**Programming Assignment 3**

**Peer to Peer File Transfer Application**

**Course: CSE-5344-003**

**Team Members:**

**Nandan Thareja(1001114230)**

**Aditya Kadam(1001096909)**

**Chetan Khullar(1001111077)**

**Sanket Bhoirkar(1001107629)**

1. **Objective**

The venture objective is to create an ad-hoc Peer to peer network for file transfer. P2P networks are overlay networks, (which is built in top of other networks) are end systems in the Internet that maintain information about a set of other nodes (called neighbors) in the P2P layer. P2P participants join or leave the P2P system frequently; hence, P2P networks are dynamic in nature.

1. **Introduction**

Peer-To-Peer communication mechanisms are used in order to get away from the drawbacks a client/server model suffers from. The following points are a few of the drawbacks that the classic client/server model suffers from:

• It has a single point of failure. Essentially this means that if the server goes down the service is not able to function properly if it is even able to function at all.

• The server needs to serve a possibly high amount of work because all requests in the system are directed to the server, so the load of the server is high.

Peer-to-Peer communication is interesting and potentially effective mainly because it takes away the single point of failure drawback that the client/server application suffers from and it gives an application a distributed communication instead. This enables peers to communicate directly with each other instead of having a server in the middle which all communication passes through.

This report includes a detailed description of the peer-to-peer architecture used in the project to build a p2p file transfer application.

1. **Tools/Technology/Platform/Libraries Requirements.**

JDK 1.8.0\_45

Netbeans IDE 8.0.2

mysql-connector-java-5.1.18.jar

1. **Implementation**

The code is implemented keeping in mind the ring architecture. As the execution starts, first we need to start the execution of the server. This server is used to authenticate the client and it gives the IP address of the first connected client. Now, as new clients are connected they send their authentication request to the main server. When new client evokes it sends a socket connection request to the main server, which in the response of this request sends a reply that the connection is open and also sends the IP address of the last connected client to the new client. Now this newly connected client replaces the previous IP address on the main server with the IP address of the newly connected client. This is done so that every new incoming client will get address of the lastly connected client in the link.

Now each client will have two IP addresses, one is the IP address of the client who joined in the network previously to that client which will become the local server to that client and the other will be the main server which is used for authentication. The new client sends a ‘join’ message in the network to other client which is ahead of it in the ring, which passes this message similarly till the first client who joined. Now the first client finds that the IP address currently it is connected to not the last client in the ring, and that a new client has joined the network so it updates the IP address in its record and in this way the ring is completed when a new user connects to the network.

As a new client is authenticated it is redirected to the file share window, where the client can see the available list of clients and the files shared by them. The client has access to four features, Firstly, to select a file from the available files list and click on *fetch file* button to download the file. Secondly, to add a file to the network on clicking the *Add file* button. Thirdly, refreshing the list of available files by clicking on the refresh button. Fourthly, clicking on the leave network button to leave the network.

When a user joins or leaves the network each peer is notified with a message, and if a peer leaves a network all the files uploaded by that peer gets released from the network and the list of files for all the connected peers gets updated.

1. **GUI at the client side.**

**Join Network Screen (JoinNetworkScreen.java)**

* Username Field: for entering the peer name to join the network.
* Join button: for joining the peer network.

**File Share and peer window (FileShare\_Peer\_Window.java)**

* Connected Peers List: shows the list of active peers on the network.
* Notification box: updated with messages when a user joins or leaves the network.
* Available File list: shows the list of available files which are uploaded by all the active peers in the network.
* Fetch File button: Download the selected file in the list.
* Add file button: to upload a file to the network.
* Refresh List button: to refresh the list of available files uploaded by all other peers in the network.
* Leave Network button: to leave the network and release all the files uploaded from the server.

1. **Challenges**

The team faced challenges in establishing communication between two clients without using an intermediate server.

1. **Limitations:**

The application is consuming unusual amount of time for downloading even small files. The application needs to optimize in terms of performance.

1. **References**

* <http://www.aisectuniversity.ac.in/Anusandhan3/16.pdf>
* <http://www2.cs.uregina.ca/~hamilton/courses/330/notes/distributed/distributed.html>
* <http://www.diva-portal.org/smash/get/diva2:524732/FULLTEXT01.pdf>
* <http://www.codeproject.com/Articles/1297/Peer-to-Peer-Communicator-and-File-Transfer>
* <http://listoffreeware.com/list-of-best-free-p2p-chat-software/>
* http://www.codeproject.com/Articles/17321/A-simple-peer-to-peer-chat-application-using-WCF-n