International Journal of MC Square Scientific Research Vol.8, No.1 Nov 2016

REMOTE DISPLAY ACCESS USING REMOTE FRAME BUFFER AND IO STREAMING

N.Snehalatha
Assistant Professor, Dept of SWE, SRM University kattankulathur ,Chennai¸ Tamilnadu-603023,India
snehalatha.n@ktr.srmuniv.ac.in
T.S. Shiny Angel
Assistant Professor, Dept of SWE, SRM University kattankulathur ,Chennai¸ Tamilnadu-603023,India
shiny Agel.t.s@ktr.srmuniv.ac.in
S.Amudha
Assistant Professor, Dept of SWE, SRM University kattankulathur ,Chennai¸ Tamilnadu-603023,India
amudha.s@ktr.srmuniv.ac.in

Abstract - This paper propose a virtual network computing (VNC) based architecture for accessing the desktops of remote computers from a cellular phone. A viewer is provided on the cellular phone that enables the user to see and manipulate the desktop of various remote systems such as MS Windows, Macintosh, and UNIX. The system to be accessed must be running a VNC server and it must be attached to a network. A proxy is used to send the image of the desktop to the cellular phone, to convert different devices, to suppress network traffics, and to support recovery from an unscheduled disconnection. To reduce user effort and solve problems inherent to the cellular phone's small screen, several functions are provided on the cellular viewer. Frequently used screen areas can be assigned and restored quickly by using the Shortcut function. The Guidance function can be used to show the current key assignments. Two areas can be viewed at the same time by the Twin view function. In this paper a prototype of the proposed architecture has been implemented using Java and has been tested on a Java-enabled cellular phone emulator.

Keywords **-** VNCserver, SVNCserver, VNC Viewer, Remote Frame Buffer Protocol (RFB) and Compact Remote Frame Buffer Protocol (CRFB)

1. Introduction

Cellular phones have shown a dramatic improvement in their functionality to a point where it is now possible to have cellular phones execute Java programs. As a result, cellular users throughout the world are now able to read and write e-mail, browse Web pages, and play Java games using their cellular phones. This trend has prompted us to propose the use of a cellular phone as a device for remotely controlling computers. For example, if a cellular user is able to remotely access computers (such as workstations in offices and personal computers (PCs) in homes) or other networked digital appliances, it would provide the user with the following capabilities:

- To see the contents of a file placed on the desktop of a remote computer.
- To reboot a remote server as an administrator.

VNC stands for Virtual Network Computing. It is, in essence, a remote display system which allows you to view a computing 'desktop' environment not only on the machine where

it is running, but from anywhere on the Internet and from a wide variety of machine architectures.

We propose a virtual network computing (VNC) based architecture for accessing the desktops of remote computers from a cellular phone. A viewer is provided on the cellular phone that enables the user to see and manipulate the desktop of various remote systems such as MS Windows, Macintosh, and UNIX. The system to be accessed must be running a VNC server and it must be attached to a network.

The main activities reside on accessing the desktop of remote computers.

The most important steps are given below:

1. Server connecting
2. Desktop viewing
3. Mouse pointer and key accessing

Server Connecting

This server connecting module is used for connecting the server to the mobile client using Stream Socket Connection. For the connection the remote desktop IP address and password is given in the mobile and then connected. Jar file is created using J2ME wireless toolkit & installed into the cellular phone through the USB port.

Desktop Viewing

The computer screen's view is fragmented into 8 to 16 parts to synchronize with the display area of the mobile. Full Screen mode can also be viewed.

Mouse Pointer And Key Accessing

The user can move the pointer on the remote desktop vertically & horizontally by pressing keys. Similarly events such as clicking, double clicking &dragging can be done by pressing keys as specified in our program. Text can be entered and edited locally on the cellular phone using the built-in text input capability of the cellular phone.

Purpose

To access the desktops of various remote systems such as MS Windows, Macintosh and UNIX systems, from a cellular phone using the Virtual Network Computing (VNC) based architecture, and do all types of processes like mouse and key accesses.

Focus of the Paper Work

- This project uses a cellular phone as a device.
- To control remote computers.
- To send the image of the desktop to the cellular phones.
- To convert different devices.
- To suppress network traffics.
- To support recovery from an unscheduled disconnection

Suppress Network Traffic: The wireless transmission bandwidth available for a cellular phone is limited. Currently, it is 384k bps, even on IMT-2000 based services (only downstream at this transmission rat

Recover From An Unscheduled Disconnection: Because of its wireless nature, stable network connectivity cannot be expected. For example, when the user goes into a tunnel or a building, established connections can be lost. In addition, In order to use the same cellular phone to talk to someone, the user must terminate the network connection.

Suppress Computational Resource Use: CPU performance and memory size are limited on a cellular phone to achieve portability and to lower power consumption. BASIC

OPERATIONS and the following functions are available on the cellular viewer as basic operations to manipulate the view port and to send events.

Panning And Zooming: The user can move the view port horizontally and vertically. The view port can be widened (zoom out) to browse its contents and narrowed (zoom in) to see the display in greater detail.

