US 20050180405A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2005/0180405 A1

Bastin (43) Pub. Date: Aug. 18, 2005

(54) SUB-PACKET INSERTION FOR PACKET LOSS COMPENSATION IN VOICE OVER IP NETWORKS

(75) Inventor: **Roger Bastin**, Carp (CA)

Correspondence Address:
F. CHAU & ASSOCIATES, LLC
130 WOODBURY ROAD
WOODBURY, NY 11797 (US)

(73) Assignee: MITEL NETWORKS CORPORATION

(21) Appl. No.: 11/110,245

(22) Filed: Apr. 20, 2005

Related U.S. Application Data

(63) Continuation of application No. 09/800,194, filed on Mar. 6, 2001, now Pat. No. 6,901,069.

(30) **Foreign Application Priority Data**

Mar. 6, 2000 (GB) 0005376.9

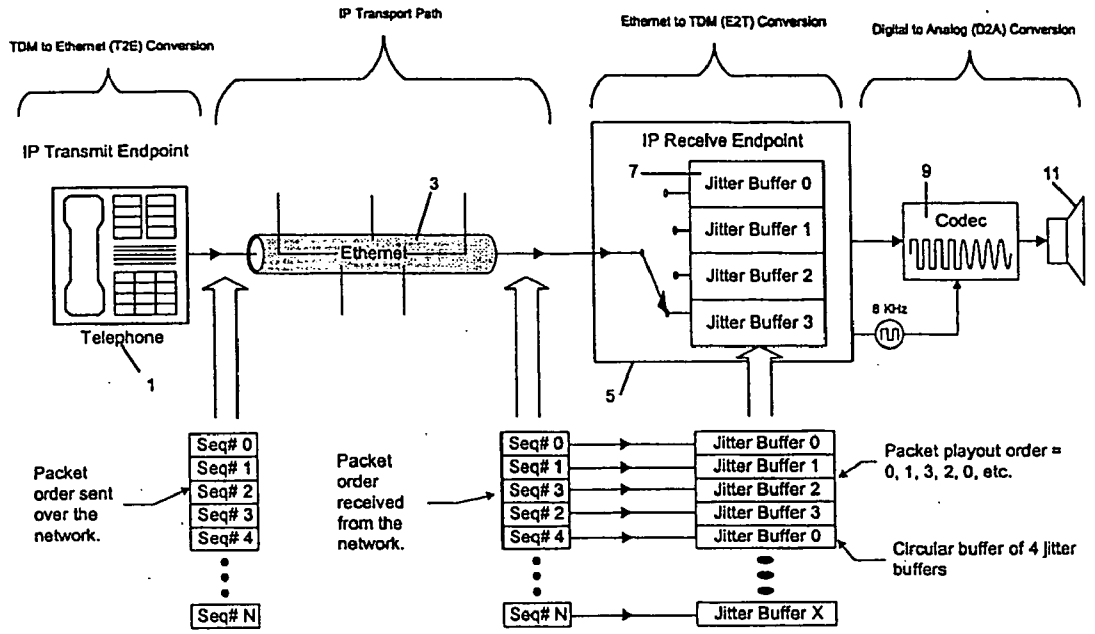
Publication Classification

(51) Int. Cl.[7] H04L 12/66

(52) U.S. Cl. 370/352

(57) **ABSTRACT**

A method of compensating within a receiving endpoint for lost audio packets transmitted across an IP network, comprising the steps of storing a packet buffer of samples as a plurality of sub packets within a jitter buffer, inserting at least one interpolated sub packet between successive sub packets in the buffer, and playing out the sub packets from the buffer, such that only small portions of the jitter buffer are replayed at specific times to minimize the negative effects on voice quality. The inventive method inserts the replayed portions to compensate for packet loss in a way that results in only a relatively low processing burden.



