TRIBHUVAN UNIVERSITY

Institute of Engineering

Pulchowk Campus

Department of Electronics and Computer Engineering

A Minor Project

On

# “**Remote PC Access [RPA]**”

Submitted By:

|  |  |  |
|---|---|---|
| Bishal Shrestha (27110) | e-mail:bishalbs@gmail.com | ph: 9841674728 |
| Laxman Kasula (27120) | e-mail:laxkas@gmail.com | ph: 9841620241 |

Submitted to:

Department of Electronics and Computer Engineering

Pulchowk Campus

02/03/2008

## Acknowledgement

For the completion of this project, many people have helped us by giving their valuable suggestions and useful ideas. We would like to extend our sincere appreciation and gratitude to our project supervisors Asst. Professor **Mr. Jaya Ram Timsina**, **Mr. Bikash Shrestha** and **Mr. Deepen Chapagain** for their valuable suggestions and ideas. We would also like to thank all our friends who have helped directly or indirectly in this project.

Any comments and helpful suggestions regarding our project are most welcomed.

Thank you!!!

**Bishal Shrestha**

**Laxman Kasula**

## Abstract

Remote PC Access [RPA] is a computer program that lets you access your PC from another PC via the Internet or LAN and work on your computer remotely.

Through the use of RPA, it is possible to access the remote computer from any computer with internet connection. The connection with the server is made by specifying its IP-address and port number**.** To use RPA you must have a network TCP/IP connection, a server (running on a remote computer) and a client to connect to the server.

RPA is ideal for accessing and performing operations on the remote file system. It can also be used for remote system administration, transferring files and monitoring the remote computer.

# TABLE OF CONTENTS

## Introduction

Let’s assume a scenario that a user is working in a project in his office and he is also working in the same project in his home. Suppose he wants the files of project in his office from his home computer then he needs to take help from someone in his office to browse the local drives and upload the files to some file sharing websites or servers, so that he can download them from there.

But there is also another good solution to directly access the file systems of his office computer using the Remote PC Access [RPA].

Next, suppose we reached to campuses or offices with pen-drives or CD and later we discovered that we forgot to include important files, then in such situation also, RPA can be helpful.

Similarly a person travelling to distant location needs a photo or a file from his home computer or he needs to defragment or format his computer, then he can simply go to a cyber café or any other computer with internet connection and access his home computer from there.

Remote PC Access [RPA] is a computer program that lets you to access your PC from another PC via the Internet or LAN and work on your computer remotely.

The user in client side can browse remote local drives graphically; perform common operations such as copying, deleting, making new folder. The files can be uploaded to or downloaded from server. The user can also start or stop the applications, give system command to command prompt of server through ‘Remote Command Prompt’. It can also be used to log off, restart and shutdown the computer over a local network or the Internet. The client can also view the remote screen by capturing the screen.

Another important feature of RPA is that the client side program can be directly run from the web page using Java Web Start technology. So he doesn’t need the program to be installed or present in his computer. All he need is a computer with internet connection so that he can access his computer from anywhere.

The Remote computer can be protected by a username and a password. Each client is provided a unique username and password as well as different permissions of access.

## Objectives

The proposed Remote PC Access [RPA] included the following objectives regarding the Access of Remote PC:-

- To browse the local drives.
- To edit, copy, rename, delete the files.
- To upload and download the files.
- To access the command prompt to give the system commands.
- To start and stop the services.
- To shut down the computer.

## Platform and Tools

JAVA Standard Edition [JDK 1.6.0] is used for the development of RPA.

Tools used are:-

- NetBeans 5.5
- Apache/2.0.59
- Mysql 5.0.27-community-nt
- mysql-connector-java-5.0.6

## Literature Review

*1. Java Web Start Technology*

➔Java Web Start provides the power to launch full-featured applications with a single click. Users can download and launch applications, without going through complicated installation procedures.

Java Web Start technology allows the users to launch Java applications by clicking a link in a web page. The link points to a JNLP (Java Network Launching Protocol), which instructs the Java Web Start to download, cache and run the applications. JNLP file is the XML file that contains elements and attributes such as title, main class, name of jar file etc. which tells Web Start how to run the applications. To deploy an application with Java Web Start, the application should be packed as JAR files which should be kept in the Web Server. Applications launched with Java Web Start are, by default, run in sandbox model so the program cannot access the local files and networks. For applications to work outside the sandbox, the Jar files have to be digitally signed.

Java Web Start verifies that the contents of the JAR file have not changed since it was signed. If verification of a digital signature fails, Java Web Start does not run the application. When the user first runs an application as a signed JAR file, Java Web Start opens a dialog box displaying the application's origin based on the signer's certificate. The user can then make an informed decision regarding running the application.

*2. Remote Method Invocation [RMI]*

➔ RMI allows an object running in one Java Virtual Machine to invoke methods on an object running in another Java Virtual Machine.

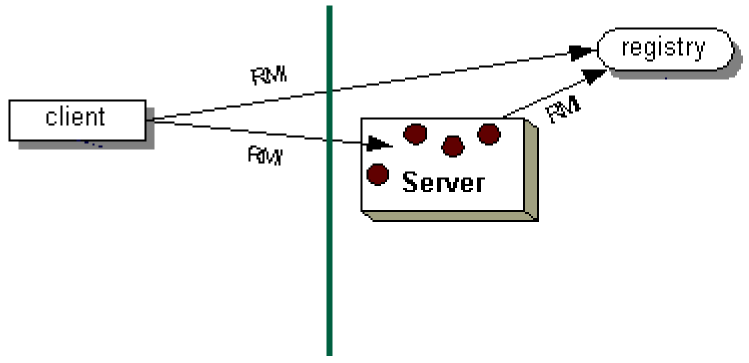


*Fig: Block diagram of RMI implementation*

In RMI application, Remote objects are created by implementing a remote interface and it has various methods that can be called across client module to perform operations and get information about the server side.

The Server calls the registry to associate or bind a name with a remote object. RMI passes a remote stub for a remote object which acts as the local representative for the remote object and is the remote reference to the caller. The client looks up the remote object by its name in the Server registry and then invokes methods on the local stub which is responsible for carrying out the method call on the remote object.

*3. File Class*

➔File class is included in java.io package.

File is an abstract representation of file and directory pathnames. Instances of the File class are immutable; i.e. once created, the abstract pathname represented by a File object will never change.

Instances of File class can be created using following constructors.

1. File (File parent, String child)

   # creates a new File instance from a parent abstract pathname and a child string.

2. File (String pathname)

   # creates a new File instance by converting the given pathname string into an abstract pathname.

3. File (String, parent, String child)

   #creates a new File instance from a parent pathname string and a child pathname string.

