MASTER THESIS

TITLE: Software support for video streaming from smartphone on small airship

MASTER DEGREE: Master in Science in Telecommunication Engineering & Management

AUTHOR: Miguel Carretero Rodríguez

SUPERVISOR: Zbynek Soban

DIRECTOR: Lukas Kencl

DATE: February, 9th 2009

Overview

This project has been created to study and to solve two goals. The main goal has been streaming real video from a smartphone to a laptop and the second goal has been to control the airship electronics from the smartphone.

To achieve the main goal of the project, firstly has been necessary realize a technologies study to select the better choice. The chosen technology has been to use of wi-fi, mainly due to the high transmission speed. Once the technology has been chosen, an ad-hoc wi-fi network has been created to be able to transmit wirelessly. After that, it has been necessary to study how to do the sending of the information. The first idea was using the User Datagram Protocol or Transport Control Protocol, at the beginning it seemed a good choice, because of was possible to achieve a communication between the smartphone and the laptop and a camera in record mode application was created. However, when the real video was to be sent to the laptop, it was not possible because it was not way to open the file while it was been created and there was not any temporal file. Then, another possibility had to be studied and it was the use of streamers. The Live555 Streaming Media is a set of C++ libraries which allows to stream real time from a device to a computer. After the hard work to run this application, it was running fine on the laptop, as well as, the laptop's webcam was working fine. However, after try to develop a smartphone's application was discovered that these libraries are not support for Windows Mobile. Therefore, a new way was discovered, the use of DirectShow which is architecture for streaming media, it seemed would work fine, but after the difficulty to compile a code, the smartphone was not detected and there weren't so much information about how to solve the related problems. Finally, a real time software has been used to stream real video from the smartphone to a laptop. There have been two possibilities, Qik or AtekSoft Web CameraPlus. However, the second one was descarted due to the obligation to pay for use the software.

The second goal was studied to develop a serial port application to be used by Infra Red port. However, it was possible that smartphone wouldn't have IR Port, so a second alternative was studied and it was using the mini USB connection to achieve through a mini USB to USB adapter and an USB to serial port converter to communicate using serial port communication, but this possibility was not support by Windows Mobile. Then, a possible solution is described, it is using the smartphone like an USB device and on the other side (electronics) the use of an USB host and through drivers allows the communication.

The smartphone used has been HTC Touch 3G which uses Windows Mobile 6.1 Operative System. The laptop used has been Dell XPS M1330, which uses Windows Vista Operative System the programming software has been Visual Studio 2008.

In conclusion, the project has helped me to improve knowledge, on the one hand in my studies and on the other hand in my foreign language. It has been a good experience to work in a real project in the RDC department.

Table of Contents

List of figures

1 Introduction

Due to the continuous technologic advances, constantly, are appearing new tools, software..., definitively, new ways to do easier and better the life.

The Software support for video streaming from smartphone on small airship project is created to take advantage of the new opportunities offered by new products, like for instance the smartphones. The goals of the project are described in the following points and are illustrated in the **Figure 1.1**:

- At present, the idea of getting stream video, for instance from internet, into a smartphone is not a new idea, because of there are different possibilities to get it and almost all the new smartphones include a solution to achieve this objective. However, the idea to stream real video from a smartphone to a computer it is not very common; thus, the main challenge in our project is to study how to develop a new solution for this requirement. Therefore, the main goal is to carry video through one wireless medium. The smartphone, the airship and the laptop are the elements which appear in this goal. The main idea is to develop an application to be used in Windows Mobile, which is the smartphone's Operative System, in the smartphone which will be joined to the airship. This application has to be able to activate the camera in record mode and send the real video through a wireless medium, the possibilities studied has been wi-fi, gprs and Bluetooth, to the laptop. Once the video is being sent, in the laptop has to be software to reproduce the video.

- The second goal is to control the airship. The elements includes in this goal are the smartphone, the airship and the airship's electronics. The main idea is to develop an application by using Windows Mobile to send some commands to the airship's electronics with the finality to control the airship movements.

