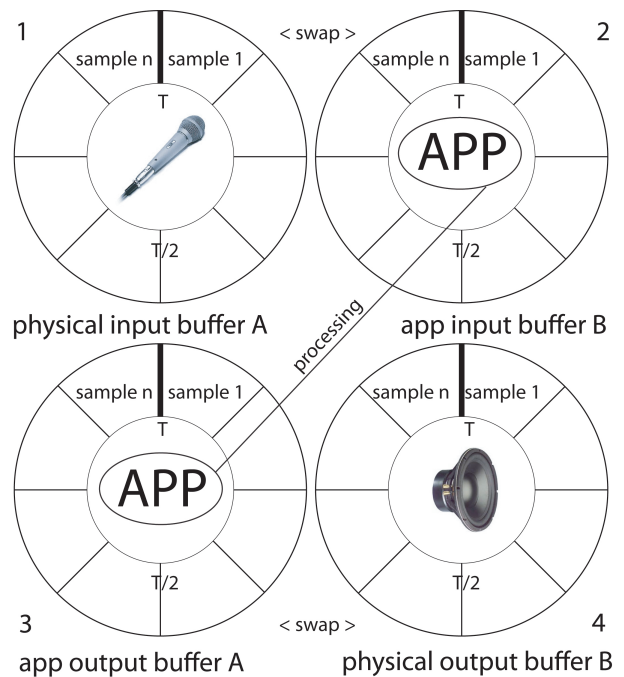


Fig. 3: Audio capture / playback process

The physical input is bound (i.e.. via DMA) to input buffer A; hence the application (APP) cannot access it. Respectively APP cannot access the physical output buffer B.

While the hardware (HW) writes block n to buffer A the APP reads from buffer B which was filled previously with block $n-1$ resulting in a delay shift of 1 block corresponding to the block size.

At that point the application can process the captured block $n-1$ and finally write it to out buffer A. At the same time the hardware reads from output buffer B, which was filled previously by output buffer

A. This again leads to delay shift of one audio block.

Once the block is read out, the buffers are swapped in order to process the next audio buffer (double-buffering system).

Regarding the whole audio capture and playout scenario the following processes happen at the same time :

- HW reads from phys. input to input buffer A
- APP reads input buffer B (1 block latency)
- APP processes the captured audio block
- APP writes to output buffer A (1 block latency)
- HW writes from buffer B to phys. output

Hence for playback, the captured audio block has to pass the playout buffers which corresponds to the same blocking delay. This in turn results in a total in/out delay of $2 * \text{blocking delay}$ plus possibly an additional millisecond for the ADC/DAC conversion depending on the soundcard's quality.

$$\text{total audio delay} = 2 * \text{blocking delay} \quad (2)$$

Table 2 shows the total introduced audio delay for the typical sampling frequency of 48kHz.

Frame size/samples	Blocking/ms	Total/ms
2048	42.7	85.3
1024	21.3	42.7
512	10.7	21.3
256	5.3	10.7
128	2.7	5.3
64	1.3	2.7

Table 2: Soundcard total delay at $f_s = 48\text{kHz}$

3.4. Dependencies

The lower the chosen frame size of the soundcard the more packets are sent in a specific time interval. In terms of payload this doesn't make any difference, but since each packet contains an IP and UDP header, and several other protocol headers depending on the actual network architecture, this results

in at least 28 bytes of packet overhead. Hence lower frame sizes result in a larger amount of packets/s, which in turn increases the overhead. Another disadvantage of lower frame sizes is that a large amount of small packages is more vulnerable to network jitter, thus the probability of any negative effect of network jitter is higher the smaller the sent packages are.

4. DSL PROTOCOL STACK OVERHEAD

In terms of home usage, internet via traditional voiceband modems works comparably simple. This is in sharp contrast to how ADSL networks with their various layers work. Though useful in terms of reliable ADSL network access, each of them imposes overhead on top of the existing net payload [4].

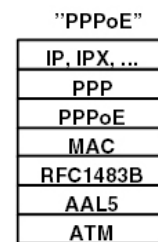


Fig. 4: PPPoE protocol stack

In comparison to "IP over ATM", "PPPoA", "RFC1483 Bridging" or other stacks the PPPoE protocol stack is the most complicated but has become the most commonly used.