4. File (URI uri)

   #creates a new File instance by converting the given file: URI into an abstract pathname.

Useful methods of File class includes:-

canRead(), canWrite(), delete(), exists(), getAbsolutePath(), getFreeSpace(), getParent(), getParentFile(), getTotalSpace(), isDirectory(), isFile(), isHidden(), lastModified(), length(), listFiles(), listRoots(), mkdir(), renameTo(File dest), toString() etc.

*4. JTree Class*

➔JTree class is included in javax.swing package.

With the JTree class, we can display hierarchical data. A JTree object doesn't actually contain data; it simply provides a view of the data. Like any non-trivial Swing component, the tree gets data by querying its data model.

JTree displays its data vertically. Each row displayed by the tree contains exactly one item of data, which is called a node. Every tree has a root node from which all nodes descend. By default, the tree displays the root node, but we can decree otherwise. A node can either have children or not. We refer to nodes that can have children — whether or not they currently have children — as branch nodes. Nodes that can't have children are leaf nodes.

Branch nodes can have any number of children. Typically, the user can expand and collapse branch nodes — making their children visible or invisible — by clicking them. By default, all branch nodes except the root node start out collapsed. A program can detect changes in branch nodes' expansion state by listening for tree expansion or tree-will-expand events.

Instances of JTree class can be created using following constructors.

1. JTree (TreeModel newModel)

   #returns an instance of JTree which displays the root node -- the tree is created using the specified data model.

2. JTree (TreeNode root)

   #returns a JTree with the specified TreeNode as its root, which displays the root node.

Useful methods of JTree class are:-

addTreeExpansionListener (TreeExpansionListener tel), addTreeSelectionListener (TreeSelectionListener tsl), getModel(), getPathForLocation (int x, int y), getToolTipText

(MouseEvent event), setShowsRootHandles (Boolean newValue), setCellRenderer (TreeCellRenderer x) etc.

*5. Runtime Class*

➔Runtime class is included in java.lang package.

Every Java application has a single instance of class Runtime that allows the application to interface with the environment in which the application is running.

Useful methods of Runtime Class are:-

getRuntime(), exec(String command) etc.

*6. FileChannel Class*

➔FileChannel is included in java.nio package.

FileChannel is a channel for reading, writing, mapping, and manipulating a file. A file channel has a current position within its file which can be both queried and modified. The file itself contains a variable-length sequence of bytes that can be read and written and whose current size can be queried. The size of the file increases when bytes are written beyond its current size; the size of the file decreases when it is truncated. The file may also have some associated metadata such as access permissions, content type, and last-modification time; this class does not define methods for metadata access. Bytes can be transferred from a file to some other channel and vice versa, in a way that can be optimized by many operating systems into a very fast transfer directly to or from the file system cache.

File channels are safe for use by multiple concurrent threads. The close() method may be invoked at any time, as specified by the Channel interface. Only one operation that involves the channel's position or can change its file's size may be in progress at any given time; attempts to initiate a second such operation while the first is still in progress will block until the first operation completes. Other operations, in particular those that take an explicit position, may proceed concurrently; whether they in fact do so is dependent upon the underlying implementation and is therefore unspecified.

This class does not define methods for opening existing files or for creating new ones. A File channel can be obtained from an existing FileInputStream, FileOutputStream, or

RandomAcessFile object by invoking that object's getChannel method, which returns a file channel that is connected to the same underlying file.

Useful methods of the FileChannel Class are:-

size(), transferFrom (ReadableByteChannel src, long position, long count), transferTo (long position, long count, writableByteChannel target) etc.

## System Designs and Implementations

Remote PC Access [RPA] can be divided into two modules viz. Server module and Client module. The server module resides in the remote computer to be accessed whereas the client module resides in the local computer. The client module is used to access the computer in which the server module is running. In order to establish the connection between them, the server should be running before the client is started.

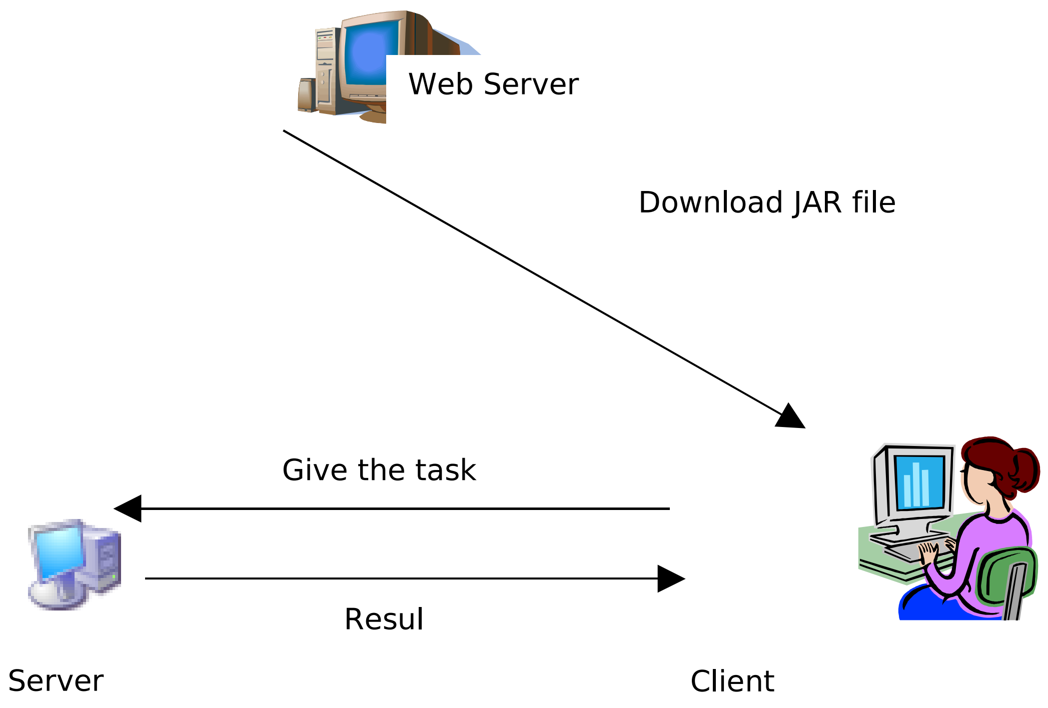


*Fig: Block diagram of Server-Client module*

## 1. Client Module

### *1.1 Starting the applications*

Initially, the client module will not be in the client computer, instead it is in the web server so that it can be accessed from any computer in the network. The user browses the respective web pages and runs the program directly from it. Web Start technology has been used for this purpose.

### *1.2 Communication between Server and Client*

For the communication between Server and Client, the mechanism called RMI (Remote Method Invocation) has been implemented.