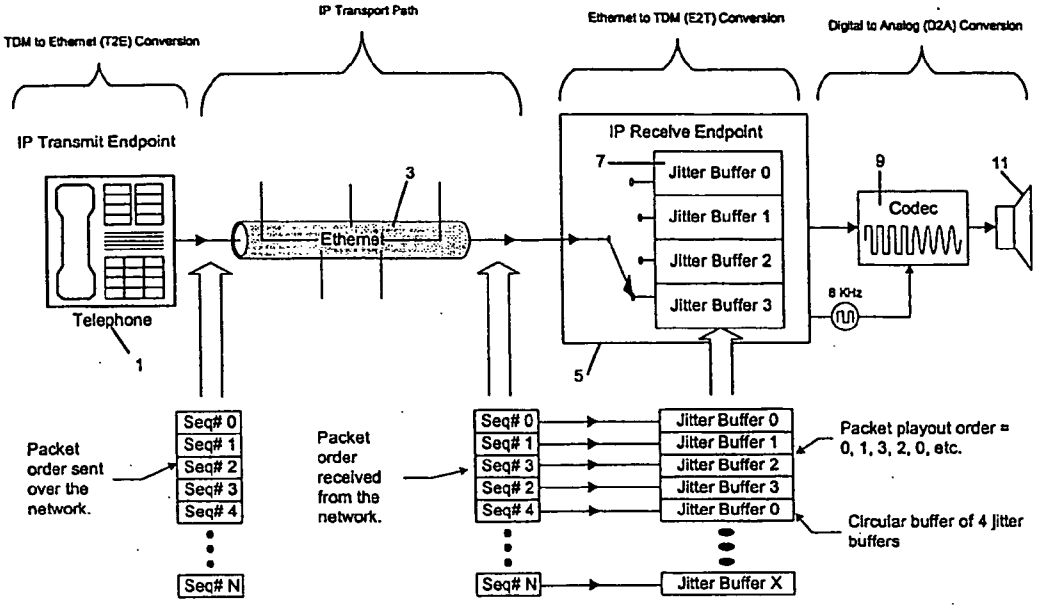


Figure 1

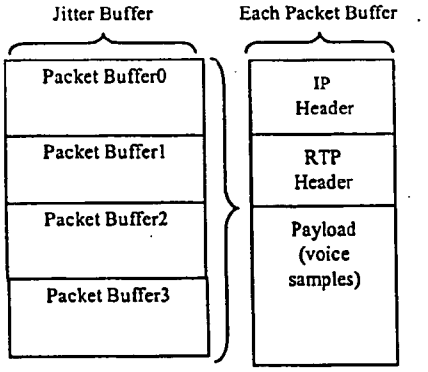


Figure 2

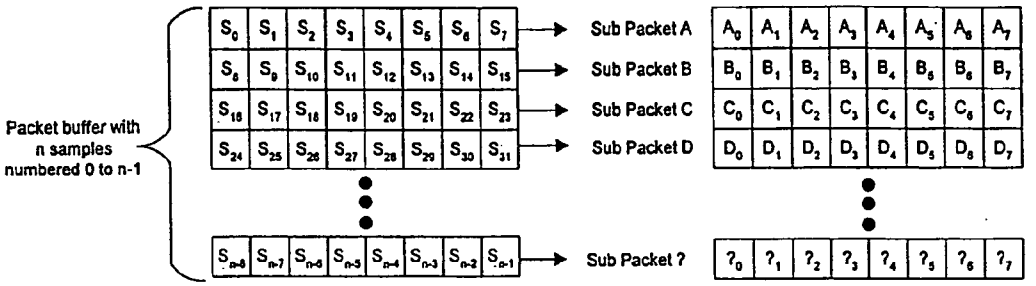


Figure 3

Portion of jitter buffer prior to compensation

Sub Packet A	A_0	A_1	A_2	A_3	A_4	A_5	A_6	A_7
Sub Packet B	B_0	B_1	B_2	B_3	B_4	B_5	B_6	B_7
Sub Packet C	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
Sub Packet D	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
Sub Packet E	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7
Sub Packet F	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7

Uncompensated playout order would be:
{A_0, A_1,... A_7, B_0, B_1, ... B_7, C_0, C_1, ... C_7, D_0, D_1, ... etc.}

Playout after compensation

	A_0	A_1	A_2	A_3	A_4	A_5	A_6	A_7
Inserted Sub Packet	AB_0	AB_1	AB_2	AB_3	AB_4	AB_5	AB_6	AB_7
	B_0	B_1	B_2	B_3	B_4	B_5	B_6	B_7
	C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7
Inserted Sub Packet	CD_0	CD_1	CD_2	CD_3	CD_4	CD_5	CD_6	CD_7
	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7
Inserted Sub Packet	EF_0	EF_1	EF_2	EF_3	EF_4	EF_5	EF_6	EF_7
	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7

Compensated playout order would be:
{A_0, A_1,... A_7, AB_0, AB_1, ... AB_7, B_0, B_1, ... B_7, C_0, C_1, ... C_7, CD_0, CD_1, ... CD_7, D_0, D_1, ... etc.}

