Abstract
Voice over IP (VoIP) technology has many advantages over the traditional Public Switched Telephone Networks. In our project, we realize a simple VoIP software, which mainly consists of a speech collection and playing back part, a speech CODEC and a networking API. For speech collection/playing part, we use the software (C++ class) and the hardware (sound card) to realize its functions. For the speech CODEC, we choose G.711 PCM $\mu$-Law voice coder. For the transmission of voice conversation, we employ the socket communication API.

1 INTRODUCTION
Voice over Internet Protocol (VoIP) is the technology used to transmit voice conversation over a network using the Internet Protocol (IP) [1]. As one of the hottest telecommunication research topics, it has many advantages over the traditional Public Switched Telephone Networks (PSTNs):

First, IP networks are more cost-efficient. Service providers can effectually reduce their operating cost from the standard equipments and cheap communication carrier. So service subscribers can get a lower price for the better or same services supplied by PSTNs.

Second, IP networks can provide integrative data and voice services. VoIP can handle data packets as well as voice packets. So it can provide new interesting services [2].

Third, IP networks are more bandwidth-efficient. Bandwidth of the communication channel is very limited. In some sense, it is a priceless source. PSTNs mainly use International Telecommunication Union (ITU) Recommendation G.711 coding scheme,
which samples the analog voice signals at a rate of 8,000 Hz, encodes one sample with 8 bits and take up 64kbps bandwidth. IP networks can use more advanced voice coding schemes and take up less bandwidth. Besides G.711, IP networks can also implement G.726, G.728, G.729 and G.723 coding schemes, which only take up 32kbps, 16kbps, 8kbps, and 6.3kbps bandwidth respectively. In our project, first we implement G.711 PCM u-law coding scheme. We will consider more bandwidth efficient coder later.

In our project, we develop an VoIP software. The software will mainly consist of three parts: a speech collection and playing back part, a speech CODEC and a networking API. Speech collection/playing part uses the software (C++ class) and the hardware (sound card) to realize its functions. For the speech CODEC, we choose G.711 PCM μ-Law voice coder. For the transmission of voice conversation, we employ the socket communication API. Our programming environment is visual C++ 6.0 under windows XP operating system.

In the following, first, we will discuss the key components and the implementation of our software. Then, we will make our conclusions and draw the future work. Finally we list all the reference and attach programming appendix.

2 KEY COMPONENTS

In a typical VoIP system (as shown in Figure 1), there are three main parts, speech collection/playing, CODEC (coder/decoder) and socket communications. The details of these three parts will be discussed in the following paragraphs. Generally speaking, at the transmitter side, the speech signal is collected and encoded before transmitted to IP networks. At the receiver side, we do the reverse processes. The received data stream is decoded and recovered into speech signal and played back.
2.1 SPEECH COLLECTION/PLAYING

Speech collection/playing function is performed by the software (C++ class) and the hardware (sound card). We will talk the details in the implementation part.

![VoIP System Structure](image)

Figure 1 VoIP System Structure

2.2 CODEC

To be able to transmit voice signal, we need to convert it from analog format to binary data first. This step is called voice coding. Voice coder is used to encode speech samples into a small number of bits so that the speech is robust in the presence of link errors, jittery networks, and burst transmission. At the receiver, the bits are decoded back to the PCM speech samples and then converted to analog waveform.

Coders are mainly classified into three types, waveform coders, vocoders and hybrid coders.

2.2.1 WAVEFORM CODERS

Waveform coders reproduce the analog waveform as accurately as possible, including background noise. They operate at high bit rate. [3]

G.711 is the waveform coder to represent 8 bit compressed pulse code modulation (PCM) samples with the sampling rate of 8000Hz. This standard has two forms, a-Law and µ-
Law. A-Law G.711 PCM encoder converts 13 bit linear PCM samples into 8 bit compressed PCM samples, and the decoder does the conversion vice versa. µ-Law G.711 PCM encoder converts 16 bit linear PCM samples into 8 bit compressed PCM samples.

G.726, based on Adaptive Differential Pulse Code Modulation (ADPCM) technique, convert the 64 kbit/s A-law or µ-law pulse code modulation (PCM) or 128Kbits linear PCM channel to and from a 40, 32, 24 or 16 kbit/s channel. The ADPCM technique applies for all waveforms, high-quality audio, modem data etc.