### *1.3 Features Implementations*

#### *1.3.1 File System Display*

File Systems have been displayed using tree. For this, JTree class has been used. First, the root node is created and sub-nodes (files and folders) are added dynamically to the node when it is expanded. The nodes contain the files icons and names. FileSystemView class of sub-package filechooser of package Swing have been used to display the system disk icons, file icons and system display names. Both remote and local file systems are displayed in a same screen for convenience.

*1.3.2 Operations on nodes*

*1. Expanding and collapsing nodes*

➔The nodes of tree can be expanded or collapsed if the node is disk or directory. The sub nodes are added dynamically when node is expanded. If the node is the file, then that file can be opened and run on the server.

*2. Refresh*

➔The refresh operation is used to refresh the contents of the tree which is the file system of the Server .When refreshing the tree, all children nodes are removed from the root node and the tree model is reloaded from the root node and other nodes are added.

*3. Transferring files and folders.*

➔Transfer operation is used to transfer file of any size from server to client or from client to server. Class FileChannel in nio package is used to enhance performance. FileInputStream and FileOutputStream object are created from the source and destination file which gives respective FileChannels. The FileChannels are used to transfer data to and from the SocketChannel.

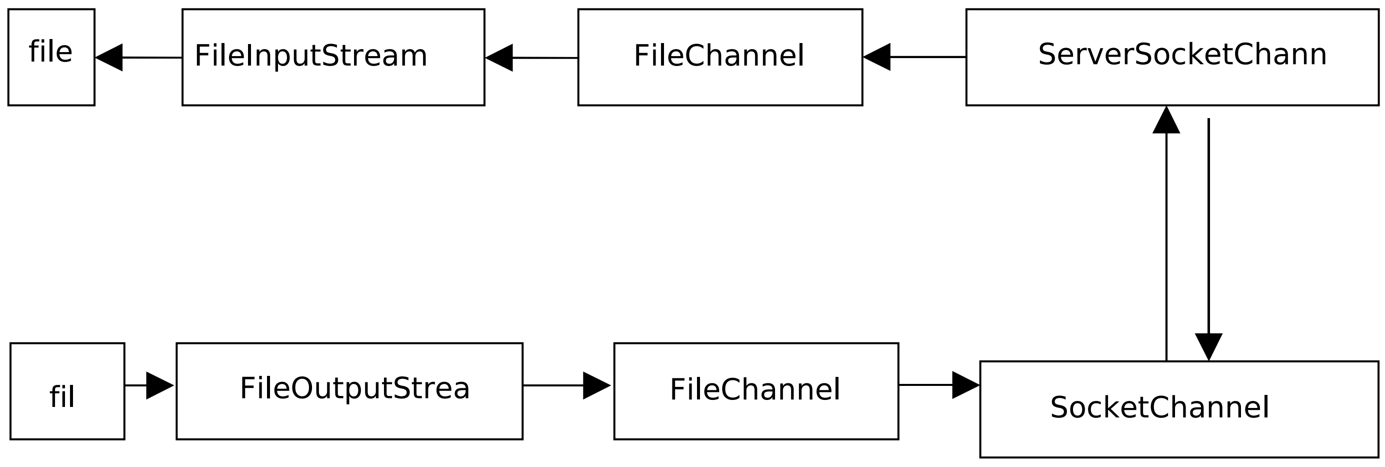


*Fig: Block diagram of File Transfer using Channels*

*4. Transfer and run*

➔The transfer and run operation transfers the file to a temporary folder and runs the file if it is executable file else opens the file through respective program. If the temporary folder doesn't exits initially, then it is created.

*5. Edit*

➔Edit operation can be used to edit files like text files, doc files, java file of the server. For this the file is  downloaded from server to client in the temporary location, opened by respective program and transferred back to the server after editing is finished to the same location replacing the original file.

*6. Creating new folder.*

➔New folder can be created in specified disk or in the folder by giving the folder name. mkdir() method of the file object has been used for this purpose.

*7. Copy/Cut & Paste*

➔A file or folder can be copied from one location to another location of the file system. FileChannel are created for source file and destination file and the data is transferred through the channels. This method gives enhanced performance than the traditional method.

For copying source directory to the destination directory, a new directory of same name as that of source directory is first created and the contents of the source directory are copied. If the contents are files, they are copied directly and if the content is directory, then the same process is repeated recursively.

For cut operation, a file or directory is first copied to the destination directory and then the source file or directory is deleted.

*8.  Renaming*

➔Using the renameTo() method of the File object, the File object can be renamed to the given new name.

*9. Deleting the file nodes*

➔Delete operation can be used to delete the files or folders permanently from the disk. A file can be deleted by creating the File object for that file and invoking the delete() method of it. The folder can be deleted only if the folder is empty, so in order to delete the folder, first all of its contents have to be deleted. For this a recursive function has been used.

*10. Displaying the properties*

➔The general properties of file and folder can be viewed.

The icons and the display names of file or folder is extracted from the system.

The size of the file is given by the length() method of File object. To calculate the size of the directory, we add the total size of all the files of that directory. The number of files and folders in the directory or drive has also been displayed.

The location of the File object is given by getAbsolutePath() method of the File object. Similarly the modified time, readable, hidden, writable of the file can be obtained by methods of File object.

*1.3.3 Restart, Shutdown and log off of remote computer.*

➔These are the windows specific commands to the remote computer. The Runtime object has been used to access the command prompt and execute the restart, shutdown and log off commands.

*1.3.4 Capturing the remote desktop screen*

➔The remote desktop screen can be viewed from the local computer. To perform this, we have used the createScreenCapture() method of Robot object of awt package. We have passed the rectangle of screen size using the Toolkit object as an argument for the createScreenCapture() method. The returned BufferedImage is then saved as the jpeg file. The jpeg file in the server is then transferred to the client side to display the captured screen.

*1.3.5 Remote Command Prompt*

➔Command prompt is a command line utility still in widely use for administrative purpose. When the commands are typed in, the respective operation is performed and results are displayed.

Remote Command Prompt is a command prompt like interface which is used in the client side to give commands to the command prompt of the server side. The respective results are displayed in the Remote Command Prompt. Some widely used commands have been kept in the JList like ping, msg, tasklist, defrag, format, taskkill, systeminfo, attrib, chkdsk, compact, ipconfig, mem, replace etc. When any of these commands is selected, the respective help is generated.

***1.4 List of Files in Client Module***

1. *CommandClient2.java*

   ➔Client side file to display the 'Remote Command Prompt'.

2. *ConnectClient.java*

   ➔To connect the SocketChannel and transfer files between server and client.

3. *ConnectToListener.java*

   ➔For input of server address and port number and user authentication.

