Internet Telephony Gateway Server
- Software Design

*B. H. Lee, S. C. Hui, S. Foo, C. K. Yeo
School of Applied Science, Nanyang Technological University, Singapore

ABSTRACT

An Internet Telephony Gateway Server is a server that integrates the Internet into the Public Service Telephone Network (PSTN). It makes use of the Internet as a new form of media to transmit speech audio between two conventional telephones. This paper examines the software design of the server and identifies the necessary Application Programming Interfaces (APIs) for the server software. Two main groups of APIs are required. The first group of APIs interfaces with the hardware interface to answer calls, decode DTMF tones, check trunk line availability, make and terminate calls. The second group of APIs interfaces with the Internet to provide functionality for establishing connection, packet loss replacement, packet ordering, packet time stamping, compression and decompression.

INTRODUCTION

The Internet is fundamentally changing telephony - just as it is changing virtually every other industry today. The telephone network and data network have coexisted side by side for decades. However, it is only now that they have reached a critical mass and have come together to deliver a whole new range of powerful and economical communication options.

By using the Internet as an alternative backbone for voice traffic, it is possible to integrate voice and data traffic on the same network – which can be much more economical than traditional circuit switched telephony. It is possible to use the Internet to control and monitor telephone costs. The telephone is everywhere: it is affordable and easy to use for basic service. The Internet can add computing power to the telephone to make it even more powerful and useful.

The objective of this research project is to design an Internet Telephony Gateway Server to bridge the Internet and the Public Switched Telephone Network (PSTN). Through these Gateway Servers, a telephone user can call another telephone user via the Internet. The gateway’s main tasks are to answer, make, and terminate telephone calls through the PSTN and transfer audio data through the Internet.
SYSTEM OVERVIEW

Figure 1 shows a system overview of the gateway server. The caller calls its local gateway via a local trunk line and is subsequently requested by the gateway to key in the area code and telephone number of the recipient via a Interactive Voice Response (IVR) system. It will attempt to contact the corresponding gateway via the Internet, failing which, it will inform the caller and await further instructions. If connection to the remote gateway is successful, this remote gateway will acquire a remote trunk line and contact the recipient. At the same time, it will negotiate a set of common communication parameters with the caller’s gateway. Once these parameters are set, audio data from the two trunk lines via the Internet will be sent to each other simultaneously. The audio at one end will first be encoded and compressed before it is sent through the Internet. At the other receiving end, the compressed audio is decompressed, decoded and sent to the trunk line. This process will continue until one party terminates the call and initiates a tear-down procedure to disconnect the telephones from the trunk lines and the gateways from each other. The reader should refer to (Lim, 1997) for a more detailed system overview description and the hardware design of the gateway server.

COMMUNICATION PROCESS

Figure 2 shows a typical communication process of the Internet Telephony Gateway Server. This can be divided into four phases: Answer Call, Call Set-up, Audio Transmission, and Call Termination.

- **Answer Call** A caller first calls the local telephone gateway by using a hunting directory number to access all the trunk lines to the gateway. The PSTN will send a ringing tone to the gateway through one of the free trunk lines. The gateway will detect the ringing tone, and off-hook the trunk line.

- **Call Set-up** With the trunk line off-hook, the gateway will request the caller to enter the area code of the recipient. If the gateway supports global telephone calls, the area code can be the standard IDD (International Direct Dialling) area code. Alternatively, if the server is used for a multinational corporation (MNC) with regionally or globally dispersed offices, the area code can be defined by a convenient user-defined extension number.

The gateway will make use of this area code to retrieve the IP address of the recipient’s gateway. Once the IP address is identified, the local gateway will try to establish a connection with the remote gateway using the TCP protocol. While waiting for gateway connection to be established, the caller is prompted by the local gateway to enter the recipient’s telephone number. This is to speed up the process as the two gateways may require some time for them to be
connected. Data transmitted to the remote gateway include the trunk line ID, audio parameters, recipient’s telephone number and hang up control signal.

