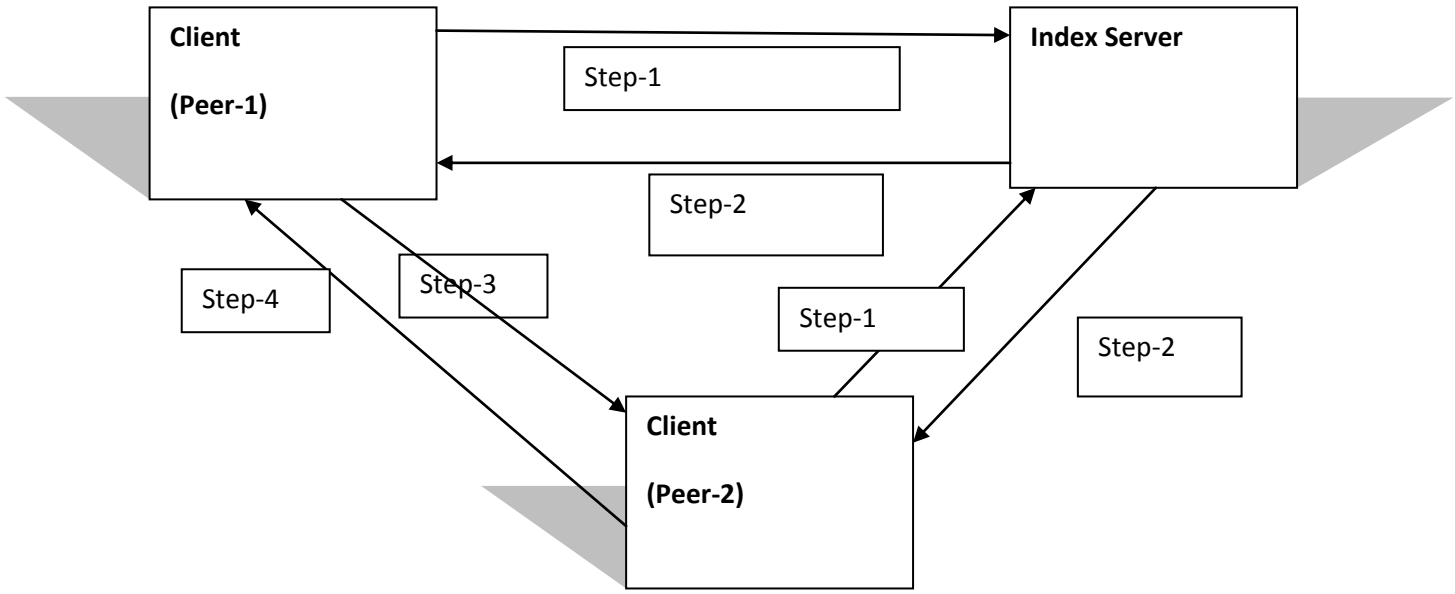
Introduction

Napster Peer-to-Peer File Sharing: The Napster Peer-to-Peer File sharing consist of an Indexing Server and Peer.

Indexing Server: This server indexes the contents of all of the peers that register with it. It also provides search facility to peers. In this code the Indexing Server (IndexServer.java) accepts the Array List by using object Streams from the client and store the Array List in the form of Hash Maps. The Server stores the list in the form of (Id,ArrayList).Multiple clients can be connected to the server.

Peer (Client) Sends all the files present in it's specific folder in the form of an ArrayList to the indexing server

Coding Platform:Java
Protocol: Blocking and Socket Programming.



Step 1:Client connects to the server and sends a list of files in a specific folder to Index_Server.The Index_Server depending on this list creates a HaspMap(key(filename),Value(PeerName)).The client then sends a filename to server which it wants to receive from anyother people

Step 2:The Index_Server depending on the filename makes a lookup and sends Peername of all those Peer having that specific file.

Step 3:Then the Peer-1 make a connection request to the peer from where it has to download the file.

Step 4:Then the file transfer takes place after which the connection between the peers is broke.But connection with Index_Server remains.

Peer : The Peer acts as both as a Client and a Server. When it acts as a Client it connects to the indexing server and sends it an Array List of files present in its specific folder to the Indexing Server. At the same time the Peer also acts as a server by continuously listening to other connections that request a download of a file from it.

