NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

INTERNET BASED AUTOMATION

Graduation Project
COM 400

Student: Shaikh Wasim Rashid

Supervisor: Assoc. Prof. Dr. Dogan Ibrahim

Nicosia - 2001

Acknowledgements

I wish to thank Dr. Suat İ. Günsel, founding president of the Near East University and Assoc. Prof. Dr. Şenol Bektaş. Many thanks to Assoc. Prof. Dr. Doğan İbrahim for being extremely generous with his advice and time for the completion of this project. Also special thanks to the Dean of Engineering Department Prof. Dr. Fakhreddin Mamedov, and to the chairman of computer engineering department Assoc. Prof. Dr. Adnan Khasman and to my advisor Mr.Tayseer Alshanableh for always being kind and informative.

I also wish to thank all the teachers of my department for imparting education and knowledge, and also to the lab assistants for the smooth running of computer and other laboratories in the university.

Abstract

The term Internet with all its connotations has become an inescapable part of our lives, we are used to e-mailing, seeking information, filling on-line survey forms, buying goods and services and all other innovations which keep happening every minute of the day.

One of the most promising of applications of the Internet is the "Internet based Automation". Individuals increasingly are being required to travel, locally as well as internationally and while doing so there is also a need for controlling various devices, apparatuses etc being geographically away from them, for example turning on a machinery. Thousands of dollars can be saved by automation applications by providing information about problems at home or factory to the individual involved at his/her location. Internet based Automation provides a relatively cheaper alternative compared to other technologies for satisfying these requirements. The existing Internet infrastructure is used; and with a few modifications actual communication can occur between two PCs or between two custom designed devices.

The actual technology involved is the same as that used for Internetworking, and is well tested giving a good score on reliability for "Internet Based Automation" vis-à-vis the cost involved.

This project studies the principles of Internet based automation and provides several examples on how the Internet can be used to control remote devices it also looks at the technology involved in the designing of applications for automation using the Internet. Most of the work is theoretical and software based but the ideas and the principles can be applied to real-life problems and applications can be developed from these ideas.

Table of Contents

List of Abbreviations

NSF	National Science Federation
IAB	Internet Architecture Board
IETF	Internet Engineering Task Force
ISP	Internet Service Provider
IANA	Internet Assigned Numbers Authority
WWW	World Wide Web
GUI	Graphical User Interface
A/D	Analog to Digital
GPS	Global Positioning System
VPN	Virtual Private Network

Introduction

Automation is one of the most important fields in industrial world. Traditionally automation has been carried out locally where one or more on-site controllers provide the required control and monitoring actions. Internet based automation provides new avenues for deploying automation, also it aids and makes possible various automation applications, which were not possible earlier or were too expensive to implement. The cost effectiveness of automation applications based on the TCP/IP protocol using the Internet makes them very attractive for automating residential and industrial processes. Future developments in this technology include the addition of networking capability in household and industrial appliances for the purposes of controlling them through on-site servers and remote severs. This type of technology will make it possible to save money and time spent on having to travel to locations for controlling the devices, savings will also be incurred by learning about problems earlier and being able to monitor remote devices not requiring manual presence.

Internet is currently very widely used for file transfer, searching, and many other office related tasks. This project looks at the remote automation applications, made possible by internetwork technology. The project looks at the TCP/IP protocol, which is the underlying protocol used on the Internet; Client-server structures are examined under the TCP/IP protocol. Visual basic language is described in terms of the client-server application development. The structure of the Internet is also presented and several examples are given for remote Internet based automation, using the visual basic language.

Chapter one provides information about the project, including the aim of the project, the technologies and the methodologies used.

Chapter two discusses the TCP/IP protocol with an intention to give a general idea of the meaning, function and use of the protocol.

Chapter three discusses the structure of the visual basic language.

Chapter Four provides an introduction to the Internet and different applications on it.

Chapter Five deals with the hardware and software aspects of implementing Internet based Automation.

Chapter Six discusses an application of Internet based Automation.

Chapter Seven presents case studies regarding the application of Internet based Automation.

Chapter Eight presents the conclusions.

Chapter One: Applications of Internet Based Automation

1.1. Aim of the Project

Internet based automation promises to be an important application of internetworking, though still in its infancy the applications of this technology can have a great impact on work and home lives of individuals. This project aims at exploring the technologies available, the hardware and software requirements and the potential applications of this technology with an intention to develop an understanding about the working of this technology.

Increasingly the Internet is being used for more than just data retrieval and information lookup. Automation of household appliances and factory appliances can be accomplished by sending the required commands using available technologies across the Internet.

1.2. Technologies

The main technologies involved are The TCP/IP protocol based communication via the Internet, interface devices between computers and the device to be controlled, for example to control a light, an electronic switch controlled from the parallel port of the PC is required. This electronic switch will be the interface between the appliance and the PC, this local PC will control the switch based on instructions received from another remote PC. Appropriate software will be required to control the switch and to transfer data between the computers. This software can be developed using different programming languages this project focuses on software developed using Visual Basic language.

1.3. Methodology

The method used to achieve the solutions is as follows; data is exchanged between computers using sockets under the TCP/IP protocol, the receiving computer then processes this data. Based on the commands contained in this data the software, along with the driver software for the electronic switches, controls the various devices. For example to turn on a residential water heater remotely the following process is done; a remote computer contacts the local computer (residential gateway) via the Internet and sends commands to turn on the water heater, the local computer analyzes the request and turns on the water heater by controlling electronic switches, which power the heater.

1.4. Commercial and Personal Applications

Many commercial and personal applications can be developed using this technology, either individual devices or a group of devices can be controlled by installing a network among these devices and the network being controlled by one computer acting as server, this server will receive information from the remote computer or computers and drive the appropriate device. Further this server can be controlled not only by remote PCs but also handheld devices such as Palm OS, Pocket PC and Mobile Phones.

Chapter Two: Transmission Control Protocol / Internet Protocol (TCP/IP)

2.1. Description

TCP /IP is a family of protocols used for computer communications. The letters stand for Transmission Control Protocol/Internet Protocol, TCP and IP are both individual protocols/that can be discussed separately, but they are not the only two protocols used in the family. TCP /IP was developed by the Department of Defense. Protocols usually are grouped into "families" (sometimes called suites or stacks) each protocol in a family supports a particular network capability. No protocol is of much use on its own and requires the use of other protocols in its family.

The TCP /IP protocol family includes protocols such as Internet Protocol (IP), Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP), Transport Control Protocol (TCP), Routing Information Protocol (RIP), Telnet, Simple Mail Transfer Protocol (SMTP), Domain Name System (DNS), and numerous others.

In TCP /IP, all protocols are transported across an IP Internet, encapsulated in IP packets. IP is a routable protocol, which means that two nodes that communicate using IP do not need to be connected to the same physical wire. To have a basic understanding of how information travels across a routed network it is important to consider the address format in the protocol, device addressing, address mapping to a physical address, router finding process by a router, topology learning by routers and service finding by users.

When a system sends data over the network using the TCP/IP protocol, it is sent in discrete units called datagrams, also known as packets. A datagram consists of a header followed by application dependent data. TCP/IP offers a reliable, full-duplex byte streaming type data communication. Data packets are re-transmitted if they do not reach their destinations reliably. TCP/IP is also connection-orientated protocol. This means that before two computers can begin to exchange data they must establish a connection with each other. One computer usually assumes the role of a server while the other computer the client. The server normally waits for a connection. The client is responsible for establishing the connection, while the server has the option of accepting (or rejecting) a connection request. If the connection is accepted then the two computers can begin to exchange data with each other.

2.2. Basic Networking Concepts

2.2.1. Addressing

The central concept of networking is addressing. In networking, the address of a device is its unique identification. Network addresses are usually numerical and have a standard, well- defined format. In routed networks, the address has at least two pieces: a network (or area) piece and a node (or host) piece. Network refers to a set of machines connected to the same physical wire (or set of wires connected only by bridges and repeaters). Internet means one or more networks connected by routers. If two devices on an internet have addresses with the same network number, they are located on the same network and thus on the same wire. Devices on the same wire can communicate directly with each other by using their data link layer protocol (Ethernet).

Correct addressing of devices on a network requires that every device connected to the same network (wire) be configured with the same network number. Also, every device with the same network number must have a different node (or host) number from every other device with the same network number. Finally, every network in an internet must have a unique network number. To rephrase, every network on an internet must have a unique network number, and every node on a network must have a unique node number within that network. This rule ensures that no two devices on an internet ever have the same network and node number and therefore have a unique address within the internet.

In addition to a unique address for every device on an internet, special addresses often are used to address multiple nodes simultaneously. These addresses are called broadcast or multicast addresses. Two important types of addresses are; network layer addresses and Media Access Control (MAC) layer addresses. These two address types are completely Independent of each other. The network layer addresses are all IP addresses. MAC addresses are used to communicate from node to node on the same wire and are built into the communications card (the Ethernet card).

2.2.2. Packets

Information sent across networks such as TCP/IP is generally broken down into pieces called packets (or datagrams) for two main reasons: resource sharing and error detection and correction. In networks that have more than two computers (Ethernet or Token Ring), the medium connecting the devices is shared. If any two devices are communicating, no other devices can communicate at the same time. If two devices want to share a very large amount of information they will block the network for a long period of time; other devices might have urgent information to transfer to other devices. If the large block of information is broken into many small blocks, each of it can be sent individually, enabling other devices to interweave their own messages with the packets of the extended conversation. As long as each piece is individually addressed to the intended destination and contains enough information for the receiver to piece back together, the fact that it is broken into pieces does not matter.

The other main use of packets is for error detection and correction. Networks are ultimately made up of wires (or radio waves or light beams) that are prone to interference, which can corrupt a signal sent across them. Many error detection and correction techniques are based on check sums. There is a practical upper limit to the amount of data that can be sent in one transmission

2.2.3. Protocols

Protocol specification provides meaning to the packets transferred on a network. A protocol is a set of rules that defines two things, the format of the packets and the semantics of their use. Most packets have a format that includes a header and a body. The header often includes information such as a source and destination address, the length of the packet, and some type indicator so that the receiver knows how to decode the body .The body can be raw data (for example, a piece of a file or an e-mail message), or it can contain another packet that has its own format defined by its own specification. Packet formats usually are depicted in a specification by a rectangular picture that gives the order, size, and names of the pieces of information that make up the packet. A protocol specification specifies the format of the information exchanged i.e. the packets and the correct sequencing of that Information as well as the additional actions (Logging mail delivery, table updates, and so on) that might be required. Figure 2.1 is an example of an Ethernet frame.

