

计算机网络编程

第12章 基于TCP的客户机/服务器程序

信息工程学院 方徽星

fanghuixing@hotmail.com

大纲

- 设计目的
- 相关知识
- 例题分析

1. 设计目的

- **通过基于TCP的客户机与服务器程序设计**
 - **了解TCP协议的基本概念与主要功能**
 - **掌握这类网络应用的设计思路与编程方法**

2. 相关知识

- TCP协议的主要特点

TCP主要用于对传输可靠性要求高的应用层协议

应用层	FTP、HTTP、SMTP、Telnet、DNS、SNMP、DHCP、TFTP...
传输层	TCP 、UDP、SCTP
网络层	IP、ARP、IGMP、ICMP

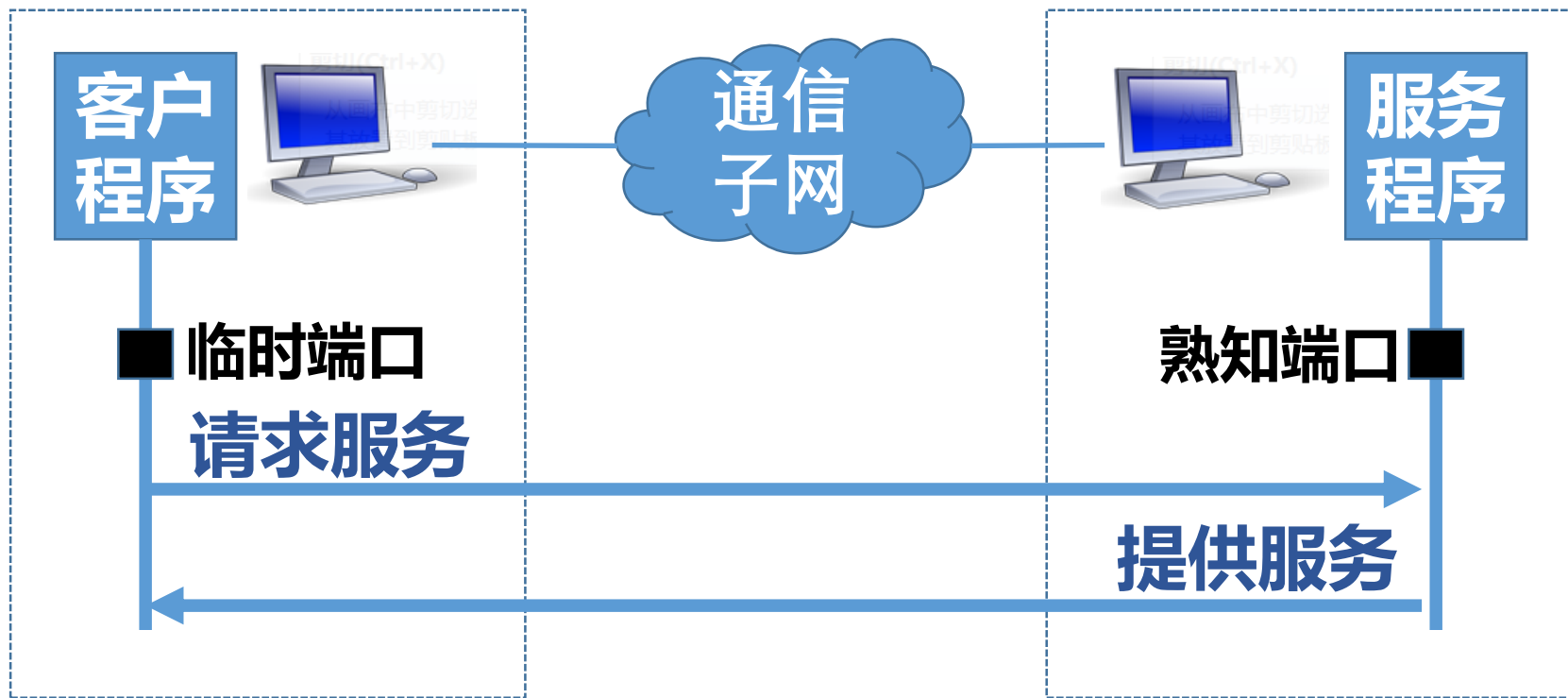