4. *FileNode.java*

   ➔Contains File object and methods on it for Remote File System.

5. *IconCellRenderer.java*

   ➔For the cell renderer of each node.

6. *IconData.java*

   ➔Contains FileNode object, its display name, system icon and methods.

7. *LocalFileNode.java*

   ➔Same as FileNode but for Local File System.

8. *LocalFileTree.java*

➔Creates and displays the Tree for the local File System.

9. *LocalProperties.java*

   ➔Displays the properties of the files of Local Computer.

10. *MenuFrame.java*

    ➔Main client side user interface.

11. *Properties.java*

    ➔Displays the properties of the files of Remote Computer.

12. *RemoteFileTree.java*

    ➔Creates and displays the tree for the remote File System.

13. *RemoteInfo.java*

    ➔To display information about the server.

14. *RemoteTreeInterface.java*

    ➔Remote interface containing the remote methods.

15. *ScreenCaptureFrame1.java*

    ➔To display the captured screen of the server.

16. *TransferFile.java*

    ➔Uses the connectClient.java to transfer file.

## 2. Server Module

The server module is implemented in the computer that is to be remotely accessed. The server waits for the connection from the client.

The communication between client and server is through the RMI. The server creates the remote object and it implements the remote interface methods. It calls the rmi-registry to bind the name for the remote object which is the stub and is the reference for the client to look up in the rmi-registry to invoke the remote methods in server.

## ***2.1 Features Implementations***

### *2.1.1 Logging*

A log has been kept in the Server side in order to know the activities performed by the client in the server side. The log is displayed in the JTextArea. Initially, the log contains server's address and port number. Then whenever client performs activities such as deleting, transferring etc, the respective log is appended to the JTextArea.

### *2.1.2 Records of Users*

Users must be authenticated before they can be given access. So, each user is given a unique username and password to identify the user. The username and password are stored in the database in the server side along with the permissions given to the user. The server can add a new user, edit the username and/or password of the existing user and delete the user. When a new user is added, unique username and password have to be specified. This recently added user initially has all the permissions which can be modified later. The permissions available are restart, shutdown, copy, delete, rename, upload, download etc.

To restrict the client from accessing all the drives, a drive or folder can be specified for each user. Then the user can only browse the specified disk or folder and its subfolders and perform the operations on them only. To perform an operation, its permission is first checked from the database and is allowed to perform only if the permission is granted; otherwise the operation can not be performed.

### *2.1.3 Connected Users*

The users connected to the server are displayed in this panel.

## ***2.2 List of Files in Server Module***

1. CommandServer.java

   ➔Server part of 'Remote Command Prompt'.

2. ConnectServer.java

   ➔Server side for transferring files.

3. DisplayQuery.java

➔To return the result of the query given.

4. EditFrame.java

   ➔To show the edit panel to edit username and/or password.

5. ExecuteQuery.java

   ➔To execute the query given.

6. MainProtocol.java

   ➔Called by CommandServer which actually executes the commands.

7. RemoteTreeInterface.java

   ➔ Remote interface containing the remote methods.

8. RmiServer.java

   ➔Creates the remote object and contains the remote methods.

9. Server.java

   ➔Main class of Server side.

10. permissionFrame.java

    ➔To display the permission panel

11. userFrame.java

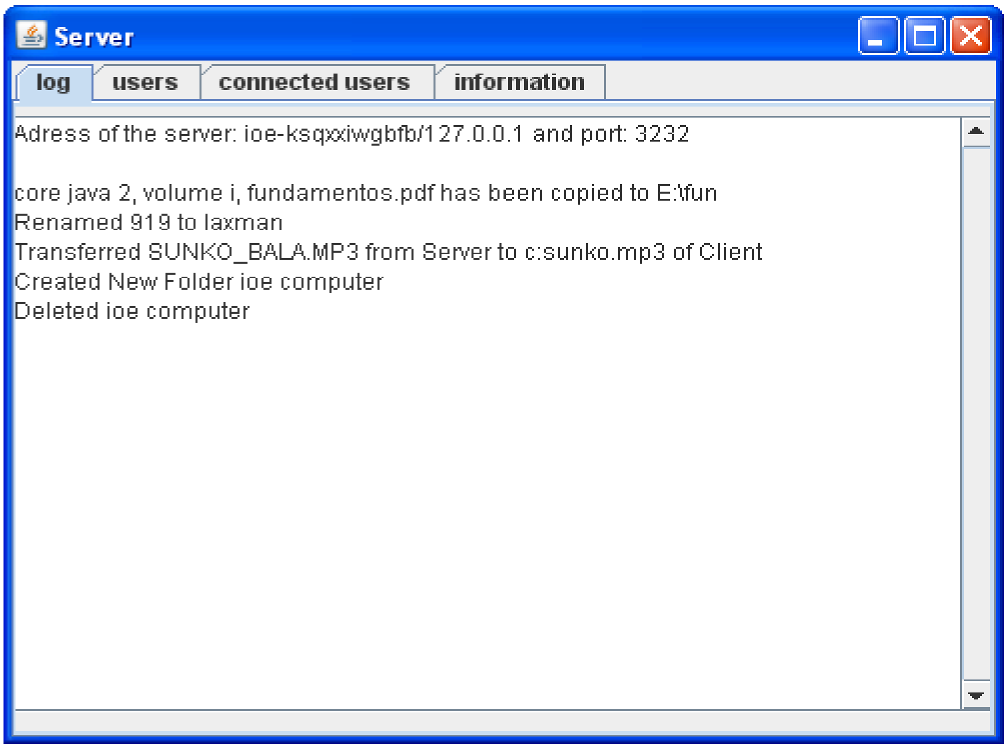
    ➔To display the users panel.

## Features of Remote PC Access [RPA]

- Support **remote file systems access** to transfer files, edit or delete files, rename files, copy/cut files, run applications etc.
- **Multiple user accounts** for accessing the same remote PC by different authorized users.
- **Usernames and Passwords** saved in databases. Correct Username and password must be given for the authentication to access remote PC.

- **Individual permissions for each account** lets to specify what each user can do and what they cannot do on remote computer. The following permissions can be granted: access file system to specific drives or folders only, operations like download files, upload files, delete files, rename files, copy/cut and paste files etc.
- **Logging** writes information about server address and port, actions taken by users in the server like copy, transfers, delete, downloading, renaming etc.
- Calculates **size** of files and folders and display the **number of subfolders and sub files** of the folder or disk of the local and remote computer
- **Remote reboot, shut down, log off and system commands** (windows specific).
- **Remote Screen capture**
- **Two-panel GUI design** showing both local and remote file systems at a time.
- **Java Web Start** technology to launch the application through the web.