In total, besides the IP overhead of 20 bytes and the UDP overhead of 8 bytes there are six other protocols to be taken into account. The PPP, PPPoE and the MAC layer introduce a fixed overhead of 26 bytes, and a minimum overhead of 23 bytes is introduced by the ATM related protocols (RFC1483B, AAL5, ATM).

Due to ATM restrictions (AAL padding), unfortunately the ATM overhead results is 70 bytes [4], which adds up in a total PPPoE-stack overhead of 96 bytes. This has a high impact on the DSL upload bandwidth due to the high amount of packets/s when sending audio with a very low frame size.

Protocol layer	Overhead / Bytes
PPP	2
PPPoE	6
MAC	18
RFC1483B	5
AAL	8
ATM	10
total	49

Table 3: Minimal PPPoE-stack overhead

$$\text{total overhead/packet} =$$

$$IP - Header + UDP - Header + \quad (3)$$

$$PPPoE - Stack - Headers =$$

$$(20 + 8 + 96) \text{ bytes} = 124 \text{ bytes}$$

It should be mentioned again, that this huge amount of overhead is for the most part contributed by the fact that we want to use DSL lines. Ironically, in high-speed networks (like between universities) a lot of this overhead is dispensable.

5. VOIP APPROACH

In comparison to NMP's EPT (ensemble performance threshold) of 25ms, VoIP deals with delay restrictions at ~ 150 ms, which makes it much less time critical and thus allows larger packet buffering in terms of providing QoS in the application level.

In order to handle unpredictable delays caused by network jitter, the typical approach in VoIP is using audio block sizes larger than 1024 samples with their appropriate compression codecs in combination with the RTP protocol. By data retransmission possible packet loss is reduced in order to provide a reliable voice stream.

Since the EPT of 25ms is significantly smaller than the VoIP equivalent, most of VoIP principles based on large packet buffering cannot be adopted.

6. NMP APPROACH

Using no compression avoids the delay associated with it, but leads to a reduced audio quality, because

the bit-rate limitation of the DSL-Uplink does not allow high-quality PCM transmission.

It leads to decreasing the audio quality to 8kHz/8Bit and results in an upstream payload bandwidth of 64kBit/s. But due to the fact that lower sampling rates cause larger delays when the block size is constant, this in turn increases the blocking delay to useless values beyond 20ms. On the other hand, decreasing the block size to lower the delay is not possible, as standard soundcards have problems handling frame sizes below 128 samples.

$$2 * (256 \text{ samples} / 8000 \text{ samples/s}) = 64 \text{ ms} \quad (4)$$

A somehow simple way to avoid the larger blocking delay of lower sample rates is to sample with 48kHz, but decimate the sampled buffer by a factor of 2, 4 or 8, which is upsampled on the receiving end. Hereby, additional latency introduced by the required low pass filtering has to be taken into account. Depending on the decimation factor this leads to a payload for one outgoing stream of

$$\begin{aligned} 768 \text{ kBit/s} / 2 &= 384 \text{ kBit/s} \\ 768 \text{ kBit/s} / 4 &= 192 \text{ kBit/s} \\ 768 \text{ kBit/s} / 8 &= 96 \text{ kBit/s}. \end{aligned} \quad (5)$$

In terms of overhead the IP and UDP header generate another 28 bytes and PPPoE stack another 96 bytes / packet = 124 bytes / packet. Assuming an audio frame size of 256 samples, 188 packets / s are sent (at 48kHz) which results in 187 kBit/s overhead / stream.

With a decimation factor of 8 the payload of 96kBit/s plus the previously calculated overhead of 187kBit/s results in 283kBit/s which only supports 384kBit/s-upstream lines and moreover generates a poor audio quality of 6kHz.

Due to the very low delay restrictions, NMP in DSL networks is feasible only with consequent optimization in audio capture / network transmission and the assumption of stable network conditions.

