基于 Socket 的边远岛传输系统设计与实现*

李　博，成方林，叶　颖，王　鹏，张翼飞

（国家海洋技术中心　天津　300112）

摘　　要：为了保证边远海岛系统监控数据在传输过程中的安全性、完整性、实时性以及传输效率，文章结合 Socket 网络编程接口和多线程技术，提出了在边远海岛数据传输系统中采用服务器端和客户端程序的系统机构。以实际业务化监控过程中进行测试，结果表明系统可以有效地保证传输数据的安全性、完整性和实时性，增加了传输数据的线程数量、发送数据缓冲区域和接受数据缓冲区域提高数据传输效率。

关　键　词：边远海岛；套接字；数据传输；多线程

1　引言

边远海岛数据的利用与监控系统由岸基数据中心、岛基子系统和通信子系统组成，岸基数据中心与岛基子系统通过无线卫星和地面专线两种方式进行通信。其中，岛基子系统由远程视频子系统、被动声呐子系统、水下地质灾害子系统、陆源地震子系统以及集成工作站组成，集成工作站负责接收、处理、显示、发送、备份各子系统传来的数据。岸基数据中心由卫星接收系统、辅助决策系统和岸基数据服务器组成。其中，岸基数据服务器完成专线数据接收、处理、备份以及为服务决策系统提供数据源。

在边远海岛数据传输系统中，利用套接字作为通信系统的接口可以实现数据的跨平台传输，采用多线程可以有效提高传输效率。本文将介绍在边远岛监控系统中以套接字和多线程为基础的数据传输子系统的设计与实现。

2　系统设计

本系统采用了 Socket 套接字和多线程技术。Socket 套接字是一种广泛的、开放的、支持多种协议的网络编程接口。该协议支持单一的通信，既 Internet 域。各通信进程通过这个域互相之间用 Internet 协议进行通信。套接字（socket）是通信的基石，是支持网络通信的基本单元。可以将套接字看做不同主机间的进程进行双向通信的端点。一个正在被使用的套接字接口都有它的类型和与其相关的进程。套接字可以分为两类：流套接字（stream socket）和数据报套接（datagram socket）。流套接字是面向连接的，通信双方进行数据交换前，必须建立一条路径，这样既确定了它们之间存在的路由，又保证了双方都是活动的并可彼此响应的。本系统采用了流套接字作为网络编程的接口[1]。

多线程是为了使得多个线程并行的工作以完成多项任务，以提高系统的效率。线程是在同一时间需要完成多项任务的时候被实现的。多线程块模型（MTA）在每个进程里只有一个块而不是多个块。这单个块控制着多个线程而不是单个线程。所有的线程都是相同的块的一个部分，并且可以共享，比单线程模型执行速度都要快。

2.1　系统设计及建模

本系统的开放环境为 Eclipse 2.6 及 JDK 6.0，开发语言为 J2EE[2]。本系统分为服务器端和客户端，其中服务器端运行在岸基数据中心的服务器上，客户端运行在海南值班室的集成工作站上。系统的设计思想是：海南值班室内将各子系统生成的数据通过辅助传输程序以文件拷贝的方式发送到客户端。客户端采用文件

* **基金项目**：国家海洋公益性项目“边远岛的利用与监控技术研究与示范”（200805079）.

夹轮询的方式对各子系统拷贝到客户端的文件所在文件夹进行扫描，再根据服务器的 IP 地址和服务器设定的端口连接到岸基数据中心的服务器端，然后经过文件压缩、计算、打包等步骤将文件发送到服务器端。服务器端连续监听自己定义的端口，当接收到客户端发送的文件后，对文件进行解压、分发、计算等。客户端以及传输辅助程序都需要有配置文件，里面记录经常需要更改的参数，如传输文件的目的地址、接收文件路径、文件类型、文件备份路径等。客户端发送数据后，根据文件类型将文件备份，服务器端接收文件后将文件放置到设定的目录下，待集成展示系统和辅助决策系统使用。服务器和客户端应用程序开启时，都会在相应的路径建立一个 txt 文件作为日志文件记录接收或发送的数据文件的信息[3]。

