Design and Implementation of Voice Interaction System Between PC Terminal and PSTN in Intranet

Intranet PC终端与PSTN语音交互系统的设计和实现

施寒潇　吕强　朱巧明　杨季文　(苏州大学计算机工程系　215006)

摘要：本文介绍了内部网内PC终端与电话语音交互的整个流程思想及具体实例。系统主要是采用了客户机/服务器模式，利用Windows的多任务机制，结合Windows MDK低层音频服务、Windows Sockets和语音压缩等技术以及现有的CTI应用产品来实现的，这大大方便了PC端用户跟PSTN的联系。

关键词：VoIP　Winsock　计算机电话集成　Internet电话网关

1 引言

Internet在全球范围内呈爆炸性增长的趋势，它的主要业务已由传统的文件传输、电子邮件和远程登录等基本服务转向视频点播、IP电话、远程医疗等应用为主的多媒体服务。以VoIP（Voice Over IP）等业务为代表的通信业务就充分体现了现代通信技术和Internet技术的融合。在逐渐由电路交换向IP交换发展过渡的过程中，CTI（Computer Telephony Integration）即计算机电话集成技术的实现无疑是最早能够体现数据网和电话网二网合一的一种新兴应用技术。

以往中小型企业的通话系统往往需要专门布设线路和交换机，而这些设备是比较昂贵的。企业内部局域网络的带宽一般都在10M以上，如果只用来收发Email、信息查询和公文流转的话，则利用率太低。本系统充分利用CTI技术和现有的内部网络，实现了语音的实时通信。

2 系统概况

2.1 系统结构

本系统的设计模式是Client/Server，客户端就是PC端，服务器端也就是IP网关。前者的主要工作是采集声卡上的语音数据，然后进行压缩和发送，同时接收和解压从服务器端发送过来的语音数据，并把它放到音频输出队列进行播放。后者的工作比较复杂，需要对连接用户进行管理，还要针对Intranet和PSTN(Public Switched Telephone Network)这两个不同网络进行协议的转换。同时对语音数据进行采集、压缩、发送，另外的过程则是接收、解压、播放。这两个进程是并行的。系统结构图见图1。

2.2 系统特点

整个系统的开发工作是基于Internet Protocol，服务器端采用电话语音卡来实现电话网关，客户端利用系统平台所提供的API来完成语音数据的采集和放音工作。

系统有以下三个特点：

(1) 能够更加高效地利用网络资源。而企业内部网络的带宽一般很高（10M以上），正是考虑到这一点，本系统采用了先进的数字信号处理技术，可以将原64kb／s的话音信号压缩成8kb/s或更低码速率的数据流，能够在同一条线路上传输比采用模拟技术时更多的呼叫，大大提高了效率。并且采用的是分组交换技术，可以实现信道的统计复用，使得网络资源的利用率更好，大大降低投入成本。

(2) 友好的用户界面，便捷的用户管理。企业所属员工每人都有一个分机号（和他的IP地址一一对应）。当用户打开客户端程序，连到服务器上之后，这时就可以在自己的PC机上拨打或接听你的电话了。