Over-viewing: In order to browse the entire area of the desktop display and to choose a specific area the over viewing mode is provided. When the user turns this mode on, the aspect ratio is changed so that the whole area is rendered to fit the screen of the cellular phone. The user can move the black rectangle horizontally or vertically and shrink or enlarge it by pressing keys to change the view port while examining the desktop display. This helps the user adjust the view port to the desired area of the desktop display.

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 view port, the view port 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 via the SVNC proxy. The same mechanism can be used to send control characters such as backspace, delete, carriage return, and line feed, by entering escape sequences.

Shortcut Assignment: Common GUI operations, such as pressing GUI buttons and opening pull-down menus become very tiresome when only basic operations are provided. For example, to push a GUI button that is not currently displayed on the viewer, the user has to zoom out, pan several times, and may then have to zoom in to show the button. To shorten the time necessary to access frequently used display areas, the SVNC viewer offers a mechanism called a shortcut. This mechanism enables the user to register the current view port to a specific numerical key by pressing a numerical key from Key-1 to Key-9 after pressing Key-Asterisk twice. For subsequent operations, the user can easily restore the viewport by pressing the designated numerical key after pressing Key-Asterisk once.

For example, when accessing a remote MS Windows sys-tem, the user may assign shortcuts to the current working area, to the upper-left corner of the current application window (i.e., the area around “File” menu), and to the lower-left corner (i.e. the area around “Start Menu”).

2. Existing System

In the existing system, we use Bluetooth connection to access the system contents in a wireless device. Desktop applications are redesigned to operate on mobile hardware platforms in a shorter and portable mode, thereby often losing functionality. To executing typical office applications users connect over a wired local area network to the central server. Mobile cloud computing can give mobile device users a number of advantages. Company users are able to share resources and applications without a high level of capital expenditure on hardware and software resources. Mobile cloud computing provides a solution to meet the increasing functionality demands of end-users, as all application logic is executed on distant servers and only user interface functionalities reside on the mobile device. The mobile device

acts as a remote display, capturing user input and rendering the display updates received from the distant server.

Limitation of Existing System

- In the existing system we use Bluetooth to access the system contents in a mobile
- Particular application can access and the files of the system can be accessed only within short distances
- In the existing system we use BLUETOOTH to access the system contents in a mobile.
- In this, only particular application can to access.
- The files of the system can be accessed only with in short distances.

3. Proposed System

Virtual Network Computing (VNC) is a desktop sharing system which uses the RFB (Remote Frame Buffer) protocol to remotely control another computer. It transmits the user events from one computer to another relaying the screen updates back in the other direction, over a network.

How VNC works on a mobile phone?

- Here, instead of PC a GPRS enabled cellular phone is used as the client.
- A viewer called SVNC viewer is provided on the cellular phone that enables the user to see and manipulate the desktop of various remote systems such as MS Windows, Macintosh, and UNIX.
- The system to be accessed must be running a VNC server and it must be attached to a network.

Existing System Vs Proposed System

- Conventional VNC System comprises accessing of remotely located computer desktop via internet by another computer.
- This system lacks ease of portability & hence in our proposed system we are using cellular device as the client which enables itinerancy across any part of the world.

4. Problem Formulation

This paper presents a virtual network computing (VNC) based architecture for accessing the desktop of various remote systems (such as MS Windows, Macintosh, and UNIX systems) from a cellular phone. It is assumed that the remote computer system is running a VNC server and that it is attached to a network. The cellular user can see and manipulate the desktop on the cellular phone.

VNC stands for Virtual Network Computing. It is, in essence, a remote display system which allows you to view a computing ‘desktop’ environment not only on the machine where it is running, but from anywhere on the Internet and from a wide variety of machine architectures.

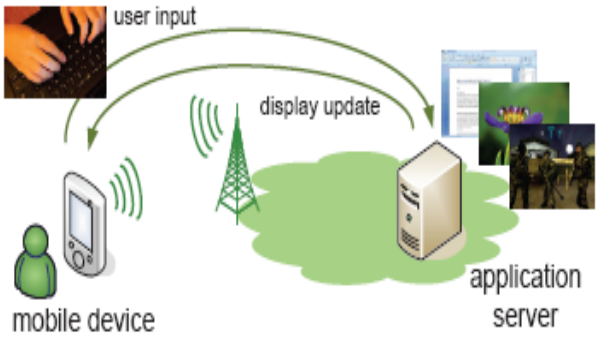


Figure.1. Description

We propose a virtual network computing (VNC) based architecture for accessing the desktops of remote computers from a cellular phone. A viewer is provided on the cellular phone that enables the user to see and manipulate the desktop of various remote systems such as MS Windows, Macintosh, and UNIX. The system to be accessed must be running a VNC server and it must be attached to a network. A proxy is used to send the image of the desktop to the cellular phone, to convert different devices, to suppress network traffics, and to support recovery from an unscheduled disconnection. To reduce user effort and solve problems inherent to the cellular phone's small screen, several functions are provided on the cellular viewer. Frequently used screen areas can be assigned and restored quickly by using the Shortcut function. The Guidance function can be used to show the current key assignments. Two areas can be viewed at the same time by the Twin view function. A prototype of the proposed architecture has been implemented using Java and has been tested on a Java-enabled cellular phone emulator.