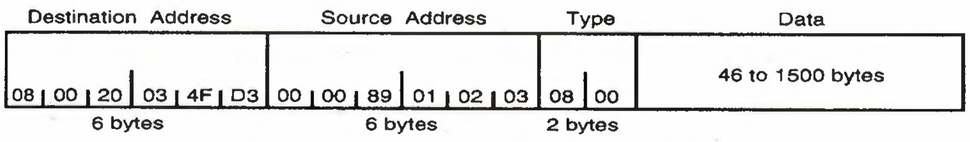


Figure 2.1 Example of an Ethernet Frame

2.2.4. Routers and End Notes

Routed networks have two classes of devices: end nodes and routers Figure 2.2 gives a typical structure of Routers and end nodes in a network.

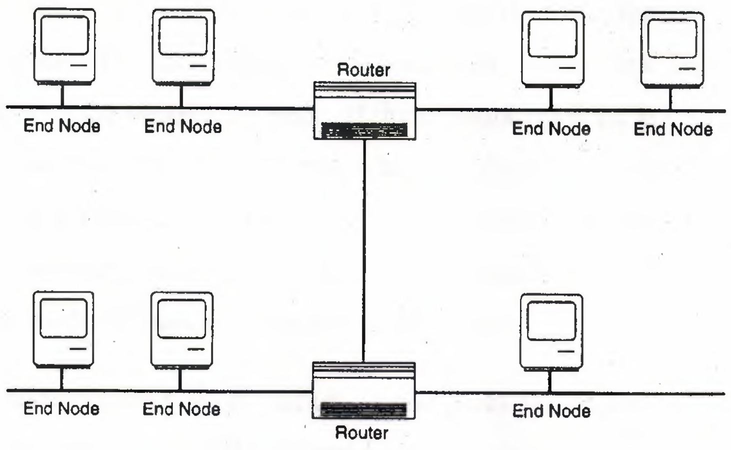


Figure 2.2 Routers and End Nodes.

End nodes are the devices with which users interact-workstations and PCs, printers, file servers, and so on. Routers are devices that connect networks. Routers have the responsibility to know how the whole network is connected and how to move information from one part of the network to another. Routers must have a separate address for every network to which they are connected. Routing is often a CPU-Intensive task and can significantly impact the performance of a machine doing tasks other than routing. For this reason, most routers are dedicated machines. Routers are introduced to networks for several reasons. Routers enable more devices to be ultimately interconnected because they extend the address space available by having multiple network numbers. Routers help overcome physical limitations of the media by connecting multiple cables. The most common reason for using a router is to maintain political isolation. Routers enable two groups of machines to communicate with each other while remaining physically isolated, which is especially important when the two groups are controlled by different organizations

2.2.5. End Node Network Send and Receive Behaviour

When a node on a TCP /IP network has an IP packet to send to another node, it follows a simple algorithm to decide how to proceed. The sending node compares the network portion of the destination address with the network portion of its own address. If the two networks are the same, it implies that the two nodes are on the same wire--either directly connected to the same cable or on cables separated only by repeaters or bridges in this case, the two nodes can communicate directly using the data link layer (for example, Ethernet). The sending node uses ARP to discover the destination node's MAC layer address and encapsulate the IP packet in a data link layer frame to be delivered directly to the destination node. If the network portions are different, the two nodes are separated by at least one router, which implies that the sending node cannot deliver the packet without using a router as an intermediary. The packet is encapsulated in a data link layer frame addressed to the MAC address of a router on the same wire. The router delivers the IP packet to the remote network.

When an end node receives an IP packet, It compares the destination address in the packet to its own address and to the IP broadcast address with which it is configured, if the destination address matches either of these addresses, the end node accepts the packet and processes it further. The way it is processed depends on which subprotocol of IP it is. If the destination address does not match, the packet is dropped (ignored).

2.2.6. Router Send and Receive Behaviour

When a node is functioning as a router and it receives an IP packet, it examines the destination IP address in the packet and compares it to its own IP address. If the addresses are the same or the destination IP address is the IP broadcast address, the packet is processed, as it would be for an end node. Unlike an end node, a router does not automatically drop packets that are received but not addressed to it. These are packets that end nodes on the network are sending to the router to be forwarded to other networks as indicated in Figure 2.3

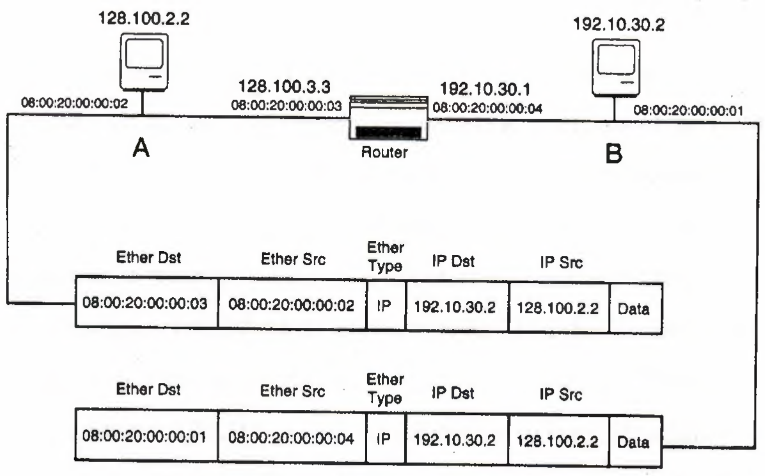


Figure 2.3 Transfer of Packets

All routers maintain routing tables that indicate how to reach other networks. The router compares the network portion of the destination address with each network in its routing table. If the router cannot find the destination network in its routing table, it checks for a default route (typically listed as a route to 0.0.0.0). If it does not find a default route, the packet is dropped (and an ICMP destination unreachable message is sent to the source IP address in the dropped packet).

When a matching route to a network is found (or a default route exists), the router checks the distance to the remote network. If the distance is listed as 0, the network is directly connected to the router. In this case, the router sends an ARP request for the destination IP address and encapsulates the IP packet in a data link layer frame addressed to the MAC address of the destination returned. In the ARP response. If the distance is greater than 0, the packet must travel through at least one more router. In this case, the router uses the next router field from this route and sends an ARP request for that router, encapsulating the IP packet in a data link layer frame addressed to the MAC address of the next router. This way, an IP packet travels across an Internet, maintaining the same source and destination IP addresses the entire time, but having the source and destination MAC addresses change for each hop.

2.2.7. Format of an IP address

For any routable protocol to be efficiently routable, the address must have two parts. TCP /IP addresses have two components-a network component and a host (or node) component. Addresses used with TCP /IP are four-byte (32-bit) quantities called simple addresses (not TCP /IP addresses). These addresses are written in standard notation, which means that each byte is written as a decimal number separated by dots (the period character), for example, 192.37.54.23 (pronounced "192 dot 37 dot 54 dot 23, because each piece of the IP address is 1 byte, its value must be between 0 and 255 inclusive, for example, the IP address 125.300.47.89 could not be a legal IP address because 300 is greater than 255 and would not fit in a single byte. Figure 2.4 depicts a typical packet with IP addresses.

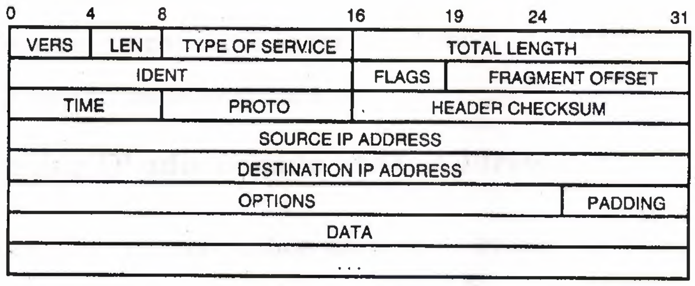


Figure 2.4. IP addresses in a typical packet.

IP addresses are composed of a network portion and a host portion determined by a graduated method of network and host division. To provide for efficient address use, IP addresses are divided into classes. The three most important classes of networks are A, B, and C. IP addresses are split into these classes according to the first few bits of the address (or the value of the first byte), as shown in Figure 2.5.

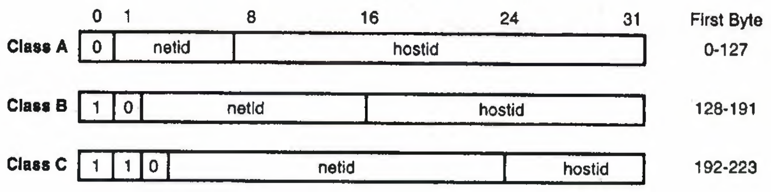


Figure 2.5 Classes of IP addresses.

2.2.8. Assigning IP addresses to TCP/IP Devices

IP addresses can be assigned in a number of ways. A good strategy is to obtain globally unique IP address; an organization that connects to the Internet can obtain an IP address thorough the Internet Service Provider. Corporations can also have internal network administrators in charge of assigning IP addresses to individual users. IP addresses must be configured into devices with the same network number as all other devices on the same wire but with a unique host portion. If two or more devices have the same IP address, they will not work reliably and will present a very difficult situation to debug. Most IP devices require manual configuration. The person installing the device must obtain a unique and correct IP address and type it in to some configuration program or console, usual ly along with other information such as IP broadcast address, subnet mask, and default gateway address. Some sites support dynamic configuration of IP devices. Protocols such as Boot P (BOOTP) and Dynamic Host Configuration Protocol (DHCP) enable the use of centralized servers to hand out unique IP addresses and other configuration information on needed basis.

2.2.9. Mapping IP addresses to MAC Addresses

Ultimately, all computer communication takes place by moving data from node over some form of link such as Ethernet, Token Ring, FDDI, and the Point-to-Point (PPP). Many links support attaching more than two nodes and therefore require data sent over them be addressed to a specific destination to be delivered. These addresses are MAC addresses, sometimes called hardware or link addresses. Unlike IP addresses that are assigned, most MAC layer addresses are built into the hardware by the manufacturer of the device or network interface (NIC). This address is a six-byte quantity usually written using hexadecimal numbers with a colon separating the bytes, for example, 08:00:20:0A:8C:6D. Ethernet addresses are assigned by the Institute of Electrical and Electronics Engineers (IEEE) and are unique among all devices. The Ethernet address is divided into two parts; the first three bytes constitute the vendor code and a different remaining three bytes to guarantee that every Ethernet device in the world has a unique address built in.

