DARE IT: File Sharing System

Rohit negi

Department of Computer Science and Engineering

School of Engineering and Technology

Sharda University, Gr. Noida, UP, India

rohit25.negi@gmail.com

Abstract

This is the era of networking and machine are capable of communicating with each other. In this time handling and working in your own machine is not enough. The need here is to connect two machines and to establish communication between them efficiently. This paper presents a software which allows two machines to connect and share information efficiently. The software also facilitates the user with the features like sharing files, navigating remote machine, all sorts of modification on files present in remote machine.

1. Introduction

Remote desktop is an application in which processing can be done on different machine while the application is actually running on a local machine. This type of application is usually used in distributed system. It is also used by company service providers for the purpose of troubleshooting the problems of their customers. It works on the concept of the sockets which is the combination of IP address and the port number of a machine on the internet or on a local network. There are many applications which are based on this concept like Team Viewer, Share it etc. This paper presents a similar kind of desktop based application (developed using java RMI) which allows to connect two computers and share files with a very high speed.

2. Features Provided

2.1 Remote Access

Remote access is the feature of a software or of an operating system which allows the user to access a remote machine without actually accessing it. It works on some networking protocols. The Software presented in the paper allows the user to be an active user or a passive user. Active user is the user who gets the power of controlling the remote machine and the passive user is the user who does all the processing on behalf of active user.

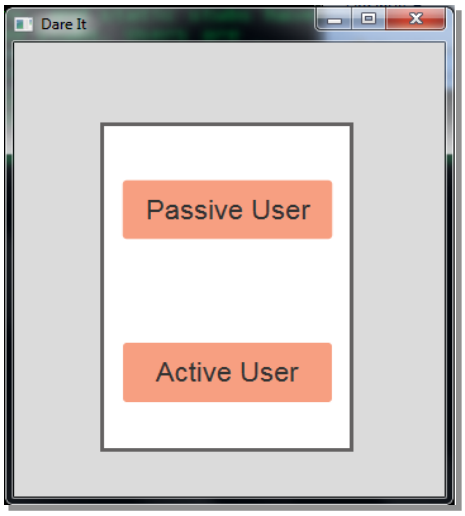

Fig 2: Dare It

2.2 Remote Navigation

This is a very powerful function that this software offers. This feature allows the user to remotely navigate other machine without getting physical access to it. This Feature can be very dangerous because it can disclose ones entire confidential information.

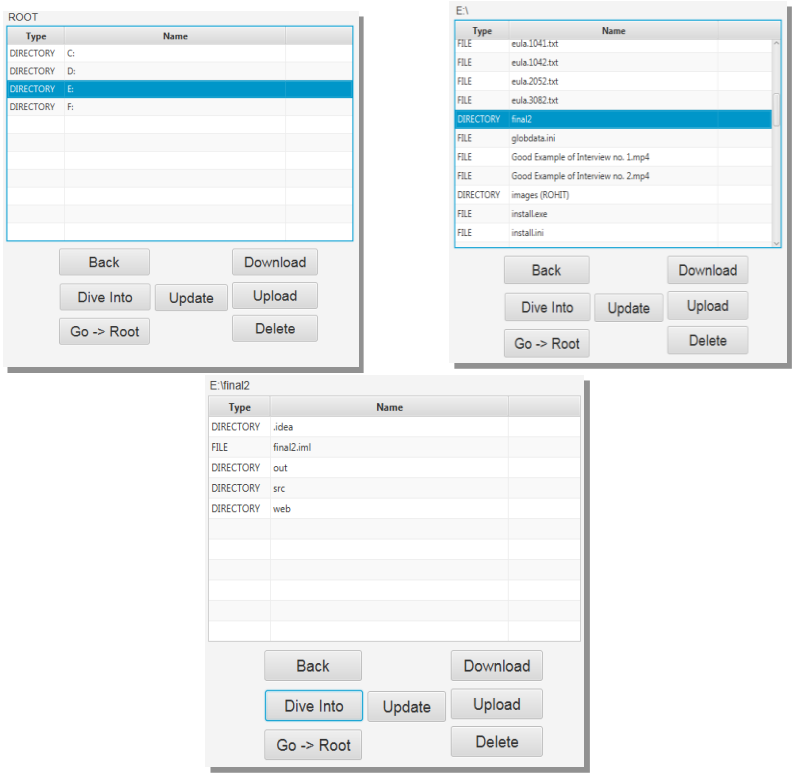

Fig 3: Remote Navigation

2.3 Downloading files from remote machine

This feature is used to download any kind of file from remote machine to local machine. The Download is performed at a speed of 1.7 MB/s on an average. The file which is to be downloaded can be selected through the navigation and the destination directory in the local machine can also be selected using file dialog.

