12-19-2003

Creating a Portable Wireless Display

Srivatsa Gundala
University of New Orleans

CREATING A PORTABLE WIRELESS DISPLAY

A Thesis

Submitted to the Graduate Faculty of the
University of New Orleans
in partial fulfillment of the
requirements for the degree of

Master of Science
in
The Department of Computer Science

by

Srivatsa J Gundala

M.Sc, Andhra University, India, 2000

December 2003

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I would like to express gratitude to Dr. Golden Richard III, my advisor for his support, encouragement and sharing his knowledge with me. It was a great experience in working with both the software and hardware aspects of the prototype.

I would like to thank Dr. Adlai DePano and Dr. Ming Hsing Chiu for being on my committee. I am infinitely indebted to my parents for giving me the very best of their values. I am grateful to my family for encouraging and supporting me. The city of New Orleans gave me a number of good friends, to who I am thankful.

Finally, I would like to thank Rupa, my wife and my best friend for always being there for me and believing in me.

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Real time computing has become a vital part in military applications. Moreover certain operations require that the soldiers carry computing devices to assist them. These devices, besides providing them with location-based information, should also be transmitting the requested data. In this thesis, we present a portable wireless display prototype, which renders the desktop of a remote computer.

The prototype functions under the range of an 802.11b or Bluetooth wireless network. The Software interfacing is done with Virtual Network Computing (VNC). This thesis is a first step towards analyzing and creating head/wrist mounted displays capable of transmitting images from a remote computer. The thesis starts with an overview and proceeds with a discussion on the concepts involved behind the functioning of the prototype. It then provides a detailed description of the how the prototype was built, followed by a performance test and its analysis and concludes by summarizing the results achieved.

Chapter 1. Introduction

The need for computers on the go has led to many innovative inventions and has given rise to an entire different perspective on the once huge and immobile computers. Personal computers are available in different sizes depending on their usages. There is a significant use of the personal computers in the defense sector. Embedded computers are used in numerous military applications like unmanned combat vehicles, missiles, GPS units, etc. They are used by the soldiers for gathering mission critical information. The computers or gadgets with embedded computers have become de facto equipment for the soldiers. A probable and easy way of carrying such devices would be to 'wear' them on the body. The computer is patched or attached to the soldier's uniform and a display is provided through a small LCD close to the soldier's eye.

In this thesis, we present a custom built wireless display prototype that can be used to view the desktop of a remote computer

1.1. Background

This project is an initial attempt towards analyzing and creating head/wrist mounted wireless display. The creation of the prototype is a proof of concept that such technologies can be created. The use of the prototype is for real time screen updates from a remote computer. A soldier on a mission needs to access critical data regarding his geographical position as well as the data and the layout of the surroundings. A notebook or a laptop computer would be useful, but it may not be an ideal tool, especially when the soldier should not be visible to the enemy or may not be in a position to input any data into the computing device.

An ideal device would be a display that a soldier can view without utilizing much of his resources. The prototype presented here displays the screen of the laptop computer in soldiers backpack via a wireless network.

1.2. Objectives

The objective of this wireless display prototype is to analyze the feasibility of creating a wrist and head mounted displays. The prototype that we present now proves that such displays can be built and has etched a path for their creation. The objective is to create a handheld or a wrist mounted wireless display that would render the contents of a remote computer. The prototype has to be rugged, safe, reliable, inexpensive (throwaways), small, lightweight (portable/wearable) remote electronic displays that consume little power (=0.5W to maximum of 1W) and have high resolution (SVGA/XVGA, with 32k colors). The prototype is to be powered by internal rechargeable batteries that last a minimum of 6 hours and may last up to 12 hours. An external DC input is used to run with an external power source and/or to charge the internal battery.

The prototype should be portable and easily carried around without much effort. The network medium should not be of a long range, as it would result in the easy detection of the person carrying the prototype. The prototype functions within a specific range and it should not go beyond certain physical boundaries from the remote computer. The prototype should be able to render static images and text on the remote computer with a decent update (refresh) rate.

1.3. Overview

The rest of the chapters are organized as follows: Chapter 2 is organized into three major sections. (i) In the first section, we will discuss in detail about the basics of wireless computing and its applicability. (ii) In the next section we will discuss in detail the features of an 802.11b wireless network. (iii) In the final section we will discuss about VNC. Chapter 3 will focus on the architecture and design of the Prototype. Chapter 4 contains the implementation of the

Prototype. Chapter 5 addresses the performance of the Prototype under different scenarios and gives us an outline of what can be expected of such wireless displays. Finally, we conclude and suggest further enhancements in Chapter6.

Chapter 2. Concepts of the Wireless Display Prototype

The wireless display prototype that we present is a tool that enables the user to view the desktop of a remote computer. This chapter explains the concepts that are behind the functioning of the prototype.

2.1. Wireless Networks

This key concept here is to transfer data between the remote computer and the prototype using a wireless network. Wireless networks eliminate the need for any possible cable connection between the client and the server. This gives mobility to the client, in our case the prototype. Though it provides mobility there are other associated trades-offs with the wireless networks like the bandwidth is reduced considerably as compared to its wired counterpart. Other issues like maintaining a constant connection with the client and a guaranteed Quality of Service are also not possible. Security also plays an important role in any application and especially in wireless networks additional measures have to be taken to implement it successfully. In this section we discuss briefly about the wireless networks and their characteristics.

A wireless network can be defined as a communication system that transmits and receives data using electromagnetic waves as its media [1]. There have been several technologies in use.

1. Local Area Network
2. Personal Area Network
3. Metropolitan Area Network
4. Wide Area Network

2.1.1. Wireless Local Area Networks (WLAN's)

These are the networks that use electromagnetic waves and make use of the spread-spectrum technology based on the radio waves, to transfer information between devices in a limited area. These can be further classified into two types:

Infrastructure WLAN's

These comprise of WLAN's connected to wired networks and contain access points to channel the network traffic.

Independent WLAN's

These comprise of a peer-to-peer network of devices connected together without the presence of a central server for administrative purpose.

WLAN Technology Options

Spread Spectrum

Spread-spectrum technology is widely used in the wireless networks. This is a wideband radio frequency initially developed by the military for use in reliable and secure systems. More bandwidth is consumed here than in the case of narrowband transmission, but this produces a signal, which is louder and easier to detect. It is assumed that the receiver knows the parameters of the spread-spectrum signal being broadcast. In case the receiver is not in the correct frequency, a spread-spectrum signal is like background noise. There are two types of spread spectrum radio: frequency hopping and direct sequence.

Frequency-hopping spread-spectrum (FHSS) is a technology where the data signal is modulated with a narrowband carrier that has a predetermined frequency pattern that is known both to the transmitter and receiver. The whole idea is to maintain a single logical channel. This

can be achieved if the transmitter and the receiver are properly synchronized. For an eavesdropper, FHSS seems like noise.

Direct-sequence spread-spectrum (DSSS) generates a redundant bit pattern for each bit to be transmitted, called a chipping code. This bit pattern increases resistance against interference. The lost data can be recovered by the redundancy of the signal. For an eavesdropper, DSSS appears as low-power wideband noise.