Where, $AB_0 = (A_0 + 3B_0) / 4$,
$AB_1 = (2A_1 + 2B_1) / 4$,
$AB_2 = (3A_2 + B_2) / 4$,
$AB_3 ... AB_7 = A_3 ... A_7$

Or the general case,
$PN_0 = (P_0 + 3N_0) / 4$,
$PN_1 = (2P_1 + 2N_1) / 4$,
$PN_2 = (3P_2 + N_2) / 4$,
$PN_3 ... PN_7 = P_3 ... P_7$,
P = Present sub packet sample
N = Next sub packet sample

Figure 4

SUB-PACKET INSERTION FOR PACKET LOSS COMPENSATION IN VOICE OVER IP NETWORKS

FIELD OF THE INVENTION

[0001] This invention relates in general to digital signal transmission, and more particularly to a sub-packet insertion method for packet loss compensation method in voice over IP (VoIP) networks.

BACKGROUND OF THE INVENTION

[0002] The transmission of voice and audio data over IP networks presents some inherent challenges regarding end to end quality of service. Specifically, packet loss, packet delays and packet jitter are characteristics that can significantly impact voice quality.

[0003] From an endpoint's (e.g. phone) perspective on an IP network, packet loss occurs in an arbitrary, unpredictable fashion. Packet loss is out of the endpoint's control and typically occurs due to a collision or some network overload (e.g. in a router or gateway). Since the packet loss can occur in the physical implementation of the network (e.g. collisions in cables) there is no guaranteed mechanism to inform the receiver when a packet is missing. Therefore, sequence numbers are used to allow the receiver to detect packet loss. Also, once lost, the packet is not re-transmitted since the associated delay in retransmission is prohibitive in real time telephony applications. Thus, the onus is on the receiving endpoint to implement some form of detection and compensation for packets lost in the network. The challenge in this respect is to adequately reconstruct the original signal and maintain a sufficient level of voice quality.

[0004] Packet delay and packet jitter are additional network phenomena that require measures of compensation to maintain voice quality. As packets travel from a source endpoint to a destination endpoint they are typically relayed through various routers or hubs along the way. As a result of variable queuing delays and variable routing paths, sequential periodic packets sent from a source can arrive out of order and with substantial delay and jitter at the destination endpoint. Typically a receiver manages these issues by implementing a buffer of packets to smooth the variable jitter and to allow the receiver to re-arrange packets into their proper order. Unfortunately such a buffer increases the nominal delay of the audio stream depending on its size, and as such must be minimized since audio delay has its own negative effect on voice quality. This minimization prevents 100% compensation for delay and jitter in the receiver and effectively increases the rate of packet loss in the system since a late packet cannot be inserted into an ongoing audio stream.

[0005] Most applications use the aforementioned buffer of packets to handle jitter and packet delay. Routines that manage this buffer monitor incoming sequence numbers and detect both lost and late packets. In telephony applications packets are usually delay constrained to 10, 20 or 30 ms in size. To compensate for a loss of this duration the receiving endpoint can replay a previous packet, decrease the playout rate (assuming the jitter buffer is of sufficient size), interpolate samples or implement a silence detection and insertion scheme.

[0006] Simple replaying of a previous packet is computationally trivial yet often yields unsatisfactory results since voice quality dramatically suffers as packet loss increases. A variation of this scheme is to replace the lost packet with an idle or zeros packet but this too is quite noticeable under even marginal packet loss.

[0007] Decreasing the playout rate and interpolation between samples are effectively the same thing; both alter the receive sample rate to reduce the consumption rate of samples. Playout adjustment is implemented in the prior art via hardware for adjusting the sample clock or the sample frame length, whereas interpolation is implemented in the prior art as a software method of inserting additional samples by means of averaging. Both methods have an undesirable side effect of causing a frequency shift of the signal due to the change in sample rate. To minimize the frequency shift only small adjustments to the sample rate can be made. However, under conditions of packet loss, small adjustments do not provide an adequate rate of compensation.

[0008] Prior art silence detection algorithms monitor the signal stream to determine the intervals between voice where the signal consists of merely background noise. Silence insertion is the process of using the silence detection information to insert additional silence periods to compensate for lost packets. This method can be effective if there are many silence intervals or if the jitter buffer is large enough to guarantee some silence intervals most of the time. Unfortunately in voice conversations silence periods are often very small (between words) and they cannot be guaranteed during the time frame of a typical jitter buffer. Furthermore, silence detection imposes an additional processing burden when compared to the other prior art methods of compensation.

