ON BUILDING AN INTERNET GATEWAY FOR INTERNET TELEPHONY

Cheng-Yue Chang and Ming-Syan Chen

Electrical Engineering Department
National Taiwan University
Taipei, Taiwan, ROC
cychang@arbor.ee.ntu.edu.tw; mschen@cc.ee.ntu.edu.tw

ABSTRACT

In recent years, the Internet has emerged as an important collaborative platform. Many applications utilize the Internet to provide new kinds of services. Among others, Internet telephony is attracting an increasing amount of attention for the reason that it has the potential to significantly reduce the cost of long distance voice communication. In view of this trend, it is very important to construct a PSTN/Internet gateway for further experiments and developments. In this paper, we explore several methodologies for the implementation of a PSTN/Internet gateway. Based on these methodologies, we devise the corresponding architecture and implement a PSTN/Internet gateway. Following the ITU H.323 recommendation, the gateway built is interoperable with other H.323 compatible terminals. We also validate the gateway built by having fluent two-way communication between a PSTN phone and an H.323 terminal. A performance study on the sensitivity of the buffer size is also conducted. It is noted that the gateway devised is very versatile and can be easily extended to support various applications.

1. INTRODUCTION

Recent technology advances have brought revolutionary impacts to our life. The Internet emerges as a collaborative platform nowadays and it is envisioned to be even more important for years to come. Lots of applications utilize the Internet to provide new kinds of services, such as information services, entertainment, and communication services [1][2]. Search engines, for instance, provide information-searching services over the Internet. Distance education, as another example, makes it possible that the instructor and the students do not need to be physically present in the same location, allowing rural students to have the same opportunity for education as urban students.

Among others, real-time transmission of voice over the Internet, also known as Internet telephony, is attracting an increasing amount of attention [3][4][5]. Many architectures and signaling protocols are proposed in [6][7][8] to provide the Internet telephony service. The reason for the importance of Internet telephony is that Internet telephony has the potential to significantly reduce the cost of long distance voice communication. Additionally, Internet telephony introduces entirely new and enhanced ways of communicating. Video conferencing, application sharing, and white-boarding are some typical applications that exploit real-time voice communication over the Internet. Note that, in addition to phone-to-phone communication, PSTN/Internet gateways can be used to provide phone-to-PC and PC-to-phone communication.

In view of the huge benefits the PSTN/Internet gateway can bring, it is of both theoretical and practical importance to construct one for further experiments and developments. The goal of the paper is thus to design and implement a PSTN/Internet gateway for Internet telephony. Through this gateway, the PSTN user can connect to any Internet user who uses an H.323 [9][10][11][12] compatible terminal (such as Microsoft NetMeeting or Intel Internet Video Phone), and vice versa. In this paper, we explore several methodologies for the implementation of a PSTN/Internet gateway. Based on these methodologies, we devise the corresponding architecture and implement a PSTN/Internet gateway. As will be explained in detail later, the host application of our implementation can be further divided into three major components, i.e., PSTN Event Retriever, State Machine Manager and Audio Channel Manager. We have successfully implemented the PSTN/Internet gateway and validated the system built by having fluent two-way communication between a PSTN phone and a Microsoft NetMeeting terminal. A performance study on the sensitivity of the buffer size has also been conducted.

It is worth mentioning that the gateway we devised is very versatile and can be easily extended to support various applications. Specifically, we extended our gateway to a call center providing voice-mail services over the Internet. The user may take advantage of this service to leave a message for the called party if the line is busy or there is no one answering the call. As a matter of fact, many other applications are conceivable. With the PSTN/Internet gateway prototype, we are able to do proper modifications to cer-

0-7803-6536-4/00/$10.00 (c) 2000 IEEE

tain software components and obtain the customized applications we need very efficiently. This is the very advantage we have by implementing this PSTN/Internet gateway by ourselves.

The paper is organized as follows. The architecture of the PSTN/Internet gateway proposed is introduced in Section 2. In Section3, we explain the implementation of the gateway, and develop a customized application for illustration. Performance analysis of the gateway buffer size is included in Section 4. Finally, we conclude this paper with Section 5.

2. METHODOLOGY FOR BUILDING A PSTN/INTERNET GATEWAY

In this section, we will present methodologies for building a PSTN/Internet gateway. We design state machines for the gateway to maintain existing calls in Section 2.A. The architecture of our gateway is described in Section 2.B.