2.2.10. Examining how end nodes find a router

To send a packet to a node on another network, an end node requires the aid of a router. To deliver a packet to a node on a different network, a source node sends the unmodified IP packet to the local router by encapsulating it in a link level packet addressed to the router's MAC address. If the link level is Ethernet, the source node needs to know the Ethernet address of the local router since nodes should not deal with Ethernet addresses directly; therefore, TCP /IP end nodes need to know how to obtain the Ethernet address of a router .By using the ARP protocol, an end node that knows the IP address of a router can obtain the Ethernet address. TCP /IP end nodes need to be manually configured with the address of at least one router (usually called a default gateway), or the address may be obtained dynamically.

2.2.11. How Routers learn the network topology

For routers to fulfill their role, they need to know which networks are reachable and how to get to them. To accomplish this, routers store information about the topology of the network. This topology is usually stored as a routing table that lists each known network, tells how "far" away the network is, and indicates which router is the next one to send a packet to reach a network not directly connected, this process is indicated in Figure 2.6 below.

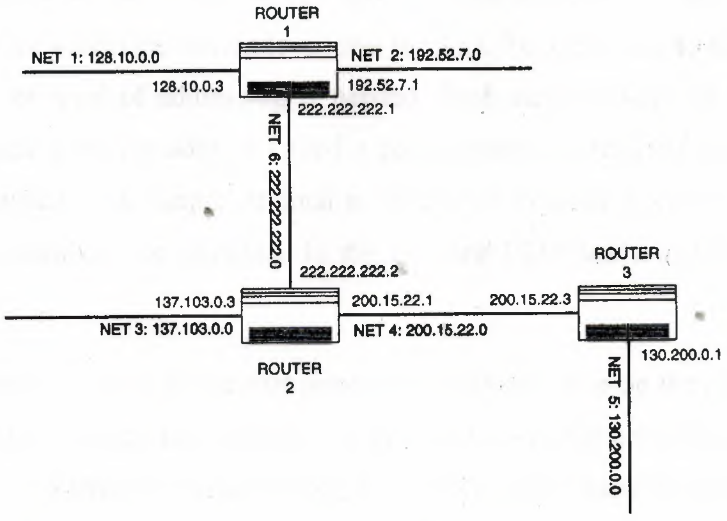


Figure 2.6. The process of locating networks by routers.

The cost of a network is most often simply a count of how many routers a packet must go through to reach a network. The cost to a network is often called the distance or number of hops to a network. If a router is not directly connected the cost is non zero in this case the router sends an ARP request for this "next hop" router and encapsulates the datagram in a data link layer packet addressed to the MAC address of the next router. When this router receives the packet, it checks its routing table and determines if the packet can be delivered locally or needs to be routed to yet another router.

2.2.12. Finding and Using Services

Since maintaining an up-to-date file on every single IP device can be difficult, usually a network administrator configures one or more servers to maintain a network- accessible database of name-to-IP mappings. Two commonly used methods are the Domain Name System (DNS) and Sun 's Network Information System (NIS). Maintaining such a database requires that one or more machines be designated as "keepers of the database," and all other machines send requests to these servers to have a name converted into an IP address or Vice versa. A network-accessible database of name-to-IP mappings is easy to maintain on a large network and requires little per-device configuration.

2.2.13. TCP and UDP

Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) travel encapsulated in IP packets to provide access to particular programs (services) running on remote network devices. After the data arrives, some mechanism is needed to enable the proper service within the device to receive the data. To direct data to the appropriate program, another level of addressing is needed. Each service available on a node is accessed through a unique address called a port (sometimes referred to as a socket). Ports are Identified by a simple decimal number .For example port 25 is the SMTP address. These numbers are contained In the TCP and UDP headers of TCP and UDP packets

A datagram-based protocol is one that imposes a maximum size on the amount of data that can be sent in a single transmission. Ethernet, IP, and UDP are all datagram-based protocols. TCP is a stream-oriented protocol, TCP breaks the transmission into smaller sizes retransmitting lost pieces, reordering data delivered out of order, and filtering out extras that might occur due to faulty retransmissions. UDP is an unreliable or besteffort protocol. TCP is a reliable protocol. Applications that invoke sessions usually use TCP to transfer data. Applications that are stateless are usually built on UDP, such as NFS. Applications and protocols that use UDP include the following: NFS, RIP, Trivial File Transfer Protocol (TFTP), Simple Network Management Protocol (SNMP). Applications and protocols that use TCP are as follows: FTP, Telnet, and SMTP.

Chapter Three: Introduction To Visual Basic

Visual basic has emerged as a widely used rapid development environment, automation using available classes and controls makes visual basic a very popular programming language.

3.1. Development Stages of Visual Basic

In the early 1990s, code reusability was an extremely vexing and important problem in the software development world, C++ permitted programmers to create classes that could be reused and extended by other programmers but C++ classes are very abstract and difficult to use. Here Visual Basic came on the scene with 16-bit '.VBX' controls - such as the text box, in the next version, VB3.0, an important control was introduced - the data control. Then with VB4.0, 32 bit '.OCX' controls were introduced that were even more powerful. These generalized visual controls were pre-built, and came with huge functionality. Programmers simply drew the pre-configured controls on a form and could then concentrate on writing the application. As such, these controls were a huge success achieving true code reusability.

3.2. Fundamentals of Visual Basic Programming

Visual Basic is a rapid application development tool for creating stand-alone software components including executable programs, ActiveX Controls and COM components.

3.2.1. The Programming Environment

There are the six parts in the standard Visual Basic environment:

- The *title bar* is the horizontal bar located at the top of the screen.
- The *menu bar* provides the tools needed to develop, test, and save your application.
- The *toolbar* is immediately below the menu bar.
- The *toolbox* contains the tools for developing your application.
- The *Initial Form Window*, the Visual Basic documentation uses the term *form* for a customizable Window.

- The *Project Window* The Project window contains a list of all the customizable forms and general code (modules) that make up an application.

Also part of the environment is The Properties window, used to customize a form or control and The Debug window, which shows up when the program is running or being debugged.

3.2.2. Help System

The online help system contains essentially all of the information in the Language Reference that comes with Visual Basic it is context sensitive, the items on the help menu are:

- *Contents* The Contents option indicates how the Visual Basic help system is organized.
- *Search* The Search option performs search for help on a specific topic.
- *Obtaining Technical Support* This option obtains online help it also contains hints on advanced topics.
- *Visual Basic Books Online* This launches the Books Online tool, searches through all the Visual Basic documentation.
- *Learning Microsoft Visual Basic* provides a simple description of Visual Basic.
- *About* The About option gives the copyright notice and serial number of the copy, also to whom the copy of Visual Basic is licensed, and system information.

3.3. Description of a Visual Basic Program

3.3.1. Code Window

Code is written in the Code window, this window opens whenever a control or form is double clicked, it also opens when the view code button from the Project window is clicked or View|Code is selected or F7 is pressed. It has a caption that lists the form, two list boxes, and an area for editing code. The left list box, called Object, lists all the controls on the form. The right-hand list box, named Proc, is the Procedure list box; it shows all the events recognized by the object selected in the Object list box.

3.3.2. Statments

Visual Basic analyzes a statement immediately after it is entered and ENTER is pressed Many typing errors are detected, the first letter of command words is capitalized and often extra spaces are added for readability. Statements in Visual Basic rarely use line numbers, each statement generally occurs on its own line. Statements can be combined on one line by placing a colon (:) between them. Lines are limited to 1023 characters and can be extended to the next line by using the underscore character (_) as the last character in the line.

Remark (or Rem) statements are put into programs to explain the code, Remark statements are neither executed nor processed. There are two ways to indicate a remark statement, the usual way to indicate a comment is with a single quote, or the older Rem keyword can also be used. When Visual Basic processes an End statement, the program stops, after the End statement, all windows opened by program are closed and the program is cleared from memory.

Visual Basic uses an equal sign for assignment and for property setting. Optional keyword Let can also be used, for example

InterestRate = .05

To change a property setting for an object, the object's name is placed followed by a period and then the name of the property on the left side of the equal sign, and the new value is put on the right-hand side, for example

object.property = value

3.3.3. Variables

Variables in Visual Basic hold information (values). Variables are named as follows:

<type><scope><name>

The name part is simply the variable name written in mixed case, and the type part is a Hungarian type. Names reserved by visual basic cannot be used for variable names, visual basic handles 11 standard types of variables, and it is also possible to define variable types. The basic variable types are:

- *String* variables hold characters String variables can theoretically hold up to 65,535 characters, although a specific machine may hold less due to memory constraints.
- *Byte* variables hold small integer values, between O and 255.
- *Boolean* variables are stored as 2-byte numbers, and can only hold the value True or False.
- *Date* variables are stored as 8-byte numbers with possible values in the range January 1, 100 to December 31, 9999.
- *Object* variables hold address information that refers to OLE Automation objects.
- *Integer* variables hold relatively small integer values, between -32,768 and +32,767.
- *Long Integers* variable hold integers between -2,147,483,648 and +2,147,483,647.
- *Single Precision* These variables hold numbers that are approximations.
- *Double Precision* variables hold numbers with 16 places of accuracy and allow more than 300 digits.
- *Currency* variables are designed to avoid certain problems inherent in switching from binary fractions to decimal fractions.
- *Variant* data type is designed to store all the different possible Visual Basic data types in one place.

Type identifiers or Dim keyword can be used for variable declaration. Scope of Variables refers to the availability of a variable used in one part of a program to the other parts of a program, variable can be local to an event procedure, available at the form level or global in scope, variables can also be static or dynamic. Static variables retain their values between procedure calls.