SUMMARY OF THE INVENTION

[0009] According to the present invention, a method is provided for packet loss compensation in real time voice over IP applications. The method of the invention allows a receiving endpoint to dynamically detect and recover from packet loss with minimal processing overhead. Specifically, a hybrid method of packet loss compensation is provided in accordance with which only small portions of the jitter buffer (referred to herein as sub packets) are replayed at specific times to minimize the negative effects on voice quality. The inventive method inserts the replayed portions to compensate for packet loss in a way that results in only a relatively low processing burden.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] A detailed description of the preferred embodiment is set forth herein below with reference to the following drawings, in which:

[0011] **FIG. 1** is a block diagram of a voice over IP (VoIP) network forming the environment in which the invention is implemented;

[0012] **FIG. 2** is a diagrammatic representation of a typical jitter buffer;

[0013] **FIG. 3** shows an arrangement of sub packets in a jitter buffer in accordance with the invention; and

[0014] **FIG. 4** shows the insertion of sub packets in accordance with the packet loss compensation method of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[0015] The basic features of any voice over IP implementation are a transmitting and receiving endpoint separated by an IP network. The IP network consists of various interconnected elements such as hubs, routers and gateways. From an endpoint perspective, however, the interface is simply a connecting IP cable which can be viewed as a dedicated connection from transmitter to receiver.

[0016] Thus, as shown in **FIG. 1**, a transmitting endpoint **1** on the IP network simply accumulates samples from its Analog to Digital process (TDM to Ethernet (T2E)) into a packet or payload buffer within the endpoint **1**, according to a sequential order. Once the buffer is full the endpoint transmitter wraps a packet header around the payload and transmits this across the network **3** with appropriate addressing and sequence information in the header, as is well known in the art. The routing information in the header describes the final destination and is attached to each and every packet (e.g. Seq#**0**, Seq#**1**, etc.). Due to multiple network routing paths and/or variable queuing delays at each routing hop across the IP network **3**, transmitted sequential packets can become out of order at the receiving endpoint. This is shown in **FIG. 1** by the fourth packet (Seq#**3**) arriving ahead of the third packet (Seq#**2**). The receiving endpoint **5** corrects these sequencing errors by buffering the packets in the correct order within a jitter buffer **7**, prior to the digital-to-analog conversion (Ethernet to TDM (E2T) process) and playback via codec **9** and speaker **11**.

[0017] The jitter buffer **7** is conventionally implemented as a simple ring buffer of sequentially ordered IP packet buffers. Each packet buffer contains an IP header section, an RTP (Real Time Protocol) header section and the packet payload as shown in **FIG. 2**. The payload comprises a buffer of samples to be played according to the sequence number in the RTP header. The jitter handling capability is determined by the size of the jitter buffer **7**. This size (in number of packets) is an architectural parameter, however, as previously mentioned, the jitter buffer **7** must be minimized to limit the end to end delay. This can lead to effective packet loss when the packet jitter exceeds the buffer's capabilities. Additionally if a packet is lost in the network **3** due to a collision or overload (e.g. if Seq#**3** in **FIG. 1** were lost instead of out of order), the jitter buffer **7** will detect the mismatch in packet order but is unable to compensate.

[0018] In prior art packet loss compensation schemes, replaying any subset of a voice stream is a form of interpolation. As indicated above, it is commonplace in the prior art to replay an entire packet to compensate for the loss of a packet. This method is noticeable for most users as a stuttering effect, since the packet size is usually tens of milliseconds in duration. It is also known in the art to either replay one sample at a time or interpolate to generate an additional sample. Both approaches suffer from the disadvantage of decreasing the frequency of the voice signal. Even interpolating every 5th or 10th sample causes a noticeable frequency shift and is often insufficient in compensating for lost packets in a timely fashion.

[0019] According to the present invention, each packet buffer is divided into smaller sub packets to allow the replaying of sub packets as a compromise between the two prior art approaches discussed above. A sub packet is simply a short sequence of samples contained in the payload of a given packet buffer. The non-obvious benefit of the sub packet approach is that the frequency shift of sample interpolation becomes less noticeable as the sub packet size increases while the stuttering effect of packet replay decreases as the sub packet size decreases. The choice of sub packet size thus becomes critical in the tradeoff between these two competing requirements.

[0020] According to the best mode of this invention, a one-millisecond sub packet is selected based on experimental results.