Since the audio blocking delay appears twice, it has to be kept as low as possible meaning a sampling rate

Frame size	Packets/s @48kHz	overhead	total
512	94	93kBit/s	190kBit/s
256	188	187kBit/s	283kBit/s
128	375	374kBit/s	470kBit/s

Table 4: Decimated payload of 96 kBit/s + overhead

of 48kHz and a frame size of 128, which causes a total audio delay of 5.3ms. In this context the network delay must not exceed a value of 20ms, which in general cannot be provided without Fastpath.

7. NMP/ULD APPROACH

Our new approach is to use high quality audio compression with a very low delay in this scenario. We use the prediction-based Ultra-Low Delay (ULD) audio coder, which we will describe briefly here. It has an encoding/decoding delay of only 5.3ms at 48kHz sampling rate (blocking/soundcard delay included).

The ULD Audio Codec enables compression of audio streams with a block size of 128 samples/frame with a constant bit rate of down to 64 kBit/s with high audio quality. This coder is used in combination with our so called "Soundjack" system to stream the audio packets from soundcard to soundcard.

Frame size	Packets/s @48kHz	overhead	total
512	94	93kBit/s	157kBit/s
256	188	187kBit/s	251kBit/s
128	375	374kBit/s	483kBit/s

Table 5: ULD 64kBit/s payload + overhead

Depending on the current network situation, Soundjack can accommodate various soundcard preferences which are as well supported by the ULD Codec. Besides this, in terms of CPU performance, a typical PC is still able to process the encoding and decoding easily.

The ULD Coder achieves a total encoding / decoding delay of 5.33 to 8 milliseconds with sampling frequencies from 32kHz to 48kHz by separating irrelevance and redundancy reduction and assigning

them to different functional units [10][11]. Fig. 5 shows the main functional blocks of the ULD Coder.

A psycho-acoustically controlled adaptive linear filter is used on the input audio signal $s(n)$ for the irrelevance reduction. In our current implementation, the perceptual model incorporates a DFT based on 256 samples with 50% overlap causing a delay of 128 samples. To synchronize the pre-filtering step with the perceptual model, a corresponding delay is introduced in the signal path. The output of the perceptual model is an estimation of the masking threshold.

This estimation is then transformed into filter coefficients and a gain factor via the Levinson-Durbin algorithm [13]. Together, the filter operation and the following filter gain (see Fig. 5) can be interpreted as a normalization of the input signal $s(n)$ to the masking threshold. The resulting prefiltered signal $p(n)$, which is much smaller than $s(n)$, is then uniformly quantized into $\tilde{p}(n)$ via a simple rounding operation.

The decoder (see Fig. 6) contains a post-filter, which is the inverse of the pre-filter, and hence has a frequency response like the masking threshold. Thus, the added quantization noise remains at or below the masking threshold, as desired.

The redundancy reduction employs a predictive lossless coding scheme using backward adaptive prediction, resulting in the prediction error signal $e(n)$, and entropy coding (with the resulting entropy code words $c(n)$). Both processing steps provide minimal coding delay and are lossless. However, both steps inherently produce a variable bit rate, because $\tilde{p}(n)$ contains a variable amount of information. To facilitate a constant bit rate (CBR), a technique has been developed that allows the user to limit the necessary data rate. This is described in detail in [2].

In a final stage, side information and entropy coded audio data are packed into a bit stream; in Fig. 5 this step is visualized as a multiplexer [2].

The total delay of NMP/ULD at 48kHz is made up of the soundcard delay (capture delay + playout delay), the additional algorithmic ULD delay (128 samples), the processing delay and the network delay.

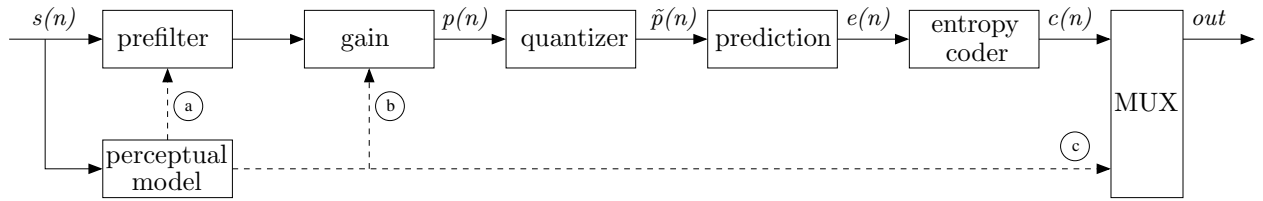


Fig. 5: Basic structure of the ULD Encoder: a) filter coefficients, b) gain factor, c) side info including a and b and additional signaling and initialization information