With the connection established, the remote gateway will check for any free trunk line available. A process ID will be sent back to the local gateway so that all transactions between caller and recipient will be respect to these two trunk lines and Internet which are uniquely identified by this process ID. The remote gateway will off-hook the free trunk line and call the recipient. If the call cannot get through to the recipient, it will continue to try for a few times before informing the local gateway. All this while, the caller at the local gateway will be kept informed of the call status through the IVR system. Once the recipient is successfully connected to the remote gateway, the Audio Transmission process is activated.

- **Audio Transmission** A set of audio parameters are negotiated between the two gateways before audio communication can take place. These include the sampling rate of audio data, number of channels, number of bits per sample, packet loss replacement technique and compression used. In order to speed up the negotiation, a series of preferred parameters are sent by the local to remote gateway. With the parameters negotiated and agreed, audio can then be transmitted between the two gateways. Audio is first digitised, compressed and time stamped into packets before being sent to the other gateway. At the receiving gateway, the data packets are ordered, decompressed before being playback to the trunk line. Audio data is transmitted simultaneously in both directions between caller and recipient using the UDP protocol.
• **Call Termination** When either party initiates a call termination by hanging up the telephone, the corresponding gateway will on-hook the trunk line and at the same time send a termination signal to the other gateway. The gateway upon detection of this signal will corresponding on-hook its trunk line. The Internet link between the two gateways is subsequently disconnected.

**APPLICATION PROGRAMMING INTERFACES**

In order to support the functionality of the gateway server, two main groups of APIs are used: one to interface with the telephony hardware and the other to interface with the Internet for audio transmission.

**Telephony Hardware Interface.** The APIs for the telephony hardware interface interacts with the trunk line to get audio data, answer, make and terminate a call. These APIs are summarised in Table 1.

<table>
<thead>
<tr>
<th>API</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>answer_call</td>
<td>• send control signal to off-hook trunk line. API is activated by an interrupt from the hardware interface when it detects a ringing tone on the trunk line. An interrupt is preferred over polling of the status of the trunk line for reason of resource optimisation.</td>
</tr>
<tr>
<td>decode_DTMP</td>
<td>• decode DTMF tone from the telephone keypad. The output corresponds to the number keyed. This is used to extract the recipient’s telephone area code and number.</td>
</tr>
<tr>
<td>check_trunk_line_availability</td>
<td>• check which trunk line is available for the in-coming call to be connected. The API will return the identification number of the available trunk line.</td>
</tr>
<tr>
<td>make_call</td>
<td>• off-hook the available trunk line and send the telephone number through the trunk line using the DTMF transceiver on the hardware interface.</td>
</tr>
<tr>
<td>trunk_line_monitoring</td>
<td>• monitor the status of the trunk line that include: line free, ringing, dialling, talking and hang-up.</td>
</tr>
<tr>
<td>record</td>
<td>• read audio data from the hardware interface. The size of the audio data read will depend on the type of compression used and the duration of audio it represented in real-time.</td>
</tr>
<tr>
<td>playback</td>
<td>• reverse of the <code>record</code> API to send audio data received from the remote gateway to the hardware interface.</td>
</tr>
<tr>
<td>terminate_call</td>
<td>• send control signal to on-hook (hang-up) trunk line. This API can also be activated by a remote signal to off-hook the trunk line when the remote trunk line has hung up.</td>
</tr>
<tr>
<td>IVR_playback</td>
<td>• playback pre-recorded audio to the trunk line for issuing instructions to the user.</td>
</tr>
</tbody>
</table>