Narrowband Technology

This technology is used for transmitting information on a specific radio frequency. Narrowband radio keeps the radio signal frequency as narrow as possible just to pass the information. Since different users are on different channel frequencies, cross talk between communications channels is avoided.

802.11 is the IEEE approved standard for WLAN's. There are three popular technologies in existence:

a. 802.11 a (54 Mbps over 5 GHz Range)
b. 802.11 b (11 Mbps over 2.4 GHz Range)
c. 802.11 g (22 Mbps over 2.4 GHz Range)

We will discuss more about 802.11b in section 2.2 as this is the most suitable technology for the deployment of the prototype. The equipment cost is relatively low compared to the other two variations of the 802.11 and its signal comes in the range of the specifications mentioned in section 1.2 of chapter 1.

2.1.2. Wireless Personal Area Network (WPAN's)

WPAN's allow the users to establish ad-hoc [2], wireless communication for devices like PDA's, cellular phones and laptops. These are effective up to a distance of 10 meters. IEEE working group for the WPAN's is the 802.15. The current working technologies are:

Bluetooth Technology

This technology is a non-expensive, short-range transmission of radio signals to achieve point-to-point communication and to establish ad-hoc networks [3]. This, in effect, makes the digital devices commonly used by us free of cables. Bluetooth can be used to access the Internet, synchronize data or even unlock a car. This makes use of the FHSS signal to connect devices using radio signals. The effective range is 30 feet (10 meters) through objects and the transfer rates are close to 1Mbps. Another characteristic to mention of the Bluetooth is that the bandwidth is shared with other Bluetooth enabled devices. Nevertheless, this literally gets rid of the cables present with the Bluetooth enabled devices. This is another technology that fits into the requirements mentioned in section 1.2 of chapter 1.

Infrared Technology

This technology needs a clear line of sight to make a connection. The Infrared Data Association (IRDA) standard 1.1 supports data transfers of 1.15 and 4 Mbps. Today, most of the new PDA's and laptops come equipped with the IRDA ports as a standard feature.

2.1.3. Wireless Metropolitan Area Networks (WMAN's)

The WMAN's enable users to establish wireless connection between multiple locations within a metropolitan area like a wireless connection between multiple office buildings in a city. WMAN's can also be used as a backup for the traditional wired networks. These use the radio waves or the infrared light to transmit data. IEEE 802.16 working group is developing specifications for Broadband wireless access. Examples include the Microwave networks, Laser Networks and Ricochet by Metricom.

Microwave Networks

Typical microwave networks have a range of 20 miles and require that there is a line of sight. They provide the 100 Mbps Ethernet speed and have an operating frequency of 7MHz to 38MHz. These are useful in creating private WMAN's.

Laser Networks

These networks work best within a range of 2-4 miles. Best suited for industries. The laser network is susceptible of being disrupted by fog. These also require a line of sight.

Ricochet by Metricom

Ricochet [4] is a high-speed Internet solution for a metropolitan area. It uses shoebox size, solar powered units mounted on the streetlights. The end-user needs a custom modem, available as internal or external. The modem could be connected to a Desktop, notebook or a PDA. This service is now limited to particular areas in San Diego and Denver. One good thing about the signal is that it is radio signal and it penetrates through the walls, buildings and other obstacles. The speed is said to be around 128 Kbps and this uses the FHSS technology.

Boingo Wireless Network

Unlike Ricochet, this is a wireless network service available in over 1900 locations spreading over 12 countries. In the US it covers 45 states and ~570 cities. It has plans for the existing ISP's to provide wireless access [5]. Apart from this it is available in most of the hotels and coffee shops.

2.1.4. Wireless Wide Area Networks (WWAN's)

The WWAN's enable a much longer range of networking than the WMAN's and the WLAN's. These provide the end-user more mobility than the other technologies. Radio signals

are used over analog, digital cellular or PCS networks. The WWAN's may also utilize the microwave signal or the electromagnetic waves. . Until now 2G (2nd Generation) technologies like the CDMA, PDC, and GSM were being used and available bandwidth for data-intensive applications was insufficient. With the advent of 3G (3rd Generation) technologies, sufficient bandwidth will be available to support wireless applications. However, we do not consider 3G further because our wireless prototype must operate in areas where the infrastructure for 3G wireless WANS is not present. Further, use of 3G networks is still relatively expensive.

2.2. Final Contenders

In the wireless technologies discussed above WLAN and WPAN are the closest to the requirements listed in section 1.2 of chapter 1. WMAN and WWAN have a long range and some even require a clear line of sight. 802.11b and Bluetooth emerge as the final choice for networking as their implementation costs are much low. Apart from this the signal range in which they are used aptly suits the prototype development.

2.3. 802.11b Wireless Network

This is the first major revision of the 802.11 standard approved by IEEE in 1999 [6]. The features of 802.11b are summarized below:

Operating Frequency	2.5 GHz ISM (Industrial, Medical, Scientific) Band
Transfer Rate (Theoretical)	1, 2, 5.5, 11 Mbps
Transfer Rate (Throughput)	4 Mbps (average)
Mechanism	DSSS
Channels Available	11 (partially overlapping, 3 [1,6,11] non-overlapping)
Maximum Range	175 feet

Table 2.1. Characteristics of an 802.11b network

The older version of this used the 1Mbps transmission. The FHSS mechanism was used with this. By the year 2000, 802.11b became the standard in Ethernet wireless technology for both business and home. There were many custom products in the market that were supporting the 802.11b. For the interoperability of these the WiFi organization was created. A realistic bandwidth around 2 - 4 Mbps was proving good for both the business and the home sector.

The equipment necessary to establish an 802.11b network consists of a wireless router, access points (AP) and the network interface cards (NIC). All these collectively are available for under $100.

IEEE 802.11 has two operating modes: Infrastructure mode and Ad hoc mode. The infrastructure mode was discussed in section 2.1.1. In ad hoc mode, also known as peer-to-peer mode, wireless clients communicate directly with each other (without the use of a wireless AP). Two or more wireless clients who communicate using ad hoc mode form an Independent Basic Service Set (IBSS). Ad hoc mode is used to connect wireless clients when a wireless AP is not present. This mode is suitable for our prototype as there may not be any possible AP when the soldiers operate in an open field.

A single wireless AP that supports one or multiple wireless clients is known as a Basic Service Set (BSS). A set of two or more wireless APs that are connected to the same wired network is known as an Extended Service Set (ESS). An ESS is a single logical network segment (also known as a subnet), and is identified by its Service Set Identifier (SSID). If the available physical areas of the wireless APs in an ESS overlap, then a wireless client can roam, or move from one location (with a wireless AP) to another (with a different wireless AP) while maintaining Network layer connectivity.