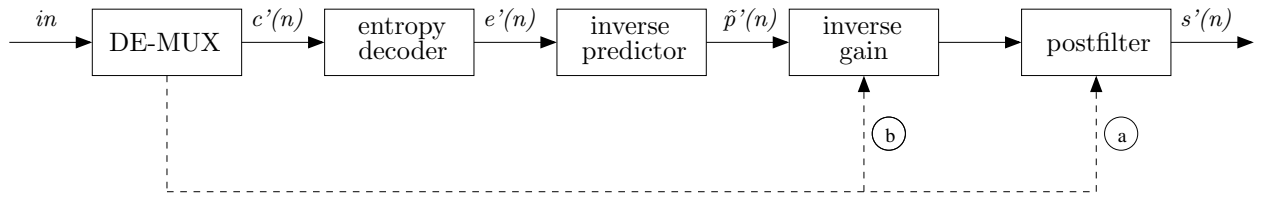


Fig. 6: Basic structure of the ULD Decoder: a) filter coefficients, b) gain factor

total delay =

$$\begin{aligned} & \text{capture delay}(2.3\text{ms}) + \text{ULD algorithmic}(2.3\text{ms}) + \\ & \text{processing delay}(< 1\text{ms}) + \text{network delay} + \\ & \text{processing delay}(< 1\text{ms}) + \text{playout delay}(2.3\text{ms}) \approx \end{aligned}$$

$$7.9\text{ms} + \text{network delay}$$

(6)

8. QUALITY OF SERVICE FOR NMP

Though the ULD Codec provides error concealment in case of packet loss or jitter, ways for avoiding unreliable transmission conditions are to be found which in turn lead to a suitable quality of service (QoS) architecture for NMP. The main criteria in this context are delay, jitter, bandwidth and reliability.

Today's internet works in best effort manner. This implies that packages are forwarded on the fastest route if the network resources are available. If this route lacks bandwidth capacity, network congestion is the result. This causes network jitter, which delays certain packages more than others and thus can mix up the packet order. Besides this, larger delays are possible in case of a route change. With best

effort no guarantee exists that the route is provided with the sufficient bandwidth capacity.

Through over-providing of the network resources, the chance for fast and error free packet transmission is higher, which is already a simple QoS approach.

Hence with best effort internet, the most reliable NMP usage is achieved in broadband LANs directly connected to an internet backbone.

On the other hand, in narrow-bandwidth networks like DSL over-providing of resources is not that simple. Higher bandwidth can only be obtained by upgrading the DSL line. So from QoS point of view, audio compression with the ULD also contributes to increase QoS, because it decreases the payload that has to be transmitted.

9. EXPERIMENTAL RESULTS

9.1. Broadband tests

In order to test the software prototype, two NMPs have been established. Connections were between Lübeck (ISNM, Germany) and Paris (IRCAM, France), and between Lübeck and Ilmenau (IDMT, Germany).

The average UDP-latency (one-way) between Lübeck and Paris was 20ms. The audio settings

were 48kHz, 256 samples/frame, which is 5.33ms blocking delay. As the blocking delay has to be added twice (capture/playout), the total latency was 30.7ms. The ULD increased the latency to 33.3ms. In terms of latency we could still play jazz tunes, but funk tunes or songs supposed to be 100% on the beat were not playable anymore, since the delay was above the EPT.

Between Lübeck and Ilmenau the average UDP-latency was 6ms. With the blocking delay, this added to 16.6ms. The total latency with ULD of approximately 19.2ms was still 5.8ms below the EPT. Hence we could also play tunes which needed more synchronicity. The only problem we encountered was occasional network jitter which was caused by an overloaded router in the local ISNM network.

