LAPDROID

(A PC Mobile Two Way Communication System)

Mangesh Bhautkar[1], Aniruddha Bansod[2], Rakesh Rangari[3],Abhinav chadrakar[4]
Computer Science of Engineering
Guru Nanak Institute of Engineering and Technology
Nagpur, Maharashtra, India
mangesh.2013.bhautkar@gmail.com

***Abstract*— This android plus desktop application is a short hand method to remotely access the files and other applications of android smart phone through a laptop via Wi-fi or internet.**

I. INTRODUCTION

The goal of this project is to develop a remote desktop application allowing a user to have full control over the computer's mouse and keyboard in a smart phone platform. The Internet is used for the communication between the server and the client. The application allows a user to control a computer without getting physically involved. This project can be used to access the computer from any part of the world using the phone.

Nowadays multiple remote monitoring software tools are available. Those are made remote view of the Personal computer from another computer or mobile. This technique takes high data transmission. In this project we implement Remote PC File Manager application. In this application user can explore the remote computer drives and files through internet.. This project is made for smart phones. Smart phones are available in three major smart phone operating systems. The operating platforms are Android, Ios, Windows. The Android is a growing technology which has started to fulfill the needs with lots of applications to make things handy. So this project will be developed in android.

The introduction of smart phones has brought a big change in the technical field related to cellular phones. Now a days , smart phones are used worldwide and provide much better facilities than previously available cellular phones. These phones provide features which were previously provided by computer system architecture. In this project ,we describe the system which can provide access to remote computer system through internet and provide features like desktop access, panning and zooming, over viewing , file transfer, playing video. This system will be implemented on Android software stack. Android software stack provides various packages for networking and it also provides high performance for android cellular mobiles. The security is maintained by providing authentication at the server side.VNC architecture and VNC protocols are used for client and server transactions. In the scope of remote control there are several projects and initiatives designed to allow remote control between devices. Although most of the architectures have the objective of control remotely PCs, there are some initiatives that aim to control mobile devices.

The project idea includes presenting android based remote control of mobile devices through VNC. This project proposes and analyzes different architectural approaches for the implementation of remote control systems of mobile devices using the Android software stack. In this work, we propose a fast screen sharing method to improve screen update rate in mobile VNC systems. In case of mobile devices, high complexity video compression techniques cannot be employed due to their strict computation limit. However, the bandwidth limitation requires a certain level of compression ratio. Thus, there exists a trade-off between encoder complexity and compression ratio for fast mobile VNC systems. We first integrate various video encoders into our prototype system, and explore their suitability for mobile VNC. Also, the existing RFB protocol for VNC is extended to easily integrate video encoders in a backward-compatible way. We additionally propose a new modified region coding method which transmits only modified regions between current and previous screen images. We implemented a prototype mobile VNC system actually, and its practical performance is widely evaluated.

A. Literature Survey

- Proposed by- Archana Jadhav, Vipul Oswal, Sagar Madane, Harshal Zope, Vishal Hatmode on "VNC Architecture based Remote Desktop access through android mobile phones", under"Intrenational Journal of Advanced Reseach in Computer and Communication EngineeringVol.1, Issue 2, April 2012, ISSN 2278-1021.
- Proposed by- Sonam Gavhane, Rasika Phanse, Monica Sadafule, B.W. Balkhande on "Remote Desktop on Mobile", under "International Journal of Innovations in Engineering and Technology (IJIET)."
- Proposed by- R. Manikanandsamy on "Remote Desktop Connection using Mobile Phone", under "International Journal of Science, Engineering and

Technology Research", Vol.2, Issue 8, Aug 2013, ISSN 2278-7798.

Design.

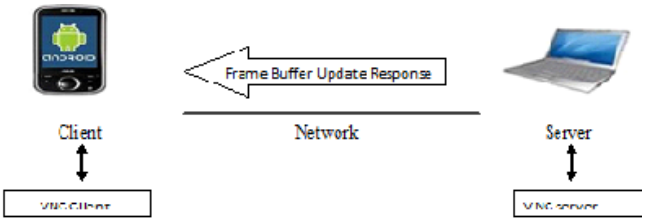
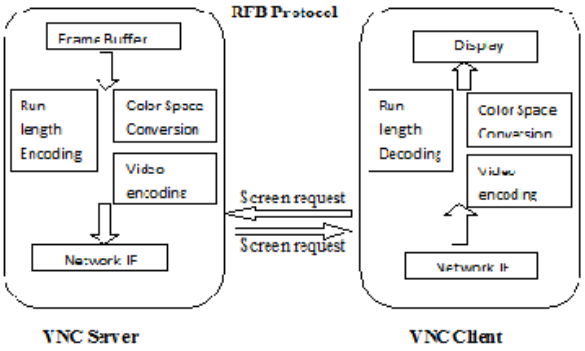


FIG.3.1 CLIENT SERVER COMMUNICATION



B. Modules.

- Server Module
- Client Module
- Communication Module

Server Module

This module runs on pc(personal computer) side. Server application (Tight VNC) is installed in pc. Server waiting for client connection, server is response to the client request when client connect to the server.