If the access points are placed in strategic positions, then the signal could be maintained uniformly throughout a building [7]. The nature of the building does have an impact on the transmission. The signal considerably comes down if it needs to pass through concrete or brick

walls. This should not affect the performance as the prototype should not be more than 50 feet. Below is a table that summarizes the signal and speed depending on the environment:

Environment	**11 Mbps**	**5.5 Mbps**	**2 Mbps**
Completely open environment	*525 ft*	*885 ft*	*1300 ft*
Semi-open environment	*165 ft*	*230 ft*	*300 ft*
Closed environment (Brick)	*80 ft*	*115 ft*	*130 ft*

Table 2.2. Range and Speed of 802.11b network in different environments

The Access points periodically transmit beacons for any possible client to connect to it. The mobile host, through its NIC, sends a probe request for synchronization information. The Access point responds with synchronization information. This forms an association of the mobile host with the access point. The mobile host may disassociate with the access point and then may associate itself with another access point, while on the move. The mobile hosts NIC also searches for an access point that has got a stronger signal.

Security is the major drawback of wireless LANs. Basic security can be provided by WEP. This is sufficient to start with, as long as the security features are enabled and the NIC support 128-bit encryption. However, WEP is only basic security and is not sufficient for any classified or mission critical data.

Additional, sophisticated devices from Cisco Systems and 3Com are required to allow more reliable authentication. These solutions are implemented by using a combination of more advanced Access Points and a RADIUS server that maintains a list of the authorized MAC addresses that are permitted to log into the WLAN.

Presently the Access Points allow the WLAN to be enabled without requiring that encryption be turned on. In addition, many Access Points allow their Service Set Identifier (SSID) code to be broadcast frequently, so outsiders may be able to gain access to the WLAN. Proper configuration can provide a reasonable level of security, but care must be taken to implement and use security options consistently.

2.4. Virtual Network Computing (VNC)

Virtual Network Computing (VNC) is a remote display system which allows viewing of 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.

The VNC software is completely free. VNC can be used to perform administrative tasks from a remote terminal with an actual view of the desktop on which the operations are done. It is portable on Windows, UNIX and mixed network environments. There are a number of third party utilities and packages that are compatible with VNC. This eliminates the need of physical presence.

The software comes in a bundle of a server and a viewer. The machine that needs to be operated remotely contains the server software and the machine that tries to access the server should have a viewer installed on it. The viewer, facilitating a realistic view of the remote machine's desktop, handles cursor handling locally.

The concept of the VNC protocol is based on the Remote Frame Buffer (RFB) protocol [8]. This protocol allows a server to update the buffer to be displayed on a viewer. The aforementioned platform independence is achieved by the protocol working at the framebuffer level.

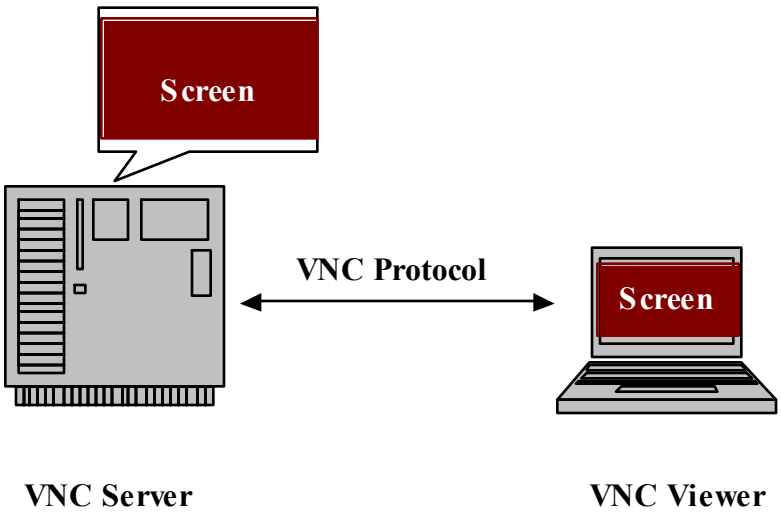

Figure 2.1. VNC Architecture

The protocol is portable to any device that has something in the shape of a communications link. The protocol will operate over any reliable transport like the TCP/IP.

Displaying pixel data on the viewer is based on the principle of putting a rectangular pixel of data at a given (x,y) coordinate. There are different schemes available for encoding pixel data. These can be used effectively for every rectangle sent in order to achieve maximum results with the network bandwidth, processing power of the server and the rendering capabilities of the client.

One example of encoding is the raw encoding, where the pixel data for a rectangle is sent in the left to right order. This is a basic encoding scheme and all the VNC servers and clients should support this encoding. Another example is a copy-rectangle encoding, which is used when the client has already a copy of the buffer that the server is supposed to send. Instead, the server sends the (x,y) coordinates of the buffer. This is very useful when a window is moved or

is scrolled, thus eliminating the need for resending a rectangle buffer and causing less flow of traffic in the network.

The client can be written in such a way that the encodings can be requested when the client requires them. For example, if the client does not have the capability to recall already sent buffers it can be written in such a way that it does not ask for a copy-rectangle encoding. Another method for optimizing is based on the fact that a typical desktop comprises of large areas of colors and text. By categorizing these into sub-rectangles of different colors and on the background color, optimized data can be sent to the client, resulting in less network traffic.

A framebuffer update contains a set of rectangles of pixel data. An update is the transition of a framebuffer from one state to another, causing a change in the rectangles. The server has the capability to choose from different encodings available for each of the changed rectangle. This depends on the content of the screen being sent and the available network bandwidth. Usually the update protocol is requested by the client. That is, for the server to send an update the client has to explicitly ask for it. The server puts together all the screen changes from the last client update and sends it to the client. Depending on the network bandwidth this appears different on a fast and a slow network connection. On a fast network connection, if a window is dragged it is seen moving smooth in a continuous fashion. On a slow connection, the movement of the window is seen in a non-continuous motion with a few stops towards its destination.

A number of compression algorithms are supported in the TightVNC [9], a variant of VNC, to provide even higher compression rates and a number of additional desirable features. Comparisons between VNC and TightVNC compression schemes are provided in Tables 2.1 and 2.2. The important issue observed is compression ratio vs. compression time. Table 2.1 shows results for a text session, while Table 2.2 is for a graphics-intensive session. The TightVNC team measured these results, using a reference platform consisting of a 350MHz Pentium II with 64MB RAM.

	Encodings					
	Raw	**Hextile**	**Zlib**	**Tight-1.0**	**Tight-1.1L**	**Tight-1.1**
Data size in screen updates, bytes	111,178,662	2,788,259	1,836,495	476,738	454,748	454,748
Bandwidth savings, Tight-1.1 vs. others	99.59%	83.69%	75.24%	4.61%	-	-
Compression time, seconds	-	5.2	231.5	29.7	21.5	21.3
Description:	This 16-bit-color session shows Xvnc compilation process started in xterm window maximized to occupy almost full screen. The window manager is IceWM.					
Explanations:	This test session is prepared in order to show Tight Encoder efficiency on compressing two-color (monochrome) screen areas. We see that compression ratio on such data may be four times (!) better than it stands for pure Zlib compression. Compression speed improvement is even more impressive: Tight encoder is about ten times faster on this session as compared to Zlib encoder. Also note raw data size: it's about 106 Mbytes, and it's only 444 Kbytes for Tight-1.1 encoding (compression ratio is 244.48). Such improvement in both compression ratio and speed is caused by replacing 16-bit true-color pixel values with 1-bit indices in 2-color palette. So the input stream size for final zlib compressor may be up to 16 times smaller. This allows zlib library to use its small (32K) dictionary much more efficiently. And compression speed is increased dramatically just because there is much less data to compress.					