面向连接 + 可靠的传输 + 数据流传输

2. 相关知识

- **客户机/服务器编程**

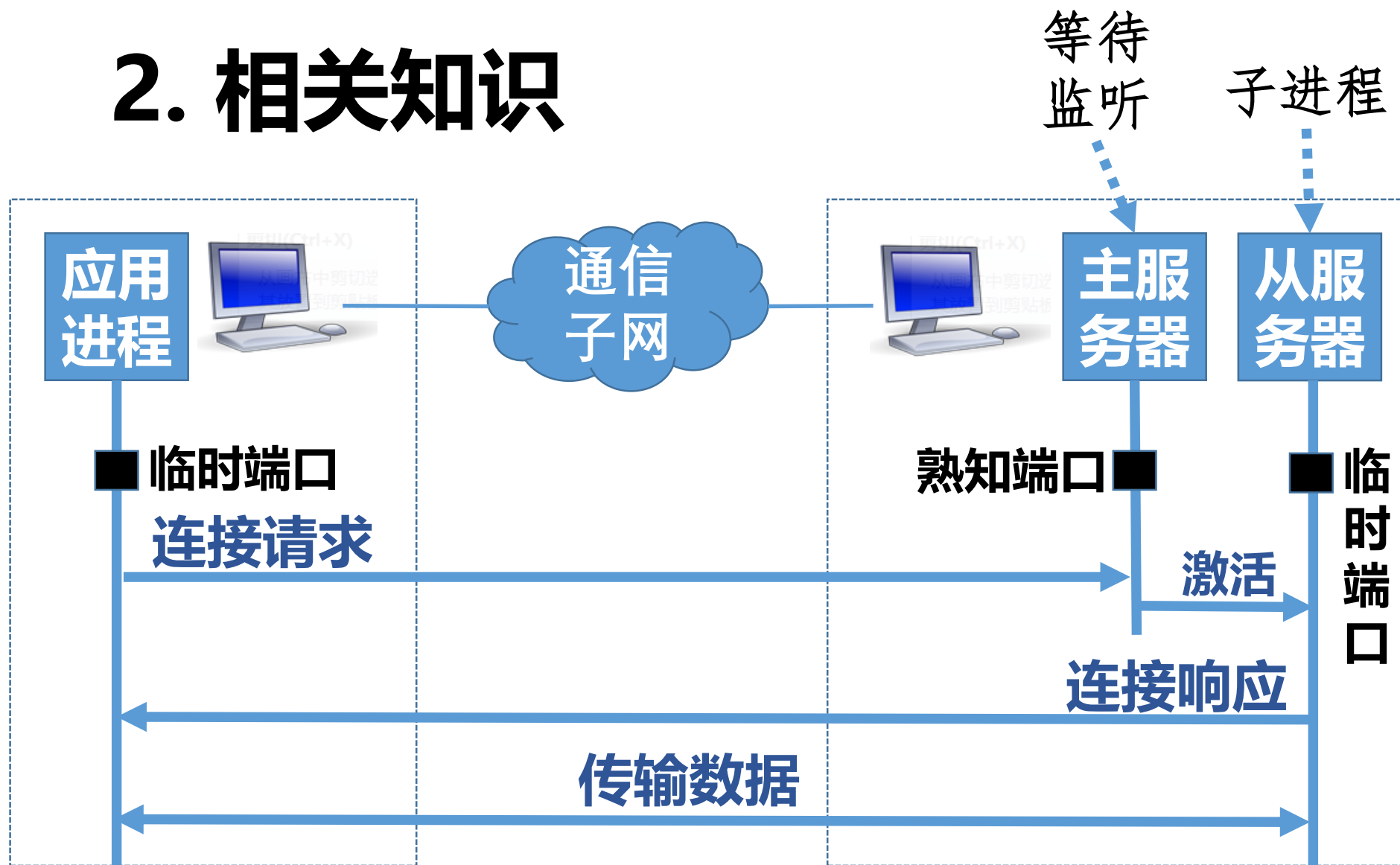
- 基于TCP的网络应用采用客户机/服务器模式
- **客户机是使用网络服务的应用进程**
- **服务器是提供网络服务的应用进程**

2. 相关知识



基于TCP的客户机/服务器结构

2. 相关知识



并发服务器工作原理

3. 例题分析：设计要求

- 根据基于TCP的客户机/服务器工作模式，编写服务器程序接收客户机的命令，并根据命令向客户机做出响应
 - 客户机向服务器发送sendfile命令
 - 服务器向客户机返回command ok响应
 - 客户机向服务器发送指定的数据