2.1. State Machine Design

A call is a point-to-point multimedia communication between two endpoints. The call begins with the call setup procedure and ends with the call termination procedure. Since each call has its own progression, i.e., Idle, Offering, Connected, Disconnected, etc, we can utilize a state machine to manage the call. The key elements of a state machine are initial state, final state, intermediate states and transitions between states. We will define the call-states, call transitions and state diagram for our PSTN/Internet gateway in this section.

A call-state is a stage of a call progression. A call is created in one of the following two conditions:

- The H.323 Protocol Stack notifies that a call from the Internet arrives.
- The PSTN interface card notifies that a call from the PSTN arrives.

For a newly created call, the initial state is set to Idle. Then, each call moves from state to state as the call connection is established, connected, and concluded. The particular state of a call is in determining what commands and actions can be performed. Such states are called valid states for a particular command. If a command is issued when the call is not in a valid state for that command, an error occurs.

Each call maintains a current state. The transition from the current state to a new state is triggered by messages or events relating to the call. Note that the determination of the occurrence of an event is important. We can divide this issue into two parts. The first one is how the events from the PSTN side are detected. Using the Dialogic API, we implement a module, PSTN event retriever, to detect the occurrence of the events. The Dialogic PSTN interface card adds a message to its own event queue while its status is changed. Hence, we can detect the events from the PSTN by using the call status transition event functions of Dialogic API to keep retrieving events from the event queue.

The second part is how the events from the Internet side are detected. To detect the events from the Internet side or H.323 system side, we first set up the event handlers by calling the library functions of the H.323 Protocol Stack, and then receive the stack callbacks.

Figure 1 shows a constructing element of a state diagram. Each state is represented by an ellipse that contains the state name. The states are connected with arrows indicating the valid call-state transitions. Each call-state transition includes two parts. The upper one is the message used to cause that state changed and the initiator of the message. The lower one is the responding actions of the host application. According to the principle we can construct the state machines for a call form the PSTN to the Internet and a call from the Internet to the PSTN.

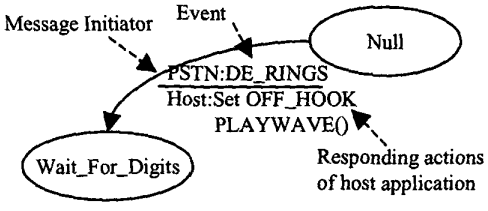

Figure 1: State Diagram Explanation

2.2. Architecture of Our Internet Telephony Gateway

Next, we construct the architecture for our PSTN/Internet gateway. The gateway consists of four layers and is depicted in Figure 2. The four layers are composed of a network interface layer, an operating system layer, an application program interface (API) layer and a host application layer.

The network interface layer is the bottom of the architecture, and is the physical layer of the architecture. It consists of two network interface cards, Ethernet interface card and PSTN interface card.

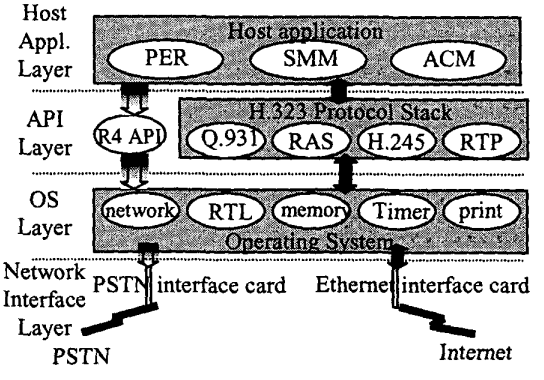

Figure 2: The architecture of our Internet telephony gateway

The operating system layer manages the network interface layer, and provides the upper layer with the common operating system services. These services include network operations, dynamic memory allocations, run-time libraries, system timer, and print. We adopt Microsoft Windows NT Server as our operating system because of the interoperability between API and the NT operating system layer and also the ability of the NT operating system to execute programs in the multi-thread mode.

The application program interface layer consists of Dialogic API [13] and the H.323 Protocol Stack API. Dialogic API provides programmers with a set of libraries to make use of the PSTN interface card while H.323 Protocol Stack API implements the mandatory elements of H.323 as function libraries.