5. Design Overview

Design Engineering deals with the various UML [Unified Modeling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product. Four Modules are to be followed,

1 RFB:

Remote Frame Buffer (RFB) is a simple protocol for remote access to graphical user interfaces. Because it works at the frame buffer level it is applicable to all windowing systems and applications, including X11, Windows and Macintosh. RFB is the protocol used in VNC (Virtual Network Computing). The remote endpoint where the user sits (i.e. the display plus keyboard and/or pointer) is called the RFB client or viewer. The endpoint where changes to the frame buffer originate (i.e. the windowing system and applications) is known as the RFB server.

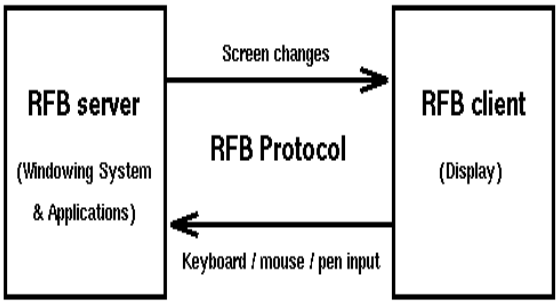


Figure.2. Remote Frame Buffer

2 Soft Grid Applications Streaming

This platform allows applications to be deployed in real-time to any client from a virtual application server. It removes the need for local installation of the applications. Instead, only the Soft Grid runtime needs to be installed on the client machines. All application data is permanently stored on the virtual application server. Whichever software is needed is streamed from the application server on demand and run locally.

The four files created and installed on the Soft Grid Application Server are accessed by the desktop. The result is the creation of a virtual application environment on the user's machine with the bare minimum of application components streamed into it. The result is a self-contained application runtime space that virtualizes the following components:

- Registry – registry changes unique to the application are not made to the main OS on the desktop. Rather, they are virtualized within the isolated application runtime space.
- File system – calls from the application for local disk access can be redirected to access DLLs and other components from a virtual file system.

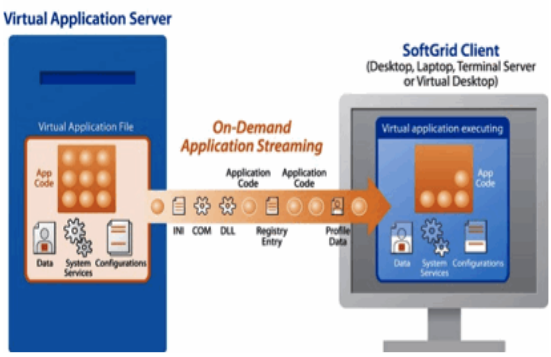


Figure.3.Soft grid Application Streaming

3 Downstream Data Peak Reduction

Interactive applications only update their display unless instructed by the user. Usually, these display updates involve a large amount of data that needs to be sent to the client in a short interval to swiftly update the display Moving or capable of moving with great speed. Their analysis of remote display protocol traffic traces reveals a lot of redundancy, caused by the repainting of graphical objects after recurring user actions. They propose a hybrid cache-compression scheme whereby the cached data is used as history to better

compress recurrent screen updates. The cache contains various drawing orders and bitmaps using Microsoft's Remote Display Protocol (RDP) and dependent on the size of the cache.

4 Optimization of Upstream Packetisation Overhead

The some network send in desecrate chunks call packets User events are the principal source of remote display traffic in the upstream direction from client to server. Individually, each user event embodies only a small amount of information: a key or button id, one bit to discriminate between the press and release action and possibly the current pointer coordinates. Nevertheless, user events induce important upstream traffic because they are often generated shortly after each other. Entering a single character results in two user events to indicate the press and release action, a large Packetization overhead is observed owing to the headers added at the TCP, IP and (wireless) link layer. The upstream Packetization overhead of three commonly used remote display protocols. The maximum buffering period is a consideration of remote display bandwidth reduction against interaction latency. The highest bandwidth reductions are achieved for interactive applications with frequent user events and lower round trip times.