3. 例题分析：设计要求

- 具体要求
 - 要求程序为命令程序

TcpServer server_port



服务器侦听的TCP端口

3. 例题分析：设计要求

- 具体要求
 - 要求将服务器的状态显示在控制台上

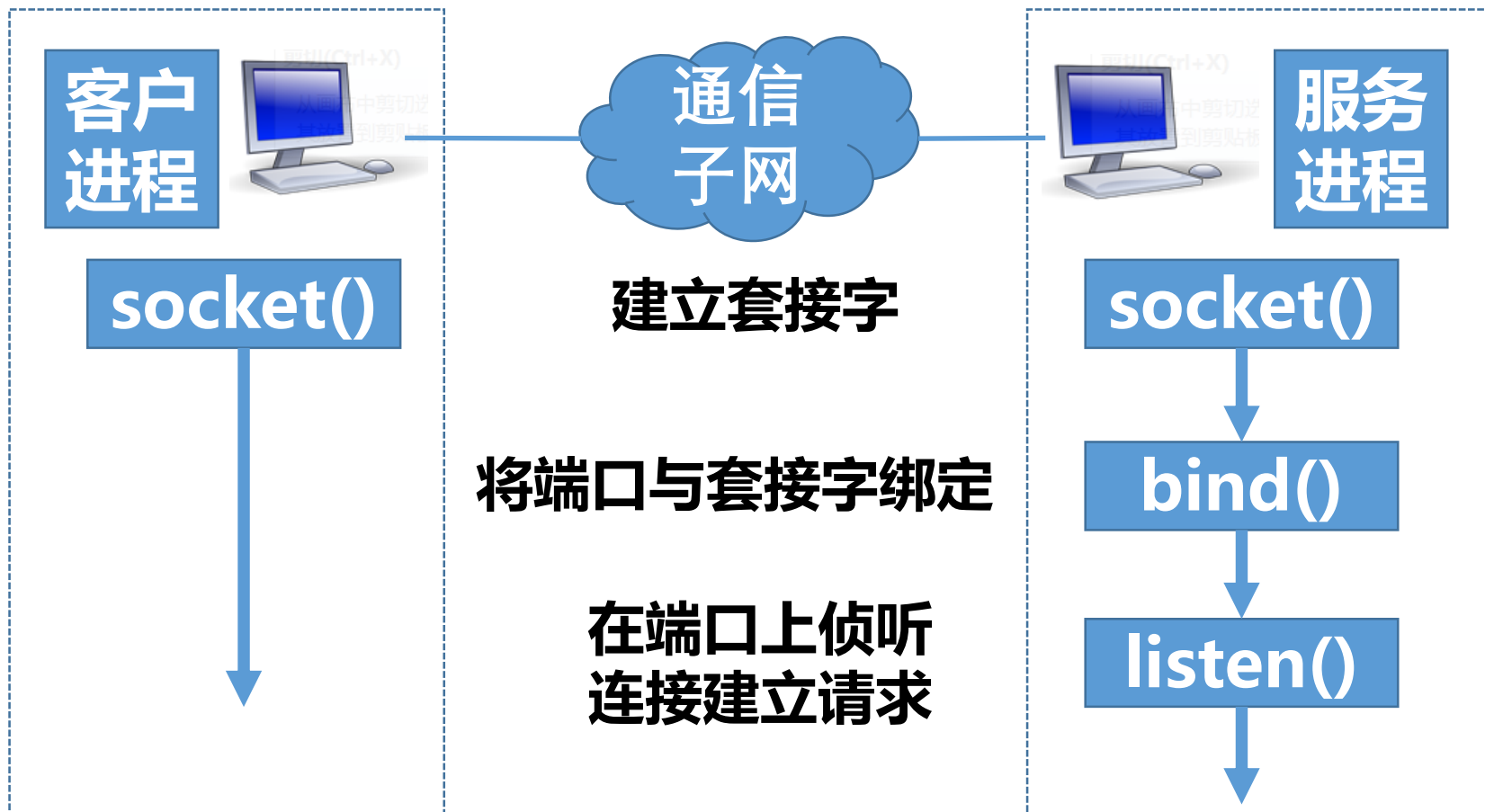
TCP Server开始侦听xx端口

TCP Server与TCP Client建立连接

TCP Server接收数据: ...