The host application consists of three major modules: State Machine Manager (SMM), PSTN Event Retriever (PER) and Audio Channel Manager (ACM). The function blocks of the host program are shown in Figure 3. For existing calls, the host application maintains independent state diagrams in the system. Hence, we design a module, State Machine Manager, to manage these state diagrams. The SMM keeps track of the call-state for each call, and triggers the call-state transitions by retrieving events from the PSTN interface card and the callback from the H.323 Protocol Stack. The responsibility of the PSTN Event Retriever is hence to keep retrieving events from the event queue of the PSTN interface card.

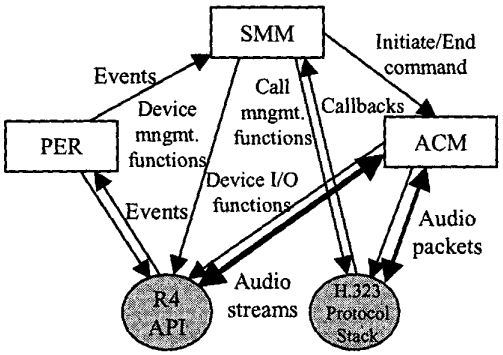

Figure 3: The function blocks of host program we devised

Audio Channel Manager (ACM) is designed to conduct the buffering between PSTN interface card and the H.323 protocol stack. ACM is created for each call after the audio channels have been opened. For each call from PSTN to Internet (explicitly, NetMeeting in this very case), it uses the I/O functions of Dialogic API to retrieve the audio data from the PSTN interface card and buffers the data with our buffering mechanism. Then it packetizes and transmits the audio data to the called party by the library functions of the H.323 Protocol Stack. For the data flow toward the other direction, ACM receives and unpacketizes the audio packet by the library functions of the H.323 Protocol Stack. Then, it buffers the audio data, and finally uses the I/O functions of Dialogic API to play it out.

3. DEVELOPING CUSTOMIZED APPLICATIONS

It is worth mentioning that the gateway we devised is very versatile and can be easily extended to support various applications. For illustrative purposes, we extended our gateway to a call center providing voice-mail services over the Internet. Consider the following example. A PSTN user wants to make a call to his friend on the Internet. However, the line is busy or there is no one answering the call. In this case he would like to leave a message for his friend. Using our gateway, a call center is designed to support this service. The call center asks the caller to leave a message whenever he fails to make a call. If the user would like to leave the message, the call center asks for the receiver's E-mail address and begins to record his message to a WAVE file. Then it encodes the file according to MIME [14] format and sends the file as an email to the receiver.

To provide this service, we insert two new states, Wait_For_Address and Wait_For_Message to the state machine for a call form PSTN to the Internet. The host application is notified that one call is disconnected, and then it prompts the PSTN user to enter the receiver's E-mail address (i.e., by calling dx_play() and dx_getdig()). The state then transits to Wait_For_Address. Once the user has finished entering the receiver's E-mail address, the host application is notified by the message, TDX_GETDIG, retrieved by PSTN Event Retriever. It then begins to record the message (i.e., by calling dx_record()), and the state transits to Wait_For_Message. Wait_For_Message waits for the completion of recording the message until the host application is notified by the message, TDX_RECORD, retrieved by PSTN Event Retriever. It then saves the recorded file for later encoding and sending. The state transits to Idle. Note that within the above two states, the PSTN user may hang up the call, and the state transits to Idle directly. As a matter of fact, many other applications are conceivable. With the PSTN/Internet gateway prototype, we are able to do proper modifications to certain software and obtain the customized applications we need very efficiently. This is the very advantage we have by implementing this PSTN/Internet gateway by ourselves.

4. PERFORMANCE ANALYSIS

The performance of our PSTN/Internet gateway is defined as the continuity of the speech from the remote party. The primary effective variable of the continuity is related to

inter-arrival jitters between received audio packets. In order to improve the performance of our gateway, we shall minimize the number of the occurrences of inter-arrival jitters. The approach we employed is to pre-buffer some audio packets in the Audio channel Manager. As a result, when an inter-arrival jitter is long, Audio Channel Manager can still consume the pre-buffered data to keep the playback speed over the active voice channel on the PSTN interface card. The more data pre-buffered, the less likelihood the system buffer will become empty. However, having more data pre-buffered means a longer delay between the speaking party and the receiving party. Therefore, we have to make a trade-off between pre-buffered data size and the number of times which the buffer is consumed to be empty.

To choose an optimal pre-buffered data size for our gateway, we measure the number of times when the buffer becomes empty under various pre-buffered data sizes. Two experiments, for the duration of 3 minutes and 5 minutes, are conducted.