**Network Interface.** The standard protocol suite used for the Internet is the Transmission Control Protocol/Internet Protocol (TCP/IP)(Stevens, 1994). The two main transport protocols in the TCP/IP
is the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). TCP is a
connection-oriented protocol with guaranteed delivery of data. UDP is a connectionless-oriented
protocol with no guarantee of arrival of data. These two protocols apply well for non-temporal data
transmission but does not have any provisions to support data transfer of real time nature. In order to
simulate a real time application using the TCP/IP protocol suite, extra mechanism must be added in the
application layer. In addition, in order to have a fast data transfer, no re-transmission of data is
possible. Therefore, UDP is used to transfer audio data and the TCP is used to transfer control data
and negotiation of communication parameters. The telephone number is also sent using TCP. This
group of APIs interfaces with the Internet to transmit audio data between the two parties. These APIs
are summarised in Table 2.

Table 2. APIs for Telephony Hardware Interface

<table>
<thead>
<tr>
<th>API</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP_address_conversion</td>
<td>• convert the telephone area code into the corresponding IP address of the remote gateway. The IP address is retrieved from a database that stores all the available IP address of all the gateways.</td>
</tr>
<tr>
<td>establish_connection</td>
<td>• establish connection between two gateways.</td>
</tr>
<tr>
<td>send_available_trunk_line_ID</td>
<td>• send available trunk line identification number to remote gateway.</td>
</tr>
<tr>
<td>Negotiation</td>
<td>• negotiate the communication parameters used for the sampling and transmission of audio data.</td>
</tr>
<tr>
<td>send_telephone_number</td>
<td>• send telephone number to remote gateway.</td>
</tr>
<tr>
<td>receive_telephone_number</td>
<td>• receive telephone number from remote gateway.</td>
</tr>
<tr>
<td>compression</td>
<td>• compress digital audio data to reduce size and speed up transmission. Compression schemes supported include A-law and μ-law (G.711)(ITU,1985), Adaptive Differential Pulse Code Modulation (ADPCM)(ETS,1992), Groupe Speciale Mobile (GSM)(Campbell, 1986), and the Linear Predictive Coding (LPC)(Campbell,1986). Hardware compression can also be optionally used in place of software compression.</td>
</tr>
<tr>
<td>time_stamping</td>
<td>• time stamped each audio datagram before transmission to remote gateway to enable incoming audio datagrams to be ordered in the correct sequence for playback.</td>
</tr>
<tr>
<td>send_voice_datagram</td>
<td>• send compressed digital audio data packets to remote gateway.</td>
</tr>
<tr>
<td>receive_voice_datagram</td>
<td>• receive compressed digital audio data packets from remote gateway.</td>
</tr>
<tr>
<td>decompression</td>
<td>• reverse of compression API to decompress audio data. A common compression scheme must be utilised for both compression and decompression APIs.</td>
</tr>
<tr>
<td>packet_ordering</td>
<td>• re-arrange received audio datagram into the correct order before audio playback.</td>
</tr>
<tr>
<td>packet_loss_replacement</td>
<td>• to replace lost audio datagram through transmission (since UDP protocol with no guaranteed delivery is used for data transmission). Replacement schemes (Hardman, 1995) supported include silence substitution, white noise substitution, waveform substitution, sample interpolation, embedded speech coding and redundancy techniques.</td>
</tr>
<tr>
<td>termination</td>
<td>• send termination signal to remote gateway to instruct gateway to hang up trunk line.</td>
</tr>
<tr>
<td>disconnect_link</td>
<td>• disconnect the network link between two gateways at the end of communication session.</td>
</tr>
</tbody>
</table>

CONCLUSION

The software design of the Internet Telephony Gateway Server is completed and a set of necessary Application Programming Interfaces (APIs) for the server software is defined. These APIs constitute the most basic set of APIs required to support the functionality of the gateway with scope for further improvement. These may involve encryption of audio datagrams, improved user security, connection accounting, voice mail facility, facsimile interface and the use of a multimedia computer to call up a conventional telephone.