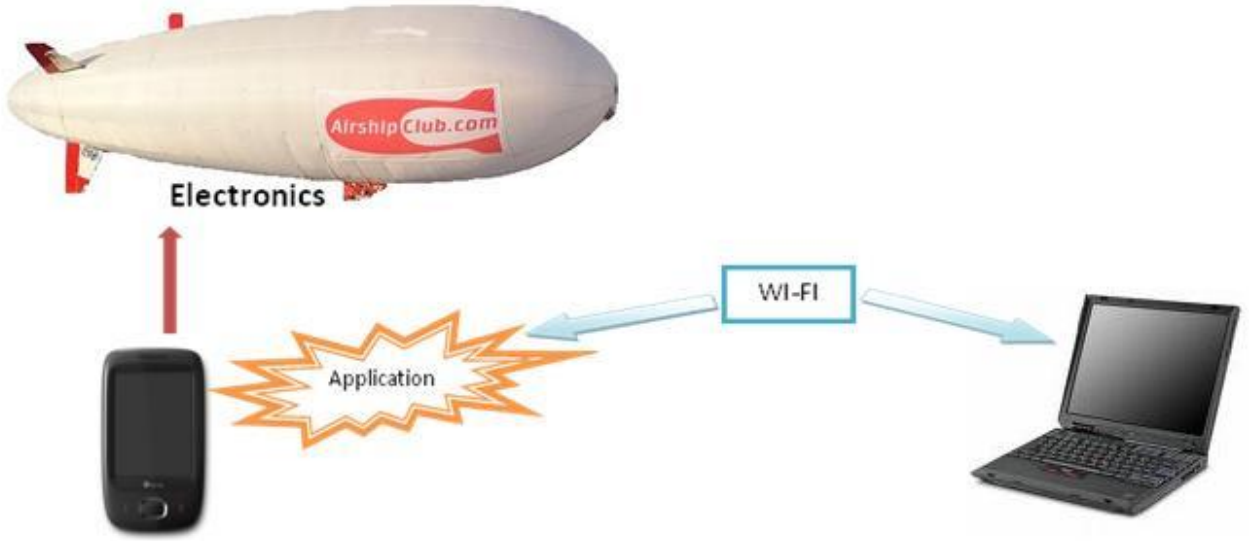

Figure 1.1 – General project diagram

2 Related technology

In this chapter is explained the main characteristics of the device used in the project, as well as, are explained the different possible technologies to realize the main project goal and the chosen technology.

2.1 HTC Touch 3G

In this section, there is a brief description of the smartphone used in the project, as well as, its main specifications.

HTC Touch 3G **Figure 2.1** runs on high-speed 3G networks with download speeds up to 18 times faster than standard 3G. Touch 3G integrated with Windows Mobile 6.1, it also has HTC's TouchFLO user interface letting users control the menu through finger gestures when scrolling through contacts, launching media and browsing the web by zooming and panning websites with one-hand. Users can search for and watch streaming video from YouTube, get directions and mapping using Google Maps for mobile and stay updated on the latest news with the integrated RSS reader.

Next, are showed the main HTC Touch 3G's characteristics:

- 2.8-inch touch screen, with four times the resolution of most phones.
- Powered by Windows Mobile® 6.1 Professional.
- Next generation TouchFLO™ user interface, responding perfectly to your finger gestures when scrolling through contacts, browsing the web and launching media.
- Small, lightweight design available in a range of vibrant colours.
- Surf and download at broadband speed with HSDPA internet connectivity.
- 3.2 megapixel camera for quality stills and video.
- microSD™ slot for expandable storage. (1)

Figure 2.1 – HTC Touch 3G

2.2 Wireless technology used

The communication between the smartphone and the computer has to be wireless because, as it is known, the smartphone is joined to the airship and this will be flying, so, it is not practical to connect a cable between the two devices. Thus, has been necessary realize a study of the different wireless technologies to be used in the project. In the following paragraphs will be explained the studied technologies, wi-fi, gprs and bluetooth.

2.2.1 Wi-Fi

Wi-Fi (short for "wireless fidelity") is a term for certain types of wireless local area network (WLAN) that use specifications in the 802.11 family. The term Wi-Fi was created by an organization called the Wi-Fi Alliance, which oversees tests that certify product interoperability. A product that passes the alliance tests is given the label "Wi-Fi certified" (a registered trademark). Originally, Wi-Fi certification was applicable only to products using the 802.11b standard. Today, Wi-Fi can apply to products that use any 802.11 standard. The 802.11 specifications are part of an evolving set of wireless network standards known as the 802.11 family. (2)

The studied has been centred in the IEEE 802.11g standard, which has the following main properties:

- Cover distance: 100m
- Speed: Up to 54 Kbps
- Frequency: 2,4 GHz

2.2.2 Bluetooth

Bluetooth is a wireless protocol for exchanging data over short distances from fixed and mobile devices, creating personal area networks (PANs). It was originally conceived as a wireless alternative to RS232 data cables. It can connect several devices, overcoming problems of synchronization. Unlike other wireless transmissions technologies, Bluetooth is intended to be used by devices that are close to each other. For instance, a PDA could transfer data such as a phone number to a cell phone. A laptop could transfer data such as e-mails to a PDA. With this technology, the need for wires and proprietary file transfer software could be eliminated.

Some examples of Bluetooth applications are:

- Wireless networking between desktops and laptops.
- Bluetooth peripherals such as printers, mice, and keyboards.
- Bluetooth cell phones have been sold in large numbers, and are able to connect to computers, personal digital assistants (PDAs) and, specifically, to hands free devices.
- Certain mp3 players and digital cameras to transfer files to and from computers.
- Bluetooth car kits that allow users with Bluetooth-equipped cell phones to make use of some of the phone's features, such as making calls, while the phone itself can be left in a suitcase or in the boot/trunk, for instance.
- For remote controls where infrared was traditionally used. (3)

Next, are showed the main Bluetooth properties:

- Cover distance: 10 m
- Speed: 1 Mbps
- Frequency: 2,4 GHz

2.2.3 GPRS

General Packet Radio Services (GPRS) is a packet-based wireless communication service that promises data rates from 56 up to 114 Kbps and continuous connection to the Internet for mobile phone and computer users. The higher data rates allow users to take part in video conferences and interact with multimedia Web sites and similar applications using mobile handheld devices as well as notebook computers. GPRS is the first high-speed digital data service provided by cellular carriers that used the GSM technology. GPRS added a packet-switched channel to GSM, which uses dedicated, circuit-switched channels for voice conversations. (4)

The main properties of the General Packet Radio Services are showed following:

- Cover distance: It is the same that the network covered by the mobile operator.