Table 2.3. TightVNC results for a Text Session

	Encodings					
	Raw	**Hextile**	**Zlib**	**Tight-1.0**	**Tight-1.1L**	**Tight-1.1**
Data size in screen updates, bytes	25,569,676	16,781,057	10,657,746	9,830,852	9,966,649	6,130,479
Bandwidth savings, Tight-1.1 vs. others	76.02%	63.47%	42.48%	37.64%	38.49%	-
Compression time, seconds	-	1.6	24.1	14.6	14.1	35.3
Description:	Test session demonstrating intensive use of full-color graphics in 24-bit mode. It is IceWM desktop with photo image used as wallpaper. Another photo is viewed in a Gimp window.					
Explanations:	The results show great improvement (about 40%) in compression ratio when the "gradient" filter is used on suitable data. When color depth is 24 bits, this filter can give greatest compression improvements. But you can see that the cost we pay for that is much slower compression. The second issue seen in results above is that the Tight-1.1L encoder operates a little worse than Tight-1.0 implementation. It's a rare situation, but it's possible. The reason is difference in algorithms used to split large screen areas into smaller sub-rectangles. The algorithm used in 1.1 version typically operates better, but that is not the point for all possible situations.					

Table 2.4. TightVNC results for a Graphics Session

Enhancements provided by TightVNC over VNC (besides better compression) include local cursor handling, to improve interactive use of VNC, a JPEG compression scheme that degrades image quality to provide faster screen updates, and SSH tunneling. SSH tunneling is especially intriguing, because this provides for high quality encryption between the computer

generating the display and the wireless display itself. Since encryption of the wireless link may be required for the wireless display, we can make use of the SSH tunneling feature of TightVNC to enable encryption (if it is needed). Further implementation details and the compression schemes ported are discussed in section 4.1 of chapter 4.

Chapter 3. Architecture and Design

This chapter describes the architecture, design and the technology used in the making of the prototype. The Prototype can be broadly classified into three major units

- The processor required to handle the client software (IPAQ)
- The display on which the results from the server are to be displayed
- The Application software required to interface the VNC server

3.1. Technical Architecture

Architecture defines the broad outlines of the system and the underlying mechanisms. The system is composed of entities and the architecture depicts their interaction. The VNC software is present in a Client/Server format [10]. This model has its own advantages

- Centralization - access resources and data security are controlled through the server
- Scalability - any element can be upgraded when needed
- Flexibility - new technology can be easily integrated into the system
- Interoperability - all components (clients, network, servers) work together
- Accessibility - server can be accessed remotely and across multiple platforms
- Ease of application development

In this project we have simple 2-tier Client/Server architecture. The server is the VNC Server listening on a pre-defined port on the computer, which is to be remotely operated. The client is an application integrated with the drivers of the display card for displaying the results, fetched from the VNC server running on a remote machine, onto the LCD.

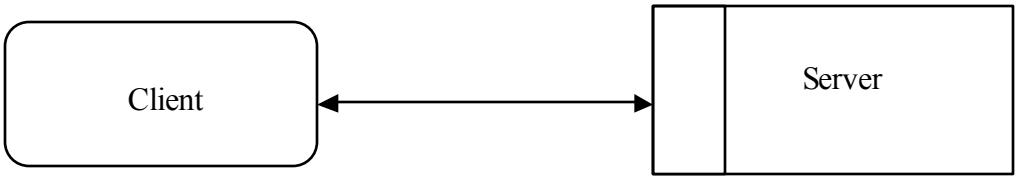

Figure 3.1. Client/Server Architecture

The rectangular box with the rounded corners represents the client and the solid rectangle represents the server. The server listens for the client on a specific port. The client connects to the server and the initial setup procedure is performed.

It may be noted here that the server can handle multiple clients. In our case, we have tested with a single client, but the server is multithreaded and can support more than one client.

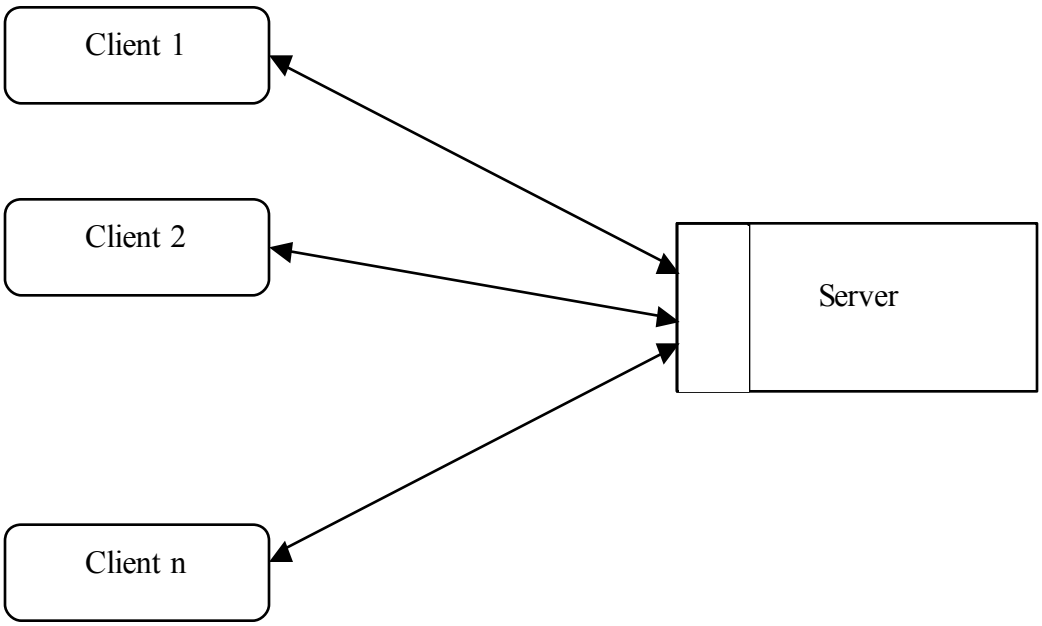

Figure 3.2. Multithreaded Client/Server Architecture

3.2. Functional Architecture

Functional Architecture outlines the way in which the components in a system interact. Figure 3.5 shows the details of the actions that happen when the prototype is powered on.