3.3.4. Operators

The following is the list of symbols for the five fundamental arithmetic operations:

- + Addition
- - Subtraction (and to denote negative numbers)

- / Division
- * Multiplication
- ^ Exponentiation

For integers and long integers, there is one symbol and one keyword for the arithmetic operations unique to numbers of these types:

\	Integer division (this symbol is a backslash)
Mod	The remainder after integer division

The following list gives the order (hierarchy) of operations:

- Exponentiation
- Negation (making a number negative)
- Multiplication and Division
- Integer division
- The remainder (Mod) function
- Addition and Subtraction

Operations on the same level are done left to right, parentheses changes the order of precedence.

The Relational Operators are:

<>	Not equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to

3.3.5. Definitions

Visual Basic's named constant feature allows mnemonic names for values that never change. Constants are declared like variables, and the rules for their names are also the same.

Looping or repeating a set of statements can be done in visual basic by using the following:

Determinate Loops i.e. loops for which the number of iterations are known beforehand.

For-Next loop.

```
For I% = 1 To 10
    Print I%
Next I%
```

For and Next are keywords that must be used together. The statements between the **For** and the **Next** are usually called the body of the loop. The keyword For sets up a counter variable. In the preceding example, the counter is an integer variable: I%. In this example, the starting value for the counter is set to 1. The ending value is set to 10. Visual Basic first sets the counter variable to the starting value. Then it checks whether the value for the counter is less than the ending value, If the value is greater than the ending value, nothing is done. If the starting value is less than the ending value, Visual Basic processes subsequent statements until it comes to the keyword Next. At that point it adds 1 to the counter variable and starts the process again. This process continues until the counter variable is larger than the ending value. At that point, the loop is finished, and Visual Basic moves past it.

To change the counter variable by a value other than 1 the Step keyword is used as follows:

```
For 1% = 10 To 1 Step -1
    Print  1%
Next 1%
```

Indeterminate Loops i.e. loops for which the number of iterations are not known beforehand.

Do-Until Loop

```
Do
    Visual Basic statements
Until condition is met
```

The Do While Loop

```
Do
    Visual Basic statements
Loop While XS = " "
```

While/Wend Loop

```
While  XS = " "
    Visual Basic statements
Wend
```

For indeterminate loops care should be taken to update the variable controlling the condition, as failure to do so may result in infinite loops.

Conditional Statements in visual basic have the following form:

If condition Then Do something else...

When an If-Then statement is encountered, the If clause is checked to determine whether it is True. If that clause is True, the statements following the Then clause are executed. If the test fails, processing skips to the next statement.

To process multiple statements if a condition is True or False. the If-Then-Else is used, called the Block If-Then, using it many statements can be processed in response to a True condition. The block If-Then looks like this:

```
If thing to test Then
    Statements
Else
    Statements
End If
```

The Else part is optional; and the block following it will be processed only if the If clause is False, whether the Else is there or not, the Block If must end with the keywords End If.

Select case control structure is used to check multiple branches, the structure of the Select-Case statement is as follows:

```
Select Case (thing to test)

        Case 1

        Case 2

End Select
```

3.3.6. Objects

What makes visual basic powerful is the ability to work with variables of the special object type these variables are declared same as the other variables. The Set keyword is used to make an object variable to refer to a specific object in a project. The New is used to create a new instance of a visual basic object at run time. The Nothing keyword is used to release the memory occupied by an object variable. Properties of objects can be manipulated via code during the execution of programs.

At the heart of VB's object creation abilities is its Class module. Each Class module in a project defines a class and exposes it for use by other applications. A class is exposed by means of its interface, this refers to the properties and methods that the class makes available to other applications. An OLE class's properties and methods are the public variables and procedures placed in the class module.

Chapter Four: Internet And The World Wide Web

4.1. Internet Description

4.1.1. Brief History of the Internet

In the early 1960s, computer scientists across the United States began exploring ways of directly connecting remote computers and their users. In the mid-to- late 60s, realizing the potential of such a network the United States decided to fund an experimental network that would enable remote research and development sites to exchange information. This network, funded by the U .S. Advanced Research Projects Agency, was named ARPANET.

4.1.2. Development of Arpanet

Among the major goals of the ARPANET research one was to develop a network on which communications would not be seriously impaired if physical sections of the network were lost. Also further additions or removals from the network should cause minimal or no impact, and allow computers of many different types to communicate easily. One of the major impacts of the ARPANET research was the development of the TCP/IP protocol described in an earlier chapter.

4.1.3. Structure and Administration of Internet

During the early 1980s, the ARPANET became the backbone of the Internet. In the mid 1980's the NSF funded a backbone network called NSFNET this connected the supercomputer centers earlier set up by NSF to universities and research centers. In the late 1980s, NSF awarded a contract to a single organization (MERIT, a consortium of educational institutions in Michigan) to be responsible for maintaining and upgrading the physical network and the network administration for the NSFNET. In the early 1990s, MERIT proposed allowing the Internet to carry commercial traffic with the profits earned to be used to improve the network infrastructure. By mid-1990's, the Internet had become completely commercial, In April of 1993, the NSF awarded five-year cooperative agreements for the management of the Network Information Services. The recipients of these agreements together manage the InterNIC (Internet Network Information Center). They are responsible for providing information about getting connected to and using the Internet. Network Solutions was chosen to provide the

Internet registration services, including the assignment of IP addresses and registration of domain names.

Any network connected to the Internet agrees to the decisions and standards set forth by the Internet Architecture Board (IAB) it also administrates several committees that provide technical support for the Internet, the main one being the Internet Engineering Task Force (IETF). The IETF is a committee of scientists and experts that work to resolve technical and related support issues for the Internet.

4.1.4. Connecting To the Internet

Regional networks pay connection fees to use the high-speed backbone of the Internet, commercial organizations, individuals and educational institutions that want to connect to the regional network pay network usage fees to the Internet Service Provider (ISP)

4.2. World Wide Web Description

The World Wide Web (WWW or Web) is one of the Internet services. The WWW allows the combination of text, audio, graphics, and even animation, various WWW browsers allow the exploration of www Internet sites, allowing quick access to hypermedia documents provided at those sites. They also allow the use of the same GUI to interface to other Internet services such as FTP, Gopher, and UseNet newsgroups.

4.2.1. History of the World Wide Web

In 1989, some researchers at CERN European Laboratory for Particle Physics wanted to develop a better way to give widely dispersed research groups access to shared information. Earlier each retrieval and look up of documents required running various applications such as Telnet, FTP etc. To save time a system was proposed that would enable the quick access of all types of information with a common interface removing the need to execute many steps. By the end of 1990, the researchers at CERN had a text-mode (non-graphical) browser and a graphical browser for the NeXT computer. During 1991, the WWW was released for general usage at CERN. Initially, access was restricted to hypertext and UseNet news articles. As the project advanced, interfaces to other Internet services were added (WAIS, anonymous FTP, Telnet, and Gopher). During 1992, CERN began publicizing the WWW project, commercial organizations started creating WWW servers to make their information available to the Internet, by the end of 1993, browsers had been developed for many different computer systems, including X Windows, Apple Macintosh, and PC/Windows.

4.3.2. Important World Wide Web Concepts

Browsers , To access the WWW, A *browser* is required, an application that knows how to interpret and display documents that it finds on the WWW. Documents may be textual or graphical.

Hypertext (and Hypermedia)

Documents on the WWW are hypertext documents. *Hypertext* is text that contains links to other text. This allows quick access to other related text on the WWW. In addition to text, many of the documents contain pictures, graphs, sounds, or even animations. Documents that contain more than just text are called *hypermedia* documents, because they contain multiple media.

HTML

A mark up language defined under the SGML (Special General Markup Language) Scheme, HTML is a set of commands or tags that describes how a document is structured. HTML allows the definition of the parts of the document, but not the formatting, so the browser that opens the document does the formatting.

Links

Links are references to other documents. There are two parts of a link; Reference and Anchor.

Chapter Five: Communication Between Devices

TCP/IP is accepted as the standard protocol to be used when two or more computers wish to exchange information over a network. Before actual communication can take place several hardware and software components must be in place before the data can actually be sent and received.

5.1. Hardware Requirements

Physical hardware connection must exist between the computers. This can be achieved through either a network interface card (NIC), or a serial communications port for dial-up type networking connections via the Internet.

5.2. Software Requirements

Computers have to be software configured to use the TCP/IP protocol over the selected transmission medium. As part of this configuration, each computer on a network should be given a unique Internet Protocol (IP) address and an IP name. Each computer is assigned a 32-bit number, which can be used to identify it over the network. This address is broken down into four 8-bit numbers, separated by dots. This is called dot-notation and looks something like "170.45.23.1".

There are two general approaches that one can take when creating TCP /IP based programs. One is to code at the lower level using the Windows Application Programming Interface (API) calls. The other option, which provides a higher-level and much simpler approach, is to use an activex component. In this chapter working examples are provided using Winsock activex component and the workings of Windows sockets Winsock activex component. Client-server based TCP/IP communications are described.

5.3. Programming Details

5.3.1. Sockets and Ports

A socket is a logical interface between two computers, which is created when using the TCP/IP protocol; a socket is a communications end-point. For exchanging information apart from a socket a port number has to be assigned to direct the data to a specific place. A port is a virtual link between two systems, which communicate with each other. Several applications use a socket, each transmitting and receiving data through a selected port. It is important that both sides of the communication ends must use the same port number. The port numbers are divided into three ranges: the Well Known Ports, the Registered Ports, and the Dynamic or Private Ports. The Well Known Ports are those from 0 to 1023. The registered Ports are those from 1024 to 49151, and the Dynamic Ports are those from 49152 to 65535. The Well Known Ports are assigned by the Internet Assigned Numbers Authority (IANA) these should be used with care when the system is connected to the Internet. Some of the well-known ports are given in table 5.1 below. TCP/IP based applications using the Winsock control, should not use any ports that are used by other applications.