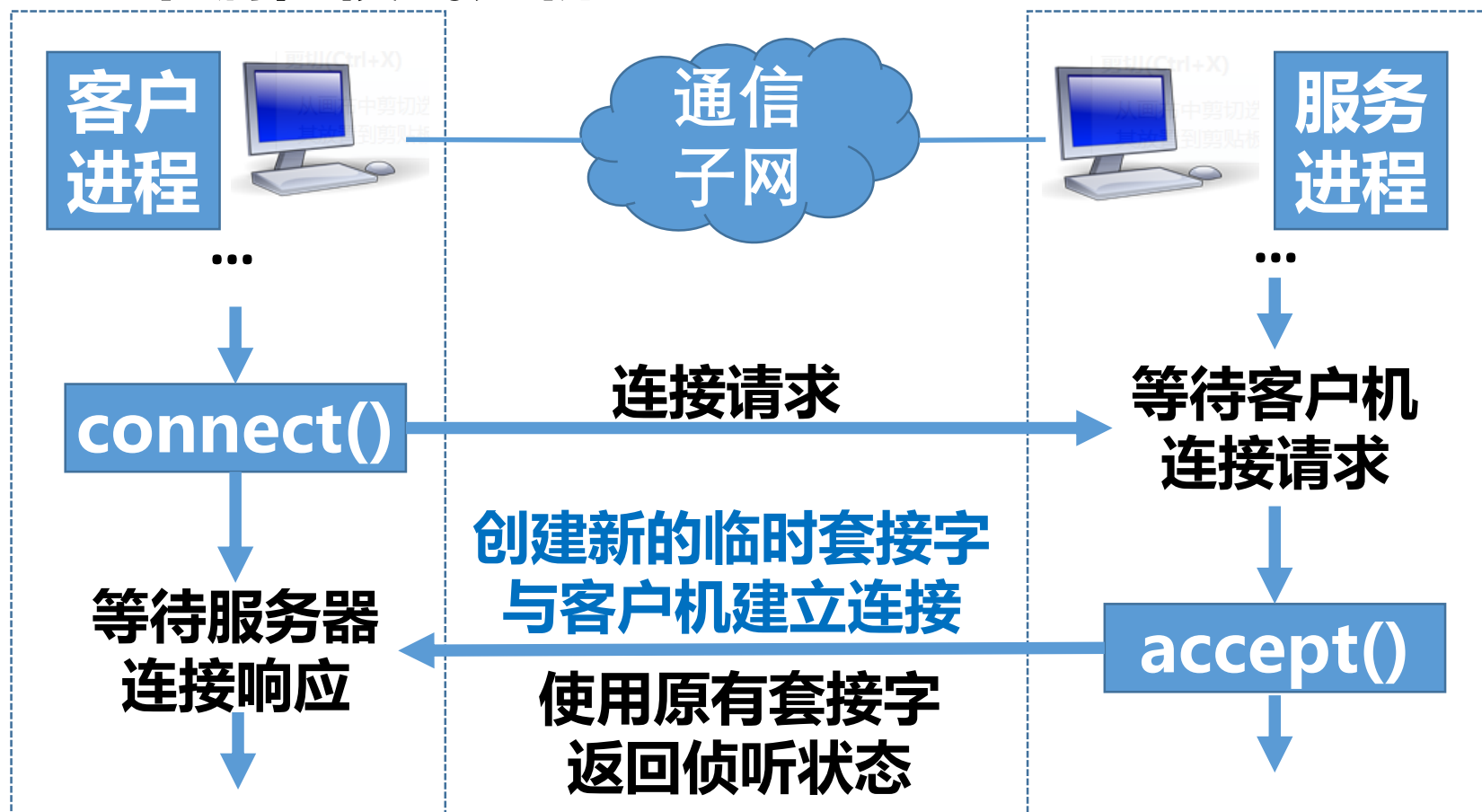
3. 例题分析：关键问题

• 基本编程模式分析



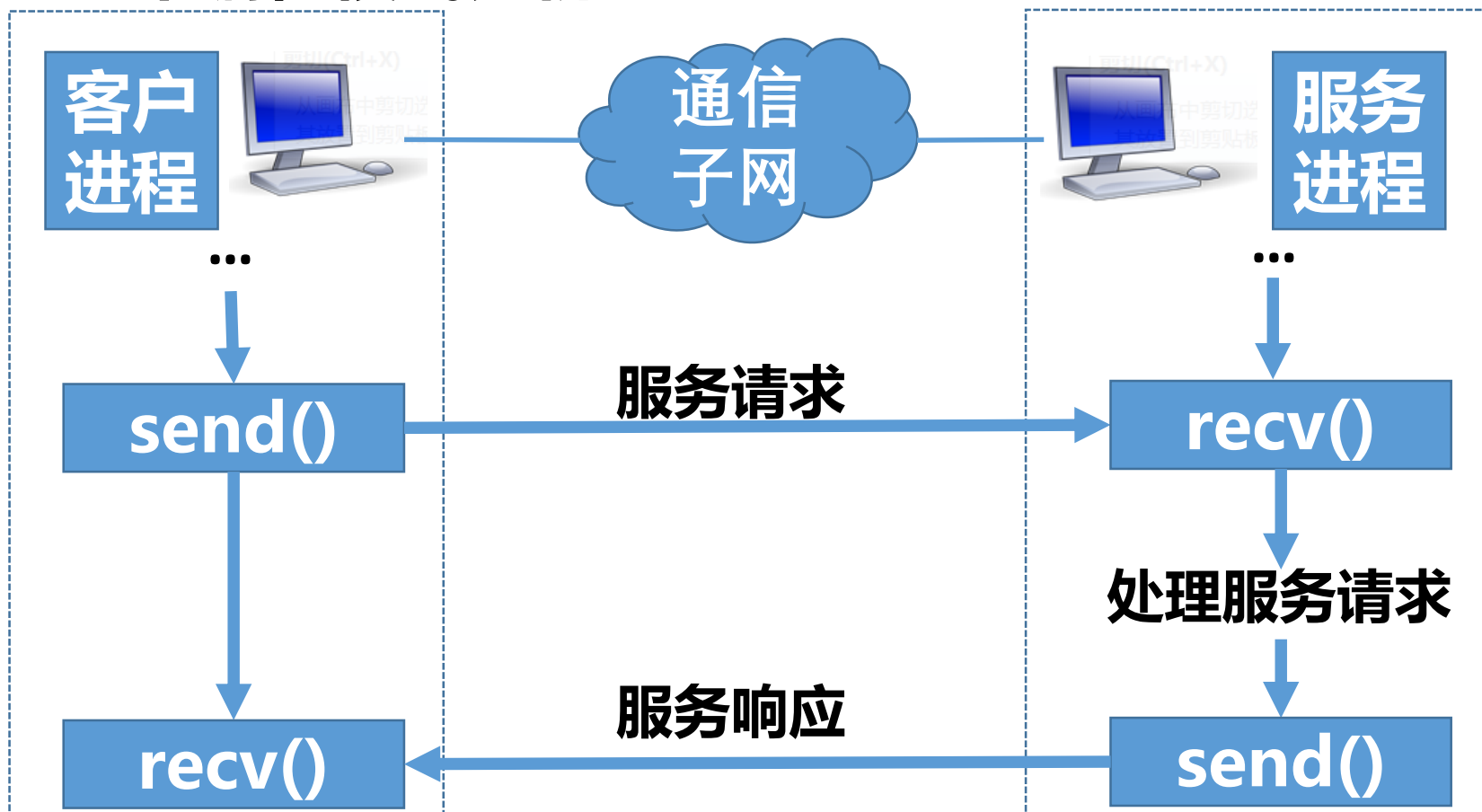
3. 例题分析：关键问题

• 基本编程模式分析



3. 例题分析：关键问题

• 基本编程模式分析



3. 例题分析：关键问题

- 创建流式套接字

```
//创建流式套接字
```

```
sock=socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
```

```
//填充本地Socket地址
```

```
sockaddr_in serveraddr;
```

```
serveraddr.sin_family = AF_INET;
```

```
serveraddr.sin_port = htons((unsigned short)  
                                atoi(argv[1]));
```

```
serveraddr.sin_addr.S_un.S_addr = htonl(INADDR_ANY);
```

0.0.0.0

3. 例题分析：关键问题

- 创建流式套接字

```
//将端口、IP地址与套接字绑定
```

```
bind(sock, (sockaddr*)&serveraddr, sizeof(serveraddr));
```

```
//在端口上侦听连接
```

```
listen(sock, SOMAXCONN);
```

```
//连接请求等待队列长度为最大值0x7fffffff
```

3. 例题分析：关键问题

- 与客户机建立并发连接，服务器需要并发处理多个连接请求
 - 需要在一个主循环中接收客户请求
 - 并创建新的服务线程与客户机建立连接
 - 侦听到连接建立请求到达时，调用accept()函数创建临时套接字与客户机建立连接
 - 服务器使用原有套接字继续侦听

3. 例题分析：关键问题

```
//创建保存线程参数的数据结构
```

```
struct ThreadParam
```

```
{
```