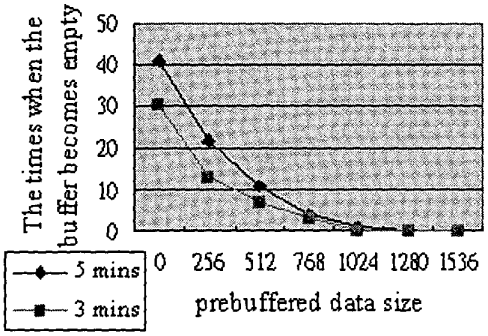

Figure 4: Experimental Result

As shown in Figure 4, the experimental result indicates that when the pre-buffered data size is set to be 1024 bytes, the number of times when the buffer becomes empty is almost 0, implying that a pre-buffered data size of 1024 bytes is proper for this experiment.

5. CONCLUSIONS

In this paper, we designed and implemented a PSTN/Internet gateway for Internet telephony. The gateway is designed to follow the ITU H.323 Recommendation, and is hence interoperable with other H.323 compatible endpoints on the Internet. By this gateway, we not only interconnect the PSTN and the Internet, but also exploit the advantages of the two wide-area-networks. We proposed the architecture of a PSTN/Internet gateway. Based on the architecture, we implemented the PSTN/Internet gateway, and conducted several experiments in this prototype. We have validated the gateway built by having fluent two-way communication between a PSTN phone and an H.323 terminal. A performance study on the sensitivity of the buffer size has also been conducted. It is noted that the gateway devised is very versatile and can be easily extended to support various applications.

6. ACKNOWLEDGMENT

The authors are supported in part by the Ministry of Education Project No. 89-E-FA06-2-4-7 and the National Science Council, Project No. NSC 89-2219-E-002-007 and NSC 89-2213-E-002-032, Taiwan, Republic of China.

7. REFERENCES

[1] B. Braden, D. Clark, and S. Shenker, "RFC 1633: Integrated Services in the Internet Architecture: An Overview."

[2] F. Fluckiger, *Understanding Network Multimedia Application and Technology*. Prentice Hall, 1995.

[3] M. Aras, J. F. Kurose, D. S. Reeves, and H. Schulzrinne, "Real-time Communications in Packet-switched Networks," *Proceedings of the IEEE, vol. 82*, pp. 122–139, January 1994.

[4] C. A. Polyzois, K. H. Purdy, P. F. Yang, D. Shrader, H. Sinnreich, F. Mard, and H. Schulzrinne, "From POTS to PANS - A Commentary on the Evolution to Internet Telephony," *IEEE Network, vol. 3*, May/June 1999.

[5] C. Reyes-Aldasoro and F. Kuhlmann, "Telecommunications and Internet in the Future Society: Myths and Realities," in *Computers and Communications, 1999. Proceedings. IEEE International Symposium on*, July 1999.

[6] C. Huitema, J. Cameron, P. Mouchtaris, and D. Smyk, "An Architecture for Residential Internet Telephony Service," *IEEE Network, Vol.13*, pp. 50–56, May-June 1999.

[7] H. Schulzrinne and J. Rosenberg, "Internet Telephony: Architecture and Protocols - an IETF Perspective," *Computer Networks and ISDN Systems vol.31*, pp. 237–255, Feb. 1999.

[8] G. A. Thom, "H.323: The Multimedia Communications Standard for Local Area Networks," *IEEE Communications Magazine*, pp. 52–56, December 1996.

[9] CCITT/ITU-T, "Recommendation G.711: Pulse Code Modulation of Voice Frequencies."

[10] ITU-T, "Recommendation H.225: Media Stream Packetization and Synchronization on Non-guaranteed Quality of Service LANs."

[11] ITU-T, "Recommendation H.323: Visual Telephone Systems and Equipment for Local Area Networks which Provide a Non-guaranteed Quality of Services."

[12] ITU-T, "Recommendation Q.931: ISDN User-network Interface Layer 3 Specification for Basic Call Control."

[13] *User's Manual: Voice Programmer's Guide for Windows NT*. Dialogic Corporation, 1996.

[14] N. Borenstein and N. Freed, "RFC 1521: MIME (Multipurpose Internet Mail Extensions) Part 1 - Mechanisms for Specifying and Describing the Format of Internet Message Bodies."