REFERENCES

European Telecommunications Standards Institute (ETS) (1992). “European digital cellular telecommunications system (Phase 1); Full-Rate Speech Transcoding (GSM 06.10)”, I-ETS 300 036, February.
APPENDIX

The existing TCP/IP protocol is used in delivering the audio packets over the Internet. However, new mechanisms such as buffer management and packets lost replacement control are incorporated to make it possible to support real-time audio communication. This approach is aimed at delivering real-time communication at an acceptable quality of service. Currently, this approach is also used by all existing Internet telephone systems and until a new protocol emerges and becomes a standard, it appears that developers are contented to make the best use of current resources and limitations.

In using this approach, it is important to realise that proper sending and receiving of audio data though important is in itself insufficient. More importantly is the on-time delivery of data which is generally dependent on network performance. This in turn is largely related to the kind of network technologies implemented which the system has basically no control over. With many different network technologies available on the Internet, the system should not be made to rely on the underlying network infrastructure. Under such varying network load conditions, the audio packets will suffer varying degrees of delay. The variance in delay produces jitters which are undesirable for real-time services. In order to attenuate the jitters, the recipient’s host computer must provide some buffering mechanism. This will prevent the distracting utterance of audio, thereby permitting continuous playback. In addition, the unreliability of the network can give rise to packets lost and duplication which will deteriorate the voice quality. Hence, additional processing is required to make the system more resilient to these ill-effects of unstable network conditions to maintain a satisfactory overall performance.

In using the existing TCP/IP protocol, a choice can be made to use either TCP and UDP entirely or in some form of combination. Using the connection-oriented TCP alone ensures guaranteed delivery of control and audio data but will result in unacceptable levels of delay and jitters. Using the connectionless-oriented UDP with no guaranteed arrival of data is clearly not suitable for transmitting control data which is required for the functionality of the system. As such, the system utilises UDP for audio data transmission and TCP for control and other important data transmission.

In order to overcome the problems of jitters, non delivery and duplication of packets, incoming audio samples are packet time-stamped before they are transmitted to the recipient. In addition, redundant audio data pertaining to previous samples are combined with the current sample to make up an audio packet. This redundant data is used for the replacement of lost packets during the delivery process. At the recipient’s end, incoming audio data is placed in the correct position within the revolving circular buffer according to the time stamp. Lost packets are detected and corresponding made up and inserted in position. Continuous audio playback is achieved by continuously playing back the contents in the buffer.

Finally, the quality of audio communication delivered by Internet telephone systems is dependent upon the compression algorithms for encoding the audio data, processing power of the host computers and network transmission factors. These three factors are inter-related and an optimal is sought to ensure satisfactory audio quality. For instance, using a high compression ratio in order to minimise network traffic will usually require high processing power and result in a potential drop in audio quality due to larger loss of original audio information. On the other hand, satisfying the processing power of the majority of the existing host computers will require the use of a less demanding compression algorithm but result in higher network traffic. This may lead to higher
potential lost or delay of data packets which will have a detrimental effect on the quality of communication. It should be obvious that there is no clear cut solution to the problem. Likewise, there is no ideal compression algorithm which should be employed in such systems. The system must have some defined minimum level of processing power to carry out the various task, a fast enough modem to transmit and receive data to support the synchronous nature of the application and a number of compression algorithms to suit different network conditions. It would be advantageous if the system is self-adapting to changing conditions and optimally adjust itself to provide an optimal level of performance. Our prototype system supports a number of compression algorithms and allows audio characteristics to be changed to the desired level. The selection of compression algorithms and other parameters are set during a negotiation process prior to connection.