Table 5.1 Some of the important Well Known Ports

1	tcpmux	tcp port service multiplexer
7	echo	echo
11	systat	systat
13	daytime	daytime
15	netstat	netstat
17	qotd	qotd
18	msp	message send protocol
19	chargen	ttytst source
20	ftp	ftp default data
21	ftp	control
22	ssh	ssh remote login protocol
23	telnet	telnet
37	time	time
42	nameserver	host nameserver
43	whois	who is
53	domain	domain name server
67	bootps	bootp server
68	bootpc	bootp client
70	gopher	ınternet gopher
79	finger	finger
80	http	world wide web http
88	kerberos	kerberos
110	pop3	post office protocol 3
123	ntp	network time services
161	snmp	snmp
162	snmp	snmp trap

5.3.2. WINSOCK Control

Similar to the other standard activex components (e.g. dialog boxes, push-buttons, text-boxes, list-boxes etc.), Winsock activex control has a number of properties, methods, and events as described below:

Properties

The Winsock's most interesting properties are described below:

- LocalHostName - IP name of local machine
- LocalIP – IP address of local machine
- LocalPort - Local port number to use for data exchange
- Protocol - Protocol to use for transmission
- RemoteHost – Remote machine IP name or IP address
- State – Current state of Winsock control

In all the examples given below the winsock control is named as winsck1.

LocalHostName returns a string, which is filled with the IP name of the local machine. Example use:

```
Dim myname as string
Myname = winsck1. LocalHostName
```

LocalIP returns a string, which is filled with the IP address of local machine in dot string format. Example use:

```
Dim myaddress as string
myaddress = winsck1. LocalIP
```

Protocol is used to set the protocol type used. Valid options are sckTCPProtocol for TCP based protocols and sckUDPProtocol for UDP based protocols. Example use:

```
Winsck1.Protocol = sckTCPProtocol
```

RemoteHost is used to set or return the remote machine to which a control sends or receives data. Either the IP name or the IP address of remote machine can be specified. Example use:

Winsck1.RemoteHost = "Charlie"

This code will define the remote machine IP name as "Charlie".

State property returns the state of the control, expressed as an enumerated type. The settings for the state property are:

▪ SckClosed	Closed (default)
▪ SckOpen	Open
▪ SckListening	Listening
▪ SckConnectionPending	Connection pending
▪ SckResolvingHost	Resolving host
▪ SckHostResolved	Host resolved
▪ SckConnecting	connecting
▪ SckConnected	connected
▪ SckClosing	Peer is closing the connection
▪ SckError	Error

Example use:

```
IF winsck1.state = sckOpen Then
  Msgbox "Socket is open"
END IF
```

This code will display the message "Socket is open" if the socket is open

Methods

The winsock's most interesting methods are as follows:

▪ Accept	Accept incoming connection (TCP only).
▪ Bind	Specify local port and IP in multiple connections.
▪ Close	Close connection (TCP only).
▪ Listen	Listen for connection (TCP only).
▪ GetDate	Retrieve received data packet.
▪ SendData	Send data packet.

In all the examples below, the winsock control is named as winsckl.

Accept is used for TCP server applications only. This method is used to accept an incoming connection in a ConnectionRequest event. Example use:

```
Private Sub Winsckl_ConnectionRequest(Byval requestID as Long)
'
'Accept the connection
Winsckl.Accept requestID
End Sub
```

The RequestID parameter identifies the request. The Accept method should be used on a new control instance and not on the one that is in the listening state.

Bind specifies the LocalPort and LocallP address to be used for TCP connections. This method is used if there are multiple protocol adapters. Example use:

```
Winsckl.Bind 500, "120.12.34.45"
```

Binds port 500 to local address "120.12.34.45". Bind is used before the Listen method.

Close method closes a TCP connection. Example use:

```
Winsckl.Close
```

Listen is used by the server TCP protocols. The machine waits in listen mode waiting for a connection request from a client. Example use:

```
Winsckl.Listen
```

GetData retrieves the received block of data and stores it in a variable. The general format is:

```
Object.GetData data,[type,][maxLen]
```

Where *data* is where the retrieved data will be stored, *type* is the type of data to be retrieved. This parameter is optional the data types are given below; *maxLen* is the size to read. This parameter is optional and if omitted, all available data will be retrieved.

Data types that can be used with GetData

▪ vbByte	Byte
▪ vblnteger	Integer

▪ vbLong	Long
▪ vbSingle	single
▪ vbDouble	Double
▪ vbCurrency	Currency
▪ vbDate	Date
▪ vbBoolean	Boolean
▪ vbError	SCODE
▪ vbArray + vbByte	Byte array

Example use:

```
Dim mydata as string
Winsck1.GetData mydata
```

SendData will send data to a remote computer. Example use:

```
Dim mydata as string
Mydata = "This is a test string"
Winsck1.SendData mydata
```

Will send the contents of string mydata to the remote computer.

The Winsock's most interesting events as follows:

In all the examples below, the winsock control is named as winsckl

Winsock control important events

▪ Close	Connection is closed
▪ Connect	A connection is established
▪ ConnectionRequest	A connection request received
▪ Error	An error has occurred
▪ DataArrival	Data is being received from remote computer

Close event occurs when the remote computer closes the connection. This event should be used to close a TCP connection properly.

Connect event occurs when a connection is established to the remote computer.

ConnectionRequest event occurs when a remote computer requests a connection. This event is available for TCP based server applications. The server should accept the connection request within this event procedure.

Error event occurs whenever an error occurs in background processing; for example, if failure is detected in receiving or sending data.

DataArrival event occurs when new data is received from the remote computer. GetData method should be used within this event procedure to retrieve the received data.

5.3.3. Architecture

Programs written to use the TCP/IP protocol are developed using the client-server model. In this model one computer assumes the role of the client, while the other computer must assume the role of the server .The server computer creates a socket and listens for connection requests from the client computer on a specified port. The client computer creates a socket and initiates a connection request to the server on the same port. The server computer then accepts (or rejects) the connection request and the two sides can start to exchange data. By accepting the connection, the server completes the virtual circuit between the two machines. It is important to note that a new socket is created by the act of accepting a connection and the original socket still continues to listen for additional connection requests. When the server computer no longer wishes to communicate, it closes the socket connection and this act breaks the virtual connection between the two machines.

The steps in establishing a connection, sending and receiving data, and closing a connection are shown diagrammatically in Figure 5.1

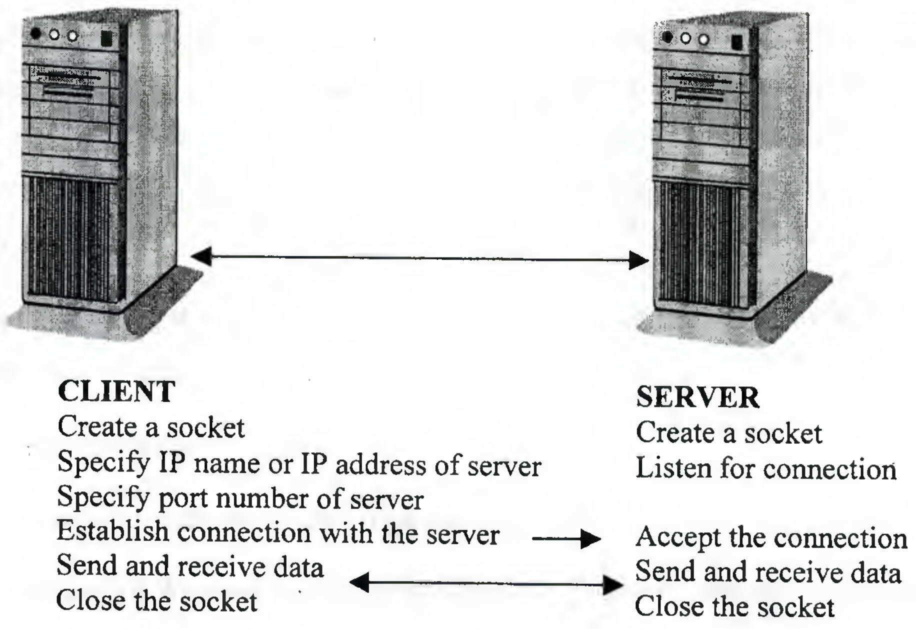


Figure 5.1 TCP/IP communication steps

5.4. Examples

5.4.1. A simple TCP/IP example

In this section a simple client-server based TCP/IP example is presented. In this example, two computers with the network names and addresses as shown in Figure 5.2

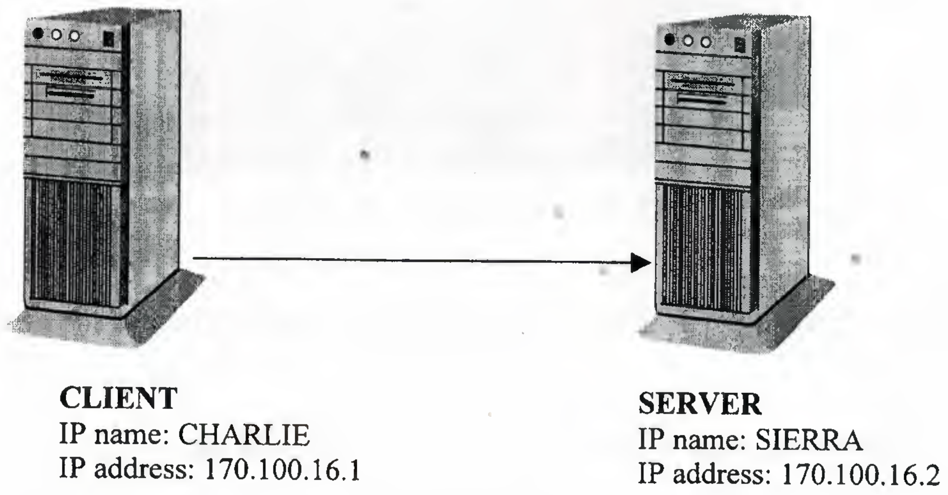


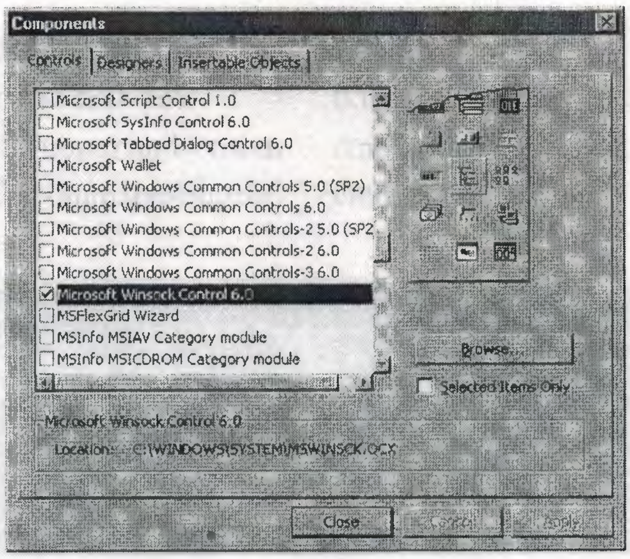
Figure 5.2 Client and server computer of the simple example

are connected to each other using a network interface card. Two programs are developed, one for the server (SIERRA) and one for the client (CHARLIE). The server computer shall create a socket and listen for a connection. The client computer shall initiate a connection and then connect to the server. Data will then be sent from the client to the server computer .The server computer will display the received data in a text-box.