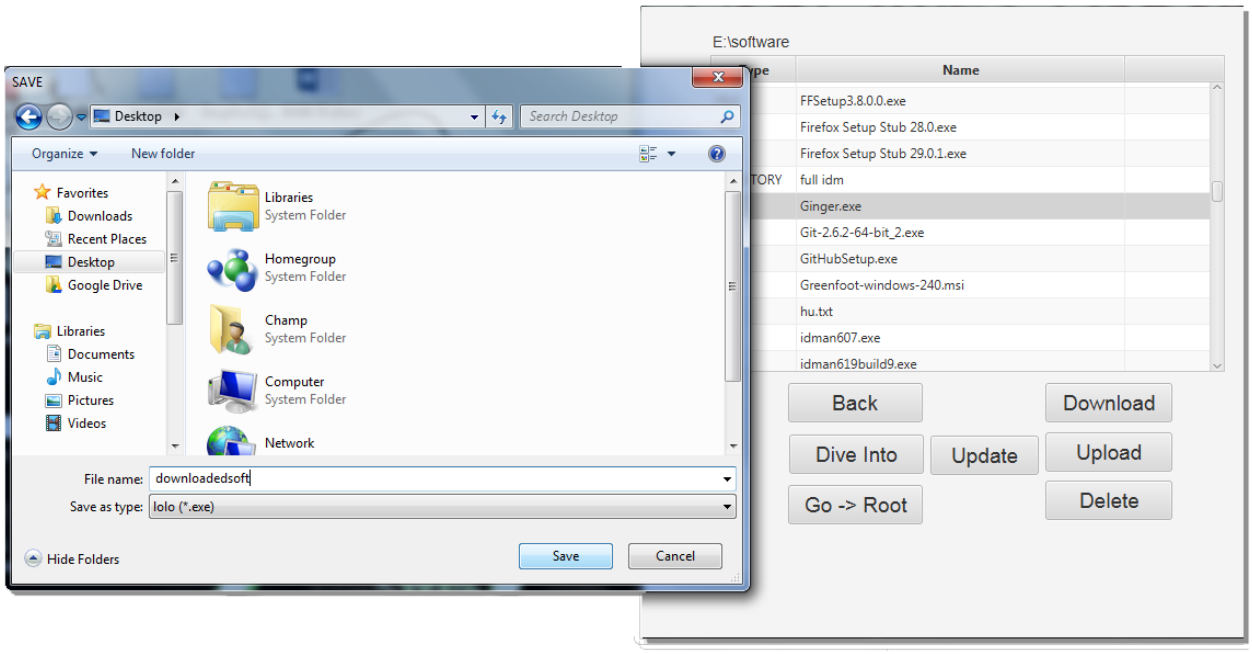

Fig 4: downloading remote file

2.4 Uploading file to remote machine

This is the opposite feature of Downloading. By using this feature you can upload any type of file the remote machine. This feature also works same as the download feature. But in the case the file in transferred from the local machine to the remote machine.

2.5 Remote Deletion

This feature is kind of dangerous because it gives users the power of deleting any file from the remote machine without asking the owner of the remote machine. The files deleted from this feature are permanently deleted from the remote machine and not back upped in the recycle bin.

2.6 Remote modification

This feature allows you to upload any file present in the remote machine. This feature can be used in many productive uses like one can help other in writing content, making codes,

helping in making project, and Remote analysis of the documents etc. Without actually accessing the remote machine. The changes are made on the local machine and the changes are reflected in the remote machine. These changes are post reflected. Local user first modify the file as per the requirement and the changes can be applied giving a final confirmation.