## Snap-shots



```
Adress of the server: ioe-ksqxxiwgbfb/127.0.0.1 and port: 3232

core java 2, volume i, fundamentos.pdf has been copied to E:\fun
Renamed 919 to laxman
Transferred SUNKO_BALA.MP3 from Server to c:sunko.mp3 of Client
Created New Folder ioe computer
Deleted ioe computer
```

*Fig 1: Server maintaining the log records*

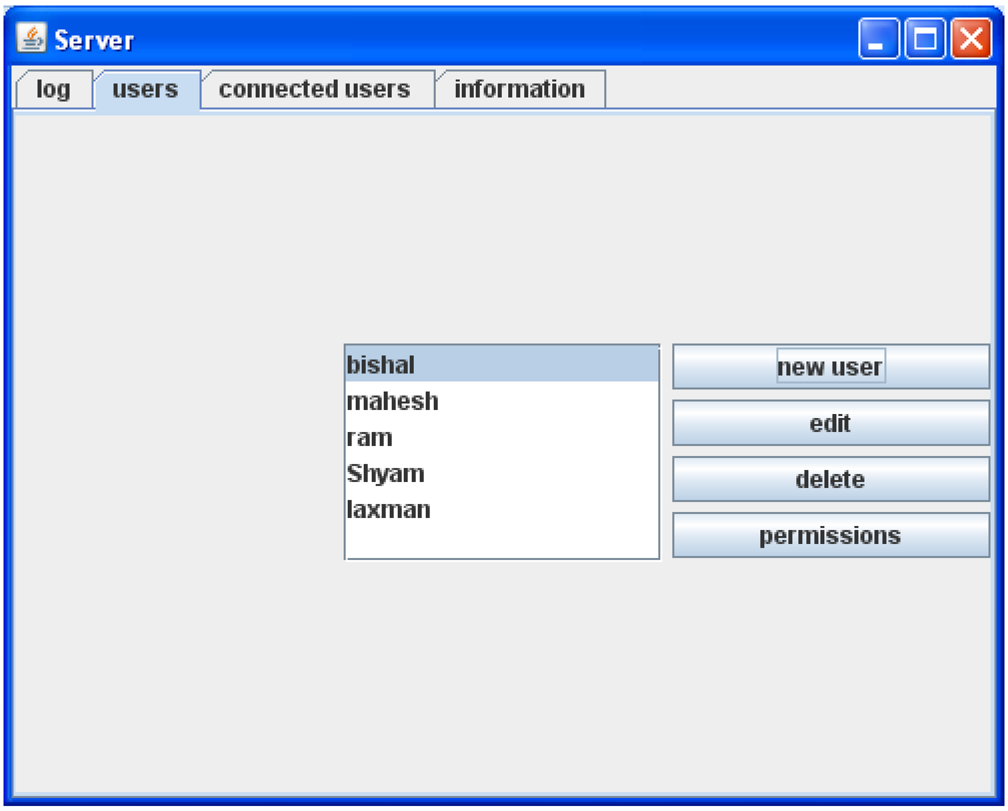
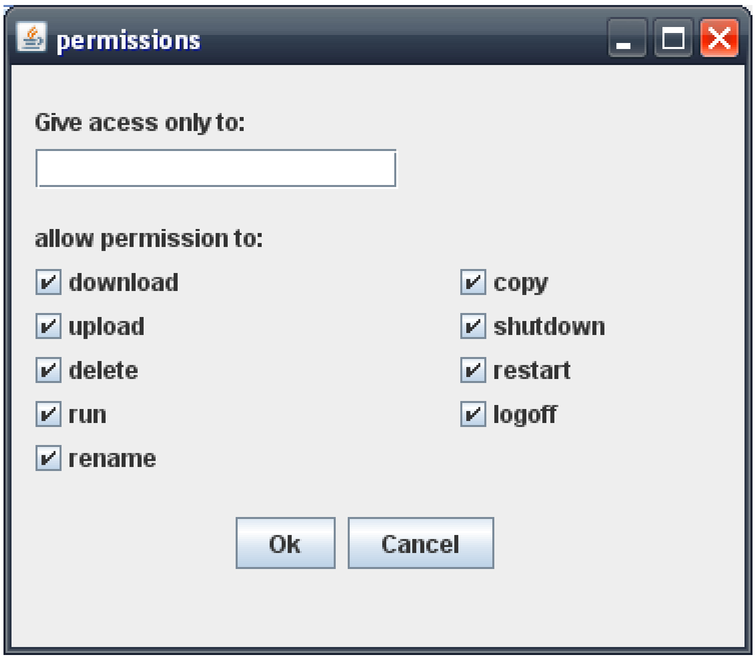


*Fig 2: Server showing available operations on user*



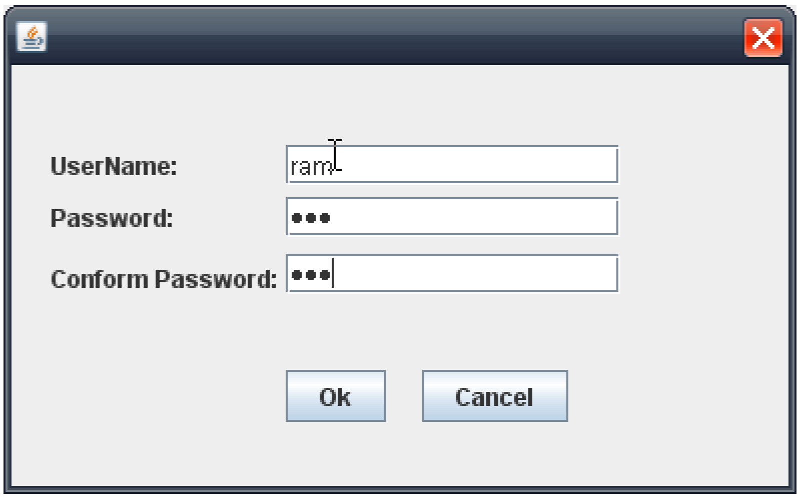
permissions

Give acess only to:

allow permission to:

- [✓] download
- [✓] upload
- [✓] delete
- [✓] run
- [✓] rename
- [✓] copy
- [✓] shutdown
- [✓] restart
- [✓] logoff

