© 2014 IJIRT | Volume 1 Issue 6 | ISSN : 2349-6002

Elements of Socket Programming in Java

Ranjeet Kumar, Vinay
Student of Information Technology
Dronacharya College of Engineering, Gurgaon

***Abstract*- Internet and WWW have emerged as global ubiquitous media for communication and changed the way we learn, live, enjoy, communicate, interact, engage, etc. This paper describes about network programming and socket elements in java. Network programming is similar to socket programming. It introduces the concepts involved in creating network applications using socket. Where socket programming is important to understand how to inter-process communication work. In this, we describe TCP/IP socket programming. There are many research papers available on the socket programming, but in this paper, we have discussed a socket program interface which will be useful in understanding the socket.**

***Index Terms*- Client-Server, Socket Elements, Socket Programming, TCP/IP, UDP**

I. INTRODUCTION

Network programming is a server side programming as well as client side. It is communicating inter-process between two or more computers. It implies that, it involves writing computer programs that communicate with other program across a computer network. Whenever we want to communicate with one computer to other computer over network then it is required to software which connects two ports in network layer that is called network programming. After connecting over network, its play a big role to send and receive data to millions of people over world. [1] Network programming uses Client-Server model, so network programming is also Client-Server Programming.

II. CLIENT-SERVER

Where one program start the communication called client process or program and other who is waiting for communication to start called the server process or program. In the simplest case, Client program sends the request to the server and Server sends that response, then Client obtains the data from the server and displays it. Network programming is also called socket programming. It uses socket for inter connecting two ports. Where socket act as the endpoint of inter-process. Socket is connected using internet protocol. [1] From last few years network programming has stopped being territorial unit of the specialist and became part of every developers work chest. [1] It was started to being used in all application. So there is need to make it simpler, So Java was started to being used for network programming because of the advantages Java provided. Java contains classes and objects that help socket program to communicate with certain types of servers and process different types of data but not all the servers and data. [1] So Java allows you to write protocol handlers to communicate with different server and process the data.

There are two network architectures widely used today:

- Peer-to-peer and
- Client/server

In peer-to-peer networks each workstation has the same capabilities and responsibilities. These networks are usually less expensive and simpler to design than client/server networks, but they do not offer the same performance with heavy traffic.

Peer to peer networks share responsibility for processing data among all of the connected devices. Peer-to-peer networking (also known simply as peer networking) differs from client-server networking in several respects.

The term ***client/server*** refers to a model utilizing networked client and server computers and application software. Web, FTP, email, DNS and many other database applications are client-server systems.

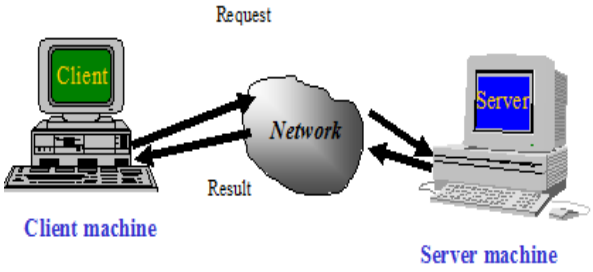

Figure1. Client-server block diagram

III. API

The application interface is the interface available to the programmer for using the communication protocols. [2] The API is depends to the OS the programming language.

We discuss the socket API. With sockets, the network connection can be used as a file. Network I/O is, however, more complicated than file I/O because:

- Asymmetric. The connection requires the program to know which process it is, the client or the server.
- A network connection that is connection-oriented is somewhat like opening a file. A connectionless protocol doesn't have anything like an open.
- A network application needs additional information to maintain protections, for example, of the other process.
- There are more parameters required to specify network connection than the file Input/output. [2]

IV. TRANSMISSION CONTROL PROTOCOL

Transmission Control Protocol (TCP/IP) is a connection-oriented protocol. [3] In 1982, the TCP/IP Internet included a few hundred computers at two dozen sites concentrated primarily in North America. [3] In 1996, over 8,000,000 computer systems attach to the Internet in over 85 countries spread across 7 continents; its size continues to double every ten months. [3] World Wide Web has replaced file transfer as the most popular global service; Uniform Resource Locators used with Web browsers appear on billboards and television shows.

Every computer on the Internet is identified by a unique, 4-byte IP address. This is typically written in dotted quad format like 174.210.25.73 where each byte is an unsigned value between 0 and 255. This representation is clearly not user-friendly because it does not tell us anything about the content and then it is difficult to remember. Hence, IP addresses are mapped to names like www.yahoo.com or www.google.com, which are easier to remember. Internet supports name servers that translate these names to IP addresses.