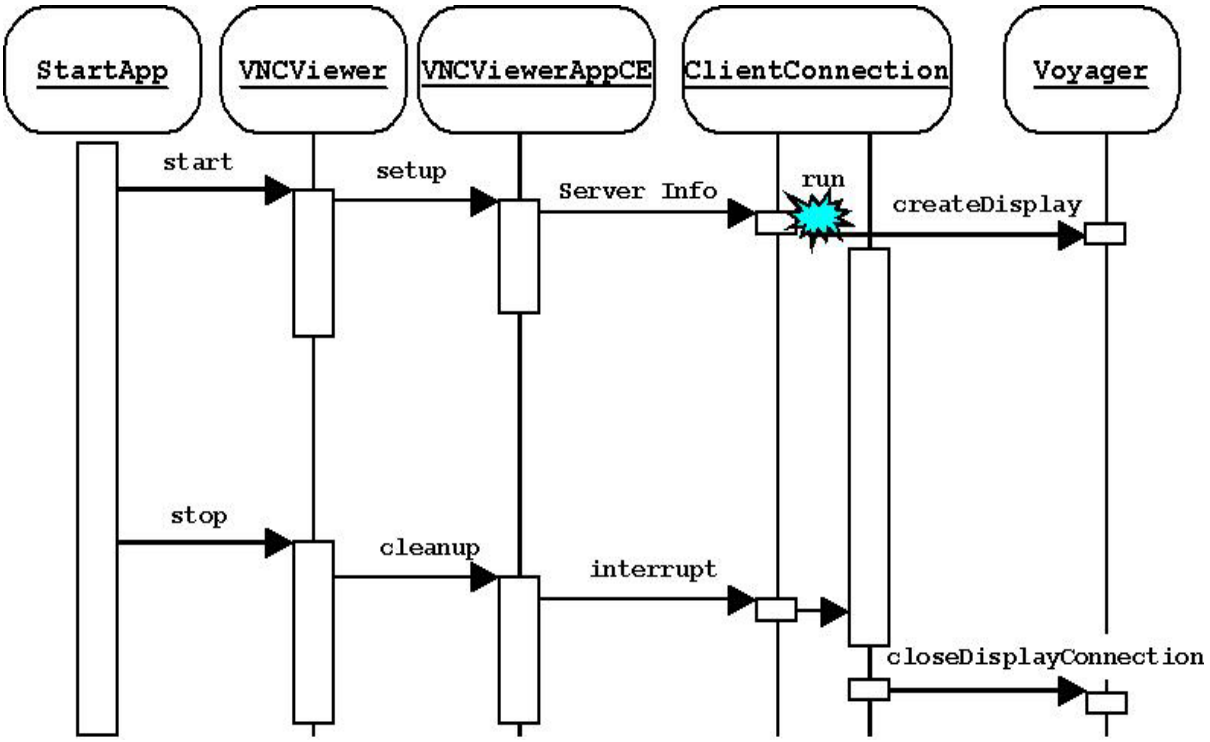

Figure 3.5. Functional Architecture of the Application

When the prototype is powered on, an initial application loads itself into memory and it loads the VNCViewer application. This is to make sure that the IPAQ runs only the VNCViewer application. After the VNCViewer gets loaded into memory, it waits for 10 seconds for the IPAQ to come up with the wireless Internet connection. The VNCViewer then calls the component VNCViewerAppCE, which contains the state of the application as a whole. This has the connection information of the server and tries to establish a connection the VNC Server.

From this point onward a new thread is created for every connection and is branched off from the main application. It is here that the coupling between the VNCViewer and the Voyager VGA CF card happens. The module ClientConnection makes sure that the display is routed to

the Toshiba LCD instead of the IPAQ's screen. When the application is ready, the initial desktop of the source computer is loaded. The prototype is switched off by the flipping off the switch.

In the next chapter we will discuss in detail about the implementation of the architecture.

Chapter 4. Implementation

This chapter discusses about how the theory on the prototype is put into practice. This forms the core of the thesis. The implementation can be divided into two parts.

- Software Implementation
- Hardware Implementation

Software implementation was done first as we did not have the required hardware in the initial stages. So, it is apt that we begin with it first.

4.1. Software Implementation

This software application combines already existing technologies to work with each other. The technologies being the VNC viewer for Windows CE and the Voyager VGA Compact Flash Card. VNC being an open source helped us a lot in hacking its code and putting in additional modules required for the functioning of the prototype. The Voyager VGA Compact Flash card was still in its development and there were very few applications that were written for it. The following were the technologies and components at the very beginning of the project:

- Source code of the VNC Viewer for Windows CE
- Device Drivers for the Voyager VGA CF Card
- Two custom built application by Colorgraphic for Voyager card, one to display symbolic graphics (rectangles, squares, lines, and bitmaps) and the other, Voyager shadow, to display the contents of the IPAQ when the Voyager card was

connected to an external monitor. This would display the exact size of the IPAQ screen onto the external monitor

- Embedded Visual C++

With the above components, the objective was obvious and clear to "make the VNC viewer for Windows CE work with the Voyager VGA CF card". That is to say, when the VNC viewer runs on the IPAQ, the desktop of the source computer is visible on the screen of the IPAQ. Since the screen of the IPAQ is only 3.8" in size, we would need scroll bars to view the un-shown portion of the source desktop. This possibility can be eliminated if the display is routed to another external monitor via the Voyager VGA CF card, resulting in a full screen display of the source desktop.

The following is a discussion of the modules present in the VNC viewer software and the integration of the Voyager drivers

Component: Vncviewer.cpp

This is the entry into the VNC viewer application. Whenever the application is instantiated it starts off here. This contains the host name and port number of the source computer. In our case, these are hard coded for demonstration purposes.

Component: VNCViewerAppCE.cpp

This contains the state of the application as a whole. It also maintains a list of the active connections to the server. Though we are concerned with a single connection to the server, the VNC server is capable of handling multiple clients.

Component: ClientConnection.cpp

This is responsible for connecting to the VNC Server and updating the screen locally. This handles the client updates and does the rendering of the server desktop at the client end.

Component: Voyager.cpp

This contains the drivers for the Voyager card and the methods to use the card with the VNC viewer application. These are called in the ClientConnection module to initialize the CF card.