3. TECHNOLOGIES USED

3.1 Java RMI

The base Technology of the software is java RMI (Remote Method Invocation). Using this technology One can execute the methods present on another machine without actually accessing it. All the processing is done on remote machine and if the method is returning any value this value is transferred through the channel and received on the local machine.

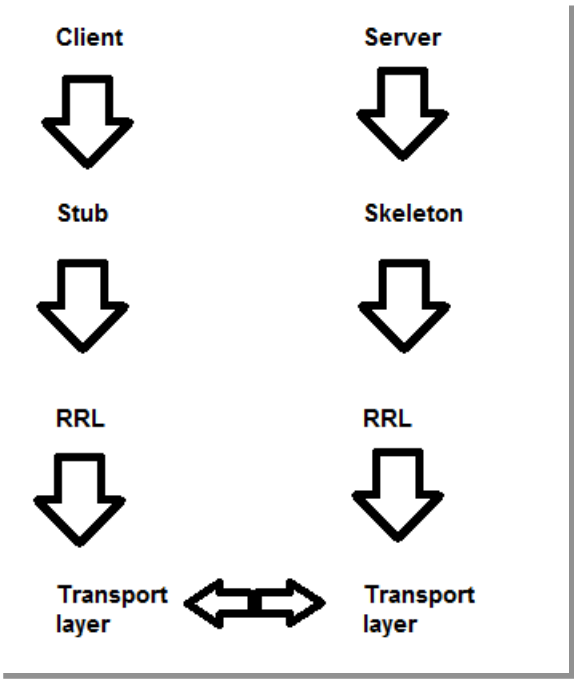

FIG 4: RMI Communication

RMI runs on Transport layer of OSI Architecture because transport layer is responsible for process to process communication. It Function on the TCP/IP protocol. There is a client which sends a request to the server side to execute a method which is the method of an object which resides on the server side.

On the Client side to access the remote object Client uses the "Stub". Stub is the proxy for the remote object. Stub is the representation of the remote object which resides on the server side. This stub object is bind with an address through which it is accessed remotely. Using this Stub, client sends the request to the server, this message is sent to the RRL (Remote reference layer) on client side which check the type of the server that is whether it is a unicast server or multicast server. RMI should be Unicast. Then this message is sent to the server side through the transport layer.

Methods which are to be invoked remotely may also take the arguments. They may take primitive type arguments or non-primitive type arguments. When the methods of remote object takes the arguments then these arguments are also transmitted to the server side with the request message. The scenario is different if the method is taking some non-primitive type parameters. Non-primitive type parameters cannot be directly transmitted because the object reference variables which are passed as arguments just represent the address of the object in the memory and that address represent something very different thing on a different machine. Hence the Object is first Serialized and converted into a byte stream and then transferred. This process is called marshaling.

On The server side the Skeleton is responsible for invoking the requested methods in the server. If the method is taking the arguments then it check whether the argument is primitive type or non-primitive type. If the argument is of primitive type it simply pass it to the method and if the argument is of non-primitive type then it performs the Deserialization and extract the object again and then pass the object to the method. The result of the method is passed to the client side in similar manner as the arguments are passed from client side to server side.

Steps to implement the java RMI

1. Make an Interface 'A' which extends the 'Remote' Interface and declare all the methods which can be invoked remotely. 'Remote' Interface is a marker interface. It does not have any methods or variables. It tells that methods in its sub interface can be invoked remotely.
2. Make a subclass 'B' of 'UnicastRemoteObject' class and implements the interface 'A'. Override all the methods of this interface.
3. Get the object of 'B' class.
4. Bind this Object with an URL using bind method of Naming class. This URL is used by the client to connect with the server.
5. Now get the 'Stub' using 'rmic' command. This command will create a stub representation of the remote object.
6. Start the listening mode on port specified in above URL.
7. Run the server.
8. On the client side connect with the server using the specified URL using lookup method of Naming class.
9. Run the method.

3.2 Java File System

Java provides a rich library to handle the file system in an operating system. It provides many features like handling file and its content, retrieving the metadata associated with file etc.