端口是一种抽象的软件结构（包括一些数据结构和缓冲区）[4]。当某一进程通过系统调用和某一端口建立连接后，传输层传给该端口的数据都被该进程所接收，同样，进程发送给传输层的数据都通过该端口输出。每个端口都拥有一个端口号，端口号用于表示同一台计算机不同的进程（应用程序）。一般 1024 以下的为系统端口，如 HTTP 的端口号为 80，FTP 的端口号为 21 等。本系统中，服务器端预留端口 1111 和 1112，分别负责监听客户端信息以及接收客户端发送的数据。客户端预留端口 1111 和 1112 分别负责监听服务器端信息反馈和发送数据包。系统在客户端以及服务器端采用多线程技术。服务器端的主线程负责接收由客户端发送的数据并在日志中记录相关信息，辅助线程负责监听客户端连接请求并建立相应连接。图 1 为数据接收端的用例图。

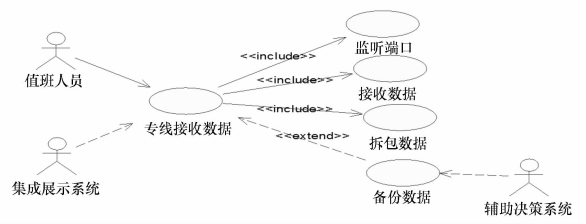

图 1 数据接收端的用例图

客户端的主线程负责发送数据并在日志文件中添加相应记录信息，辅助线程负责发起与服务器端的连接请求并建立与服务器的连接。图 2 为数据发送端的用例图。

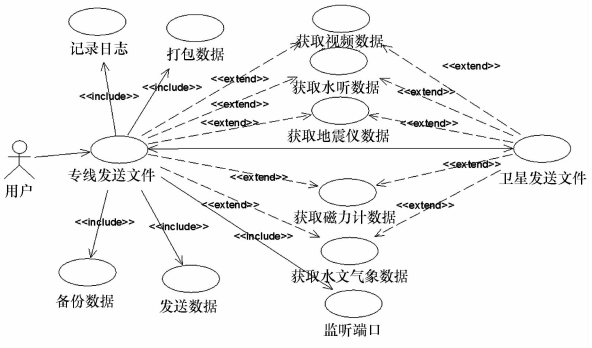

图 2 数据发送端的用例图

2.2 数据接收端程序设计

数据接收端程序主要功能函数如下：

(1) 监听端口函数：

```
Socket sock = new Socket ("100. 100. 100. 20", 1111); //监听 100. 100. 100. 20 的 1111 端口//情况
TCPListener tcpls = new TCPListener (127. 0. 0. 1, 1111) ; //利用本机的 1111 端口进行//监听
tcpls=sock. connect ();
```

(2) 接收文件数据：

```
//接收文件
File f=new File ();
f = sock. receive (buffer, totalrec, left, SocketFlag, None);
//写日志文件，记录数据块号
StreamWriter stw=null;
stw=new StreamWriter (m _ rec _ log _ dir, true);
stw. Writelog (tmplog);
stw. close ();
```

(3) 解压

```
ZipInputStream s;
s = new ZipInputStream (File. OpenrRead (FiletoUnZip));
int sz = s. Read (data, 0, data. length ());
StreamWriter. write (0, data, sz);
```

(4) 接收信息反馈

sock. send (FileNameandblocknum, Totalsend, left, SocketFlag. Non);

2.3 数据发送端程序设计

(1) 连接数据接收端

ServerSocket serv = new ServerSocket (1111); //启动本地 1111 端口建立 socket 连接

Socket sock=serv. accept (); //接受连接

(2) 数据压缩

ZipInputStream s;

s = new ZipOutputStream (File. Create (ZipfileName));

BufferedOutputStream buffer = new BufferedOutputStream (1024);

s. Write (buffer, 0, buffer. length);

(3) 数据发送

File f=new File (String path); //path 设定为传输文件的路径，文件夹

File ff [] =f. list ();