V. TCP/IP SOCKET PROGRAMMING

A socket is an endpoint of a two-way communication link between two programs running on the network. A server program creates a specific type of socket that is used to listen for client requests (server socket), In the case of a connection request, the program creates a new socket through which it will exchange data with the client using input and output streams. The socket abstraction is very similar to the file concept: we have to open a socket, perform I/O, and close it. Figure 2 illustrates key steps involved in creating socket-based server and client functions of the programs. Following figure will shows the functional structure of TCP/IP protocol in socket programming. Figure2 shows how socket works with the protocols.

The streams used in file I/O operation are also applicable to socket-based I/O. Socket-based communication is independent of a programming language used for implementing it. That means, a socket program written in Java language can communicate to a program written in non-Java (say C or C++) socket program.

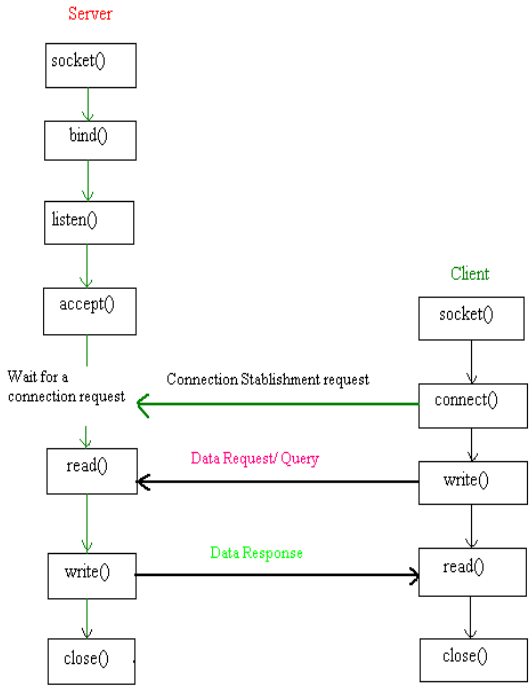

Figure2. Block diagram of TCP/IP Socket Programming [2].

1. Socket () function creates a socket on client as well as server side. It is initial require for socket programming. It initialized the type of communication stream, family of protocol and which Protocol used. Opening of socket:
   ServerSocket server = new ServerSocket(PORT No);
2. Bind () assigns the local host protocol address to the socket.
3. Listen () function is used only in TCP/IP protocol to connection establishment on server side. It is used to convert active socket into passive socket. [2] Specifies the maximum number of connections (backlog) that the kernel should queue for this socket.
4. Accept () function is called by the TCP/IP server. It accepts the request, when a port is ready to make a connection. Waiting for client Request:
   Socket client = server.accept();
5. Connect () function is used on client side. It is used to establish connection to the server. Creating I/O stream for communicate to client:
   DataInputStream is = new DataInputStream(client.getInputStream());
   DataOutputStream os = new DataOutputStream(client.getOutputStream ());
6. Read () function is used to both side. It receives data line by line from server to client and vice-versa. Reading data from client:
   Receive from client: String line = is.readLine();
7. Write () function works in same way as the read. It writes something query or response respectively client and server. Writing data to client:
   Send to client: os.writeBytes("Hello\n");
8. Close () function is used to terminate the connection on both side. Close the connection:
   client.close();

VI. CONCLUSION

In this paper, we have concluded that the socket programming generally holds the communication between the client and the server. Also discussed different client-server architecture using TCP/IP can be interfaced through the communication network. In TCP socket calls, the client sends the request to the server and the server performs all the functions i.e. socket (), bind (), listen () and accept (). One server can handled multiple clients at the same time, where the IP address and port no of server should fed in client side programming. The Internet Protocol breaks all data into packets and adds header to packets which contain source and destination address of the host. It is individually routed from source to destination. This paper is a review of the function of socket

programming. There are many areas where we can research in the Java socket programming.

REFERENCES

[1]. Network programming in Java using Socket by Abhijit A. Sawant, Dr. B. B. Meshram / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com Vol. 3, Issue 1, January -February 2013, pp

[2]. Sockets and Socket Address Structure by Mohit Mittal and Tarun Bhalla: International Journal of Engineering Research & Technology (IJERT) Vol. 1 Issue 8, October - 2012 ISSN: 2278-0181

[3]. Internetworking with TCP/IP Vol III: Client-Server Programming and Applications
BSD Socket Version Second Edition by DOUGLAS E. COMER and DAVID L. STEVENS
Department of Computer Sciences Purdue University West Lafayette, IN 47907