```
    SOCKET sock;
```

```
    sockaddr_in addr;
```

```
};
```

```
//在主循环中接收客户机的连接并创建服务线程
```

```
while(true){    ...    }
```

```
//在主循环中接收客户机的连接并创建服务线程
```

```
while(true){
```

```
//接受客户机的连接请求
```

```
    tempsock = accept(sock,  
                      (sockaddr*)&tempaddr,  
                      &tempflen);
```

```
//当线程数达到上限，停止接受客户机连接
```

```
    if(ThreadCount >= 10) {  
        closesocket(tempsock);  
        continue;
```

```
    }
```

```
...
```

```
}
```

**存放客
户端地
址信息**



```
while(true) {  
    ...  
    //设置传递给线程的参数  
    ThreadParam Param;  
    Param.sock = tempsock;  
    Param.addr = tempaddr;  
  
    DWORD dwThreadId;  
    //为每个客户机创建服务线程  
    ...  
}
```

```
while(true) {  
...  
CreateThread(  
    NULL, //线程安全相关信息, 使用默认设置时传递NULL  
    0, //要分配给线程的栈大小, 传递0时生成默认大小的栈  
    ServerThread, //执行函数指针  
    &Param, //调用执行函数时传递的参数信息  
    0, //指定线程创建后的行为,  
        //传递0时, 线程创建后立即进入可执行状态  
    &dwThreadId //用于保存线程ID的变量地址值  
);  
  
}
```