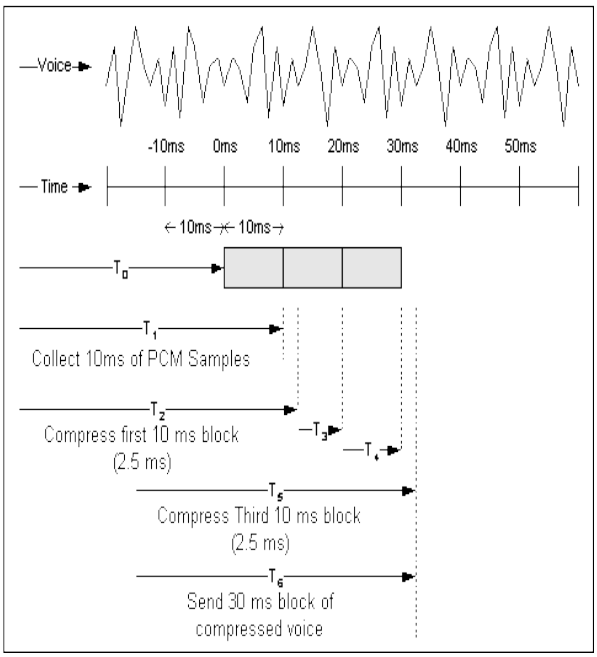


Figure.4.Optimization of Upstream Packetization Overhead

6. System Architecture

1. The Vnc-Based Architecture

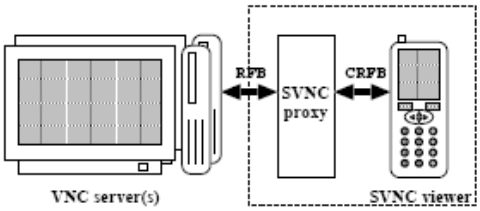


Figure 3. VNC-based architecture

To achieve the above-mentioned goals while at the same time considering portability and generality, we propose a VNC [1] based architecture. VNC is an implementation of a remote display system based on a Remote Frame Buffer (RFB) protocol [6].

2 Structure

Figure 3 depicts the VNC architecture. It consists of VNC servers running on one or more remote computers, a Smart VNC (SVNC) proxy, and a SVNC viewer on a cellular phone. A VNC server sends a remote desktop display as bitmap images in RFB protocol. A SVNC proxy converts (crops, shrinks) the display image and then transfers the converted image to a SVNC viewer in response to a user request that was received from that SVNC viewer. The transfer is performed in our own Compact RFB (CRFB), our simplified RFB protocol. Then, the SVNC viewer displays the transferred images. Key events received by the SVNC viewer are transmitted to a SVNC proxy that coverts them and sends them to the server. When the user first tries to connect to a remote computer, he must specify his user name and password for authentication as well as the host name of the computer that is running a VNC server. If authentication succeeds, the SVNC proxy establishes a session with the VNC server and the SVNC viewer starts user services[1]. To suppress network traffic, encoding is changed depending on contexts. Usually, colored display images are transferred from the SVNC proxy to the SVNC viewer. However, while the user is manipulating the remote desktop, such as scrolling and moving the pointing device, the display images are gray-scaled to reduce the number of bytes required to encode the image.

3 Maintaining The States Of A Session

In order to recover quickly from an unscheduled disconnection, the SVNC proxy maintains its own database containing each session's unique information such as the user name, the password, the target host name, and other internal states. When it is first connected to a SVNC viewer, the SVNC proxy searches its own database using the user name and the target host name pair as a key. It determines whether or not there has been any previously established session. If such a session did exist, the proxy restores the stored states. A session's state in the proxy is discarded when the user explicitly terminates it on the viewer.

4 Benefits From The Architecture

The VNC protocol is an image-based protocol in which updates to a screen by applications are captured. Therefore, we can manipulate the applications running on the

remote system by browsing the same image that we would be browsing if we were sitting at the remote computer. We can utilize the general availability of VNC servers[2]. VNC is becoming widely available as an infrastructure for controlling remote computers and for linking home appliances with PCs due to the portability of the RFB protocol.

5 RFB (Remote Frame Buffer)

RFB ("remote frame buffer") is a simple protocol for remote access to graphical user interfaces. Because it works at the frame buffer level it is applicable to all windowing systems and applications, including X11, Windows and Macintosh. RFB is the protocol used in VNC (Virtual Network Computing). The remote endpoint where the user sits (i.e. the display plus keyboard and/or pointer) is called the RFB client or viewer. The endpoint where changes to the frame buffer originate (i.e. the windowing system and applications) is known as the RFB server.

RFB is truly a "thin client" protocol. The emphasis in the design of the RFB protocol is to make very few requirements of the client. In this way, clients can run on the widest range of hardware, and the task of implementing a client is made as simple as possible. The protocol also makes the client stateless. If a client disconnects from a given server and subsequently reconnects to that same server, the state of the user interface is pre- served. Furthermore, a different client endpoint can be used to connect to the same RFB server[3]. At the new endpoint, the user will see exactly the same graphical user interface as at the original endpoint. In effect, the interface to the user's applications becomes completely mobile. Wherever suitable network connectivity exists, the user can access their own personal applications, and the state of these applications is preserved between accesses from different locations. This provides the user with a familiar, uniform view of the computing infrastructure wherever they go.

6 Display Protocol

The display side of the protocol is based around a single graphics primitive: "put a rectangle of pixel data at a given x,y position". At first glance this might seem an inefficient way of drawing many user interface components. However, allowing various different encodings for the pixel data gives us a large degree of flexibility in how to trade off various parameters such as network bandwidth, client drawing speed and server processing speed.