2.2.2 VOCODERS

Vocoders do not reproduce the original waveform. The encoder builds a set of parameters, which are sent to the receiver to be used to drive a speech production model. Linear Prediction Coding (LPC), for example, is used to derive parameters of a time-varying digital filter. This filter models the output of the speaker’s vocal tract. The quality of vocoder is not good enough for use in telephony system.

2.2.3 HYBRID CODERS

The prevalent speech coder for VoIP is the hybrid coder, which melds the attractive features of waveform coder and vocoder. It is also attractive because it operates at a low bit rate as low as 4-16 kbps. They use analysis-by-synthesis (AbS) techniques. An excitation signal is derived from the input speech signal in such a manner that the difference between the input and the synthesized speech is quite small. An enhancement to the operation is to use a pre-stored codebook of optimized parameters (a vector of elements) to encode a representative vector of the vector of the input speech signal. This technique is known as vector quantization (VQ).

G.728 is a hybrid between the lower bit rate linear predictive analysis-by-synthesis coder (G.729 and G.723.1) and the backward ADPCM coders. G.728 is a LD-CELP coder and operates on five samples at a time. CELP is a speech-coding technique in which the excitation signal is selected from a set of possible excitation signals through an exhaustive search. While the lower rate speech coders use a forward adaptation scheme
for the sample value prediction filter, LD-CELP uses a backward adaptive filter that is updated every 2.5 ms. There are 1024 possible excitation vectors. These vectors are further decompressed into four possible gains, two possible signs, and 128 possible shape vectors. Therefore G.728 is a suggested speech coder for low-bit-rate ISDN video telephony. Because of its backward adaptive nature, it is a low-delay coder, but it is more complex than the other coders because the fiftieth-order LPC analysis must be repeated at the decoder. It also provides an adaptive postfilter that enhances its performance. [4]

G.729 is designed for low-delay applications, with a frame size of only 10 ms, a processing delay of 10 ms, and a lookahead of 5 ms. This yields a 25 ms contribution to end-to-end delay and a bit rate of 8 kbps. These delay performances are important in an Internet, because we have learned that any factor decreasing delay is important.

In evaluating the performance of coders, several factors come into play. A summary of these factors is shown as follows:

1. Complexity: It may not be the major concern of the commercial VoIP software. But, for us, we only have limited time and the major goal of our project is to get better understanding of real-time speech processing. So, we choose **G.711 PCM μ-Law** voice coder.

2. Algorithm Delay: This represents the delay incurred at the codec to run the voice and coding algorithm on one frame. Comparing to other voice coders, G.711 PCM have smaller algorithmic delay (Table-1 Algorithm Delay of Voice Coders).

3. Bit rate quality: Bit rate quality contributes significantly to bandwidth-efficiency. For our project, first we will implement our software in Ethernet network. With the 10 M/s bandwidth, PCM is not a bad choice. Later, we will consider other bandwidth-efficient voice coders. (Table-2 Bit Rate of Voice Coders)

<table>
<thead>
<tr>
<th>Voice coder</th>
<th>Algorithmic Delay (msec)</th>
</tr>
</thead>
</table>

---

Hong Xiong, Yuan Guo, Fang Zhu
<table>
<thead>
<tr>
<th>Voice coder</th>
<th>Bit Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPEG L3</td>
<td>56-128 kbps</td>
</tr>
<tr>
<td>G.711 PCM A-Law/u-Law</td>
<td>64 kbps (DS0)</td>
</tr>
<tr>
<td>G.726 ADPCM</td>
<td>16, 24, 32, 40 kbps</td>
</tr>
<tr>
<td>G.728 LD-CELP</td>
<td>16 kbps</td>
</tr>
<tr>
<td>G.729 CA-CELP</td>
<td>8 kbps</td>
</tr>
</tbody>
</table>

Table 2: Bit Rate of Voice Coders

2.3 SOCKET COMMUNICATIONS

Socket is a communication endpoint of a communication link managed by the transport services (Figure 2 Socket Communications Structure).