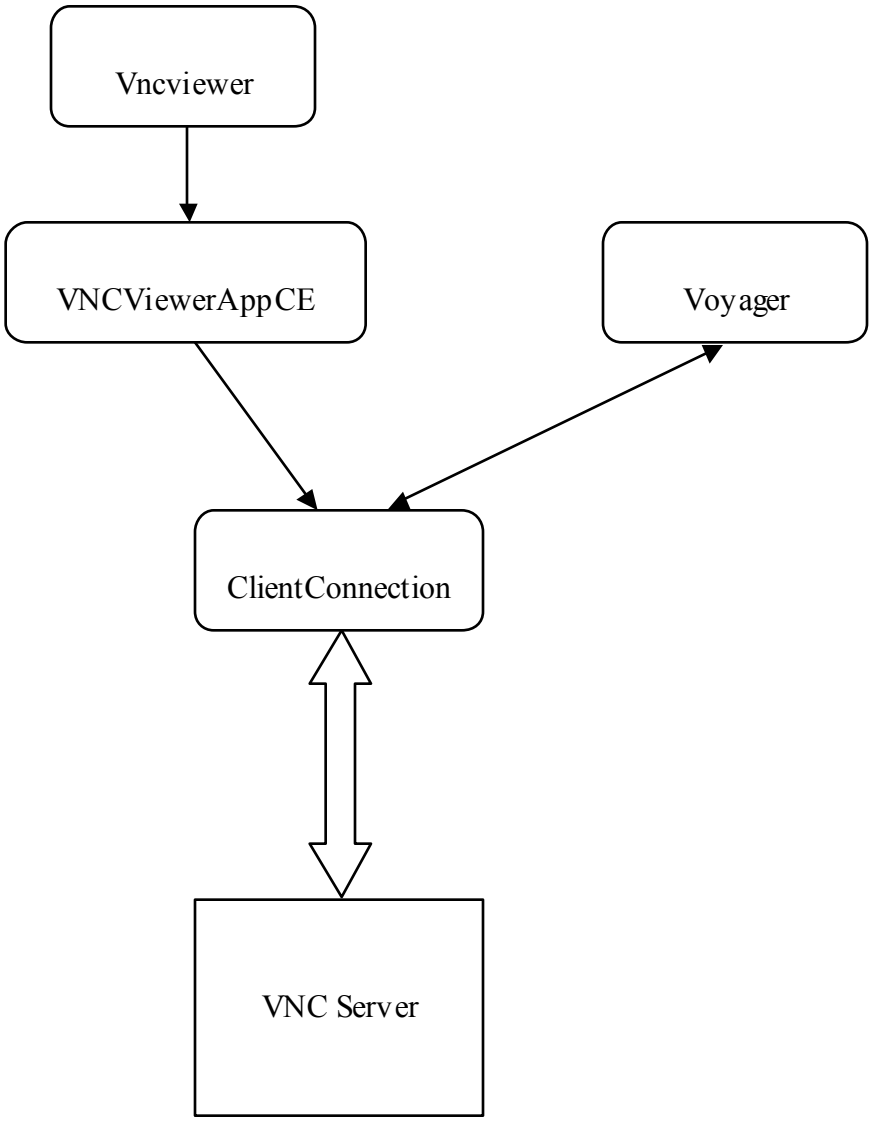

Figure 4.1. The flow of the modules in the Application

These components when integrated work in order to show the output of the source desktop through the Voyager CF card.

Figure 4.2. The VNC port to the IPAQ

A Compaq IPAQ 3870 with 802.11b and Bluetooth drives the 18" desktop LCD in figures 4.2 and 4.3. The display adaptor in the IPAQ is the Voyager CF-form factor VGA card. Software is our VNC port, utilizing portions of both the standard VNC distribution and TightVNC, with our optimizations and a modification to support the Voyager card. The laptop, whose screen is being mirrored, is a 1GHz ThinkPad A22p running an unmodified, off-the-shelf version of the TightVNCserver. The map being displayed was generated by Delorme Street Atlas USA, version8.0.

Figure 4.3. The VNC port to the IPAQ

Now, that the porting of VNC on to the IPAQ is accomplished, the next step is to improvise on the speed of the application so that it takes the minimum time to reflect the changes on the remote screen. For this we need to consider the different compression techniques offered by the VNC. As the current wireless technologies do not provide sufficient bandwidth to update high-resolution displays at the 60+fps required to provide a flicker-free image, compression is a key component of VNC.

For laptop computers, which have powerful processors, compression schemes with higher computational requirements (e.g., TightVNC 1.1, TightVNC 1.0 vs. Hextile compression. Results shown in Tab 2.1,2.2 in Chapter 2) are appropriate, but these schemes are too computationally intensive for the 206MHz Compaq IPAQ. Due to limitations of the IPAQ, primarily the high overhead imposed by Windows CE and the clock speed of the StrongARM processor, we have settled on a modified version of the Hextile compression scheme in the standard VNC distribution. While “tight” compression can reduce network traffic between the wireless display and laptop by a factor of 3 (or more), there is a much higher CPU burden placed on the wireless display. Our tests indicate that this burden is too great for the limited CPU power provided by the IPAQ.

Even though network traffic is reduced, refresh rate drops by a factor of 3-5 when "tight" is used (over Hextile). Note that this is in contrast to typical desktop applications of TightVNC—on powerful computers (Pentium III and IV class), TightVNC performs much better. We have been able to successfully incorporate JPEG compression support on the wireless display side. TightVNC JPEG compression performs with a loss in compression on the server side to reduce the amount of information needed to update the wireless display.

4.2. Hardware Implementation

The objective was to have the prototype as a single functional unit that is portable. For this it was necessary that the LCD be integrated with the IPAQ. Moreover, an easy mechanism for operating was absolutely essential. The startup of the application should be without any user input and it should be done with a minimal effort. The prototype has a flip-on mechanism to power it on and to start the software. This is essential as this is intended for soldiers in a mission.

The following are the components and technology used in the construction of the prototype:

- LTM06C310 6.3" Toshiba LCD display (1024×768 resolution)
- TRD-6702-AW6 backlight inverter circuit for the LCD
- PVX1603-114 LCD controller with auto-scaling
- Compaq IPAQ 3870, handling wireless communication and video decompression 802.11/Bluetooth communication (Bluetooth is integrated into the 3870, 802.11b is provided by a PCMCIA or CF card)
- Colorvision Voyager Compact Flash form factor VGA card.
- 9.6V Radio Shack NiMh remote-controlled racing car battery. This battery provides good power density at a reasonable cost.
- The interface was written using the Microsoft Embedded Visual C++ platform

Figure 4.4. Top view of the Prototype

The prototype is completely enclosed in a single Radio Shack project case. The prototype works with a flip switch. It is a one-action startup; by just flipping the switch the prototype comes into action. No hardware needs to be attached to the computer that is the source. The only requirement of the computer acting as the display source is that it needs to run an unmodified version of the VNC or the TightVNC server software.

Figure 4.5 Side view of the Prototype

The first move was to find out a box that would fit in all the components. A RadioShack project box was chosen to hold the entire system. The box first contains a circuit board of the LCD in it as the first layer. Every part is described in Fig 4.4.