- Speed: Up to 114 Kbps.
- Cost: It is imposed by the mobile operator, but it is charged by the data traffic.

Once studied the three different technologies, the chosen option has been the use of Wi-Fi, due to:

- The highest transmission speed.
- Free cost for data transmission.
- The airship won't be very far of the receiver, laptop.

2.3 Creating an ad-hoc Wi-Fi network

One of the goals of the project is to send the video in real-time through a wireless medium. The best idea thought has been about to create an ad-hoc Wi-Fi network. In this paragraph is demonstrated how to create this network.

The network in the computer has been created following the next steps:

1. Select "configure a connection or network" inside networks centre and share resources, which is inside networks and Internet which is inside the control panel.
2. Choose the option "configure an ad-hoc wireless network".
3. In the network settings a network name has to be set up and if it is necessary a network security also has to be set up.
4. Now, an ad-hoc network has been created, which can be used for whoever user.

Once, the network is created, we have to configure the computer's network card settings, to be able to create a LAN (Local Area Network) between the computer and the smartphone.

1. On the wireless network connection we have to choose the Internet version4 protocol properties and it will appear the window showed in the **Figure 2.2**:

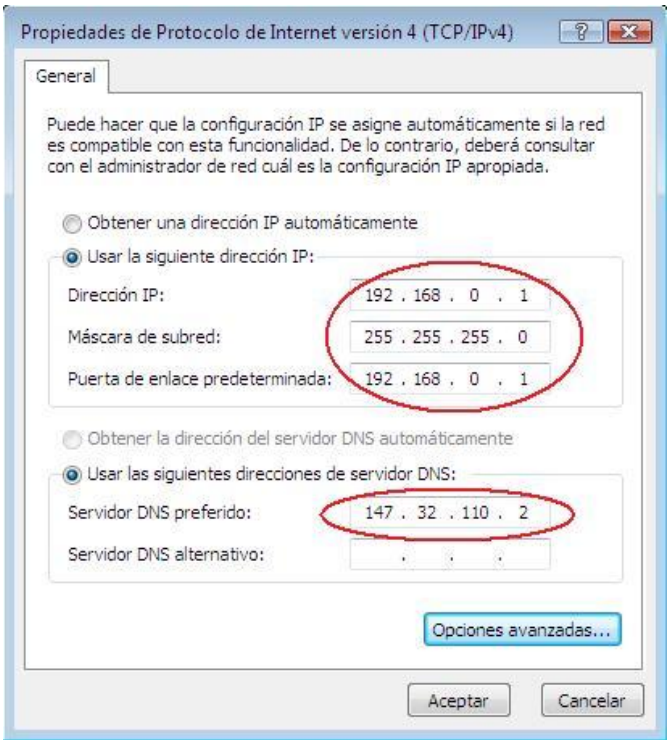

Figure 2.2 – IP addresses assignation

2. To configure to network we have put the addresses showed in the last step as well as the domain name server, if we want to have internet in the smartphone.
3. After the last step, we have to select properties of the network which we are connected to Internet. We have to choose the second label, called shared use and allow the connection of other users from this computer, like the **Figure 2.3** is showing:

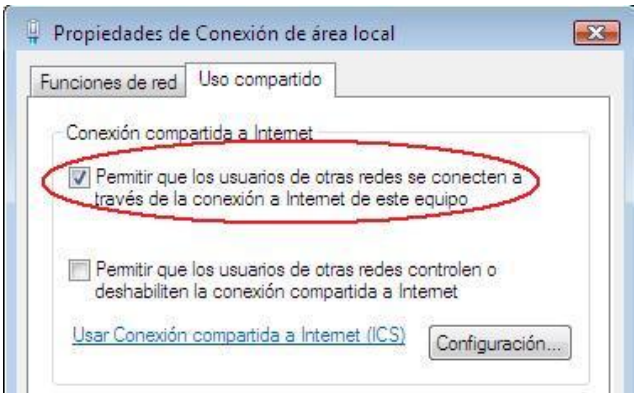

Figure 2.3 – Shared use

4. Once done, the network created in our smartphone has to be configured.

5. Firstly, the network has to be detected and the ad-hoc network properties have to be introduced in it.
6. After that, the “network adapters” option has to be selected and select the option “IEEE 802.11b/g Compatible Wi-Fi Ad-Hoc”, select IP Address and the following addresses have to be introduced:
    a. IP Address: 192.168.0.2
    b. Subnet mask: 255.255.255.0
    c. Default gateway: 192.168.0.1
7. The name server option has to be selected and introduce the same address which was introduced in the second step.

Finally, the ad-hoc Wi-Fi network is ready to be used and to accede to Internet through the Smartphone.

3 Solution

In this chapter are showed different ideas which are been developed or studied with the finality to get the main goal of the project.