Client Module

This module runs on mobile (android smart phone) side. Client application developed in android. This application is run on android 2.2 (froyo) and above version of android operating systems. In this module connect initiate to the server and get response for the file systems through internet.

Communication Module

In this module server and client connect through the internet protocol TCP/IP. This protocol enables the connection via the internet. Once connection enabled between the client and server, client send the login request to the server. Server is check and response to the client. Once username and password are authorized client enter to the File system Activity.

C. Implementation

VNC (an abbreviation for Virtual Network Computing) is a great client/server software package allowing remote network access to graphical desktops. With VNC, you can access your machine from everywhere provided that your machine is connected to the Internet. VNC is free (released under the GNU General Public License) and it's available on most platforms. Here you can find an **enhanced version of VNC**, called TightVNC, which includes a lot of new features, improvements, optimizations and bugfixes over the original VNC version, see the list of features below. Note that TightVNC is still free, cross-platform and compatible with the standard VNC. Many users agree that TightVNC is the most advanced free remote desktop package. And it's being actively developed so you can expect that TightVNC will become even better.

TightVNC can be used to perform remote control and administration tasks in Windows, Unix and mixed network environments. It can be very helpful in distance learning and remote customer support. Finally, you can find a number of additional VNC-compatible utilities and packages that can extend the areas where TightVNC can be helpful.

TightVNC is a project maintained by Constantin Kaplinsky. Many other individuals and companies participate in development, testing and support.

D. Operations to be Performed.

- **DESKTOP SHARING:**

In this module the remote desktop screen will be shared. This can be implemented with the help of the VNC protocol. VNC protocol is based on the concept of a remote frame buffer(RFB). The protocol simply allows a server to update the frame buffer displayed on a viewer.

- **PANNING AND ZOOMING:**

The user can move the viewport horizontally and vertically. The viewport can be widened (zoom out) to browse its contents and narrowed (zoom in) to see the display in greater detail.

- **POINTING AND CLICKING:**

The user can move the pointer on the remote desktop display vertically and horizontally by pressing keys. Dragging can be executed by pressing a key to specify the start of the dragging operation, then moving the pointer, and finally pressing the same key to indicate the end of the dragging operation. When the pointer approaches the edge of the viewport, the viewport is automatically panned to follow the pointer. Clicking mouse buttons can be performed by pressing

the corresponding keys on the cellular phone. Double-clicking can be executed by pressing a specific key as a prefix.

- **INPUTTING TEXT:**

Text is entered and edited locally on the cellular phone using the built-in text input capability of the cellular phone. After editing on the cellular phone, the text is transmitted to the VNC server. The same mechanism can be used to send control characters such as backspace, delete, carriage return, and line feed, by entering escape sequences.

- **PLAYING VIDEO/AUDIO:**

Thus, there exists a trade-off between encoder complexity and compression ratio for fast mobile VNC systems. We first integrate various video encoders into our prototype system, and explore their suitability for mobile VNC. Also, the existing RFB protocol for VNC is extended to easily integrate video encoders in a backward-compatible way.

- **OPENING DRIVES AND FOLDERS:**

As the virtual environment provides us to visit each and every drive and folder, we can easily search for any folder by entering any corresponding drive and do the required task.

- **MANIPULATION OF FILES:**

The word files can easily be opened stored at any location and various operations like editing, appending, copying and pasting can be performed. VNC allows to do these jobs by providing us with the virtual environment it provides us with.

E. Medhodology.

HOW THE SERVER SENDS THE UPDATED SCREEN TO THE CLIENT IS IMPLEMENTED IN SEVERAL STEPS:

- VNC server keeps the client requests in a separate frame buffer which is called as RFB (Remote Frame buffer) protocol.
- After receiving the screen update request from client, the server captures the screen image from frame buffer.
- It **can** be used for further processing.
- VNC server uses one of the improved encoding technique, the color space of the screen images is then converted into YUV format before compression because most video encoders supports YUV format which is easy to compress.
- The captured screen image data is encoded and send form server to client with a frame buffer Update message.
- The client then decoded and receives the updated bit stream.
- The procedure continues until the connection between client and server is finished or the connection is lost.