From the connection view, there are 2 types of socket communications, connection-oriented and connectionless socket communications. For the first type, a connection must be set up before transmission. It is more reliable than connectionless socket communication. TCP is the transport protocol for the connection-oriented socket communication. For the second type, there are no connection setup, packet acknowledgement and retransmission services. UDP is the transport protocol for the connection-oriented socket communication. So, we choose TCP as our transport protocol.

From the synchronization view, there are also 2 types of socket communications, asynchronous and synchronous socket communications. For the first type, the execution of the client application is not suspended while waiting for the server to return a response.
For the second type, the execution of the client application is suspended until the server returns a response. For our real time voice communication, we choose asynchronous socket communications.

There are 2 types of socket interfaces, Berkeley Socket Interface for Unix system and Windows Socket. The current version of Windows Socket is Winsock 2. Our software is working under windows XP operating system. So, we choose winsock2 as our socket interfaces [5].

Here we introduce some basics about Winsock 2. Comparing to Winsock 1.1, it is expanded version. Some new functions have been added. It is an interface, not a protocol. So, we can use different socket interface in the 2 ends of the communication links. In our project, we use winsock2 in the both ends of the communication link. As we can see in Figure 3, Winsock defines a standard service provider interface (SPI) between the application programming interface (API), with its functions exported from WS2_32.dll and the protocol stacks.
3 IMPLEMENTATION

Our programming environment is visual C++ 6.0, under windows XP operating system. First, we will show our system flow chart and the PCM coder/decoder performance we simulation in MATLAB. Then, we will discuss the “Real Time” processing issues. At the end, we will show the interface and functions of our program.

3.1 SYSTEM FLOW CHART AND PARAMETERS SELECTIONS

Figure 4 is a simplex real time voice communication flow chart. First, the client makes a Call request, the server responses this request. Then, the client begins to speak. In the client side, the voice gets recorded and encoded, then get transmitted to the decoder on the server side. Then it get decoded and played back in the server side. To realize our duplex function, we will use the coder and decoder in the both ends of the communication links. Of course, we will need 2 communications sockets, 1 for each direction.
For the parameters selection, we choose the sample rate 8000Hz, 8 bytes for one sample and Frame duration 22.5ms.

### 3.2 PCM CODER/DECODER PERFORMANCE

![Figure 5 PCM Coder/decoder Performance](image)

We use the famous sentence “we were away a year ago”, which we get from one of our homework. As shown in figure 5, the performance of PCM coder/decoder is very good.

### 3.3 “REAL TIME” PROCESSING ISSUES

We encounter 2 major problems from the essence of the real time processing of our project. First, we will discuss multi-thread problem, then the data buffer problem.
When the previous voice is encoded, the current voice must be recorded. So, record process and encode process must work under different threads. As in figure 6, we use different flag for the record process and encode process. We also need to consider this problem for the decode process and playing process.

Buffer handling is another most important programming issue. There are 2 types of linked lists to handle data buffer, the ordinary linear linked list and the circular linked list (Figure 7). The ordinary linear linked list is simpler, but consumes a lot of memory. The circular linked list is complex, but saves the memory.
3.4 PROGRAM INTERFACE AND FUNCTIONS

The program interface is shown in Figure 8. When placing a call, first you input the destination’s IP address. Then, you click the “Connect” button to setup the socket connection with the destination. If successful, you can click the “Call” button to begin the voice conversation. When you finish the call, you use the “Stop” button to end the call and “Exit” button to end the program. To receive a call, you need to click the “Server on” button to listen to the socket connection requests.

![Figure 8 Program Interface](image.png)

4 CONCLUSIONS

In our project, we developed our VoIP software, which realized the voice communications between two hosts. Putting the theory knowledge into practice, we gained a better understanding of real time speech processing and windows socket communication techniques, as well as the knowledge of hardware and software interaction. Our C++ programming skills are also improved during this project.
5 FUTURE WORK
In the future, we will consider the bandwidth efficiency problem. In scenarios with less bandwidth available, such as dialup networks, hybrid coders with lower data rate can be implemented. Other factors, e.g., noise cancellation, echo cancellation, memory usage efficiency might also need to be taken into account later. More functions such as Remote control might also be added up to this software.

6 REFERENCES