The **Figure 3.1** shows a global architecture of the system when is used the Camera in record mode application. However, as we can see in the figure, appears the used hardware, Operative System and application. In the different used solutions, the hardware and the Operative System are the same that in this example. Then, the difference between the purposed solutions is the used application, because, as it is showed in the following sections, the application in the smartphone and in the laptop won't be the same for each case, because of the necessities are different.

.

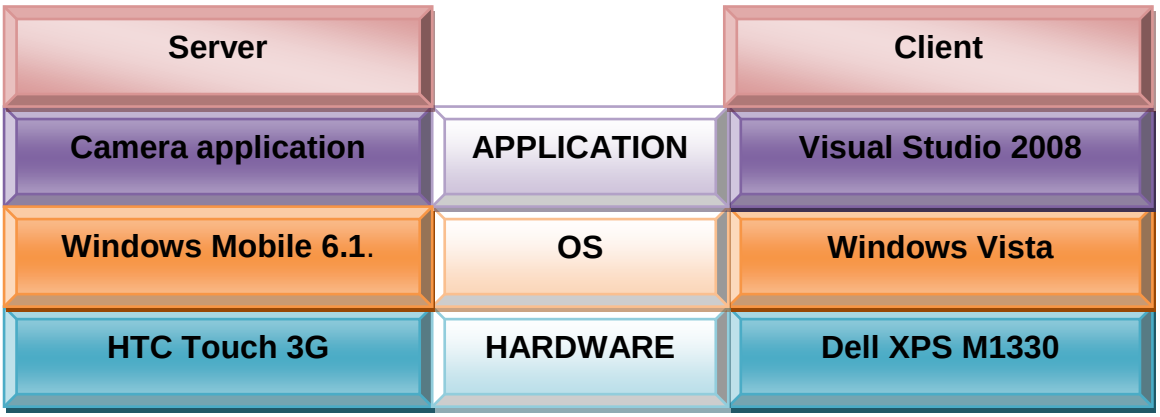

Figure 3.1 – Global architecture

3.1 Camera in record mode application

The first objective of this project has been to stream video from the smartphone's camera to a computer. Therefore, the first goal to get it has been to have access to the camera. The method used has been to use a generic camera API (Application Programming Interface) provided by Windows Mobile 5.0 which has allowed to utilize the CameraCaptureDialog which is included in the "Microsoft.WindowsMobile.Form" namespace. The application has been developed in C#. Next, the main properties of the CameraCaptureDialog class are showed in the following code:

```
public void takepicture()
{
    CameraCaptureDialog cameraCaptureDialog = new
    CameraCaptureDialog();
    cameraCaptureDialog.Owner = null;
    cameraCaptureDialog.Title = "Take Video";
```

```
        cameraCaptureDialog.Mode = CameraCaptureMode.VideoOnly;
        cameraCaptureDialog.Resolution = new Size(176, 144);
        cameraCaptureDialog.DefaultFileName = @"videos2.mp4";
        cameraCaptureDialog.InitialDirectory = @"\My Documents";
        cameraCaptureDialog.VideoTypes =
        CameraCaptureVideoTypes.Standard;
        cameraCaptureDialog.ShowDialog();
    }
```

The code showed has been the used in the application, however, a general vision about the CameraCaptureDialog class is showed in **Figure 3.2**, where it is possible to observe the different properties that the class allows us to use, as well as, the methods. There are different properties where the developer has to introduce a name, like for instance, the Title property, although there are properties where the developer has to choose between different options, like for instance, the Mode property where it is possible to select the mode of capture. In the following paragraph are explained all the properties that are been used in the camera in record mode application.

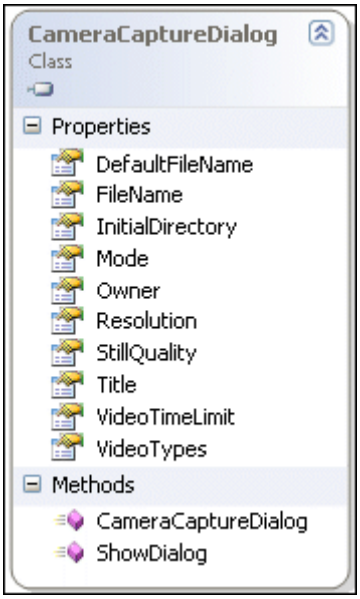

Figure 3.2 – CameraCaptureDialog class

The code showed allows set up different options necessary to take the video:

- Define the title of the video which will be recorded.
- Set up the mode of the camera, there are three kinds of modes:
  - Still -> This mode takes a snapshot.
  - VideoOnly -> This mode records without audio and it has been the mode selected in the application developed due to initially it is not necessary the audio.
  - VideoWithAudio -> This mode allows to record video and audio.
- Define the resolution of the video.
- Define the name which will be saved the video file.

- Establish the path where will be saved the file.
- Set up the format of the video. There are two kinds of formats:
  - All -> Used to make the Resolution property determine which type of video to record.
  - Messaging -> Used for recording MMS videos
  - Standard -> Used for recording WMV videos. It has been the option selected but the video is stored using the 3GP (3rd Generation Partnership) which store files using the standards H.263 and mpeg4.
- Finally, the ShowDialog() method launches the Camera dialog.