*Fig 3: Server granting permissions*

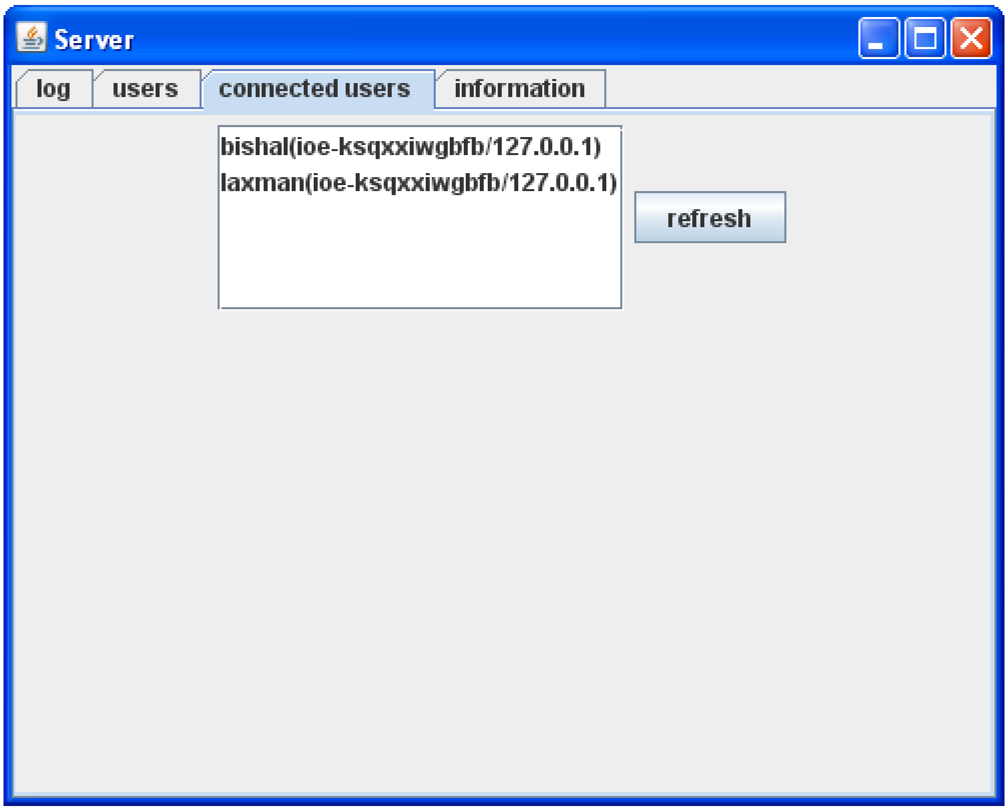


UserName: ram

Password: ●●●

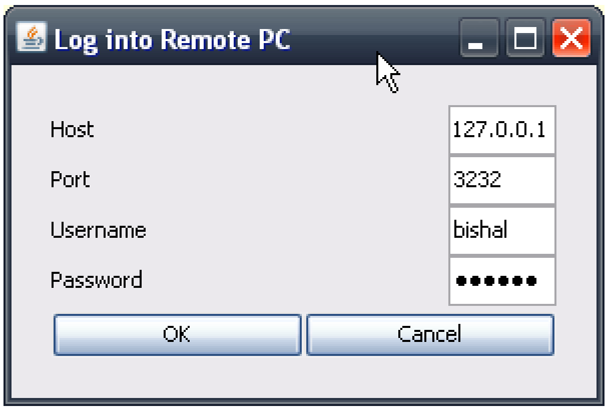
Conform Password: ●●●

*Fig 4: Server creating a new user account*



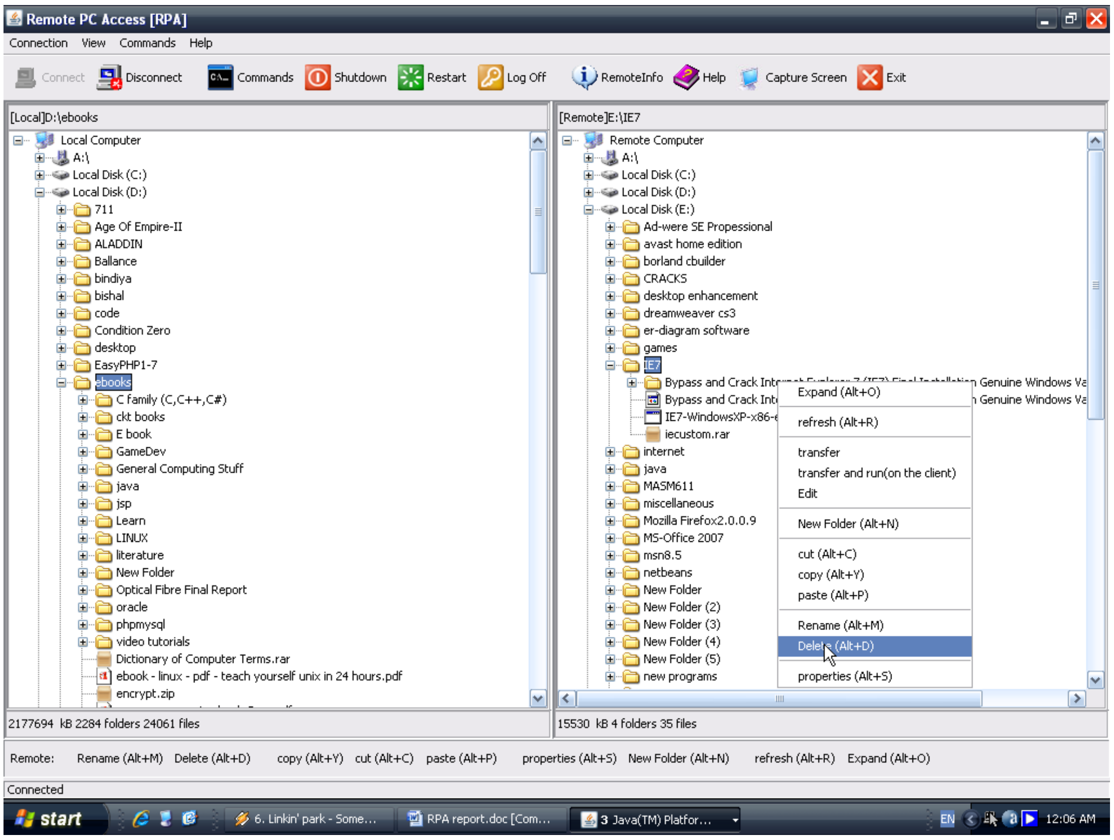
bishal(ioe-ksqxxiwgbfb/127.0.0.1)
laxman(ioe-ksqxxiwgbfb/127.0.0.1)