3. 例题分析：关键问题

- 在线程中发送与接收数据

```
DWORD WINAPI ServerThread(LPVOID lpParam)
{
    //从线程参数中获得临时套接字
    SOCKET tempsock =
        ((ThreadParam*) lpParam) -> sock;
    sockaddr_in tempaddr =
        ((ThreadParam*) lpParam) -> addr;
    ...
}
```

```
DWORD WINAPI ServerThread(LPVOID lpParam)
{
...
//通过端口接收客户机命令
recv(
    tempsock, //连接的套接字
    recvbuf, //保存接收数据的缓存地址
    sizeof(recvbuf), //缓存字节数
    0 //接收数据时指定的可选项信息
);
...
}
```

send & recv函数的可选项及含义

可选项	含义	send	recv
MSG_OOB	用于传输带外数据	Y	Y
MSG_PEEK	验证输入缓冲区中是否存在接收的数据		Y
MSG_DONTROUTE	数据传输过程中不参照路由表, 在本地网络中寻找目的地	Y	
MSG_DONTWAIT	调用I/O函数时不阻塞	Y	Y
MSG_WAITALL	防止函数返回, 直到接收全部请求的字节数		Y

DWORD WINAPI ServerThread(LPVOID lpParam)

{

...

if(strcmp(recvbuf, "sendfile") != 0)

...

//通过端口向客户机返回响应

**send(tempsock,
 "command ok" ,
 sizeof("command ok"),
 0**

);

//通过端口接收客户机数据

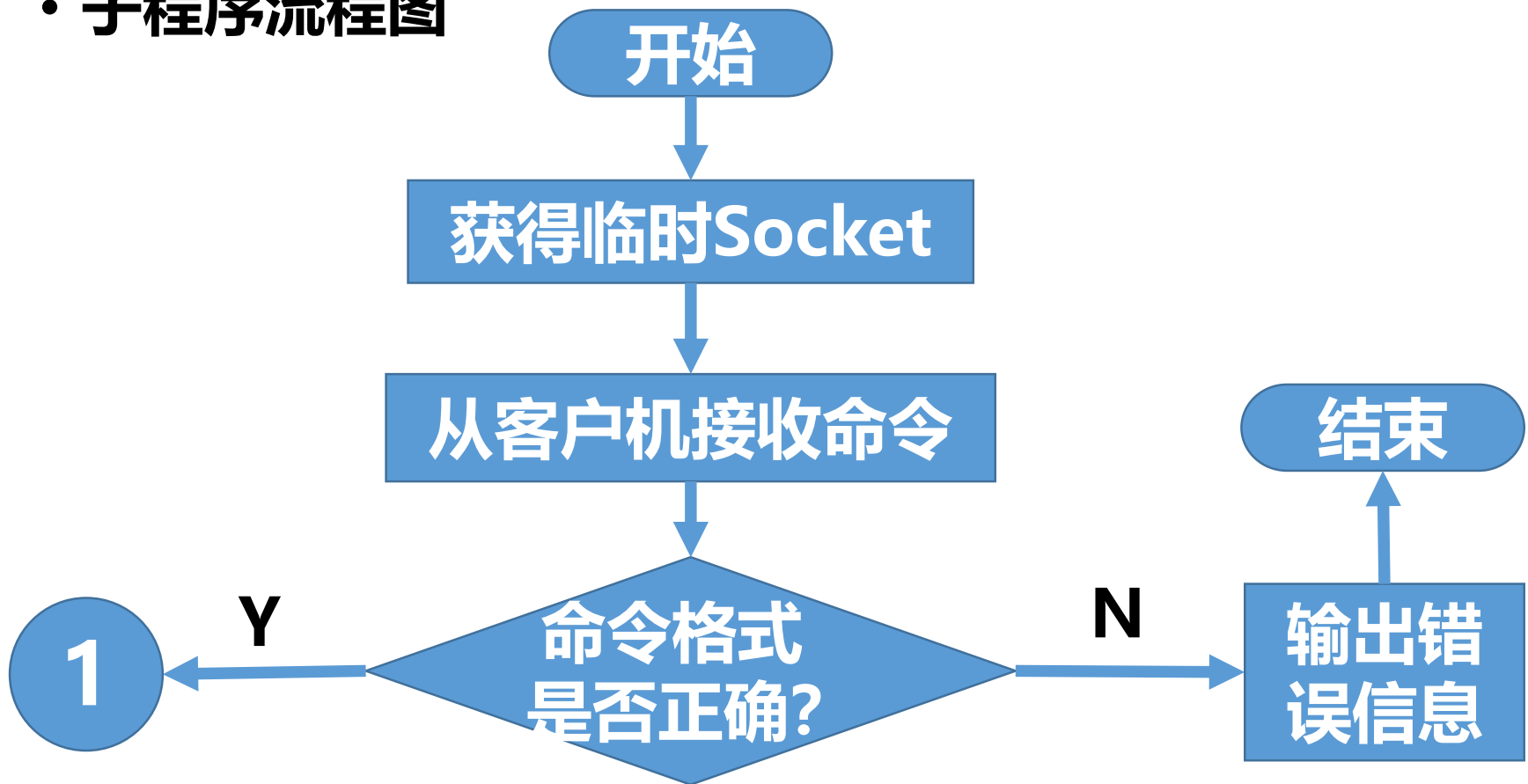
recv(tempsock, recvbuf, sizeof(recvbuf), 0);