In this point, the access to the camera is successfully, the code showed, activate the Camera Dialog, but to start to record the video it is necessary to press the “start button”. For this reason, has been necessary to research about how to simulate the press button action automatically.

Firstly, the SendKeys class was used to simulate the press button action, however a lot of problems appeared and it was not possible to work fine. Therefore, another function has been used, it is called Keybd_event() and it allows to simulate a keystroke.

Afterwards, the application has been tested, but there was the problem that while the Camera dialog was open, the application was locked and the code did not advance until the keybd_event(). So, the use of threads has been necessary to allow to the application’s functions run independently.

The following code is showing the process followed to activate the camera in record mode automatically:

```
public Form1()
{
    InitializeComponent();
    Thread th1 = new Thread(new ThreadStart(takepicture));
    Thread th2 = new Thread(new ThreadStart(accept));

    th1.Start();
    th2.Start();

    th1.Join();
    th2.Join();
}
```

3.2 The sending of the information

After the camera application worked fine, has been necessary to think about the method to send and receive the information between the smartphone and the computer. The used method has been the implementation of sockets. The socket allows the exchange of data between two applications. It is defined by three points:

- IP address
- Protocol
- Port number

UDP Socket and TCP Socket client-server applications have been developed in the project. Next, the main properties and the developed applications of both sockets are explained.

3.2.1 Socket UDP (User Datagram Protocol)

The User Datagram Protocol was designed by David P. Reed in 1980 and formally defined in RFC 768 (5). Next, the main properties of the sockets UDP are showed:

- Connectionless socket. The data can be sent without preview agreement between the two end points.
- Don't guarantee that the datagram will be delivered, but if it is delivered, it will arrive without errors.
- The data could arrive in different order that it was sent.

The first client-server application has been developed using socket UDP. It has been checked using a client and a server in the local machine, because at the beginning of the project the smartphone was not available. The first application has consisted in an exchange of strings, just to demonstrate that the communication exists. However, when the smartphone arrived, it application was adapted to be used by a smart device project. Next, the application code is showed:

```
        /*---------------------- SERVER -------------------------*/
        private void menuItem1_Click(object sender, EventArgs e)
        {
         EndPoint ip = new IPEndPoint(IPAddress.Any, 9999);

         Socket socket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

         socket.Bind(ip);

         EndPoint Remote = new IPEndPoint(IPAddress.Any, 0);

         byte[] data = new byte[25];
         int receivedDataLength = socket.ReceiveFrom(data, ref
Remote);
         MessageBox.Show(Encoding.ASCII.GetString(data, 0,
receivedDataLength));

         byte[] data2 = new byte[25];
         string welcome = "Hi, I'm the Smartphone";
         data2 = Encoding.ASCII.GetBytes(welcome);
         socket.SendTo(data2, data2.Length, SocketFlags.None, Remote);
```

```
         byte[] data3 = new byte[25];
         int receivedDataLength2 = socket.ReceiveFrom(data3, ref
Remote);
         MessageBox.Show(Encoding.ASCII.GetString(data3, 0,
receivedDataLength2));

         byte[] data4 = new byte[25];
         string goodbye = "Goodbye laptop";
         data4 = Encoding.ASCII.GetBytes(goodbye);
         socket.SendTo(data4, data4.Length, SocketFlags.None, Remote);
        }

        private void menuItem2_Click(object sender, EventArgs e)
        {
            this.Close();
        }

/*-------------------------- CLIENT -------------------------*/
        static void Main(string[] args)
        {
          IPAddress ipad = Dns.GetHostByName("HTC79").AddressList[0];
          EndPoint ip = new IPEndPoint(ipad, 9999);

          Socket socket = new Socket(AddressFamily.InterNetwork,
SocketType.Dgram, ProtocolType.Udp);

          byte[] data = new byte[25];
          string welcome = "Hi, I'm the laptop";
          data = Encoding.ASCII.GetBytes(welcome);
          socket.SendTo(data, data.Length, SocketFlags.None, ip);

          byte[] data2 = new byte[25];
          int receivedDataLength = socket.ReceiveFrom(data2, ref ip);
          MessageBox.Show(Encoding.ASCII.GetString(data2, 0,
receivedDataLength));

          byte[] data3 = new byte[25];
          string goodbye = "Goodbye Smartphone";
          data3 = Encoding.ASCII.GetBytes(goodbye);
          socket.SendTo(data3, data3.Length, SocketFlags.None, ip);

          byte[] data4 = new byte[25];
          int receivedDataLength2 = socket.ReceiveFrom(data4, ref ip);
          MessageBox.Show(Encoding.ASCII.GetString(data4, 0,
receivedDataLength2));
        }
```

As it observed in the showed code, the client starts the communication sending a welcome message to the the server, the server replies it and finally, the client sends a goodbye message and the server replies it again.

3.2.2 Socket TCP (Transport Control Protocol)

The Transport Control Protocol is defined in the RFC 793 (6). Next, the main properties of the sockets TCP are showed:

- Oriented to connection. It is necessary to establish the communication between both applications before send the data.
- The packets are sent without errors. There are methods to correct possible errors in the transmission.
- The packets are delivered in the same order that these were sent.

