**Peer to Peer file sharing system**

**-SAGAR S PATIL**

**INTRODUCTION:**

Peer to Peer technology is the one which is used for sharing the data between two computers or more computers without the need of server. In this kind of network one of the peer acts as server and other peer acts a client.

In this Project I have made use of Java Remote Method Invocation (RMI) which is a built-in framework. The project mainly consists of the following ideas:

1). Registering the Peers to the Server (Central Indexing Server) and registering the files.

2). Searching for a particular file in the network and then selecting one of the peers from the list of peers having that file.

3). Once the peer is selected then we download the file from the peer and then registering the peer who has downloaded the file to the Server (Central Indexing Server).

4). Deregistering the file from the Server whenever the user deletes a file from the directory.

5). Concurrently running multiple peers which requests the server for the list of peers and download the file from the same peer concurrently.

Diagram:

PEER2  
dwnldFrmPeer();

PEER1  
dwnldFrmPeer();

Central Indexing Server (Server)

Register();  
Search();  
Deregister();

PEER3  
dwnldFrmPeer();

There are three entities when running a program using RMI.

1). Client: We use this class to access remote methods on a server.

2). Object Registry: To Create a registry and here server first creates a registry on some port.

3). Server: Once the Server creates the registry on some port. Client first contact registry and then using the details will connect to the Server.

**Implementation has following steps:**

1. **Defining a Remote Interface:**

Here the ServerInterface extends remote, and I have declared three remote methods i.e. Register (), Search() and Deregister() and all these methods throws remote exception.

1. **Implementation of the Server:**

Here the ServerImpl has the implementation of the methods declared in the ServerInterface. Here I have declared a global array list and then updating it as and when required.

Register(All details)- Register method takes all the input i.e. PeerID, Port No, File name, directory of the Peer from the user and adding the record to the globally declared ArrayList.

Search(File Name)- This method declares a new local ArrayList, and it compares File Name given by the user to the list of filenames registered(Present in the global arraylist) and matched records are pushed to the local arraylist and returned to the user.

Deregister(All details)-This method takes all the input i.e. PeerID, Port No, File name, directory of the Peer from the user and it deletes the record of file name of particular peer only.

1. **Implementation of Peers:**

Here I have implemented a Peer in Client.java and I have implemented the dwnldFrmPeer() inside this class itself as I have only one method. I am taking the portnumber and directory from the user and spinning up new thread for each client/peer. Here the File Input Stream reads the file content from the file present in the source directory and File Output Stream writes the file content to the file present in the Destination directory. In addition, I have made use of watchservice which keeps on polling the directory and whenever a file is deleted from Peer’s directory then that file is automatically deregistered from the Server.

**4). Measure the response times when multiple Peer clients are concurrently searching for the list of peers having the file from the server and downloading file from the same Peer Server.**

Here I have computed the total time taken (in milliseconds) to search and download file from the Peer Server for different number of peers i.e. 1 PEER, 2 PEERS and 3 PEERS. I have implemented this in Concurrent.java

**Tradeoffs:**

We can use Data Structures like Hash Map instead of using Array List for Indexing.

**Possible Improvements:**

We can move the function “dwnldFrmPeer()” function to ClientImpl.java and move the WatchService code to ClientImpl.java so that it monitors the directory from very first when the PEER gets registered.