当客户敲该门时，程序为 serverSocket 调用 accept()，这在服务器中创建了一个称为 connectionSocket 的新套接字，由这个特定的客户专用。客户和服务器则完成了握手，在客户的 clientSocket 和服务器的 serverSocket 之间创建了一个 TCP 连接。借助于创建的 TCP 连接，客户与服务器现在能够通过该连接相互发送字节。使用 TCP，从一侧发送的所有字节不仅确保到达另一侧，而且确保按序到达。

```
connectionSocket.close()
```

在此程序中，在向客户发送修改的句子后，我们关闭了该连接套接字。但由于 serverSocket 保持打开，所以另一个客户此时能够敲门并向该服务器发送一个句子要求修改。

我们现在完成了 TCP 套接字编程的讨论。建议你在两台单独的主机上运行这两个程序，也可以修改它们以达到稍微不同的目的。你应当将前面两个 UDP 程序与这两个 TCP 程序进行比较，观察它们的不同之处。你也应当做在第 2、4 和 7 章后面描述的套接字编程作业。最后，我们希望在掌握了这些和更先进的套接字程序后的某天，你将能够编写你自己的流行网络应用程序，变得非常富有和声名卓著，并记得本书的作者！

2.8 小结

在本章中，我们从概念和实现两方面对网络应用进行了学习。我们学习了被因特网应用普遍采用的客户 - 服务器模式，并且知道了该模式在 HTTP、FTP、SMTP、POP3 和 DNS 等协议中的使用。我们已经更为详细地学习了这些重要的应用层协议以及与之对应的相关应用（Web、文件传输、电子邮件和 DNS）。我们也已学习了日益流行的 P2P 体系结构以及它如何应用在许多应用程序中。我们还探讨了使用套接字 API 构建网络应用程序的方法。我们考察了面向连接的（TCP）和无连接的（UDP）端到端传输服务中的套接字应用。至此，我们在分层的网络体系结构中的向下之旅已经完成了第一步。

在本书一开始的 1.1 节中，我们对协议给出了一个相当含糊的框架性定义："在两个或多个通信实体之间交换报文的格式和次序，以及对某报文或其他事件传输和/或接收所采取的动作。"本章中的内容，特别是我们对 HTTP、FTP、SMTP、POP3 和 DNS 协议进行的细致研究，已经为这个定义加入了相当可观的实质性的内容。协议是网络连接中的核心概念；对应用层协议的学习，为我们提供了有关协议内涵的更为直觉的认识。

在 2.1 节中，我们描述了 TCP 和 UDP 为调用它们的应用提供的服务模型。当我们在 2.7 节中开发运行在 TCP 和 UDP 之上的简单应用程序时，我们对这些服务模型进行了更加深入的观察。然而，我们几乎没有介绍 TCP 和 UDP 是如何提供这种服务模型的。例如，我们知道 TCP 提供了一种可靠数据服务，但我们未说它是如何做到这一点的。在下一章中我们将不仅关注运输协议*是什么*，而且还关注它*如何工作*以及*为什么*要这么做。

有了因特网应用程序结构和应用层协议的知识之后，我们现在准备继续沿该协议栈向下，在第 3 章中探讨运输层。

课后习题和问题

复习题

2.1 节

R1. 列出 5 种非专用的因特网应用及它们所使用的应用层协议。