After getting the communication with socket in the local machine, the sockets TCP have been implemented in the same way. But, after checking that the TCP application was running fine, the smartphone arrived and an application to send files was developed with the finality of send the real video. If a file was created it could be sent from the server to the client, however, while the video is recording, it has not been possible to accede to the file to send the data through the sockets. A temporal file has been looked to be sent, but it has not been found. Therefore, this method to stream the video has not been useful, so, another way to stream data has had to be explored. This way has been the use of streamers, which will be explained in the following chapter. The following code shows the action of open the file which is being recording to get the necessary information to send to the receiver, like the name, the size and the data that is the real video. This code is preceded by which active the camera in record mode and the function which press automatically the enter button. Thus, the whole code of the server and the whole code of the client, which receives the video, are showed in 9.1.

```
string name = @"\My Documents\videos2.mp4";
FileStream filenew = new FileStream(name, FileMode.Open,
FileAccess.Read);

FileInfo finfo = new FileInfo(name);

string filename = finfo.Name;//send name
byte[] bytefilename = new byte[filename.Length];
bytefilename = Encoding.ASCII.GetBytes(filename.ToCharArray());
client.Send(bytefilename, SocketFlags.None);

long filesize = finfo.Length;//send filesize
byte[] bytefilesize = new byte[filesize];
bytefilesize = BitConverter.GetBytes(filesize);
client.Send(bytefilesize, SocketFlags.None);

int bytesSize = 0;//send data
byte[] buffer2 = new byte[2048];
while ((bytesSize = filenew.Read(buffer2, 0, buffer2.Length)) > 0)
       {
           client.Send(buffer2, SocketFlags.None);
       }
client.Close();
socket.Close();
```

3.3 Streamers

The researching has been looked in the use of streamers due to the non-possibility of transmit real-time video from the smartphone using the explained protocols (TCP and UDP). The streamers researched are a set of libraries which have been developed with the finality to allow the users to stream data from a device or a file in real time, with a wide variety of format data.

Next, the main details of two different streamers are explained, as well as, the procedure to run these.

3.3.1 Live555 Streaming media

The Live555 Streaming media forms a set of C++ libraries for multimedia streaming, using open standard protocols (RTP/RTCP, RTSP, SIP). These libraries, which can be compiled for Unix (including Linux and Mac OS X), Windows, and QNX (and other POSIX-compliant systems), can be used to build streaming applications. The libraries are already being used to implement applications such as "the LIVE555 Media Server" (a RTSP server application), "liveCaster" and "playRTPMPEG" (for streaming MP3 audio using RTP/RTCP), and "vobStreamer" (for streaming DVD content using RTP/RTCP/RTSP). The libraries can also be used to stream, receive, and process MPEG, H.263 or JPEG video, and several audio codecs. (7)

The project source code is available - as a ".tar.gz" file, so, it has not been an easy way to compile the code, because it requires generate “makefiles” to be used by Visual Studio, but the used version (Visual Studio 2008) cannot open this kind of files, so there was a problem because the whole information about how to compile this code was for ancient version, but mainly for Unix systems. However, after look for a lot of information about this, a member from the Live555 developer’s mailing-list helped sharing the solution files, which allowed compiling the code.

The first program checked was the testMPEG4VideoStreamer which allows read from a MPEG-4 Elementary Stream video file located in the same machine and streams it using RTP multicast. The VLC software was used to open the file and watch the stream file. It worked fine, but the goal is stream from the smartphone, so other solutions were thought, for instance, the possibility to stream from an external device, instead of a file. However, it just was possible to devices like camera records, dvd connected to the laptop, but the smartphone has to stream wirelessly, so the only alternative was to create and smartphone project using this application. Unfortunately, this code is not supported by Windows Mobile, and then it cannot be used like a smartphone application.

3.3.2 VLC media player

VLC media player is a highly portable multimedia player for various audio and video formats (MPEG-1, MPEG-2, MPEG-4, DivX, mp3...) as well as DVDs, VCDs, and various streaming protocols. It can also be used as a server to stream in unicast or multicast in IPv4 or IPv6 on a high-bandwidth network and can stream all it can read. (8)

In the **Figure 3.3** is showed a global diagram with the VLC streaming solutions.