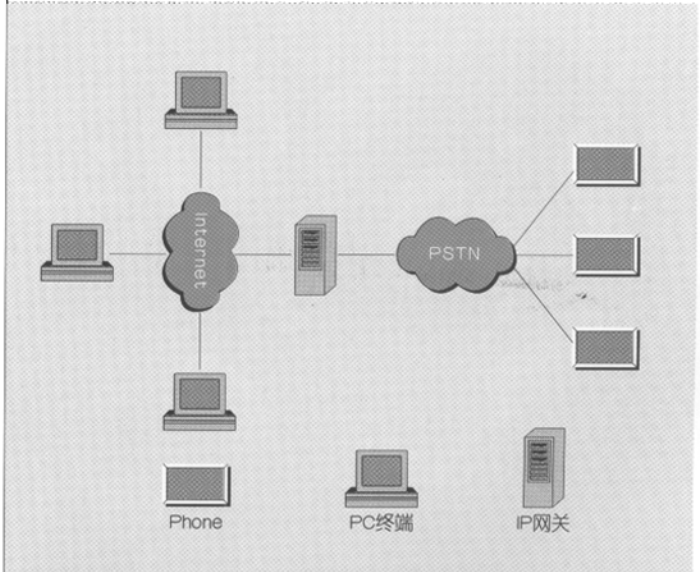


图1　系统结构图

(3) 节约了人力资源.本系统可以把接线人员从值班岗位上解放出来，减轻了他们的工作强度。因为系统可以对拨入电话进行自动处理，通过服务器连接到所对应的目标用户。

3 系统设计

3.1 PC端的实现

PC终端上的工作主要分用户管理和通话过程的管理。用户管理主要实现对用户的登录及多媒体设备的管理和用户通话状态的管理。多媒体设备管理主要是检测多媒体资源是否冲突，以及分配相应的资源给PC终端的应用程序。用户通话状态的管理主要是记录当前用户的通话状态(忙，还是空闲)，并把此信息及时反馈给服务器端。

通话过程的管理是PC终端设计的主要内容，它分两个方面来实现，一是完成语音信息经声卡及相关处理之后交于网络发送，二是接收从网络上传送过来的语音数据，经过一定的处理送声卡播放。

第一方面的工作需要以下两个步骤来完成：

(1) 数据采样及压缩。对从麦克风传来的语音进行数字化，接下来不仅要排除无用信号，而且对数字化后的语音信号进行压缩。对于语音数字化和排除无用信号的工作一般都交于声卡去完成。而对于语音信号的压缩工作，则需要采用一定的算法来实现，编解码器是这一过程的关键部件。

(2) 数据打包及发送。压缩后的数据在进行网络传输之前，必须做打包工作。除此之外，还需要在数据包加入一些协议信息。协议信息的目的是为了更好的保障完成数据的传输，比如为防止各个数据包没有按顺序到达而加上包的顺序号，以及数据校验信息。做完这些工作之后就可以准备发送了。

对于第二方面的工作也需要两个步骤来完成：

① 数据接收及解压，网络上传来的数据都是一个个压缩数据包，同时由于每个包可能通过了不同的路由到达，所以它们到达目的地的顺序和原始的顺序有一定的差别。因此当每个包到达主机时，要检查包的序号并将它放到正确的位置。在交给相应音频播放器之前，必须先解压，解压算法其实就是压缩算法的逆工作。

② 语音播放。根据实时工作的要求，把解压后的语音数据包连续放到播放序列进行不间断播放，从而还原了语音信息。

3.2 服务器端的实现

服务器端的工作就是要实现一个电话网关。它主要实现用户管理、通话管理和带宽管理。

用户管理主要负责用户的注册、用户的登录和用户通话状态的观察等工作。为了便于在服务端统一用户的管理，用户在第一次使用时必须进行注册，由服务端分配一个唯一的虚拟分机号和相应的用户帐号及密码，外部人员可以通过拨打该分机号码与该用户通话。其次服务器端会记录下当前用户的信息，包括用户是否登陆以及登录所在机器的信息，通话的次数和所用时间。还有服务端会不断检测各登录用户的通话状态，是空闲、正在通话还是转接中，及时把信息反馈给想跟他通话的对象。

通话管理包含电话语音数据的处理和话音数据的实时传输。电话语音数据处理需要完成电话信号和计算机信号的相互转换，在用户和外部电话进行通话时，通话管理将PC终端发送过来的语音数据压缩包进行一定处理转换成电话语音数据，同时实时采集电话语音数据并转换成语音数据压缩包发送给通话的PC终端。这些信号数据和协议的转换工作是由语音卡来完成。