F. Basic Types of Encoding.

Basic types of VNC encoding can be used: Raw, RRE, Hextile, Zlib and Tight.

- **RAW:**

RAW is the simplest encoding form. In this server sends all graphical pixels to the client and data consist in the form of width*height pixel values (where width and height are the width and height of the rectangle). This encoding method must be supported by clients. The process time used is minimal and the performance is very high when the server and the client are on the same machine. If the client is hosted in a remote device the performance is reduced due to the transfer Of large amounts of data. This encoding is specifically designed for the low performing devices.

- **RRE:**

RRE stands for Rise-and-Run-length-Encoding which consists of grouping consecutive identical pixels in order to send only the information of one pixel and the number of replications of that pixel. The basic intension behind RRE is the partitioning of rectangle of pixel data into sub regions each of which consist of the single pixel value and union of which comprises the original rectangular region. It is an most effective method when large blocks of the same color exist, like in patterns are to be send. There is a variant of the protocol which uses a maximum of 255x255 pixels to reduce the size of the packages.

- **COPYRECT ENCODING (COPY RECTANGLE):**

This encoding is a very simple and efficient encoding. Which is used when the client already has the same pixel data elsewhere in its frame buffer. The encoding on the wire simply consists of an X,Y coordinate. This gives a position in the frame buffer from which the client can copy the rectangle of pixel data. This can be used in a variety of situations, the most obvious of which are when the user moves a window across the screen, and when the contents of a window are scrolled. A less obvious use is for

optimizing drawing of text or other repeating patterns.

- **HEXTILE:**

This encoding divides the rectangles in the 16*16 tiles, allowing the dimensions of the sub resctables to be specified in 4 bits each and 16 bits in total. The rectangle is split into tiles starting at the top left going in left-to-right, top to-bottom order. This encoded data contents of the tiles simply follow one another in the predetermined order. If the width of the whole rectangle is not an exact multiple of 16 then the width of the last tile in each row will be smaller than that of previous tiles. Each tile is either encoded as raw pixel data, or as a variation on RRE. Each tile has a background pixel value, as before. No need to specify for any given tile if it is same as background of the previous tile.

- **ZRLE:**

ZRLE and Zlib that use lossless compression, ZRLE (Zlib Run length Encoding) is a more optimized version of Zlib. The optimizations in ZRLE include tiling, palettisation and run length encoding that improve the processing speed of Zlib compression. Thus, we finally short-list ZRLE full, ZRLE 256, ZRLE 64, and ZRLE 8 encodings.

G. VNC Requires The Following Functionalities to do the Processing.

- User first downloads and installs Remote VNC application on mobile device.
- Start the application.
- Provide unique IP/Code/URL to Facilitator (other User).
- User will enter the IP/Code/URL in to browser and get the access of remote server.
- User will request different services available on client's mobile device.
- Service will respond to client's request.

There are three stages to the RFB protocol implementation. First is the handshaking phase, the purpose of which is to agree upon the protocol version and the type of security to be used. The second stage is an initialization phase where the client and server exchange *ClientInit* and *ServerInit* messages. The final stage is the normal protocol interaction. Handshaking begins by the server sending the client a *Protocol Version* message.

- **ClientInit:**

Shared-flag is non-zero (true) if the server should try to share the desktop by leaving other clients connected, zero (false) if it should give exclusive access to this client by disconnecting all other clients.

- **ServerInit:**

After receiving the *ClientInit* message, the server sends a *ServerInit* message. This tells the client the width and height of the server's framebuffer, its pixel format and thename associated with the desktop.

- **Client to server messages:**

The client to server message types defined are:

Number	*Name*
0	*SetPixelFormat*
1	*SetEncodings*
2	*FramebufferUpdateRequest*
3	*KeyEvent*
4	*PointerEvent*
5	*ClientCutText*

Table 1:Client Server Message Types

References

- Eason, B. Noble, and I.N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529-551, April 1955. (references)
- J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68-73.
- I.S. Jacobs and C.P. Bean, "Fine particles, thin films and exchange anisotropy,"
- Android. http://www.android.com Retrieved March 1st, 2011.
- www.realvnc.com/docs/rfbproto.pdf, reviewed on June 20th,2011