DataOutputStream dos = new DataOutputStream (sock. getOutputStream ());

dos. writeLong (ff [0] . byte [1024]); //设置文件输出流，设置每次传输数据的大小为 //1024

(4) 发送信息反馈

sock. receive (FilenameandBlockNum, Totalrecv, left, Socketflag. Non);

3 实际业务化测试与运行

3.1 系统配置与运行结果

专线和 Vsat 是数据传输的途径，对于实际业务化运行的系统而言可以认为其是透明的。在此我们以专线传输为基础对传输系统进行实验。根据边远海岛项目实际的应用需求我们在文件发送端进行系统配置，其配置文件为一个 Access 数据库，表 1 中录入的信息如表 1 所示。

表 1　数据传输系统配置信息

server _ ip	sourpath	destpath	localbackup	dirtype
100. 100. 100. 20	D: \ txxt \ XDT \ realtime \	d: /txxt/xdt/	e: /b/rt/	实时信息
100. 100. 100. 20	D: \ txxt \ XDT \ perclock \	d: /txxt/xdt/	e: /b/pl/	一般信息
100. 100. 100. 20	d: /a/vd/	d: /video/	e: /b/vd/	实时信息
100. 100. 100. 20	d: /a/cl/	e: /2suo/cl/	e: /b/cl/	一般信息
100. 100. 100. 20	d: /a/dz/	e: /2suo/dz/	e: /b/dz/	一般信息
100. 100. 100. 20	d: /a/st/	d: /txxt/stxt/	e: /b/st/	实时信息
100. 100. 100. 20	D: \ EDAS \ data \ XDT \ 001 \	d: /fio/hndata/	e: /b/lz/	一般信息

其中：第一列为传输接收端的 ip 地址；第二列为数据发送端所传输文件所在路径；第三列为数据接收端接受文件的路径；第四列为数据发送端的数据备份路径；第五列为信息存储类别，实时信息按照文件形成的时间存储到备份路径，一般信息统一存储在备份路径。

3.2 测试结果及其分析

对 2011 年 6 月 19 日至 12 月 16 日的数据进行传输实验，其运行结果如下所述：① 水文气象系统数据传输：实现实时传输，每个文件的平均传输时间为 0. 5 s，文件大小约 10 k。② 水听声呐数据传输：实现实时传输，每个文件的平均传输时间为 1 s，文件大小约 37 k。③ 视频数据传输：实现连续不间断传输，每个文件的平均传输时间为 8 s，文件大小从 1 k～7 M 不等，其中多数文件在 200 k 左右。④ 磁力计数据文件：实现连续不间断传输，每个文件输出时间为 4 s，文件大小为 50 k。⑤ 水下地震仪文件：实现连续传输，每个文件传输时间 15 s，文件大小为 6. 8 M。⑥ 陆源地震信息文件：实现连续传输，每个文件传输时间 9 s，文件大小 2. 1 M。

由于系统中的水文气象信息、水听声呐信息的生成周期为每分钟 1 个文件，磁力计、水下地震仪信息为每 20 分钟 1 个文件，陆源地震信息文件每小时 1 个文件，视频系统根据扫描范围内经过船只数量确定文件，且平均每小时 15 个文件，因此本系统基本实现边远岛文件传输需求。同时，利用多线程技术有效保障了不同信息的同步传输。

4 结论

本系统的发送端与接收端之间通过 Socket 进行连接，因此不需要考虑系统所处的运行环境。将多线程引入系统的传输设计中，提高了网络的数据传输效率。直接应用 J2EE 提供的类大大简化了程序设计与实现。

参考文献

[1] 汪晓平，俞俊，李功．精通 JAVA 网络编程[M]．北京：清华大学出版社，2005.

[2] 耿详义，张跃平．JAVA 2 实用教程(第二版)[M]．北京：清华大学出版社，2004.

[3] 徐小蕾，苏厚勤．分层系统 Socket 通信软件的设计与实现[J]．计算机应用与软件，2009，26(1)：173－175.

[4] 周炎涛，李立明．TCP/IP 协议下网络编程技术的实现[J]．航空计算技术，2002，32(3)：122－125.