如果现在需要两台PC终端来进行通信，那么通话管理只要完成把分机号转换成对应的IP地址，然后转由两台终端进行PC to PC的直接通话，从而减轻了服务端的工作。话音数据的实时传输指的是PC终端和外部电话通信的时候，服务端和PC终端所进行的数据传输要求必须是连续实时的。它包括了数据的压缩和解压缩，以及打包发送。这方面的工作跟PC终端数据的压缩打包类似。

语音通信对带宽非常敏感，由于带宽的限制，会导致语音包的丢失，而在通话中出现时有中断和停顿的情况，这对用户是不满意的。当多对用户同时进行通话的时候很可能会因为带宽的不足而导致网络发生阻塞。在服务端提供了带宽管理来解决这个问题，它将限制同时发生在网络中通话的连接数目，从而限制保证了通话的质量。

4 系统关键技术

4.1 电话语音卡

系统主要通过电话语音卡来完成信令转换、音频处理等工作。选择电话语音卡的主要标准是要满足系统需求，并且满足今后系统的升级及扩充。基于当前的系统要求，首先电话语音卡在同一条电话线路连接下，既要对PC端传送过来的语音数据进行一定的处理并向电话端播放，同时对电话端的话音进行录制、处理再发送到PC端，这是一个全双工的工作，也就是说电话语音卡必须支持全双工的通话工作状态。其次根据实时性的要求，语音卡必须能够提供缓冲机制，将电话端采集到的音频数据存放到内存后再进行发送，同时能够对从PC端传输过来的语音数据经过处理之后，通过内存缓冲区来实现播放。第三，针对不同的需求，能够提供支持多种语音格式。不同的语音数据格式对于网络的传输速率、语音信号的质量都有不同的影响，因此希望语音卡能够支持尽量多的语音格式。

4.2 多媒体低层音频服务

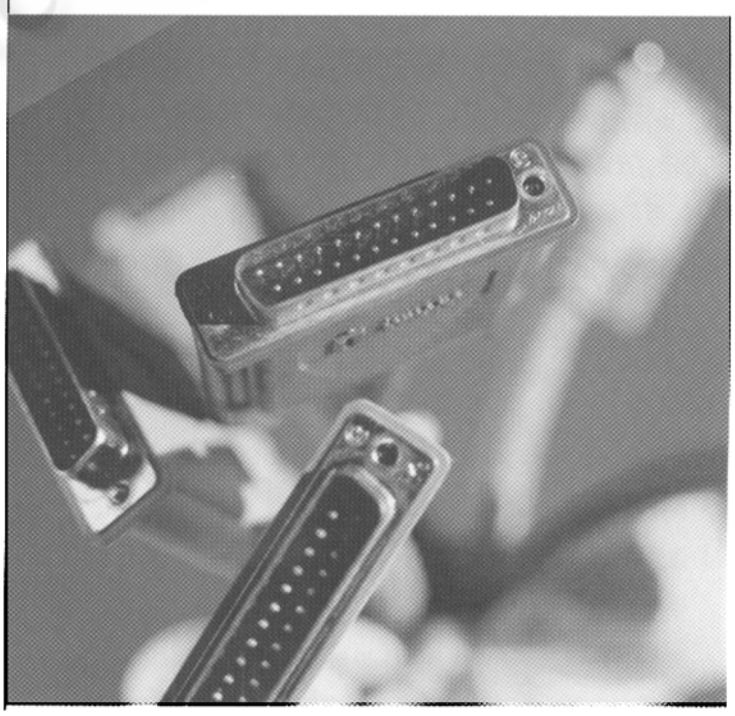
系统是要把语音直接转换为数据，存放在内存中，而不是存为语音文件，同时播放语音时，也是直接播放语音数据，而不是播放语音文件。这样的好处是省略了读写硬盘的费时操作，提高了语音通话的实时性。如果想完成上述操作，编程语言中提供的容易使用的高级多媒体语音函数是无法胜任的，只能通过 Windows MDK（Multimedia Development Kit）中的多媒体低层音频服务来实现的。Windows下录制或播放音频数据，其主要操作就是将音频数据读出到音频设备驱动程序和从音频设备驱动程序写入的操作。低层波形音频函数通过WAVEHDR结构的音频数据块对设备驱动程序的音频数据进行上述控制。以录音为例，其准备工作主要有几点，打开录音设备，获得录音句柄，指定录音格式，分配若干用于录音的内存。开始录音时，先将所有内存块都提供给录音设备用来录音，录音设备就会依次将语音数据写入内存，当一块内存写满，录音设备就会发一个Window消息MM_WIM_ DATA给相应的窗口，通知程序作相关的处理，这时程序通常的处理是把内存中的数据进行复制，在此我们的处理是把数据进行压缩和通过网络发送，然后把内存置空，返还给录音设备进行录音，这样就形成一个不断循环录音的过程。结束录音时就释放所有内存块，关闭录音设备。关键的录音函数和顺序如下：

```
WAVEFORMATEX m_pWaveFormat;
m_pWaveFormat.wFormatTag= WAVE_FORMAT_PCM;
m_pWaveFormat.nChannels=1;
m_pWaveFormat.nSamplesPerSec=8000;
m_pWaveFormat.nAvgBytesPerSec=8000;
m_pWaveFormat.nBlockAlign=1;
m_pWaveFormat.cbSize=0;
m_pWaveFormat.wBitsPerSample=16; // 指定录音格式
int res=waveInOpen(&m_hWaveIn,WAVE_MAPPER, &m_pWaveFormat,
(DWORD) m_hWnd,0L,CA
LLBACK_WINDOW); // 打开录音设备
waveInPrepareHeader(m_hWaveIn,m_pWaveHdr [i],sizeof
(WAVEHDR)); // 准备内存块录音
waveInAddBuffer(m_hWaveIn,m_pWaveHdr [i],sizeof(WAVEHDR)); /
/ 增加内存块
waveInStart(m_hWaveIn);// 开始录音
waveInStop(m_hWaveIn); // 停止录音
waveInReset(m_hWaveIn); // 清空内存块
waveInClose(m_hWaveIn); // 关闭录音设备
```