The steps in creating the two programs are given below. The TCP/IP protocol should be installed and configured on these computers.

CLIENT COMPUTER

- 1 Load up Visual Basic and create a new project.
- 2 Microsoft Winsock Control is selected from Projects|Components as shown in Figure 5.3

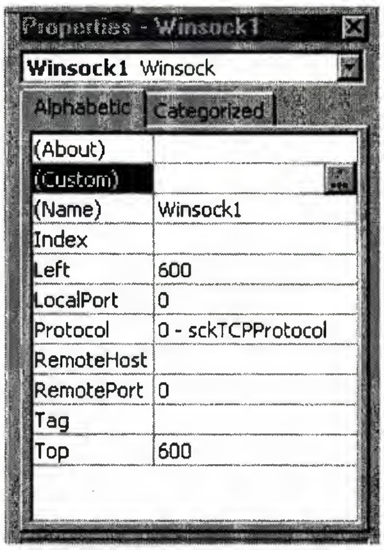


☐ Microsoft Script Control 1.0
☐ Microsoft SysInfo Control 6.0
☐ Microsoft Tabbed Dialog Control 6.0
☐ Microsoft Wallet
☐ Microsoft Windows Common Controls 5.0 (SP2)
☐ Microsoft Windows Common Controls 6.0
☐ Microsoft Windows Common Controls-2 5.0 (SP2
☐ Microsoft Windows Common Controls-2 6.0
☐ Microsoft Windows Common Controls-3 6.0
☑ Microsoft Winsock Control 6.0
☐ MSFlexGrid Wizard
☐ MSInfo MSIAV Category module
☐ MSInfo MSICDROM Category module

Microsoft Winsock Control 6.0
Location: C:\WINDOWS\SYSTEM\MSWINSCK.OCX

Figure 5.3 Selecting the Winsock Control

- 3 The control is placed on the form. (Note: The form is not visible during run-time).
- 4 Click on the Winsock control. The properties window becomes visible as shown in Figure 5.4 (Note: The name of the control is set to Winsockl).


Properties - Winsock1

Winsock1 Winsock

(About)	
(Custom)	
(Name)	Winsock1
Index	
Left	600
LocalPort	0
Protocol	0 - sckTCPProtocol
RemoteHost	
RemotePort	0
Tag	
Top	600

Figure 5.4 Winsock Properties Window

- 5 The text-box and the command-buttons are to be created on the form as shown in Figure **5.5** and names are as follows:

Message label	lblMsg
Message text-box	txtMsg
SEND command-button	cmdSend
EXIT command-button	cmdExit

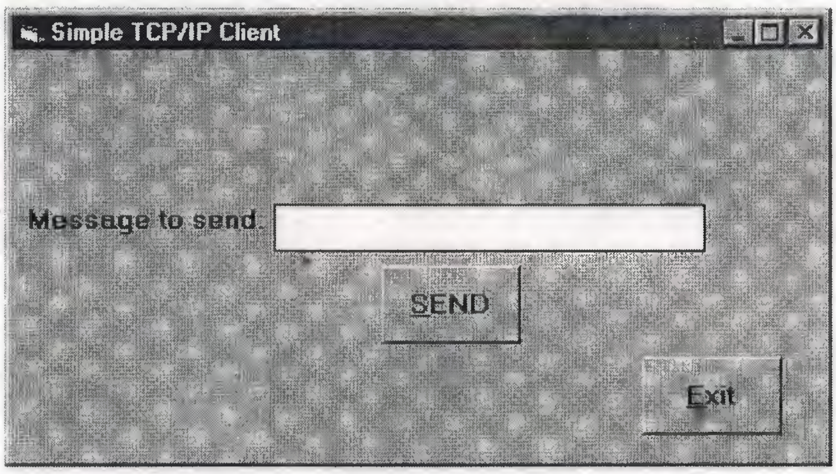


Figure 5.5 Client Form for example 1

- 6. The following code is entered in appropriate procedures:

```
'
'CLIENT PROGRAM
'Connect to server and send messages
'
Option Explicit

Private sub Form_load()
WinSock1.Protocol = sckTCPProtocol
Winsock1.RemoteHost = "SIERRA"
Winsock1.RemotePort= 1112
Winsock1.Connect
End Sub

Private Sub cmdSend_Click()
Winsock1.SendDate txtMsg.Text
TxtMsg.Text = ""
End Sub

Private Sub cmdExit_Click()
Winsock1.Close
End
End Sub
```

In procedure *Form_Load,* the protocol type is specified as TCP/IP, the remote computer name is specified as SIERRA, and the port number is set to 1112 (any other unused port number can be used). When the Send button is pressed, procedure *cmdSend_Click* is activated and this procedure sends the message, which is already in the text-box. The text-box is then cleared ready for the next message. Procedure *cmdExit_Click* closes the connection.

SERVERCOMPUTER

- 1 Visual Basic is loaded and a new project is created.
- 2 Microsoft Winsock Control is selected from Projects|Components.
- 3 The winsock control is placed on the form and the control is named as Winsock1.
- 4 The Text boxes and Command buttons are created as shown in Figure 5.6 and named as follows:

Message label	lblMsg
Received data text-box	txtRecieved
Exit command-button	cmdExit

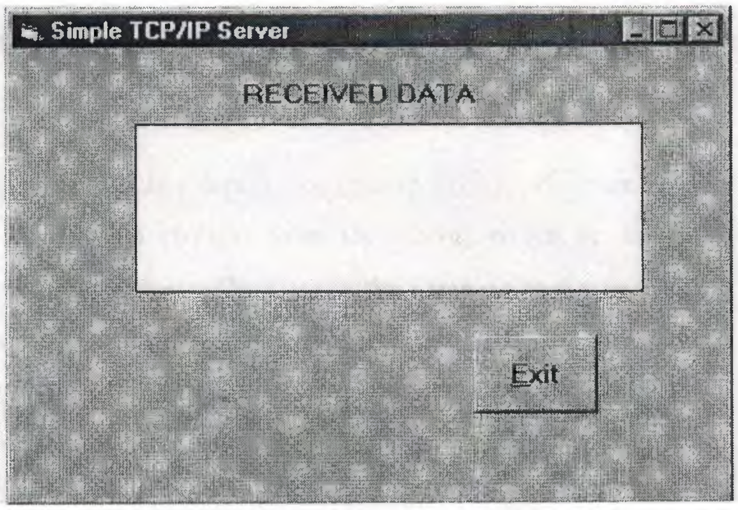


Figure 5.6 Server form for example 1.

- 5 The following code is entered in the appropriate procedures:

```
'
'SERVER Program
'
Option Explicit

Private Sub Form_Load()
Winsock1.Protocol = sckTCPProtocol
Winsock1.LocalPort = 1112
Winsock1.Listen
End Sub

Private Sub Winsock1_ConnectionRequest(ByVal requestID As Long)
If Winsock1.State <> sckClosed Then
      Winsock1.Close
End If
Winsock1.Accept requestID
End Sub

Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)
Dim msg As String
```

```
Winsock1.GetData msg
TxtReceived.Text = msg
End Sub

Private Sub cmdExit_Click()
Winsock1.Close
End
End Sub
```

Procedure *Form_Load* sets the protocol type to TCP/IP, the port number to 1112, and listens for a connection request from the client, which is accepted in procedure *Winsockl_ConnectionRequest.* The state of the listening socket is checked and closed if open. Procedure *Winsockl_DataArrival* retrieves the received data and displays this data in text-box *txtReceived.* Procedure *cmdExit_Click* is activated when the exit button is pressed and this procedure closes the TCP/IP link.

To test the program, the server program is first executed on the server machine and the client program is next executed on the client machine, the IP names used in the programs should be changed to match the TCP/IP configurations of the machines on which the programs are run. When a message is entered on the client computer and the Send button is pressed. The message is displayed on the server computer. The exit button is pressed to terminate the application.

5.4.2. Two Way Communication Example

The example in the previous section demonstrated the basic principles of TCP/IP based communication. In this section a more complicated example where both the server and the client can send and receive data is demonstrated. In addition, the program also allows the user to specify the name of the remote computer at run time so that the program can be used to connect to any server.

CLIENT Computer

The form details as shown in Figure 5.7 are created, the Winsock control is named as Wİnsockl. The other controls are named as follows:

Data to be sent label	lblSend
Received data label	lblReceİved
Remote host name label	lblHost

Send command-button	cmdSend
Connect command-button	cmdConnect
Exit command button	cmdExİt

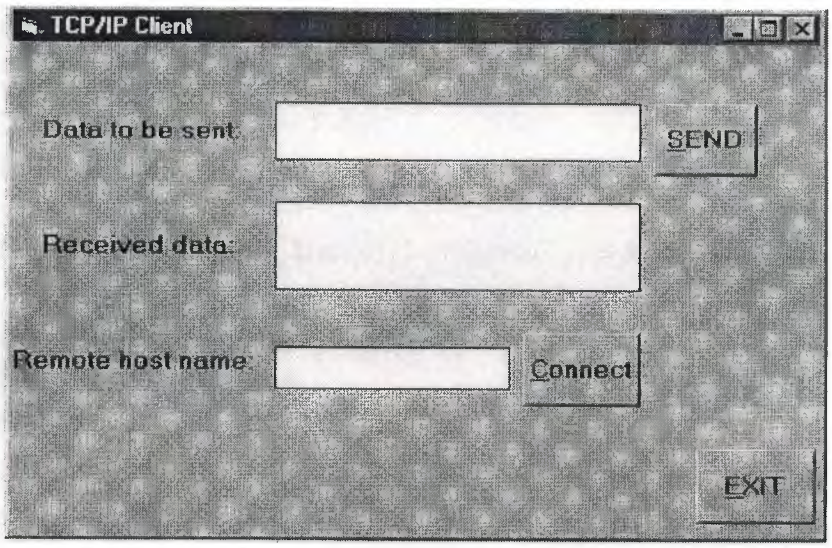


Figure 5.7 Client form for example 2

The following code is entered into the appropriate procedures:

```
'CLIENT Program
"Connect to Server and send and receive messages
'
Option Explicit

Private Sub Form_Load()
Winsock1.Protocol = sckTCPProtocol
CmdSend.Enabled = False
End Sub

Private Sub cmdConnect_Click()
Winsock1.Conncect txtRemote, 1112
End Sub

Private Sub cmdExit_Click()
Winsock1.Close
End
End Sub
```

```
Private Sub cmdSend_Click()
Winsock1.SendData txtSend.Text
TxtSend.Text=""
End Sub

Private Sub Winsock1_Connect()
CmdSend.Enabled = True
CmdConnect.Enabled = False
End Sub

Private Sub Winsock1_DataArrival(ByVal bytes Total As Long)
Dim msg As String
Winsock1.GetData msg
TxtRecieved.Text = msg
End Sub
```