A sequence of these rectangles makes a frame buffer update (or simply update). An update represents a change from one valid frame buffer state to another, so in some ways is similar to a frame of video. The rectangles in an update are usually disjoint but this is not necessarily the case.

The update protocol is demand-driven by the client. That is, an update is only sent from the server to the client in response to an explicit request from the client. This gives the protocol an adaptive quality. The slower the client and the network are, the lower the rate of updates becomes. With typical applications, changes to the same area of the frame buffer tend to happen soon after one another. With a slow client and/or network, transient states of the frame buffer can be ignored, resulting in less network traffic and less drawing for the client.

7 Input Protocol

The input side of the protocol is based on a standard workstation model of a keyboard and multi-button pointing device. Input events are simply sent to the server by the client whenever the user presses a key or pointer button, or whenever the pointing device is moved.

These input events can also be synthesized from other non-standard I/O devices. For example, a pen-based handwriting recognition engine might generate keyboard events.

8 Representation Of Pixel Data

Initial interaction between the RFB client and server involves a negotiation of the format and encoding with which pixel data will be sent. This negotiation has been designed to make the job of the client as easy as possible.

The bottom line is that the server must always be able to supply pixel data in the form the client wants. However if the client is able to cope equally with several different formats or encodings, it may choose one which is easier for the server to produce.

Pixel format refers to the representation of individual colors by pixel values. The most common pixel formats are 24-bit or 16-bit "true color", where bit-fields within the pixel value translate directly to red, green and blue intensities, and 8 - bit "color map" where an arbitrary mapping can be used to translate from pixel values to the RGB intensities.

Encoding refers to how a rectangle of pixel data will be sent on the wire. Every rectangle of pixel data is prefixed by a header giving the X,Y position of the rectangle on the screen, the width and height of the rectangle, and an encoding type which specifies the encoding of the pixel data. The data itself then follows using the specified encoding. The encoding types defined at present are Raw, Copy Rect, RRE, Hextile and ZRLE.

In practice we normally use only the ZRLE, Hextile and CopyRect encodings since they provide the best compression for typical desktops.

9 Protocol Extensions

There are a number of ways in which the protocol can be extended.

They are

- **New Encodings**

A new encoding type can be added to the protocol relatively easily while maintaining compatibility with existing clients and servers. Existing servers will simply ignore requests for a new encoding which they don't support. Existing clients will never request the new encoding so will never see rectangles encoded that way.

- **Pseudo Encodings**

In addition to genuine encodings, a client can request a "pseudo-encoding" to declare to the server that it supports a certain extension to the protocol.

A server which does not support the extension will simply ignore the pseudo-encoding. Note that this means the client must assume that the server does not support the extension until it gets some extension-specific confirmation from the server.

10 New Security Types

Adding a new security type gives the ultimate edibility in modifying the behavior of the protocol without sacrificing compatibility with existing clients and servers. A client and server which agree on a new security type can effectively talk whatever protocol they like after that - it doesn't necessarily have to be anything like the RFB protocol.

11 Protocol Messages

The RFB protocol can operate over any reliable transport, either byte-stream or message- based. Conventionally it is used over a TCP/IP connection. There are three stages to the protocol.

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 Client in it and Server Init messages.

The final stage is the normal protocol interaction. The client can send which ever messages it wants, and may receive messages from the server as a result. All these messages begin with a message-type byte, followed by any message-specific data.

1 Sequence Diagram

A sequence diagram in UML is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a message sequence chart. Sequence diagrams are sometimes called Event-trace diagrams.

2 Use Case Diagram

A use case diagram is a type of behavioral diagram created from a Use-case analysis. The purpose of use case is to present overview of the functionality provided by the system in terms of actors, their goals and any dependencies between those use cases.

In the below diagram eleven use cases are depicted. They are used to search result using CST methods. The use case diagram can be seen in the figure6.