开发人员可以充分利用Windows操作系统的多任务机制和低层音频服务中的回调机制（callback mechanism），在原始声音进行采样的同时对采样数据进行实时的音频处理，并实时播放处理后的音频数据，使录音、音频处理和放音三个原本独立的过程异步并行处理，以达到原始声音采样和处理后的声音同步播放的实时效果。

4.3 语音数据的压缩及解压

音频数据是大多数多媒体应用程序向用户提供信息的主要方式，这些数据一般具有较高的采样速率，如果不经过压缩的话，保存它们需要消耗大量的存储空间，在网络上进行传输的效率也很低，因此音频数字压缩编码在多媒体技术中占有很重要的地位。目前常用的压缩方法有很多种，不



同的方法具有不同的压缩比和还原音质，编码的格式和算法也各不相同，其中某些压缩算法相当复杂，普通程序不可能去实现其编解码算法。所幸的是，Windows 9x/ NT 4.0/Windows 2000为多媒体应用程序提供了强大的支持，引入了ACM（Audio Compression Manager，音频压缩管理器），它负责管理系统中所有音频编解码器（Coder-Decoder，简称CODEC，是实现音频数据编解码的驱动程序），应用程序可以通过ACM提供的编程接口调用这些系统中现成的编解码器来实现音频数据的压缩和解压缩。Windows 9x/NT 4.0/2000系统自带的音频CODECs支持一些音频数据压缩标准，如Microsoft ADPCM、Microsoft Interactive Multimedia Association (IMA) ADPCM、DSP Group TrueSpeech(TM)等。通过多次实验比较，系统最终选用了DSP Group TrueSpeech(TM)，因为它压缩率很高，可达10：1，非常适合对在Internet上传输的数据的压缩，大大降低了对带宽的要求，同时其还原效果也不错。