*Fig 5: Server showing connected users*

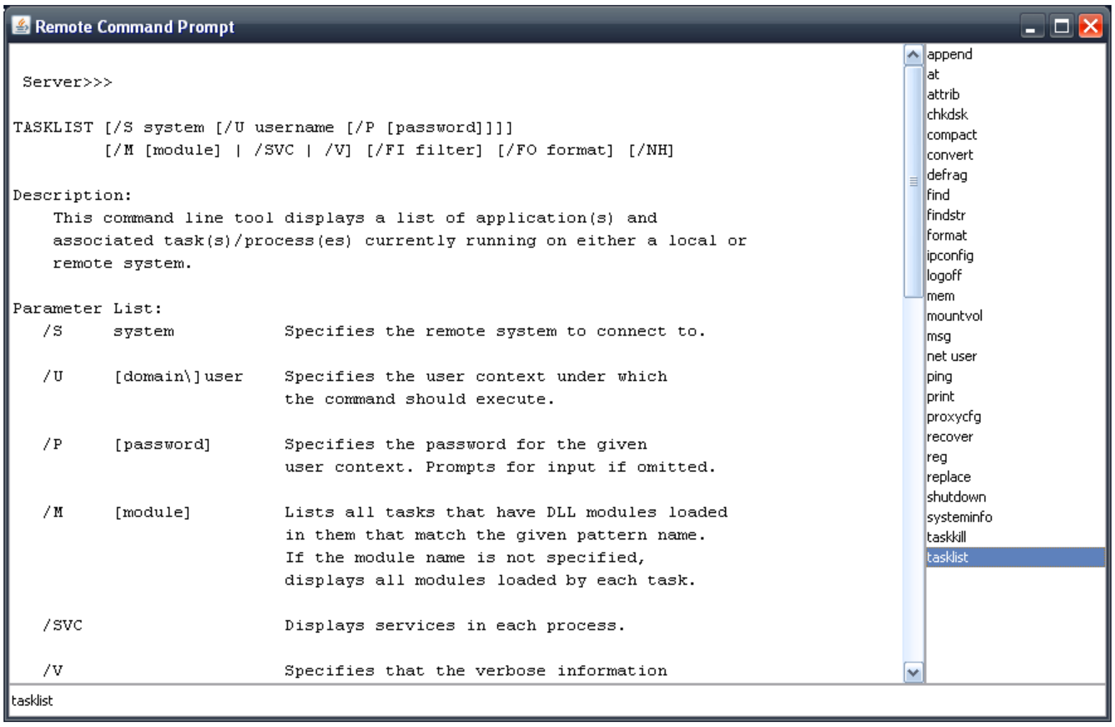


Host 127.0.0.1
Port 3232
Username bishal
Password ●●●●●●

*Fig 6: User Authentication by username and password*



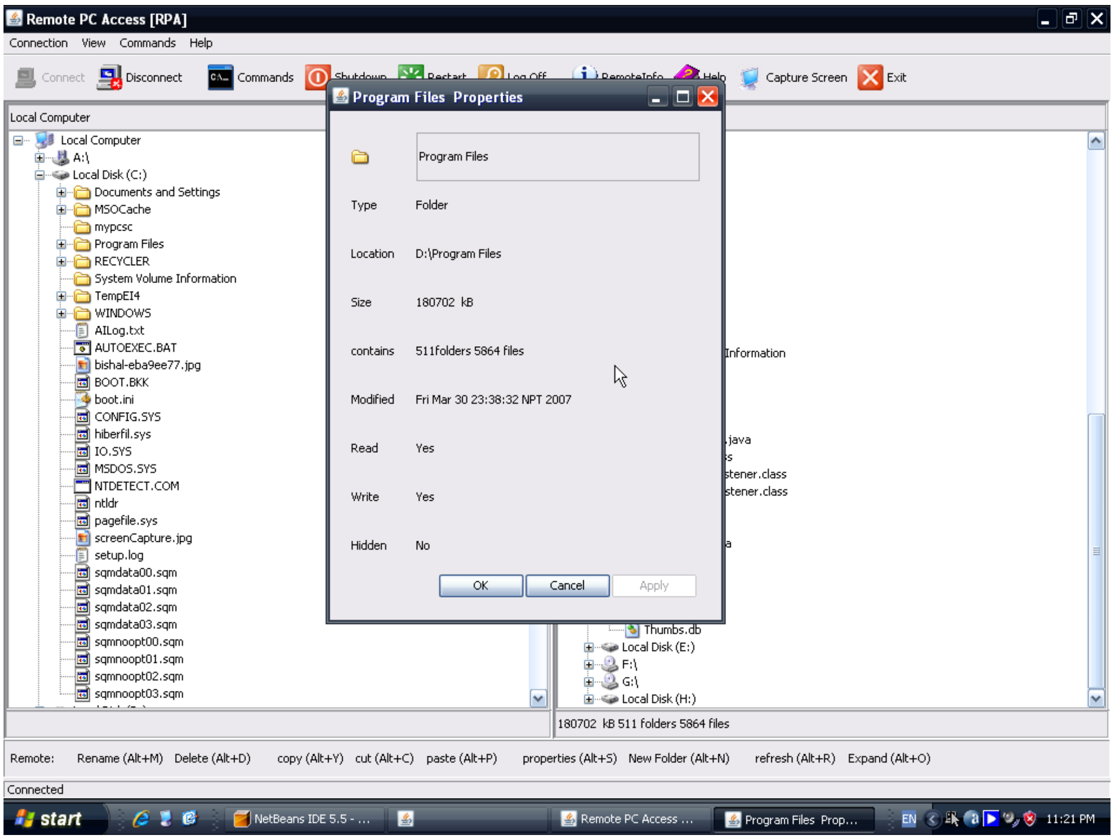
<u>*Fig 7: RPA Client Screen showing both local and remote file systems*</u>



```
 Server>>>

TASKLIST [/S system [/U username [/P [password]]]]
         [/M [module] | /SVC | /V] [/FI filter] [/FO format] [/NH]

Description:
    This command line tool displays a list of application(s) and
    associated task(s)/process(es) currently running on either a local or
    remote system.

Parameter List:
   /S     system           Specifies the remote system to connect to.

   /U     [domain\]user    Specifies the user context under which
                           the command should execute.

   /P     [password]       Specifies the password for the given
                           user context. Prompts for input if omitted.

   /M     [module]         Lists all tasks that have DLL modules loaded
                           in them that match the given pattern name.
                           If the module name is not specified,
                           displays all modules loaded by each task.

   /SVC                    Displays services in each process.

   /V                      Specifies that the verbose information
```

```
tasklist
```

<u>*Fig 8: Remote Command prompt*</u>


Remote PC Access [RPA]
Connection View Commands Help
Connect Disconnect Commands Shutdown Restart Log Off RemoteInfo Help Capture Screen Exit
Program Files Properties
OK Cancel Apply
Remote: Rename (Alt+M) Delete (Alt+D) copy (Alt+Y) cut (Alt+C) paste (Alt+P) properties (Alt+S) New Folder (Alt+N) refresh (Alt+R) Expand (Alt+O)
start NetBeans IDE 5.5 - ... Remote PC Access ... Program Files Prop... EN 11:21 PM