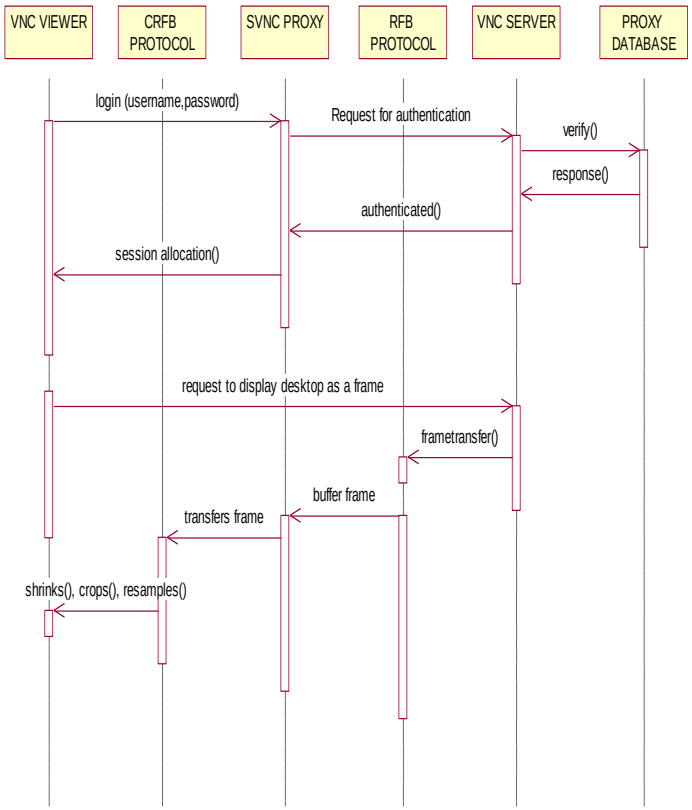


Figure.5.Sequence Diagram

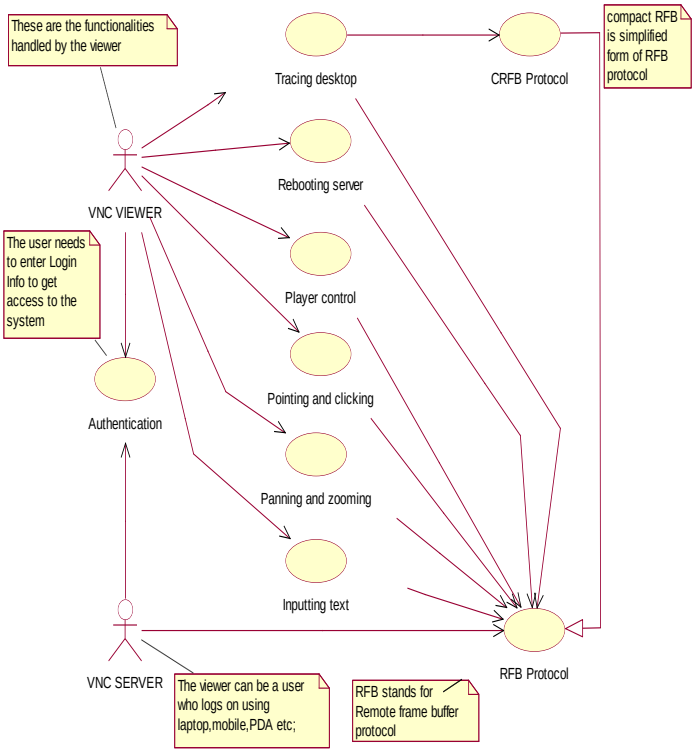


Figure.6. Use Case Diagram

Collaboration Diagram

A collaboration diagram show the objects and relationships involved in an interaction, and the sequence of messages exchanged among the objects during the interaction. The collaboration diagram can be a decomposition of a class, class diagram, or part of a class diagram. It can be the decomposition of a use case, use case diagram, or part of a use case diagram.

The collaboration diagram shows messages being sent between classes and object (instances). A diagram is created for each system operation that relates to the current development cycle (iteration).

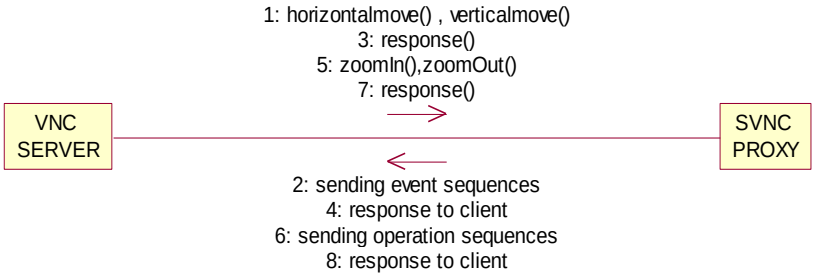


Figure.8.Collaboration diagram

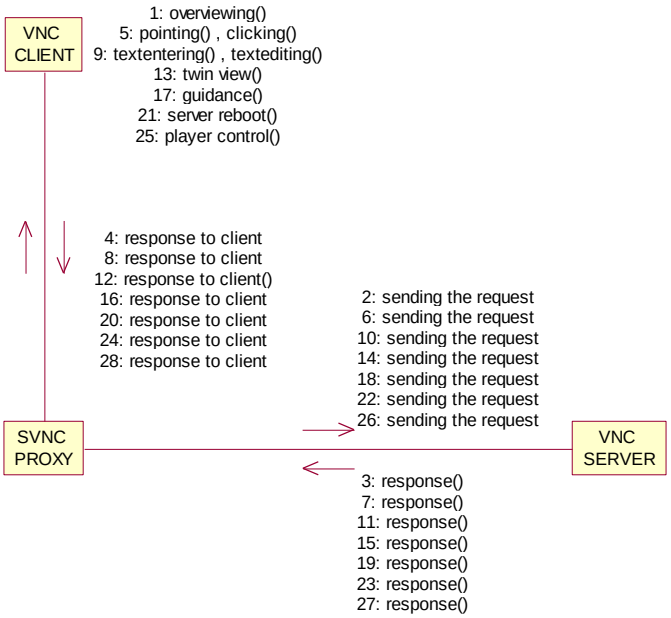


Figure.7.Collaboration Diagram

Class Diagram

A class diagram in the UML is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes. Private visibility hides information from anything outside the class partition. Public visibility allows all other classes to view the marked information.