The Class which are used in the project are:

- File Class: This provides a large number of methods to retrieve metadata associated with the files. It also allow the programmer to perform the operation related to the file itself not with the content of the file.
- FIleInputStream class: This class represent the Bytestream. This is the Subclass of InputStream class which is used for taking input from file i.e reading the content of the file. Object of File class is bind with the Object of FIleInputStream class for robust performance.

- FileOutputStream class: This class is the Subclass of the OutputStream class which is used to send the data to a File. It can be used to modify the content of the file or to create a new file depending on the requirement.

3.3 Connection with Selected machine in the Network

Java provides InetAddress class which has a long list of method that can be used for networking purposes. It contains a method 'getLocalHost()' which is used to retrieve the information including the Ip address of the local machine. Second method of this class used in this project is 'getByName(String name)' which retrieve the information of a machine with the specified name in the network. You can set the timeout to check whether the machine with the specified name id alive or not and Once you get the IP address of that machine you can connect using the IP address. That IP address is a part of URL which is passed in the lookup method of Naming class.

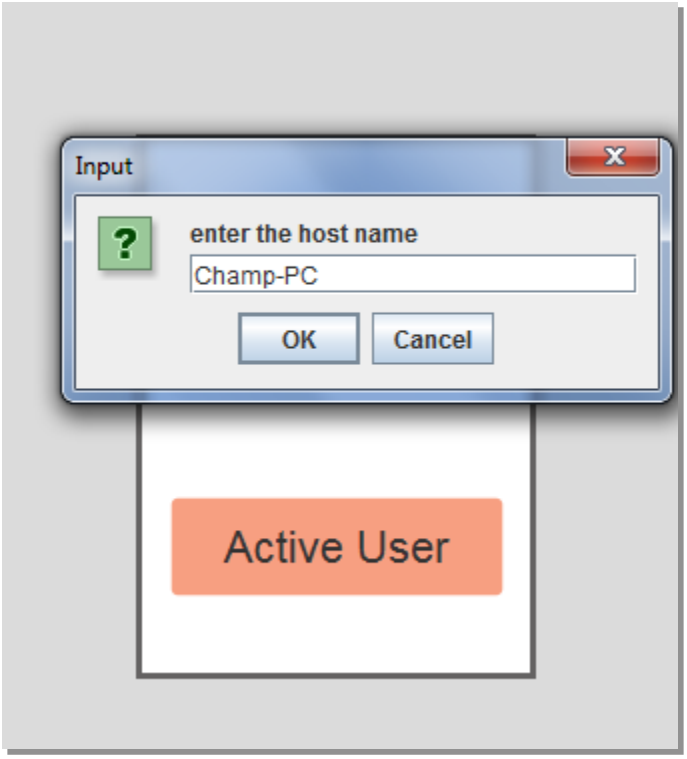

Fig 5: Connecting to Remote machine by writing name

4. Scope of the Project

The Project and the Features provided by the software can be used to share files between computers. Students can use this software to take help from other students as this software provide the feature through which they can update the files present on remote machine. The project can also be extended to Desktop sharing system where desktop activity of remote machine is shown on the local machine.

5. Issues to be considered with the project

Since the software is capable of accessing the remote machine without getting the physical access and the information on the remote could be very confidential and private so proper security measure must be applied with the software so that the user of the remote machine have the full information what the other user is doing with the data of his machine.

6. Conclusion

The software presented in this paper is successfully tested and the features provided by the software worked perfectly fine. Although the software is capable of sharing the file information at a very high speed but still the performance of the software can also be enhanced since the Java RMI which is the base of the software works on the TCP/IP protocol and the poor performance of TCP/IP and JAVA RMI protocol reduces the efficiency of the software. So to improve the quality of performance the underlining protocols must be modified so that good performance can also be achieved in bad network condition.

7. References

1. Stefano Campadello, Oskari Koskimies and Heikki Helin, Wireless Java RMI, online: https://www.cs.helsinki.fi/research/monads/papers/edoc2000/edoc2000.pdf

2. Java RMI from Java Specification
   http://docs.oracle.com/javase/6/docs/technotes/guides/rmi/hello/hello-world.html

3. Java FX from java Specification
   http://docs.oracle.com/javase/8/javase-clienttechnologies.htm