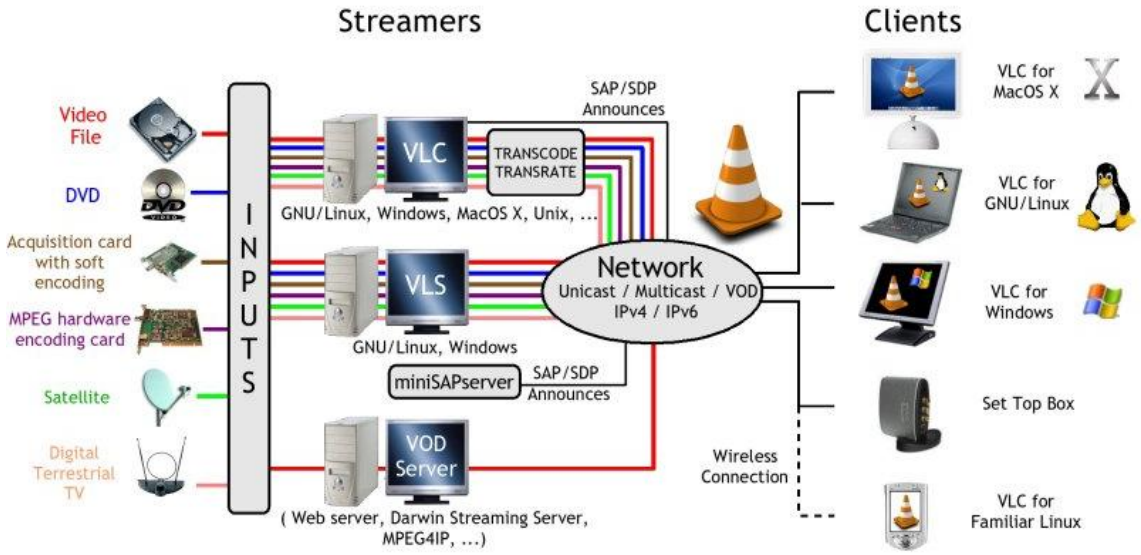

Figure 3.3 – Videolan Streaming Solution

The streaming features list, where, for instance, it is detailed the different protocols supported by the different Operative System can be found in the VLC website. (9)

In the diagram showed it is possible to see that this software is not supported by smartphones. At present it cannot be used to stream video from the smartphone, therefore, in can be only used to read stream from a server in the project. However, there is information providing the VLC forums, which says that a smartphones application is being developed. This is a good new, because in a close future, could be possible to install a VLC software and to realize video stream using the commands described at the VLC website (10), where due to it is free software, it is possible to accede to the developed code.

3.4 Directshow

DirectShow for Windows Mobile 5.0 software is architecture for streaming media that provides both high-quality playback and capture of multimedia

streams. It supports many formats, such as waveform (WAV), MP3 (MPEG Audio Layer-3), AVI, Advanced Streaming Format (ASF), and MPEG.

3.4.1 Filter graph

The main building block of DirectShow is a component called a filter. A filter is a software (actually a COM) component that performs an operation on a multimedia stream. For example, filters can read files, get video from a video capture device, decode various stream formats, and pass data to the graphics or sound card. Filters receive input and produce output, and the information is passed between filters though the filter pins. A pin is a filter port, and it can be either an input port or an output port. If a filter decodes WMV video, the input is the WMV–encoded stream and the output is a series of uncompressed video frames. In DirectShow, an application performs any task by connecting chains of filters together, so the output from one filter becomes the input for another. A set of connected filters is called a filter graph, and **Figure 3.4** shows a filter graph for playing a video file with sound.

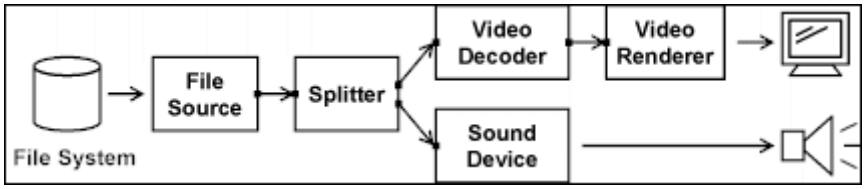

Figure 3.4 – Filter graph

A filter graph must follow certain guidelines, and the first guideline is the need of a source filter. This is the initial source of the data, in the project it is the smartphone's camera. The output of the source filter is then run through any number of transform filters. Transform filters are any intermediate filters that take a certain type of input data, modify the data coming in, and then pass the modified data to its output. The final piece of a graph is a renderer filter. Renderer filters are the final destination of any data handled in a filter graph. Renderers can represent things such as a window for displaying video on the screen, a sound card for emitting sound, or a file writer for storing data to disk.

In **Figure 3.4**, the filters are as follows:

- The file source filter reads the video file from the file system.
- The splitter filter parses the file content into two streams, a compressed video stream and an audio stream.
- The video decoder filter decodes the video frames.
- The video renderer filter draws the frames to the display by using DirectDraw or graphics device interface (GDI).
- The Sound Device filter plays the audio stream by using DirectSound.

The small squares on the edge of the filters in **Figure 3.4** represent the pins of each filter. (11)

Once, the main characteristics in Directshow for Windows Mobile have been studied, there have been two different ways to develop the application to get video from the smartphone.

3.4.2 DShowNET

The first option has been the use of a class library for capturing audio and video in .Net. This selection has been chosen because of the ease to work with this kind of code. As it is showed in the following code, it is very intuitive to capture data from another device using the DirectX.Capture library. (12)

```
// Remember to add a reference to DirectX.Capture.dll
using DirectX.Capture

// Capture using the first video and audio devices available
Capture capture = new Capture( Filters.VideoInputDevices[0],
                               Filters.AudioInputDevices[0] );

// Start capturing
capture.Start();

// Stop capturing
capture.Stop();
```