Protected visibility allows child classes to access information they inherited from a parent class A with three sections. The upper part holds the name of the class. The middle part contains the attributes of the class. The bottom part gives the methods or operations the class can take or undertake. The login and the admin parts can be seen in the figure 4.6.

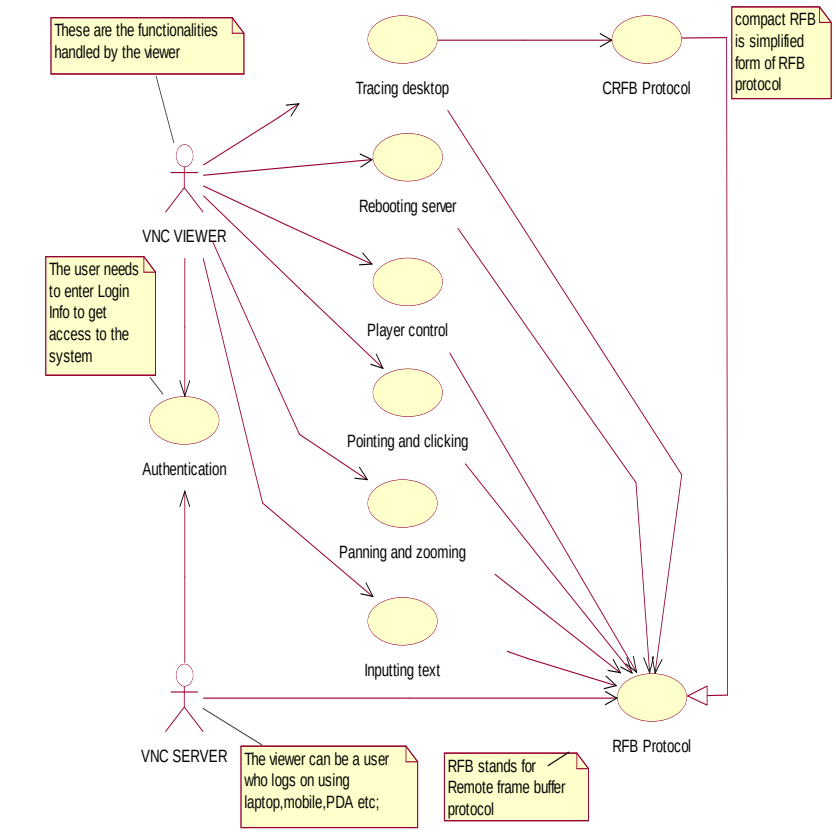


Figure.9.Class Diagram

6. Experimental Results

GENERAL METHODOLOGY

We implemented the proposed architecture including the SVNC proxy and the SVNC viewer using Java.

Platform

The SVNC proxy was implemented by modifying the Java version of the VNC viewer released by AT&T Lab-oratories, Cambridge. The proxy runs as a servlet on an HTTP server with the servlet API. We are currently using Apache Tomcat 4.0 2 as the HTTP server. We have installed the proxy on a PC with a Windows 2000 work-station.

The SVNC viewer has been implemented using the J2ME Wireless SDK released by NTT DOCOMO. The code of the SVNC viewer has been placed on the HTTP server of the SVNC proxy and is downloaded in response to requests from the cellular phone. We have tested our viewer using a real cellular phone and have successfully ACCESSED REMOTE COMPUTERS.

Handshakes In Crfb Protocol

The viewer periodically requests the SVNC proxy to send the desktop display of the remote computer as a frame. This polling action is required to ensure that the Java running on

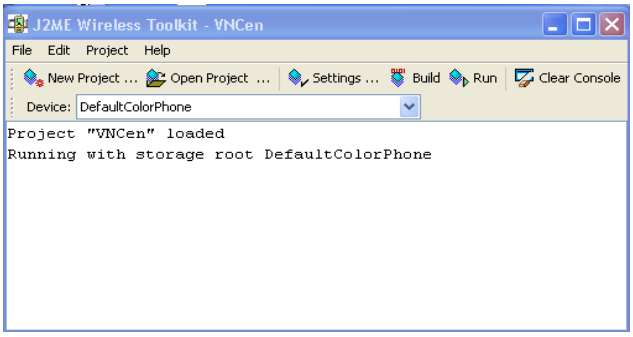
a NTT DOCOMO device follows the definition of Doja3 This definition requires that applications on a cellular phone must explicitly send a request to the proxy to start communication. Moreover, the proxy can only return one message in response to one request from the viewer. For each frame, the viewer sends the position and size of the desired viewport with its zoom level. It should be noted that the proxy can generate a frame by shrinking the original image with anti-aliasing depending upon the zoom level. Usually, bitmaps transferred from the proxy to the viewer are encoded in 120x130x8 bits with compression. However, during scrolling and dragging, bitmaps are gray-scaled into 120x130x3 bits with compression.