Procedure *Form_Load* defines the protocol type as TCP/IP and disables the Send command button on the form. This button will be enabled after a connection is made to the client. Procedure *cmdConnect_Click* initiates a connection to the server on port number 1112. Procedure *cmdExit Click* closes the connection and terminates the program. Procedure *cmdSend_Click* sends the data in the text-box *txtSend* to the remote machine. Procedure *Winsockl_Connect* is a winsock event driven procedure and is activated whenever a connection is established. The Send command-button is enabled and the Connect command-button is disabled in this procedure. Procedure *Winsockl_DataArrival* is activated when the program receives a data packet. The received data is displayed in the text-box called *txtReceived.*

SERVER Computer

The form is created as shown in Figure 5.8 two identical Winsock controls are loaded on the form both are named Winsock1, the index of one of the controls is set to 0 and the other one to 1. The various controls on the form are named as follows:

Received data label	lblReceived
Received text-box	txtReceived
Data to be sent label	lblSend
Data to be sent text-box	txtSend
Send command-button	cmdSend
Exit command-button	cmdExit

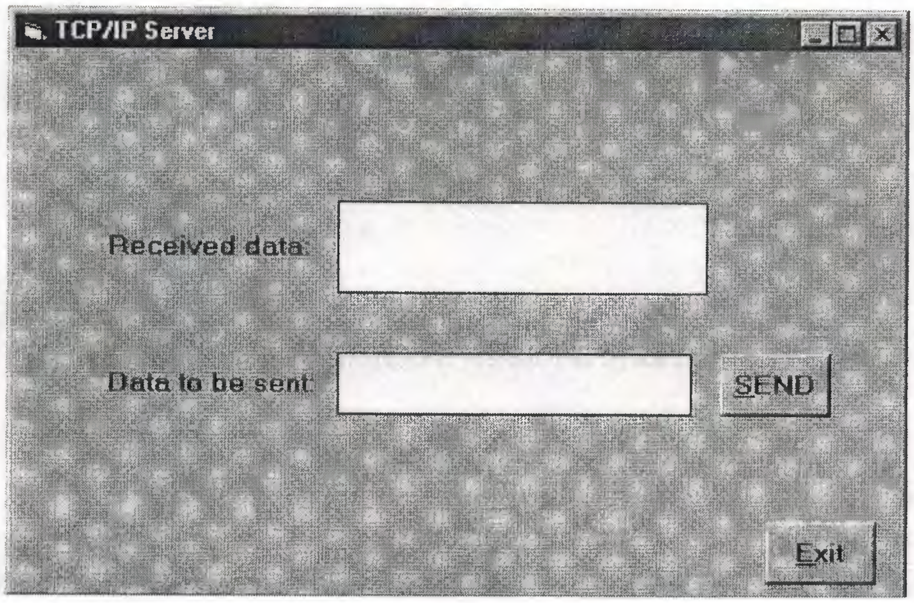


Figure 5.8 Server form for example 2

The following code is entered in appropriate procedures

```
'SERVER Program
'Accept connection from the client and display received data
'
Option Explicit

Private Sub Form_Load()
cmdSend.Enabled = False
Winsock1.Protocol = sckTCPProtocol
Winsock1.LocalPort = 1112
Winsock1.Listen
End Sub

Private Sub cmdExit_Click()
Winsock1.Close
End
End Sub

Private Sub cmdSend_Click()
Winsock1.SendData txtSend.Text
txtSend.Text=""
End Sub
```

```
Private Sub Winsock1_ConnectionRequest(By Val requestID As Long)
If Winsock1.State <> sckClosed Then
Winsock1.Close
End IF
Winsock1.Accept requestID
CmdSend.Enabled = True
End Sub

Private Sub Winsock1_DataArrival(By Val bytesTotal As Long)
Dim msg As String
Winsock1.GetData msg
TxtRecieved.Text = msg
End Sub
```

Procedure *Form_Load defines* the protocol as TCP/IP, defines the local port and then listens for a connection. Procedure *cmdSend_Click* sends the text in text-box *cmdSend* to the remote computer. Procedure *Winsockl_ConnectionRequest* accepts the connection request from the client. Procedure *Winsockl_DataArrival* retrieves the received data and displays it in text-box *txtReceived.*

For testing the programs the server program should be executed first on the server and then the client program should be executed on the client. Messages from both sides of the connection can then be sent.

These above examples demonstrate the use of TCP/IP protocol, the use of visual basic programming language and network connections between the machines. In these examples no special hardware connection was required as no interface was designed with external devices. These same principles of data exchange can be applied to drive electronic switches and other devices attached to computers on remote locations, to automate residential or industrial processes.

Chapter Six: Application Development Process

In this chapter the Application principles of Internet Based Automation are discussed using as an example monitoring system of ovens in a factory from a remote site. The ovens are located at a site geographically far from the monitoring site, data indicating the state of the ovens is transmitted via the Internet. Packets are generated by client software and the server analyzes the data on receipt of these packets using analyses software. The server displays graphs and text data regarding the state of the ovens to the users monitoring the system, the process is depicted in Figure 6.1

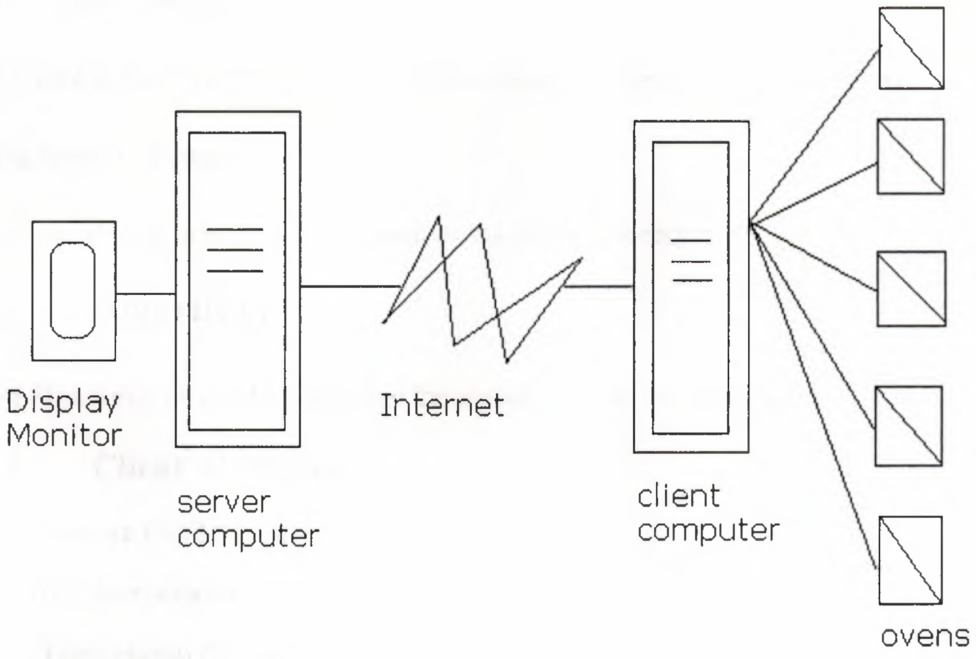


Figure 6.1 Remote Monitoring System for Ovens

6.1. Hardware

The type of card used depends on the applications, serial I/O card can be used to control serial device and digital I/O card to control parallel devices depending upon the application one can also use A/D (Analog to Digital) or D/A controllers, Timers, Watchdog interrupt devices etc. One popular digital I/O card is the PC 200 series designed and manufactured by Amplicon Limited this card has many variants but it basically provides a minimum of 24 digital input outputs and a number of timers, this card can be used to control and monitor most on-off type applications. Another card from Amplicon is the PC30, which includes up to 8 A/D converters these converters can be used to monitor and control analog devices such as temperature pressure, fogs and lights etc. In the application discussed in this chapter the A/D card is used to monitor five ovens, which are located hundreds of miles away from the monitoring site, this application uses temperature sensors to read the temperature of the ovens to provide an analog signal proportional to the temperature to the A/D converter. The client computer can then read the temperature and send the data to the server computer where it can be analyzed remotely.

On the Client Computer

PC 30 card is used for interfacing with the temperature sensors from the ovens.

On the Sever Computer

No special device is used, a connection to the Internet is required.

6.2. Algorithm

Using Structured Definition language the algorithms can be described as follows:

6.2.1 Client Algorithm

```
Do Forever or Until Stopped
    Get temperature
    Time stamp the data
    Construct packet
    Send packet to server
    Save data in database
    Wait a second
End
```

6.2.3 Server Algorithm

```
Do Forever or Until Stopped
        Wait for packet
        Receive packet
        Check accuracy
        Separate data for ovens
        Extract temperature
        Display Temperature
End Do
```

6.3. Software Description

A custom software has to be written in visual basic, the software using the input from the A/D converters constructs a packet, a possible structure of this packet is shown in the Figure 6.2, and transfers it to the server over the Internet using the TCP/IP protocol. The server then processes this data.