This appendix discuss some of the issue used during the design of the Internet Telephony Gateway Server. The issue include:

- TCP/IP Protocol
- Packet loss Reconstruction Technique
- Reconstruction Delay
- Packets Time-stamping
- Buffer Management
- Packet Ordering
- Compression And Decompression

TCP/IP PROTOCOL

TCP/IP is a protocol suite that allows computers of all sizes, from many different computer vendors, running totally different operating systems, to communicate with each other. TCP/IP is considered to be a 4-layer system, as shown in Figure 1.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Telnet, FTP, e-mail, etc.</td>
</tr>
<tr>
<td>Transport</td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>Network</td>
<td>IP, ICMP, IGMP</td>
</tr>
<tr>
<td>Link</td>
<td>device driver and interface card</td>
</tr>
</tbody>
</table>

Figure 1 - The four layers of the TCP/IP protocol

Each layer has a different responsibility:

1. The *link* layer, sometimes called the *data-link* layer or *network interface* layer, normally includes the device driver in the operating system and the corresponding network interface card in the computer. Together they handle all the hardware details of physically interfacing with the cable (or whatever type of media is being used).
2. The *network* layer (sometimes called the *internet* layer) handles the movement of packets around the network. Routing of packets, for example, take place here. IP (Internet Protocol), ICMP (Internet Control Message Protocol), and IGMP (Internet Group Management Protocol) provide the network layer in the TCP/IP protocol suite.

3. The *transport* layer provides a flow of data between two hosts, for the application above. In the TCP/IP protocol suite, there are two vastly different transport protocol: TCP (Transmission Control Protocol), and UDP (User Datagram Protocol).

TCP provides a reliable flow of data between two hosts. It is concerned with things such as dividing the data passed to it from the application into appropriately sized chunks for the network layer below, acknowledging received packets, setting time-outs to make certain the other end acknowledges packets that are sent, and so on. Because this reliable flow of data is provided by the transport layer, the application layer ignore all these details.

UDP, on the other hand, provides a much simpler service to the application layer. It just sends packets of data called *datagrams* from one host to the other, but there is no guarantee that the datagrams reach the other end. Any desired reliability must be added by the application layer.

Both TCP and UDP use IP as the protocol at network layer. Every piece of TCP and UDP data that gets transferred around an Internet goes through the IP layer at both end systems and at every intermediate router.

4. The *application* layer handles the details of the particular application.

**PACKET LOSS RECONSTRUCTION TECHNIQUE**[^2]

The speech audio is sent through the Internet using the UDP protocol. UDP does not guarantee the arrival of data. Thus there will have packet loss during the data transferring process. Packet loss also can occur for a number of reasons:

- congestion of routers and gateways, which lead to packet being discarded;
- delays in packet transmission, with packet arriving too late at the receiver to be played back;
- heavy loading of the work station, leading to scheduling difficulties in multi-tasking operating system.

Due to the packet loss through the transmission, reconstruction of lost audio packet is necessary. The aim of reconstruction of lost audio packet is to construct a suitable dummy packet at the receiver side, so that the loss is as imperceptible as possible. The repair methods for packet loss are also known as voice reconstruction techniques. With compressed speech, voice reconstruction mechanisms not only have to produce a suitable fill-in packet, but also have to maintain the decoder tracking, since the algorithms transmit difference information. Voice reconstruction techniques can be split into two categories: receiver-only, and combined source and channel techniques.
### Table 1 Voice Reconstruction Techniques

<table>
<thead>
<tr>
<th>Receiver-Only</th>
<th>Combined Source And Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Silence</td>
<td>Embedded Coding</td>
</tr>
<tr>
<td>White Noise</td>
<td>Redundancy</td>
</tr>
<tr>
<td>Waveform Substitution</td>
<td></td>
</tr>
<tr>
<td>Sample Interpolation</td>
<td></td>
</tr>
</tbody>
</table>

Receiver-only techniques are those that try to reconstruct the missing segment of speech solely at the receiver, possibly from the correctly received packets preceding that which was lost. Combined source and channel techniques are those that try to make the system robust to loss by either arranging for the transmitter to code the speech in such a way as to be robust to packet loss, or by transmitting extra information to help with reconstruction.