Working: The Peer connects to the Indexing Server it sends a file list the server stores the list in a Hash Map. When any other Peer request for a specific file it will search for the file in the Hash table and on success return a list of Peer which have the file. Then the Peer which has requested the file will download the specific file from one of Peers in the list.

The Code in here has 2 basic file Index_Server.java and Client.java.

Indexing Server

The Peers connect to the Indexing Server. Multiple Peers can connect to the Indexing Server using Sockets.

Task: Store the File list of in the form of Hash Map and on request from the Peer(Client) sends a list of Peers which have the file.

Server HashMap: The Server HashMap consist of 2 HashMaps:

1.HashTable(FileName(key),PeerName(Value))

2. HashTable_IP(PeerName(Key),PeerIP(Value))

The Index_Server lookups in the HashTable and finds the peername associated with the specific file.It then using this PeerName lookups in the HashTable_IP Table where it finds the IP and returns the Client with the specific ArrayList.

Architecture: The Server is a multi threaded Indexing Server. Hence multiple client connections are possible.The Server follows a blocking Protocol.

- The Server Consist of a Main Class Index_Server.
- The Run method is called whenever a Client connects to the server, assigning a thread to every client.
- The Server accepts the incoming Objects of Array list from the client using objectStream.
- The Server before communicates with the client using the Input and Output stream methods of socket.
- The Connection between Client and Server is been made using Sockets.
- The GetFileList method( ) is used to store the Array list of file provided by the Peer(Client)in the Hash map table of the client.
- The Search_Peers_WhereFileExsist( ) method is used to find the peer in the hash map table where a particular file exsist.

Working of Server Code:

- Whenever a Peer(Client) connects to the server a thread is intilized associated to the client.The client sends it file list to the server so the server can store and update the entries to the specific client in the hash map table.
- Whenever the Peer(Client) sends a filename the server searches the hasp map and returns the list of clients to the Peer which have that file.The server fist finds the peer name from the hashmap table and looks to find the relevant IP address from the Hash_IP table where in it has stored the Name,IP pair.

HASH_MAP:FileName,PeerName Pair

HASH_IP: PeerName,IP Pair

Peer

The Peer functions as both Client and Server(for other Peers).

There are 2 roles played by Peer 1)Client 2)Server(For File Download)

Peer As Client:

The Peer when as a Client connects to the Index Server and sends the file list Array to the server.

Peer As a Server:

The Peer works as a server for other Peers to connect to it and download the file from it.

Working and Architecture of Code:

- The communication between Peer and Index Server and Peer-Peer communication is done using sockets.
- The Client Connects to the Index Server. Once the Peer is connected to the Index server it opens another connection where it is in a continuous listening mode to accept connection from other peers and allow them to download the requested file.
- The Peer(Client) sends its file list of those files added or deleted form its folder to the Index Server on making a server connection and receiving a file.
- Whenever the Peer(Client) wants a list from Index Server of all the associated peers having a specific file it sends the request to the Index Server where the index server returns an array list of associated Peers.The list send by Index Server is of IPs of the specific peers having the file.
- The PeerProcessing( ) method used in the code accepts the array list sent by Index Server.

- The ConnecttoDownloadFile( ) method is used to connect to other Peers and download the file from them. This method will NOT connect to the Index Server.
- The Server_Connection class is used to create a Peer(Server for file download) which can accept multiple request from other peers and act as a server for other peers which request the particular peer to download a file.
- The file Downloading is done using Buffered Streams of Sockets.

File Types: .txt,.docx,.bin,.properties,.dat

SUPPORT FOR BIN FILES PROVIDED.

Conclusion and Enhancements

The Peer-to-Peer Napster like system is made using the concepts of Distributed OS.Here the Index_Server just has a registry where it maps the files with the location of the Peers and the Peers transfer the files among themselves on request.This system is useful whenever a file transfer is required between 2 distant machines without server requiring any over head of transfer of files.

Future Enhancements in Code:

- To improve System performance a proper threading algorithm is required on client and server side.
- The Index_Server must have multiple hash Maps to store the value so as to avoid bottle neck problems when the data and request are high.
- Support for .mp3 and .dll files would also be needed.
- The Index_Server code Architecture must change and it can have an robust code architecture like MVC or Layered architecture.
- Server Replication would be the most important enhancement of all.