4.4 数据传输

本系统采用基于Socket的TCP/IP协议通信，通过在网络传输层建立两端计算机的连接来实现实时通信。为了便于在Windows平台上开发，微软公司推出了一套Windows Sockets，它不仅包含人们所熟悉的Berkeley Socket风格的库函数，也包含了一组针对Windows的扩展库函数，以使程序员能充分地利用Windows消息驱动机制进行编程。根据传输数据类型的不同，Windows Sockets可分为数据流Socket（SOCK_STREAM）和数据报Socket（SOCK_DGRAM）两类。

数据流Socket提供了双向的、有序的、无差错、无重复并且是无记录边界的数据流服务，TCP/IP协议使用该类接口。数据报Socket提供双向的，但不保证是可靠的、有序的、无重复的数据流服务，也就是说一个从数据报Socket接受信息的进程有可能发现信息重复了，或者和发出的顺序不同。

针对本系统的特点，我们采用数据流Socket和数据报Socket相结合的办法。在语音通信开始之前的所有工作，如用户登录、状态信息的交流等，我们采用的是数据流Socket，因为这部分工作必须保证无差错。语音通信数据的传输则采用的是数据报Socket，因为语音数据的传输必须保证实时、快速。同时系统是基于企业内部网络，网络传输性能较高，出现丢失多个连续的数据包的可能性很小，所以在语音数据的传输上采用数据报Socket。

同时还要指出的是系统使用的版本是Windows Socket2，相对Windows Socket1.1来说，Windows socket2提供了快速、多线程数据传送的能力，性能更加先进；并支持对多种网络传输方式的一致性访问和独立于协议的多点传输/组播。更为重要的是，它提供了在新的网络介质（ATM、ISDN等）上QoS(Quality Of Service)的接口，这样多媒体应用开发人员可以请求指定传输速率，能够根据传输的吞吐量建立或者拆除连接，当网络暂时不可用时应用程序应该能自动得到提示，同时还方便以后系统的升级。基于以上考虑，我们在系统的实现过程选择采用Windows Socket2来实现对于网络数据的传输，实现过程虽然相对复杂，但是给系统带来了良好的性能和扩展性。

5 小结

尽管Internet最初的设计目的是为了处理数据业务，但随着科技的发展和技术的进步，把语音和数据服务融为一体已不再是奢望，而是实实在在已出现在我们的面前了，同时它符合三网合一（电话网、广播电视网、数据网）的发展方向。要实现PC to Phone最关键的问题是如何实现协议之间以及模拟信号与数据信号之间的转换，这在前几年是很难做到集成的。

但如今CTI技术飞速发展，我们完全可以利用现有的硬件产品结合相应的软件开发技术来实现它。

随着公司的壮大，可能办公地点不只是局限于同个地方，这样的话，可以更加体现系统的优越性。我们只要对公司职员进行统一管理，一个职员只拥有一个帐号和一个分机号。同时随着公司电话门数的增多，我们可以对程序进行简单的升级，充分利用语音卡所支持的多路电话接口和通道来完成电话的扩容。

通过本系统的实现过程，我们完全有理由相信基于CTI技术和IP技术会得到越来越广泛的应用。

参考文献

1 施寒潇、朱巧明、吕强，基于Internet的语音交互系统的设计和实现［J］，电子工程师，2002，(3)：20-22。

2 苗兰波、冯志勇、吕廷杰，IP电话网络技术［M］，电子工业出版社，2001。

3 周敬利、余胜生，多媒体计算机声卡技术及应用［M］，电子工业出版社，1998。

4 潘爱民、王国印译，Visual C++技术内幕［M］，清华大学出版社，1999。

5 侯俊杰，深入浅出MFC［M］，华中科技大学出版社，2001。

6 李维春，一个基于CTI技术的VoIP解决方案的设计与实现［D］，苏州大学硕士学位论文，2000。