9.2. DSL Tests

After the successful ULD implementation into Soundjack, a DSL test connection was established. A DSL line with a 384kBit/s-Upstream and a DSL line with an upload of 512kBit/s were interconnected in mirror mode. That means that only on one end an active user controlled the application; on the other end the decoded audio data was simply used as input for the encoder. This allowed an easy delay measurement on the user side of the connection. For completeness, the download capacity of both lines was larger than 2 MBit/s.

The one-way UDP-ping indicated a value of 13ms, but due to the mirror connection we measured a roundtrip delay of 26ms.

With the frame size of 256 samples this resulted in a blocking delay of 10.6ms. This was increased to 13.2ms due to the additional ULD Delay. Hence the total round-trip latency was 39.2ms, and a one-way delay of 19.6ms.

With a CBR of 64kBit/s, each stream caused a traffic bandwidth of 251kBit/s. Due to a very stable connection, the reflected stream had a brilliant audio quality without audible dropouts or distortions. In terms of latency the one-way UDP ping increased by 1ms when transmitting audio data, but both lines appeared to handle the two streams easily and without any increase of jitter.

10. CONCLUSIONS

The approach of capturing audio with very small

frame sizes and sending them across the network with a minimal buffer size is the key to achieve one-way latencies below the EPT of 25ms in order to play NMP sessions. Through the integration of the ULD codec, the upload bandwidth restrictions for narrow band networks (upload bandwidth ≥ 256 kBit/s) are fulfilled without a significant increase of latency and without audible loss of audio quality. Assuming DSL connections with the service quality required for NMPs, we found live play is possible, the same way as in broad band networks. Due to extremely small network buffers and low packet sizes, the network jitter is the most relevant problem with NMP which can either be solved by the best effort approach of simply using larger bandwidth endpoints and/or ideally by applying Guaranteed QoS on Demand.

11. REFERENCES

- [1] John Lazzaro, John Wawrzynek, "A Case for Network Musical Performance," presented at the 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2001) June 25-26, 2001, Port Jefferson, New York
- [2] Ulrich Krämer, Gerald Schuller, Stefan Wabnik, Juliane Klier, Jens Hirschfeld, "Ultra Low Delay audio coding with constant bit rate," presented at the 117th AES convention, San Francisco, CA, USA, 2004, October, 28th-31th
- [3] Alexander Carôt, "Diplomathesis: livemusic on the internet," FH Lübeck, May 2004
- [4] Dirk Van Aken, Sascha Peckelbeen, "Encapsulation Overhead(s) in ADSL Access Networks," June 2003
- [5] Nathan Schuett, "The Effect of Latency on Ensemble Performance," May 2002
- [6] Manfred Lutzky, Gerald Schuller, Marc Gayer, Ulrich Krämer, Stefan Wabnik, "A guideline to audio codec delay," presented at the 116th AES convention 2004 May 8-11 Berlin, Germany
- [7] GANT project managed by DANTE, <http://www.geant.net/server/show/nav.159>

- [8] Xiaoyuan Gu, Matthias Dick, Ulf Noyer, Lars Wolf, “NMP- A new networked music performance system,” NIME04, June, 2004
- [9] Mark Elias and Sven Ooghe, “Multi-Service Architecture & Framework Requirements,” Technical Report DSL Forum TR-058, September 2003
- [10] Bernd Edler, Christof Faller and Gerald Schuller, “Perceptual Audio Coding Using a Time-Varying Linear Pre- and Post-Filter,” presented at the AES 109th convention, Los Angeles, CA, USA, 2000 September 22-25
- [11] Gerald Schuller, Bin Yu, Dawei Huang and Bernd Edler, “Perceptual Audio Coding using Adaptive Pre- and Post-Filter and Lossless Compression,” IEEE Transactions on Speech and Audio Processing, September 2002, pp. 379-390
- [12] Leif Arne Rønningen, “Adaptive Video Resolution and Traffic Control in Packet Networks,” Guir, UC Berkeley, March 2004
- [13] Wolfgang Hess, “Digitale Filter,” 2nd Edition, 1993, ISBN 3-519-16121-4, B.G. Teubner Stuttgart