#### Receiver-Only Techniques

The simplest voice reconstruction techniques were receiver-only, and used either silence, white noise, repetition of part of the last correctly received speech waveform, or sample interpolation as the substitute.

**Silence Substitution**  This is the most favoured reconstruction technique as it is simple to implement, and it gives adequate performance for small packet size. However, it is not able to maintain an acceptable quality of playback audio in the event of high loss rate and large packet size. In this technique, lost packets are simply replaced with silence. Hence, applications using this technique do not incur much additional processing power and is therefore suitable for an environment where probability of losing packets is low and the computers do not have much processing power.

**White Noise Substitution**  Instead of just replacing the lost packets with silence, this technique replace the lost packet with white noise. Human is more comfortable to noise compared to total silence and the phonemic restoration (the ability of the human brain to subconsciously repair the missing segment of speech with the correct sound) occurs for the noise situation, and does not occur for silence period. Therefore, white noise substitution is a better means of voice reconstruction, since it is as easy to generate as silence.

**Waveform Substitution**  In this method, the last correctly received packet is repeated until the period of loss ends. The assumption of the technique is that the speech characteristic have not changed from a preceding segment of speech and use this preceding segment information to reconstruct the missing part. This mechanism has an inherent draw-back when the lost packet is the last in a talk-spurt; the last packet will be repeated until a new talk-spurt starts. The solution to this problem lies in realising that there should be a limit on the length of ‘waveform substituted’ speech, before the underlying assumption of no change in the speed characteristics breaks down; a suitable figure for this is 80ms.

**Sample Interpolation**  This technique is similar to Waveform Substitution. However, it does not outrightly replace all missing audio segments with the previously received segments. It modifies the previous audio packets before substituting the missing audio segments with it. The assumption taken in using this method is that the audio characteristics change slightly over a short period of time.
Combined Source And Channel Techniques

Combined source and channel techniques generally show significant improvement over the receiver-only techniques. The techniques either transmit extra information within the speech packets (to help with reconstruction at the receiver side), or alter the speech coding algorithm and network operation (to make system as a whole more robust to packet loss).

Embedded Speech Coding  This technique used with Adaptive Differential Pulse Code Modulation (ADPCM), and Cod Exited Linear Prediction (CELP) have shown significant performance improvement during packet loss. Embedded speech coding technique allow the bit rate to be adjusted from 40 to 32 or 23 kbps, without the introduction of large amount of noise; essentially the feed-back loops in the encoder and decoder operate at a lower resolution than usual. The standard was designed to ease the problem of packet loss in packet network; the code words are segmented into high and low priority bits, and then placed in different packets. The mechanism relies on arranging for the network to drop packets containing LSBs only, which means that the mechanism is not applicable to Internet as it does not provide this support.

Redundancy  This technique make use of redundancy to improve voice reconstruction at the receiver. The redundant information is the output of a synthetic quality speech coding algorithm (LPC), which is very low bit-rate (4.8kbps).

LPC as the redundant information adds only a small amount of overhead to a audio packet (12 bytes per 160 bytes of PCM or 80 bytes of ADPCM). The information is piggy-backed to the packet following (or a few packets later for heavy load condition) that containing the primary speech code words; that the loss of an individual packet can be repaired using the redundant information in the following packet. This mechanism is unique to packet networks, and is only feasible when there is a reconstruction delay introduced at the receiver. The use of this redundancy technique means an increase in the reconstruction delay by the time equivalent of the distance of the redundancy component after the primary component; this implies an extra delay of one packet for light and medium loading conditions.

The provision of LPC redundant information for use in voice reconstruction is intended to be used with per-packet state information; this prevents decoder mistracking in the case of loss. When a packet has been lost, the receiver decodes the redundant information, and feed the samples to the audio hardware. Consequently, the output speech waveform consists of periods of toll quality speech, interspersed with periods of synthetic quality speech.