closesocket(tempsock);**//关闭临时套接字**

}

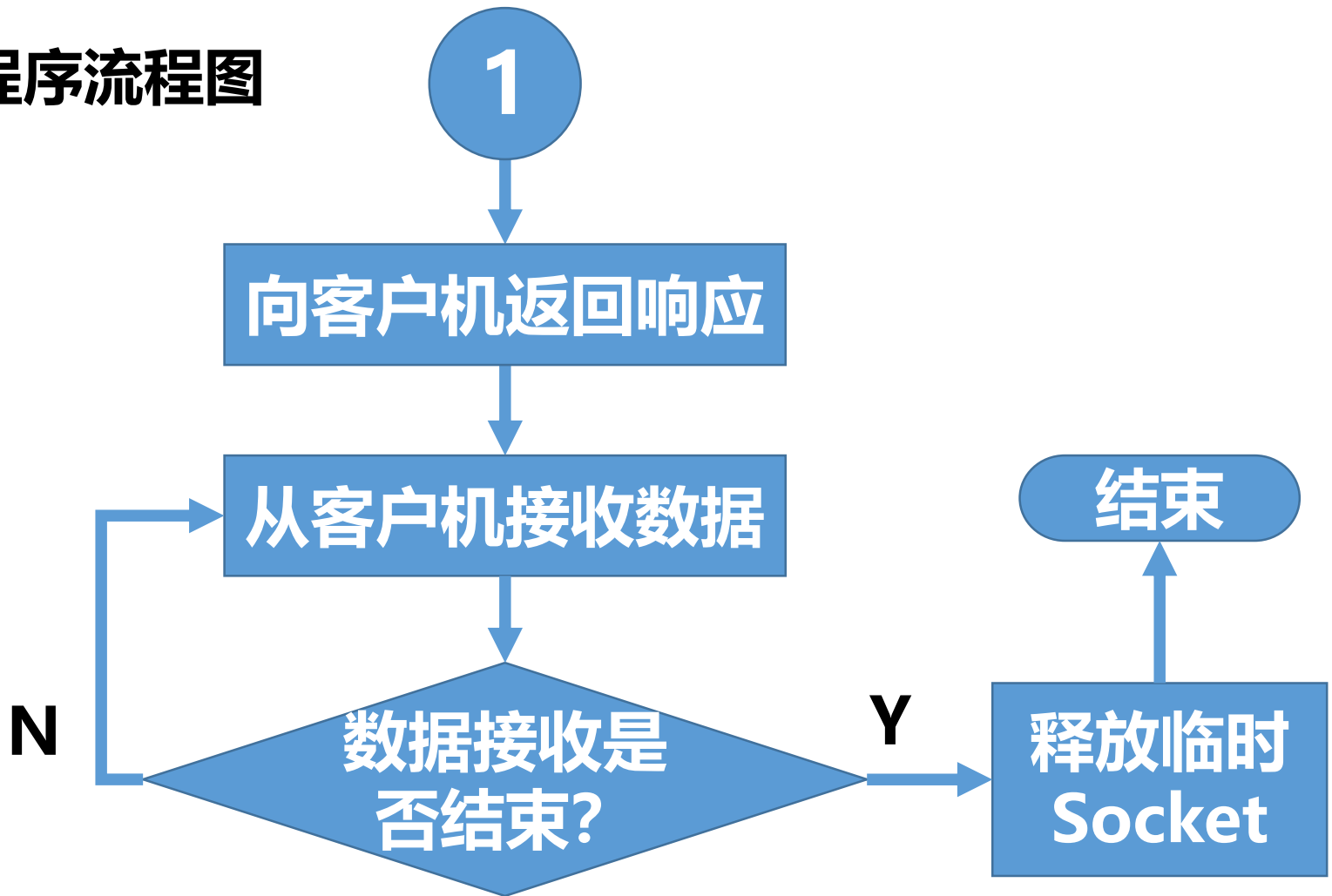
3. 例题分析：关键问题

- 子程序流程图

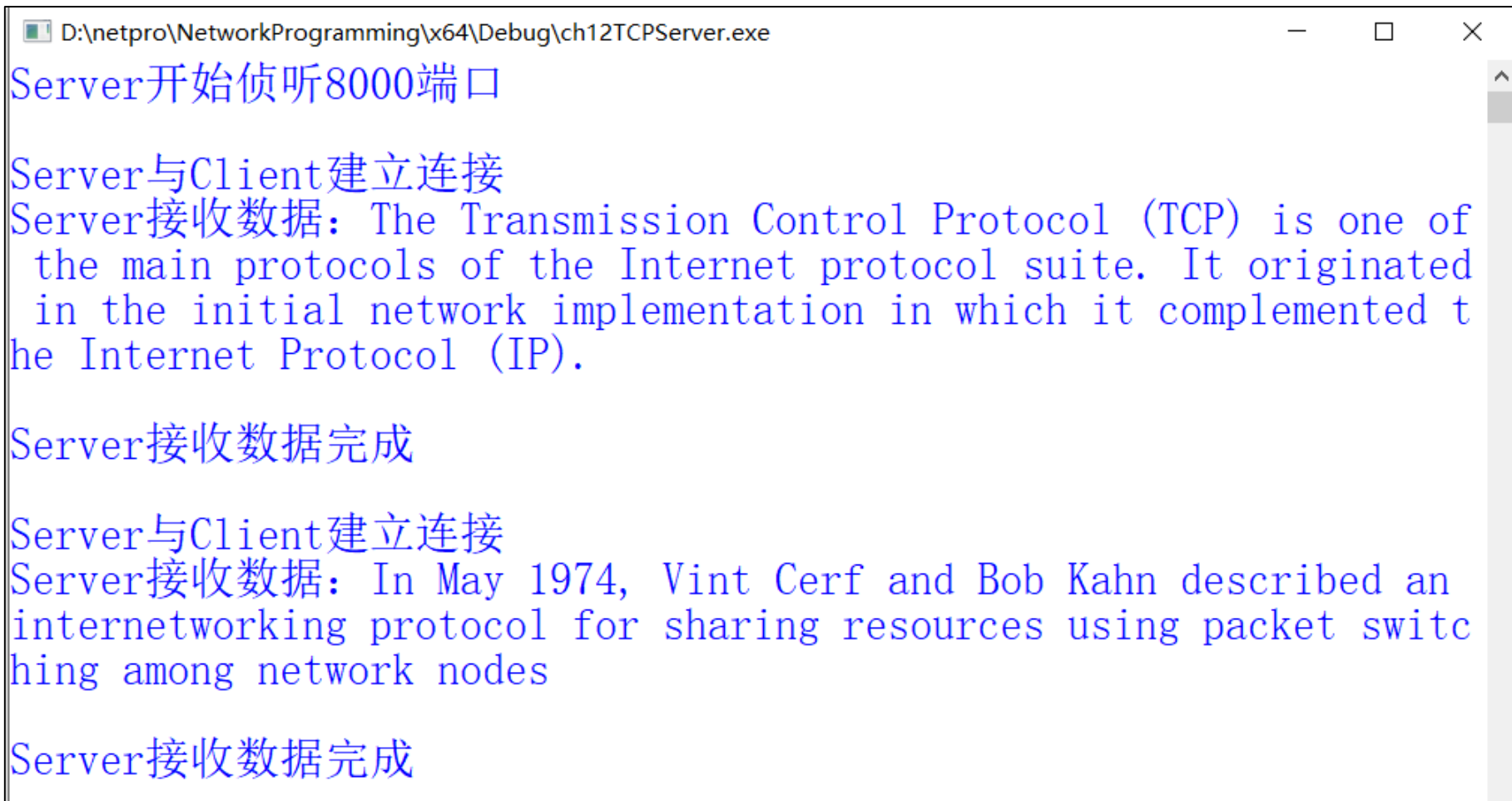


3. 例题分析：关键问题

- 子程序流程图



3. 例题分析：程序演示



The screenshot shows a Windows command prompt window titled "D:\netpro\NetworkProgramming\x64\Debug\ch12TCPServer.exe". The window contains the following text:

```
Server开始侦听8000端口  
  
Server与Client建立连接  
Server接收数据: The Transmission Control Protocol (TCP) is one of  
the main protocols of the Internet protocol suite. It originated  
in the initial network implementation in which it complemented t  
he Internet Protocol (IP).  
  
Server接收数据完成  
  
Server与Client建立连接  
Server接收数据: In May 1974, Vint Cerf and Bob Kahn described an  
internetworking protocol for sharing resources using packet switc  
hing among network nodes  
  
Server接收数据完成
```

作业

- **P143-练习题，说明文档发送到
fanghuixing@hotmail.com**

本章小结

- **设计目的**

- 了解TCP协议的基本概念与主要功能
- 掌握这类网络应用的设计思路与编程方法

- **相关知识**

- TCP主要特点、客户机/服务器编程模式

- **例题分析**

- 基本编程模式分析、流失套接字、并发连接、线程中发送和接收数据