Pointing and clicking mouse buttons are achieved by the translation of these events on the proxy side. When the proxy accepts a request from the viewer, it generates corresponding event sequences and sends the sequences to the VNC server.

SCREEN SHOTS

Snapshot is nothing but every moment of the application while running. It gives the clear and elaborated application. It will be useful for the new user to understand for the future steps.

Various Vnc Screen Shots



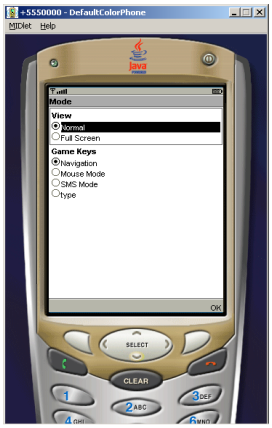
OPENING THE PROJECT IN J2ME5.3.1.1 MOBILE NODE: 01 LAUNCHING THE CONNECTION

CONNECTING MOBLE CLIENT TO THE REMOTE SERVER

3 MOBILE NODE: 03

TO CONNECT THE MODE

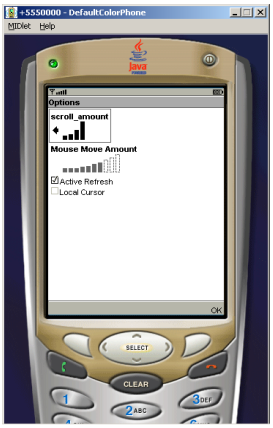
4 MOBILE NODE: 04



VIEW THE DESKTOP IN NORMAL VIEW
5 MOBILE NODE: 05

FROM MENU SELECT THE OPTIONS

6 MOBILE NODE: 06

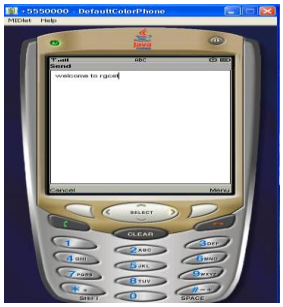


CHANGE SPEED USING OPTIONS

7 MOBILE NODE: 07

SELECT THE TEXT MENU

8 MOBILE NODE 08



8. CONCLUSION

By physically separating the user interface from the application logic, the principle of mobile cloud computing allows to access even the most demanding applications in the cloud from intrinsically resource-constrained mobile devices. In this article, we have surveyed contemporary remote display optimization techniques specifically tailored to the short mobile

device battery lifetime, the varying and limited bandwidth availability on wireless links and the interaction latency.

Although each of these solutions adequately addresses specific challenges of mobile cloud computing, an overall approach is currently lacking. The context of mobile cloud computing is highly dynamic, owing to the user mobility, the wide diversity of applications, and the varying wireless channel status. Future research should therefore be devoted to the design of an overall framework, integrating all the presented solutions, and activating the most appropriate solutions dependent on the current device, network and cloud server status.

Figure.10.Accessing system through mobile

REFERENCES

[1] Pendyala V. S. and Shim S. S. Y., "The Web as the Ubiquitous Computer," COMPUTER, vol. 42, no. 9, pp. 90–92, SEP 2009.

[2] Lamberti. F and Sanna. A, "A streaming-based solution for remote visualization of 3D graphics on mobile devices," Ieee Transactions On Visualization And Computer Graphics, vol. 13, no. 2, pp. 247–260, MAR-APR 2007.

[3] H. Kawashima, K. Koshiba, K. Tuchimochi, K. Futamura, M. Enomoto, and M. Watanabe, "Virtual PC-type thin client system" Nec Techni-Cal Journal, vol. 2, no. 3, pp. 42–47, SEP 2007.

[4] P. Simoens, F. Ali. A, Vankeirsbilck. B, Deboosere. L, De Turck. F, Dhoedt. B, Demeester. P, and Torrea-Duran. R, "CrossLayer Optimization of Radio Sleep Intervals to Increase Thin Client Energy Efficiency" IEEE Communications Letters, vol. 14, no. 12, pp. 1095–1097, DEC 2010.

[5] Tan K. J, Gong J. W, Wu B. T, Chang D. C, Li H. Y, Hsiao Y. M, Chen Y.-C,. Lo S. W, Chu Y. S and Guo J. I., "A remote thin client system for real time multimedia streaming over VNC" in 2010 IEEE International Conference on Multimedia and Expo (ICME), 2010, pp. 992–7.

[6] Mitrea. M, Simoens, P. Joveski. B, Marshall. J, Taguengayte. A, Pre-teux. F and Dhoedt. B, "BiFS-based approaches to remote display for mobile thin clients" in Proceedings of the SPIE - The International Society for Optical Engineering, vol. 7444, 2009, p. 74440F (8pp.).

[7] Paravati. G, Celozzi. C, Sanna. A and Lamberti. F, "A Feedback-Based Control Technique for Interactive Live Streaming Systems to Mobile Devices," IEEE Transactions on Consumer Electronics, vol. 56, no. 1, pp. 190–197, FEB 2010.