Local Computer

```
Local Computer
  A:\
  Local Disk (C:)
    Documents and Settings
    MSOCache
    mypcsc
    Program Files
    RECYCLER
    System Volume Information
    TempEI4
    WINDOWS
    AILog.txt
    AUTOEXEC.BAT
    bishal-eba9ee77.jpg
    BOOT.BKK
    boot.ini
    CONFIG.SYS
    hiberfil.sys
    IO.SYS
    MSDOS.SYS
    NTDETECT.COM
    ntldr
    pagefile.sys
    screenCapture.jpg
    setup.log
    sqmdata00.sqm
    sqmdata01.sqm
    sqmdata02.sqm
    sqmdata03.sqm
    sqmnoopt00.sqm
    sqmnoopt01.sqm
    sqmnoopt02.sqm
    sqmnoopt03.sqm
```

Program Files Properties

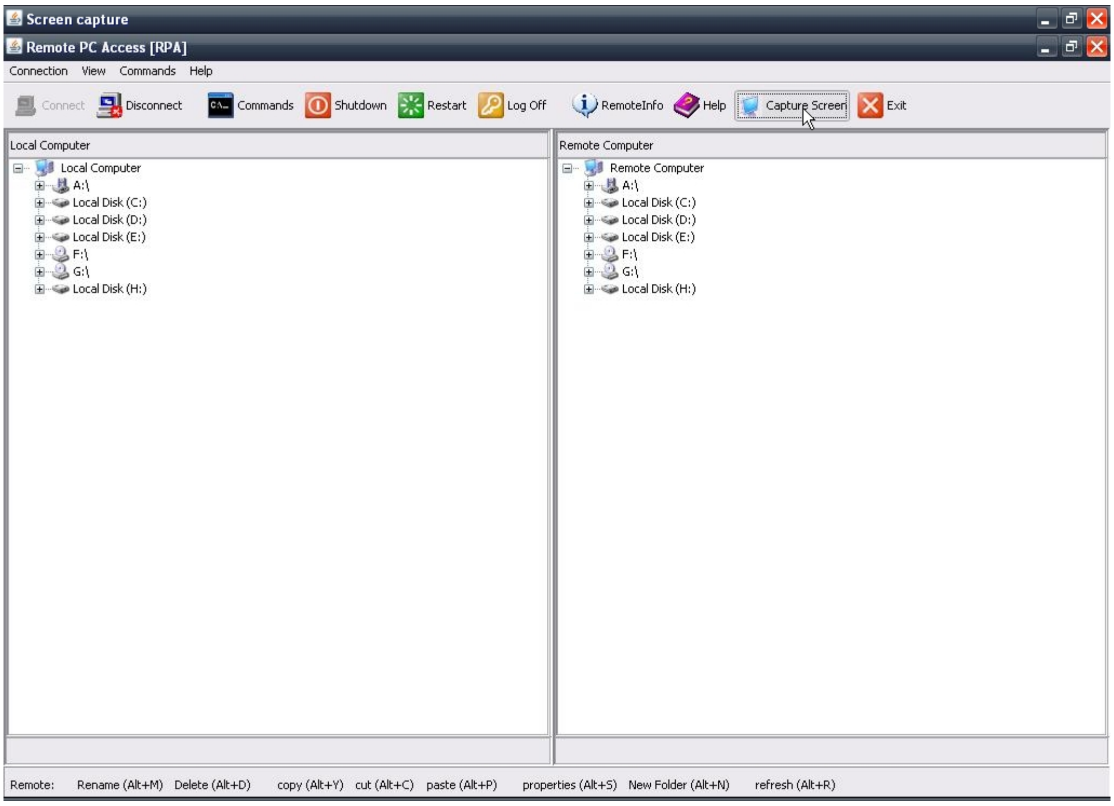
| | |
|---|---|
| | Program Files |
| Type | Folder |
| Location | D:\Program Files |
| Size | 180702 kB |
| contains | 511folders 5864 files |
| Modified | Fri Mar 30 23:38:32 NPT 2007 |
| Read | Yes |
| Write | Yes |
| Hidden | No |

```
Information
.java
s
stener.class
stener.class
a
    Thumbs.db
  Local Disk (E:)
  F:\
  G:\
  Local Disk (H:)
```

180702 kB 511 folders 5864 files

Connected

*Fig 9: Properties of a file*


Screen capture
Remote PC Access [RPA]
Connection View Commands Help
Connect Disconnect Commands Shutdown Restart Log Off RemoteInfo Help Capture Screen Exit
Remote: Rename (Alt+M) Delete (Alt+D) copy (Alt+Y) cut (Alt+C) paste (Alt+P) properties (Alt+S) New Folder (Alt+N) refresh (Alt+R)

Local Computer

```
Local Computer
  A:\
  Local Disk (C:)
  Local Disk (D:)
  Local Disk (E:)
  F:\
  G:\
  Local Disk (H:)
```

Remote Computer

```
Remote Computer
  A:\
  Local Disk (C:)
  Local Disk (D:)
  Local Disk (E:)
  F:\
  G:\
  Local Disk (H:)
```

*Fig 10: Captured Remote Screen*

## Scopes of RPA

Remote PC Access [RPA] is ideal for accessing the file system of the remote computer. It is fast, efficient, flexible and portable. It allows users to connect to their own PC while being physically away from their computer. Thus RPA has following scopes:-

- Accessing the remote file system.
- For administration and control purposes.
- For monitoring purposes.

## Limitations of RPA

- In order to connect to a remote computer, the IP address and port of the server should be known initially.
- Remote restart, shutdown, log off and system commands in the command prompts like defrag, format, tasklist, find, print etc. are windows platform specific.
- Single file or folder selection.

## Future Enhancements in RPA

- Selection and Operations on multiple files and folders  at a time, like calculating total size of selected items, copy/cut and paste, transferring, deleting all the selected items at a time.
- Support of drag and drop while transferring files between remote and local computers.
- Using encryption methods for the secure connection and transfers.
- The IP address and port can be stored in a separate web server so that the client can connect easily.

## Conclusion

Thus, the networking software [RPA] is developed successfully which lets to access one’s PC from another PC via the Internet or LAN and work on that computer remotely.

All the objectives proposed during the proposal submission have been met and we became familiar with JAVA language. We mainly get the knowledge of RMI, JTree class, File class, Java Web Start, networking, and GUI design etc.

## References

1. “Java: How to Program”, Deitel and Deitel

2. “The Complete Reference”, Herbert Schidlt

3. “Java Network Programming”, Elliotte Rusty Harold

4. www.ftponline.com

5. www.wikipedia.com

6. www.sun.java.com

7. www.forum.sun.java.com