下的数据，还需要再进行4次调用。用户可以使用第三种类型的套接字来访问原始网络。这种类型的套接字尤其适用于实时应用和用户想要实现特定错误处理模式的情况。数据包可能会丢失或者被网络重排序。和前两种方式不同，这种方式没有任何保证。第三种方式的优点是有更高的性能，而有时候它比可靠性更加重要（如在传输多媒体时，快速比正确性更有用）。

在创建套接字时，有一个参数指定使用的协议。对于可靠字节流通信来说，使用最广泛的协议是TCP（传输控制协议）。对于不可靠数据包传输来说，UDP（用户数据报协议）是最常用的协议。这两种协议都位于IP（互联网协议）层之上。这些协议都源于美国国防部的ARPANET，现在成为互联网的基础。没有可靠数据包流类型的通用协议。

在一个套接字能够用于网络通信之前，必须有一个地址与它绑定。这个地址可以是几个命名域中的一个。最常用的域为互联网（Internet）命名域，它在V4（第4个版本）中使用32位整数作为其命名端点，在V6中使用128位整数（V5是一个实验系统，从未成为主流）。

一旦套接字在源计算机和目的计算机都建立成功，则两个计算机之间可以建立起一个连接（对于面向连接的通信来说）。一方在本地套接字上使用一个listen系统调用，它创建一个缓冲区并且阻塞，直到数据到来。另一方使用connect系统调用，并且把本地套接字的文件描述符和远程套接字的地址作为参数传递进去。如果远程一方接受了此次调用，则系统在两个套接字之间建立起一个连接。

一旦连接建立成功，它的功能就类似于一个管道。一个进程可以使用本地套接字的文件描述符来从中读写数据。当此连接不再需要时，可以用常用的方式，即通过close系统调用来关闭它。

10.5.3 Linux的输入/输出系统调用

Linux系统中的每个I/O设备都有一个特殊文件与其关联。大部分的I/O只使用合适的文件就可以完成，并不需要特殊的系统调用。然而，有时需要一些设备专用的处理。在POSIX之前，大部分UNIX系统有一个叫作ioctl的系统调用，它在特殊文件上执行大量设备专用的操作。数年之间，此系统调用已经变得非常混乱。POSIX对其进行了清理，把它的功能划分为主要面向终端设备的独立的功能调用。在Linux和现代UNIX系统中，每个功能调用是独立的系统调用，还是它们共享一个单独的系统调用或者其他的方式，都是依赖于实现的。

在图10-20中的前4个系统调用用来设置和获取终端速度。为输入和输出提供不同的系统调用是因为一些调制解调器工作速率不同。例如，旧的可视图文系统允许用户在家通过短请求以75位/s的上传速度访问服务器上的公共数据，而下载速度为1200位/s。这个标准在一段时间内被采用，因为对于家庭应用来说，输入输出时都采用1200位/秒则太昂贵了。网络世界中的时代已经改变了。不对称性仍然存在，一些电话公司提供8Mbps的入站服务和512kbps的出站服务，称为ADSL（非对称数字用户环线）。

函数调用	描述
s=cfsetospeed (&termios,speed)	设置输出速率
s=cfsetispeed (&termios,speed)	设置输入速率
s=cfgetospeed (&termios,speed)	获取输出速率
s=cfgtetispeed (&termios,speed)	获取输入速率
s=tcsetattr (fd,opt,&termios)	设置属性
s=tcgetattr (fd,&termios)	获取属性

图10-20 管理终端的主要POSIX系统调用

列表中的最后两个系统调用主要用来设置和读回所有用来消除字符和行以及中断进程等功能的特殊字符。另外，它们可以使回显有效或无效，管理流控制及其他相关功能。还有一些I/O功能调用，但是它们都是专用的，所以这里就不进一步讨论了。此外，ioctl系统调用依然可用。

10.5.4 输入/输出在Linux中的实现

在Linux中I/O是通过一系列的设备驱动来实现的，每个设备类型对应一个设备驱动。设备驱动的功能是对系统的其他部分隔离硬件的细节。通过在驱动程序和操作系统其他部分之间提供一层标准的接口，使得大部分I/O系统可以被划归到内核的机器无关部分。

当用户访问一个特殊文件时，由文件系统提供此特殊文件的主设备号和次设备号，并判断它是一个块特殊文件还是一个字符特殊文件。主设备号用于索引存有字符设备或者块设备数据结构的两个内部散列表之一。定位到的数据结构包含指向打开设备、读设备、写设备等功能的函数指针。次设备号被当作