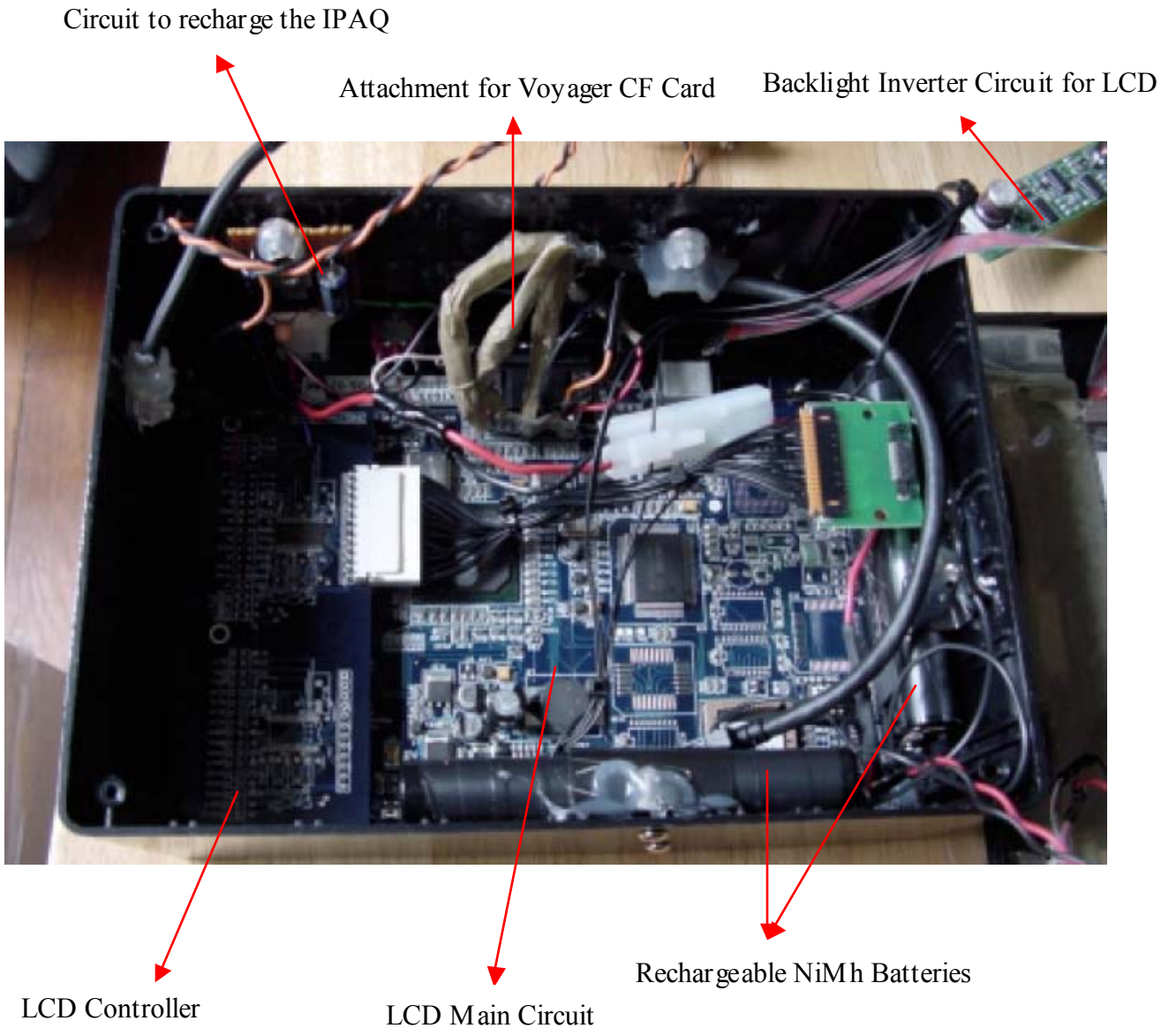

Figure 4.6. Parts of the Prototype

For the prototype to start the modified VncViewer application, another application was required whenever the prototype was powered on. This was accomplished by the code shown in fig 4.5. This application is present in a Flash memory card and inserted into the IPAQ. This makes sure that the whenever the IPAQ is powered and after it brings up the networking features, it runs the default application in the CF card.

```
int WINAPI WinMain(HINSTANCE hInstance,
                   HINSTANCE hPrevInstance,
                   LPTSTR lpCmdLine,
                   int    nCmdShow)
{
        MSG msg;
        HACCEL hAccelTable;

        // Perform application initialization:
        if (!InitInstance (hInstance, nCmdShow))
        {
                return FALSE;
        }

        hAccelTable = LoadAccelerators(hInstance, (LPCTSTR)IDC_STARTAPP);



        // REGISTER AN APPLICATION to run at either wakeup or application of AC power

        CeRunAppAtEvent(TEXT("\\Windows\\Start Menu\\vncview.exe"),
        NOTIFICATION_EVENT_WAKEUP);

        // Main message loop:
        while (GetMessage(&msg, NULL, 0, 0))
        {
                if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))
                {
                        TranslateMessage(&msg);
                        DispatchMessage(&msg);
                }
        }

        return msg.wParam;
}
```

Figure 4.7. Code for starting the vncview.exe application

Chapter 5. Performance

This chapter deals with the performance of the Prototype. Performance testing is necessary to determine the way in which a product functions after it is built. Performance testing also gives us a fair idea of what can be expected of a product.

5.1. Introduction

The prototype that is designed here was subject to the performance test and the results are tabulated in this chapter. Before moving to the details, it would be apt to put in a foreword about the environment in which the prototype should work, and the results achieved after the prototype was complete. As mentioned in the Chapter 1, the idea behind this project is to design and develop a portable wireless display prototype that could be mounted on the head or the wrist of the soldiers. This display should be communicating with a laptop or any other computing device present in the soldier's backpack.

A 4" Toshiba LCD was initially to be put in the prototype. But there was a 12-week delay in procuring the order for this display. The next available display was a 6.3" LCD, which resulted in a marginal increase in the size of the prototype. The batteries that power the LCD are NiMh batteries. All this equipment including the IPAQ was put into a RadioShack project box.

The LCD display has a resolution of 1024x768 pixels. We found that the best display results when the source computer desktop is also set to 1024x768 resolution.

5.2. Testing

We first provide the details on which the performance testing was done and the tabulated results, in terms of the time it takes to update an image, that were obtained by the test. The tests were conducted on two different images in two resolutions, 1024x768 and 800x600.

In the first test, the image in fig 5.1 was used. This is a JPEG image with a size of 351 Kbytes. The image was loaded and again reloaded to find out the time taken for the image to be refreshed and rendered on the LCD.

Figure 5.1. Image #1 used in Performance testing

The following is the data obtained from testing the image with the source desktop resolution at 1024x768 pixels:

S.No	Load (secs)	Reload 1	Reload 2	Reload 3	Reload 4	Reload 5	Average
1	65.87	67.39	68.9	66.86	71.48	66.1	67.77
2	69.00	71.82	66.29	67.8	68.2	72.53	69.27
3	68.53	68.16	71.83	65.9	69.72	67.55	68.62
4	67.52	65.84	64.8	70.25	66.24	65.4	66.68
5	66.58	65.13	63.59	70.19	63.71	66.68	65.98
						Total Time	67.66

Table 5.1. Results for Image #1 for a 1024x768 pixel resolution

The following is the data obtained from testing the image with the source desktop resolution at 800x600 pixels:

S.No	Load (secs)	Reload 1	Reload 2	Reload 3	Reload 4	Reload 5	Average
1	56.01	53.68	54.51	53.26	53.83	54.16	54.24
2	56.82	55.59	54.77	55.33	55.52	55.45	55.58
3	56.48	56.3	55.12	55.28	55.18	56.81	55.86
4	57.63	55.96	54.75	54.28	54.17	54.15	55.16
5	58.34	59.91	56.06	58.27	57.04	56.92	57.76
						Total Time	55.72

Table 5.2. Results for Image #1 for a 800x600 pixel resolution

The ‘Load’ in the table is the time taken to load the entire desktop immediately after the prototype was powered on. The ‘Reload’ was done by minimizing and maximizing the picture on the source desktop. For each test case 1 initial load and 5 reloads were done. And five such test cases were observed.