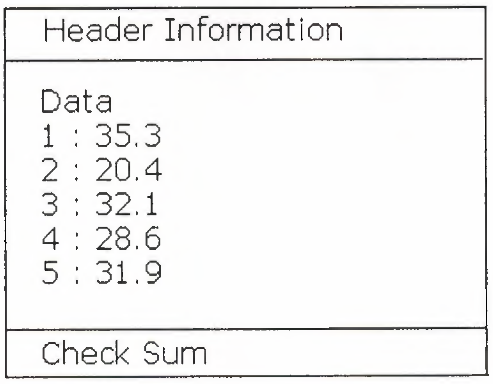


Figure 6.2. An example of the constructed Packet.

The client software is written using the visual basic language. Timer control will be used to read the data from the A/D converter at regular intervals (one second). After reading the data, it will be time stamped and a packet to be sent over the Internet will be constructed using the TCP/IP protocol, appropriate header information will be added also a checksum will be added at the end to control the accuracy of the transmission, after sending the packet the data will also be saved in a local database. After waiting for a second the process starts again. This process continues forever until it is manually stopped.

At the receiving side the server waits for the packet, on receipt of the packet it extracts the temperature and displays the temperature in text as well as in the form of a graph. If there is over heating detected in any of the ovens or if any oven does not work due to malfunctioning warning signals are displayed and user action is requested.

Chapter Seven:Case Studies

7.1. Intrusion Detection System.

Cheryl is a homeowner who travels frequently for business. One night while she is sleeping in a hotel out of town, an intruder attempts to break into her home. She immediately gets this information through Internet based automation and is aided in preventing any loss.

The intruder breaks a door window attempting to gain entry into Cheryl's home.

The Glass breakage and entry sensors immediately notify the Internet Based Automation appliance.

Cheryl has programmed her system to follow this routine:

- The house lights turn on.
- A wireless color camera near the back door captures a video clip.
- A 105-decibel alarm sounds.

The appliance sends a pager notification to Cheryl and an e-mail message (including photograph of the intruder's attempt) to her laptop.

The automation appliance also notifies a security monitoring service that dispatches police to Cheryl's house in case the intruder was not scared away by the alarm.

Cheryl further logs on to her PC and sees photos and videos of her home.

Briefly security devices using Internet Based Automation worked well when Cheryl was away from home by doing the following:

- Glass breakage and entry sensors detected an Intruder.
- They notified the "Internet Based Automation" Appliance.
- The house lights turned on.
- A camera captured video of the intruder.
- An alarm sounded.
- A pager was sent to Cheryl.

- An e-mail including the picture was sent to Cheryl's Laptop.
- A security-monitoring device was notified.

7.2. A Day in the Life

Glen and Jenny Sheppard rent a house, they both work full-time and lead active social lives. Here's how "Internet Based Automation" fits into their typical workday.

Glen is the last to leave the house in the morning. He presses a button on his key chain home control to set the house to the "Away" mode, which activates the following routine:

- ❑ Lights turn off.
- ❑ Security devices engage
- ❑ Temperature Lowers

Several Hours Later, both Jenny and Glen receive a pager notification of motion detected at their front door. They also receive an e-mail with an attached picture showing their neighbor putting a package near the door.

Jenny makes plans to go out for dinner with friends that night and enters her plans into the automated calendar. She notices on her calendar that her nephew's birthday is coming up, so she will need to get a gift. She then synchronizes the calendar with her palm pilot.

At 5 p.m., as a scheduled weekday routine, the in-home appliance automatically, raises the thermostat setting.

As Glen approaches his home, he presses a button on his key chain home control that sets the house to the "Home" mode.

- ❑ Lights turn on.
- ❑ Motion and entry sensors disengage

Briefly on a typical day, Jenny and Glen use Internet Based Automation to do a number of things:

- Set the house to "Away" mode.

- Engage the security devices.
- Turn off the lights
- Lower the temperature.
- Be notified of motion detected at their door.
- Synchronize their desktop PC calendar with their Palm Pilot.
- Raise the thermostat setting.
- Set the house to "Home" mode.
- Disengage motion and entry sensors.
- Turn on lights.

7.3. Cabin Preparation for the Weekend

Tim lives in the city and owns a lake cabin, during the week he is busy in the city with his job and on the weekends he goes fishing at his lake cabin. Tim uses Internet Based Automation to save time on the weekend.

On a typical weekend, Tim logs on to his PC at the cabin and checks the weather at the cabin, the status of the security system and safety devices.

He looks at the recent pictures and videos taken from inside and outside the cabin and sees that the lake is perfect for fishing.

While driving to his cabin, Tim uses his web phone to easily initiate a routine that automatically changes conditions at his cabin. He does the following:

- Turns on the water pump.
- Starts the water heater.
- Sets the thermostat to 68 degrees

When Tim approaches his cabin, he uses the hand-held home control to:

- Turn on the lights.
- Deactivate the entry and motion sensors.

To summarize on a weekend visit to his lake cabin Tim could save time by automating the following activities:

- Log on to his cabin PC and check the weather.
- See the status of the home's security and safety devices.
- View photos and video via camera.
- Turn on the water pump.
- Start the water heater.
- Set the thermostat.
- Turn on the lights.
- Deactivate the entry and motion sensors.

7.4. Emergency Situation

John has recently purchased a home and renovated it, he has also installed Internet Based Automation devices in his house.

One night he presses a button on his hand-held control to set the "Asleep" mode for his house and goes to sleep.

Early in the morning a pipe bursts in the basement, the water sensor located on the basement floor detects the water immediately.

The moisture sensor alerts the home automation devices, which initiate a routine to alert and awaken John as follows:

- ❑ The house lights flash three times and stay on
- ❑ A chime alert goes off.
- ❑ An alert notification is sent to his PC's display.

John checks the display by his bed and turns off the water supply preventing damage from a flooded room.

In brief the home automation system helped the homeowner by preventing a pipe burst from becoming a major loss by performing the following actions:

- A moisture sensor detects water.
- It immediately alerts the automation server that initiates a routine.
- The house lights flashed three times and stayed on.

- A chime alert went off.
- An alert notification was sent to the PC display.

7.5. Asset Tracking

MinTime Corp is a transportation company that specializes in moving high value machinery all over the globe, via ships, trucks and other modes of transport. It uses Internet based Automation using GPS (Global Positioning System) system to keep track of its cargo.

Each piece of equipment to be transported is fitted with wireless GPS pinpointing devices. These devices continuously feed the location of the equipment via GPS positioning data to on-site servers.

When any equipment leaves a warehouse, a path for its travel is defined. If during travel it deviates from its path, the following process is started:

- ❑ A message is send immediately via pager to the person in charge of transportation of that equipment and to his/her supervisor.
- ❑ An email message is also sent to the person in charge and to his/her supervisor and also to the owner of that equipment, describing the current location of the equipment and the deviation from its course.

To summarize track can be kept of assets during transportation by using Internet Based Automation as follows:

- Tagging each asset with GPS pinpointing device.
- Defining a course for their travel.
- Deviations from the course can lead to broadcast of emergency notification messages.

7.6. Industrial Automation.

Delco Corp a manufacturing company requires a large, expensive piece of test equipment to support its processes. Since they do not need the equipment regularly they want to reduce expenses associated with its use by allowing other companies access to this resource.

They have designed a solution using Internet Based Automation by allowing remote users control of this equipment over the Internet, using this a sample can be sent by mail to Delta Corp where it can be analyzed remotely. This saves the testing company travel cost and loss of employee time.

Delta Corp has used the following mechanism to set up the system:

- Installed interface devices to control the testing equipment via the Internet.
- Installed video cameras to relay the complete testing process.
- Data is collected via Sensors and transmitted to the testing company.

The testing company can change the test conditions by controlling the testing equipment remotely.

In brief Delta Corp has improved profitability by using Internet based Automation by using the following process:

- Samples to be tested can be sent by mail.
- The testing process is captured on video and sent to the testing company by Internet.
- The testing company can change the test conditions by controlling the testing equipment via the Internet.

Chapter Eight: Conclusion

The project has described the design principles of Internet based remote automation. It is shown that the TCP/IP protocol and the Visual Basic programming language can be used to design Internet based Automation for household and Industrial applications. This project also describes the TCP/IP protocol, the visual basic language and the Internet and World Wide Web. Network applications such as those described in this project can save money and time for individual and corporations. Increasingly all house hold and Industrial appliances are being built with network interface devices this allows the device to be connected with and controlled by a server, this server can then receive instructions over a network. In the near future a larger proportion of the Internet will be used for Internet Based Automation applications.

Chapter one presents a background to this project, chapter two presents the details of TCP/IP protocol, chapter three is an introduction to the Visual Basic programming language, and chapter four describes the Internet and the World Wide Web. The aim of the project was to describe Internet Based Automation technologies, methodologies, commercial and personal application possibilities the same were demonstrated in chapter five, chapter six and chapter seven.

Developments in this field are taking place rapidly and new products and ideas are being presented. Notable among these developments are the Blue Tooth specifications, which are a defacto standard; it requires all devices to have wireless network connectivity. Many commercial products are currently available for home and factory automation, including monitoring systems. Hardware and software for GPS asset tracking systems, failure detection systems for processes etc. are being designed.

Security for Internet based Automation among others involves security of the data transferred over the Internet various options are available and can be used depending on the situation. Some of the available options are: building appropriate firewalls so that only appropriate date can pass through, Encryption of data is another method that can be used for securing data between client and server, also VPN (Virtual Private Networks) can be used, a VPN uses software or hardware to encrypt all the traffic over the Internet between two predetermined end points. Another solution that can be used is Digital

Certificates, issued by a certification authority (CA) server or external CA service these certificates can be used to provide access to users who possess these certificates.

References

Reference to books

[1.]Gary Cornell and Troy Strain, *Visual Basic 4 Nuts & Bolts: For Experienced Programmers*, Osborne McGraw-Hill, California 94710.

[2.]Mary Anne Pike et al, *Using the Internet with Windows 95*, Que Corporation, Indianapolis 201 W.

[3.]Dr.Dogan Ibrahim, *Using TCP/IP from Visual Basic*, Under Print.

[4.]Frank J.Derfler,Jr., *Guide to connectivity*., ZIFF – Davis Press, USA.

[5.]Douglas. E. Comer., The Internet Book, Prentice Hall International.,N.J 07632.

Reference to Electronic Sources – Online Sources from Web

[1.]Peter G. Aitken,*Objects in Visual Basic*, Retrieved from http://www.pgacon.com

[2.]The Blue Tooth Specifications, Retrieved from http://www.bluetooth.com/