[0021] **FIG. 3** shows how a packet payload buffer with n samples is divided into sub packets according to the present invention. A typical value for n is 160 samples (i.e. 20 milliseconds of voice). Thus, in accordance with the best mode of the invention, choosing a 1 millisecond sub packet yields 20 sub packets per packet buffer.

[0022] When either the receiver **5** has either detected a packet loss or detects that the sample count of jitter buffer **7** is beginning to underflow, it enables the packet loss compensation algorithm according to the invention. It has been determined by experimentation that one packet remaining in the jitter buffer **7** represents a sufficient threshold for detection. The packet loss compensation method comprises inserting an interpolated sub packet for playout after every other sub packet period (in this case 1 ms), as shown in **FIG. 4**. This replay period is chosen to minimize both the stuttering effect and the frequency shift while quickly reclaiming the lost packet (i.e. the remaining samples in the jitter buffer **7** are "expanded" by 50%). The inserted sub packet is interpolated to minimize the transition effects between sub packets. This is accomplished by a simple weighting scheme to make the first samples of the replayed sub packet resemble the first samples of the next sub packet to be played (which would have been what the first playout would have flowed into without the compensation).

[0023] To further minimize the stuttering effect, the compensation method of the present invention is only invoked when the underflow situation is critical. Thus, if compensation has occurred for several sub packets and a new, subsequent packet arrives, the compensation algorithm is suspended until the sample count again decreases to the critical threshold. This automatically spreads the compensation out at a decreasing rate, which is less noticeable to the human ear.

[0024] The computational burden of the sub packet insertion scheme is relatively lightweight compared to prior art silence detection/insertion schemes. The insertion of a sub packet into the playout stream every other sub packet period is straightforward since the previous sub packet buffer can often be re-used and only three samples require modification to implement the smoothing process. The inserted sub packet results from a simple scaling of two samples followed by averaging (usually implemented as a shift if the scaling ratio is chosen as a power of two).

[0025] The following pseudo code shows a preferred implementation of the sub packet insertion method according to the invention:

```
#define SUB_PACKET_SIZE   8        /* In samples */
int current_sp[ SUB_PACKET_SIZE ];
int nex_sp[ SUB_PACKET_SIZE ];
void smooth_sub_packet( void )
{
   current_sp[0] = (current_sp[0] + 3 * next_sp[0]) >> 2;
   current_sp[1] = (2 * current_sp[1] + 2 * next_sp[1]) >> 2;
   current_sp[2] = (3 * current_sp[2] + next_sp[2]) >> 2;
}
If ( packet compensation mode AND in_odd_sub_packet )
{
   CALL smooth_sub_packet and RESEND current_sp
}
```

[0026] By inserting weighted sub packets after every other sub packet period, the packet loss compensation method of the present invention can compensate for lost packets at a 50% compensation rate. Thus, if 20 ms of data is remaining when compensation begins, the receiver **5** will play out data for 30 ms before suffering data starvation, which is ample time to receive a subsequent packet.

[0027] Alternative embodiments and variations of the invention are possible.

[0028] The preferred embodiment sets forth one example of specific weighting factors and sub packet sizes. Variation of these parameters may yield better results for specific applications as packet size, sample size, sample rate and type of audio vary according to system architecture. Additionally, the method of smoothing can vary according to the rate of packet loss. For example, in some applications packet loss may be quite infrequent so all that may be necessary is to interpolate one sub packet per packet buffer in order to provide adequate packet loss compensation.

[0029] Furthermore, whereas the principle usage of the packet loss compensation scheme of the present invention is in Voice over IP architectures (VoIP)—that is, in traditional telephony applications and services, the principles of the invention may also be applied to applications where other audio sources (such as music) are sent across the IP network **3**. Thus, the general application of the invention is to compensate for packet loss in audio sent over IP networks, where the audio is destined for the human ear to receive and interpret.

[0030] All such alternatives and variations are believed to be within the sphere and scope of the invention as set forth in the claims appended hereto.

1. A method of compensating within a receiving endpoint for lost audio packets transmitted across an IP network, comprising the steps of:

storing a packet buffer of samples as a plurality of sub packets within a buffer;

inserting at least one interpolated sub packet between successive sub packets in said buffer; and

playing out said sub packets from said buffer.

2. (canceled)

3. (canceled)

4. (canceled)

5. The method of claim 1, further comprising the step of inserting interpolated sub packets between every other one of said sub packets in said buffer.

6. (canceled)

7. (canceled)

8. (canceled)

* * * * *