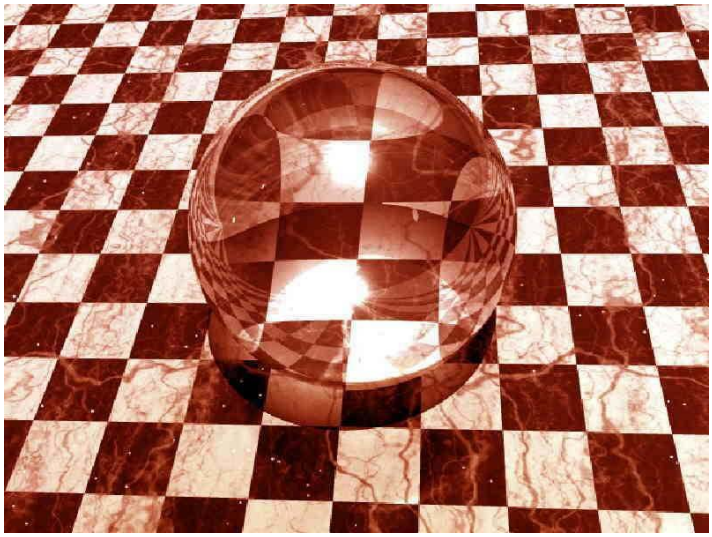

Figure 5.2. Image #2 used in Performance testing

The image in the above figure was of size 94 Kbytes. The following were the results obtained for similar test cases at 1024x768 resolution.

S.No	Load (secs)	Reload 1	Reload 2	Reload 3	Reload 4	Reload 5	Average
1	47.38	48.15	46.92	46.99	46.15	46.69	47.05
2	48.84	48.17	47.02	49.13	46.24	50.85	48.38
3	47.65	46.19	47.08	47.12	46.7	46.17	46.82
4	49.7	48.77	47.97	48.34	47.79	48.32	48.48
5	47.53	46.57	47.35	45.66	45.84	45.94	46.48
						Total Time	47.44

Table 5.3. Results for Image #2 for a 1024x768 pixel resolution

The following are the results obtained in the 800x600 resolution:

S.No	Load (secs)	Reload 1	Reload 2	Reload 3	Reload 4	Reload 5	Average
1	46.2	46.69	45.16	45.17	45.02	44.78	45.50
2	46.79	45.68	43.76	43.91	44.26	43.43	44.64
3	47.28	46.2	46	45.93	45.86	48.84	46.69
4	47.5	44.93	46.71	44.95	45.97	44.07	45.69
5	44.65	46.22	46.12	45.45	44.37	44.26	45.18
						Total Time	45.54

Table 5.4. Results for Image #2 for a 800x600 pixel resolution

5.3. Analysis

The refresh rates are good enough for static images to be displayed. This is not a good refresh rate for streaming video and multimedia rich applications. The average results obtained in the tables 5.1 – 5.4 provide statistics that static images are displayed at a normal rate. Part of the delay can be attributed to the computing power of the IPAQ. The 206 MHz StrongArm processor was a bottleneck. One would come to a conclusion that it could be the 11Mbps speed of the 802.11b network. But besides that the processing power of the IPAQ is insufficient.

The best quality came in when both the remote computers desktop and the settings in the software are in the 1024x768 pixel resolution. The optimum time was tabulated when the remote computer's desktop was in the 800x600 pixel resolution. The prototype build with this equipment should be best useful for soldiers in displaying static image or text, like a map of an area or a blueprint of a building.

Another hardware related performance issue is the duration of the batteries that power the LCD and the IPAQ. During the testing it was observed that the prototype runs continuously for 2

hours. This is a fair amount of battery time considering the nature and off-the-shelf components used.

Chapter 6. Conclusion

The wireless display prototype is a useful tool for soldiers who are in a mission to view the desktop of a remote computer (in this case a laptop in their backpack). It renders the remote desktop on a 6.3" LCD.

The prototype uses 802.11b wireless network as the medium and Virtual Network Computing as the software interface. No hardware is needed to be present on the remote computer. A TightVNC compatible server running on the remote computer is necessary.

The application is built on a simple Client/Server architecture. The VNC Server is capable of handling multiple clients. This makes the purpose of the display more deployable. Multiple soldiers can carry similar wireless displays connecting to one remote computer, thus eliminating the need for every soldier to carry a laptop.

The prototype is built with completely off-the-shelf components. It is contained in a RadioShack project box and the LCD is powered by rechargeable NiMh batteries. The processing is done by an IPAQ with 802.11b and Bluetooth wireless capabilities. The prototype has a flip switch on and off mechanism. No other user input need to be given.

The prototype performs well in handling static images. It can be powered continuously up to two hours. The refresh rate is not suited for multimedia rich applications and streaming videos. Though a high performance CPU and a wireless network with a bigger bandwidth will increase the performance, the prototype is good for displaying static images and text.

The prototype created is a proof of concept that wireless displays can be constructed with limited resources. Moreover, use of plasma display technologies and different platform other

than Windows might prove advantageous. Linux is a good alternative. Though VNC is free software, Windows should be purchased. Linux, being freeware, can eliminate the cost in every unit built. On-board CPU's with a higher clock speed can replace IPAQ.

References

[1] Andrew S. Tanenbaum: Computer Networks, Prentice Hall 1989

[2] Charles Perkins: Adhoc Networking, Pearson Professional Education 2001

[3] Bluetooth Technology Whitepaper – Taking Handheld computers to the next level
http://www.pico.net/white_papers.html

[4] Ricochet Metropolitan Area Network
http://www.ricochet.com

[5] Boingo Wireless Network
http://www.boingo.com

[6] IEEE Standards on Local Area Networks
http://standards.ieee.org/catalog/olis/lanman.html

[7] Dr. Golden Richard III: Topics in Mobile Computing, 2002
http://www.cs.uno.edu/~golden/Tutorials/PDCS2000/pdcs2000-1.ppt

[8] How VNC works.
http://www.realvnc.com/howitworks.html

[9]] Tight VNC documentation
http://www.tightvnc.org/docs.html

[10] Eckerson, Wayne W: Three-Tier Client/Server Architecture: Achieving Scalability, Performance, and Efficiency in Client Server Applications. Open Information Systems Volume 10 Number 1, January 1995, Page 3

Vita

Mr. Srivatsa Gundala was born in Mahasamund, a small town in Central India. He did his B.Sc in Computer Science from Andhra University, a reputed university in India. Later he started working as a Computer Instructor at the National Institute of Information Technology (NIIT), Visakhapatnam. He also completed his M.Sc in Computer Science from Andhra University.

He joined the MS program in Computer Science at University of New Orleans in Spring 2001. Mr. Srivatsa likes to read books on mythology and watch sitcoms.

UNIVERSITY OF NEW ORLEANS

MASTER'S EXAMINATION REPORT
Thesis

CANDIDATE: Srivatsa J. Gundala

MAJOR PROGRAM: Computer Science

TITLE OF THESIS: Creating a Portable Wireless Display

APPROVED

Golden G. Richard III — Signature

Major Professor (typed)

Adlai Depano — Signature

Committee Member (typed)

Ming Hsing Chiu — Signature

Committee Member (typed)

Committee Member (typed) — Signature

Robert C. Cashner — Signature

Dean of the Graduate School

DATE OF EXAMINATION:

11/07/2003