RECONSTRUCTION DELAY

Packet speech systems usually employ the standard speech coding algorithms, and group the emerging stream of code-words into packets for transmission over the network. At the receiver, the packets may be received: out of order, not at all, at non-uniform intervals. Consequently, a reconstruction delay must be used at the receiver to repair the network effects; this enables sample play-out to be smoothed.

In a packet speech system, the end-to-end delay is always a critical factor in the usability of a real-time voice system, and should be kept below 600ms[3] in the absence of echoes (the figure may be in fact be less than 400ms), if conversation patterns are not to break down. The size of the packets (in ms) chosen for a packet speech system directly impacts the end-to-end delay. A delay equal to the size...
of one packet is incurred at the transmitter, since the samples in the packet have to be collected before a packet can be sent. At the receiver, a rough estimate of two packets worth in ms, although the true value may be substantially in excess of this rule of thumb. Consequently, a minimum of three packets worth of delay is incurred on an end-to-end basis, before the network propagation delay has been taken into account.

Figure 2 - Reconstruction Delay

Figure above show the timing diagram of the audio packet transfer. At time T1, the transmitter start to collect the digitised audio into a packet. At time T2, the packet is compressed. At time T3, the compressed packet is send through the Internet to the receiver side. At the receiver side, the system will wait until it receive two packets of data before it start to decompressed (T5) and playback (T6). This delay introduced is to cater for the delay jitter when transmitting through the Internet and allow the system to manage the reconstruction of lost packets. If the reconstruction technique used is redundancy technique, extra reconstruction delay, of time equivalent to the distance of the redundant component after the primary component, must be added.

PACKTETS TIME-STAMPING

Each packet of audio sample obtained from the external device is linearly time-stamped to indicate the instant of sampling. This allows the packets arriving at the recipient’s host computer to be ordered in the correct time sequence before being played back. As each audio packet contains 20ms of audio samples, the time-stamp mechanism uses 20 as the base value with increments of 20 for each new audio packet generated. Although, the transmission is carried out in sequential order, the packets may arrive out of order due to the different network paths traversed and varying network delays. The time-stamp is stored in the field preceding the audio data within each audio packet. It is recorded using a 32-bits field in each packet. With a value ranging from 0 to 4294967291, it will exhaust itself after 49 days before it is reset and restarted from 0 again. Both audio samples and the associated time-stamp are forwarded to the next stage to compress the packet size before transmission.
BUFFER MANAGEMENT

The buffer size to be allocated at the receiver side depends on the size of a packet and the reconstruction delay. That is, the buffer must at least able to store two packets of audio data. If the redundancy reconstruction technique is used, the buffer size need to be increased to cater for the long reconstruction delay used. The buffer used is in a ring configuration so that it can be

PACKET ORDERING

As mentioned, the packets may arrived at the receiver out of order since the Internet will introduce a non-uniform delay on each packet. Therefore, the received packets need to be put in order before it is able to playback. The in-coming packets are identified by the packet time-stamp, and placed accordingly into the correct location in the buffer before the packets are decompressed.

COMPRESSION AND DECOMPRESSION

The compression algorithms reduce the audio sample size in raw audio PCM format by encoding it in another format. Compression algorithms supported by the system include A-law, µ-law [4], Adaptive Differential Pulse Code Modulation (ADPCM) [5], Groupe Speciale Mobile (GSM) [6] and Linear Predictive Coding (LPC) [6]. The first four formats are able to maintain almost the same quality as the raw format. However, as LPC produces a lower audio quality at a lower information rate, it is used to compress the previous audio packet into ‘redundant’ audio information and bundled together with the current sample to form an audio data packet. This information is used for lost packet replacement if necessary (see subsequent description). Decompression performed at the recipient’s end reverts the compressed audio samples (both redundant and non-redundant audio) to its uncompressed format that is ready to be played. (Please refer to the hardware paper for detail description of the compression schemes)

REFERENCES


