我们使用同样简单的客户－服务器应用程序来展示 TCP 套接字编程：客户向服务器发送一行数据，服务器将这行改为大写并回送给客户。图 2-30 着重显示了客户和服务器的主要与套接字相关的活动，两者通过 TCP 运输服务进行通信。

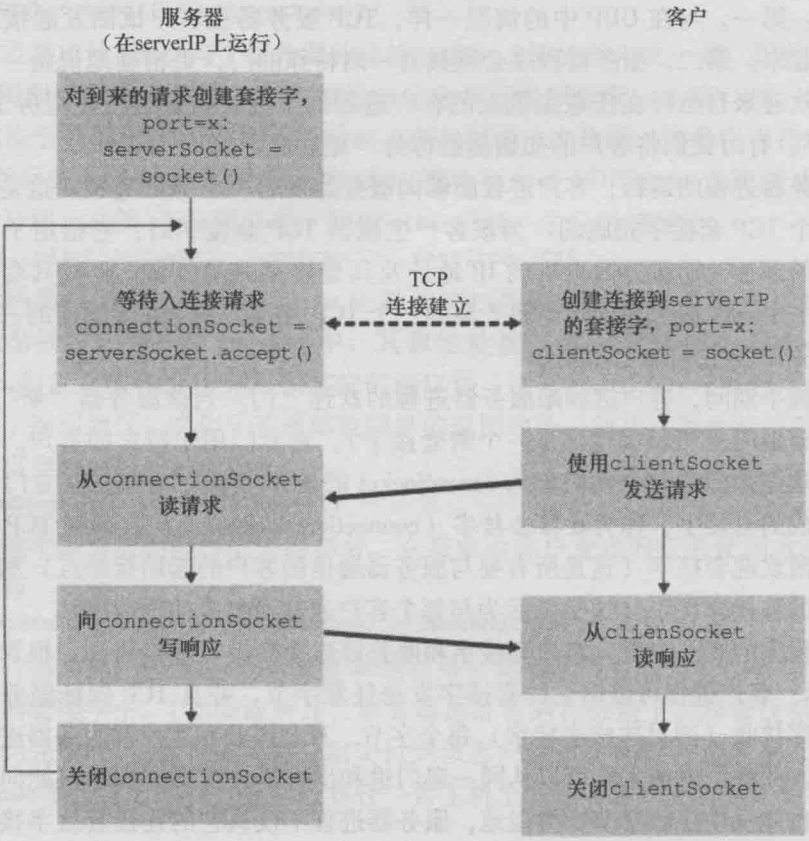

图 2-30 使用 TCP 的客户－服务器应用程序

1. TCPClient. py

这里给出了应用程序客户端的代码：

```
from socket import *
serverName = 'servername'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = raw_input('Input lowercase sentence:')
clientSocket.send(sentence)
modifiedSentence = clientSocket.recv(1024)
print 'From Server:', modifiedSentence
clientSocket.close()
```

现在我们查看这些代码中的与 UDP 实现有很大差别的各行。首当其冲的行是客户套接字的创建。

```
clientSocket = socket(AF_INET, SOCK_STREAM)
```

该行创建了客户的套接字，称为 clientSocket。第一个参数仍指示底层网络使用的是 IPv4。第二个参数指示该套接字是 SOCK_STREAM 类型。这表明它是一个 TCP 套接字（而不是一个 UDP 套接字）。值得注意的是当我们创建该客户套接字时仍未指定其端口号；相反，我们让操作系统为我们做此事。此时的下一行代码与我们在 UDPClient 中看到的极为不同：