Firstly, this code was compiled and run on the laptop and it was working fine, because it captured the video from the laptop webcam. However, when a smartphone project was created to compile the code, was discovered that this code is not supported in Windows Mobile, so it was necessary to use the second alternative, the native code. Despite of this, in the following code are showed more DirectX.Capture properties that can be used to select the desired capture parameters, but these have to be defined between the creation of the Capture class and the capture.start().

```
//Use of compressor to reduce the file size
capture.VideoCompressor = Filters.VideoCompressors[0];
capture.AudioCompressor = Filters.AudioCompressors[0];


capture.FrameRate = 29.997;                 // NTSC
capture.FrameSize = new Size( 640, 480 );   // 640x480
capture.AudioSamplingRate = 44100;          // 44.1 kHz
capture.AudioSampleSize = 16;               // 16-bit
capture.AudioChannels = 1;                  // Mono
//Set up the file folder destination
capture.Filename = "C:\MyVideo.avi";
```

3.4.3 DirectShow API

One of the most interesting developments is the inclusion of the DirectShow API in Windows Mobile 5.0 software. Because of the API showed in 3.1 is not sufficient for meeting the requirements of the goal, DirectShow provides more control and greater flexibility. DirectShow is a native API and is therefore available through the native toolset (Visual C++).

A developed filter graph capture has been used to get the video from the smartphone. The description of the properties of this filter graph capture can be observed in the **Figure 3.5**.

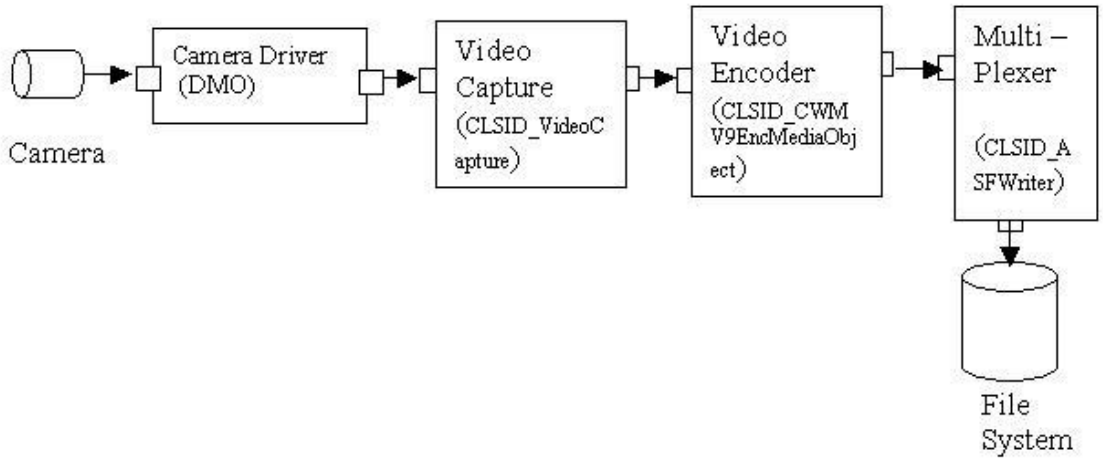

Figure 3.5 – Filter graph Capture

The camera is the hardware component. For an application to interact with the camera, it would need to talk to its drivers. Next, the video capture filter enables an application to capture video. After capture, the data is encoded using WMV9EncMediaObject, a DirectX Media Object (DMO). A DMO inside a filter graph is used with the help of a DMO Wrapper filter. Next, the encoded video data needs to be multiplexed. A Windows Media ASF writer filter is used for this task. The ASF writer multiplexes the video data and writes it to an .asf file. With that, the filter graph is ready. (13)

The specific descriptions about how to set up the environment, build the filter graph, as well as, the code used can be checked in the article's website. (13)

There have been a lot of problems to run this code, because, despite of in the source website, there is information about how to do it, there were missing steps, so it was necessary to look for help by using different forums, as well as, sending e-mails directly to developers. Firstly, the development of this application has been to create the source filter, also called capture filter, with the finality to get the video from the device. However, it has not been possible to detect the smartphone, despite of the code, seemingly is working fine. By the

moment, there is not so much information, about possible problems developing filters graphs in Windows Mobile, but a way to continue researching in this objective has been opened.

4 Real Time software

Looking for information, some real time software has been found. These are able to transmit real time video through a wireless medium. This is not the desired solution to stream video from the smartphone, but while the researching is continuing, this software could be useful to do some experiments.

4.1 Qik

Qik is developed software by a company, which allows streaming real time video from a smartphone and sharing it on a website (14). The procedure to get this goal is explained in the following steps:

- Download the software which is supported by the smartphone, in our project, is not the HTC Touch 3G, but the HTC Touch Dual's software has been installed and it is working fine.
- Realize a registration on the website's link *www.qik.com*, where the username is chosen and it will be used to accede to watch the real time video, because of, it is necessary to watch the video through the Qik's website, therefore, the link will be *www.qik.com/username*.
- Once, the installation and the register are completed, it necessary to be connected to a Wi-Fi network in our project, in spite of it can be possible with GPRS or 3G, and just push the stream key.
- Finally, whether the last steps are successful, the real time video will be showed on the username's website.

This software has been the chosen option, it has been tested and it is working fine, so it is possible to stream real time video from the smartphone to a server through a wireless medium, in our project is used wi-fi.

4.2 Ateksoft WebCamera Plus

This software, like the previous, allows streaming real time video from the smartphone to a computer. Next, are showed the main Ateksoft WebCamera Plus properties:

- Allows the user to use the smartphone's camera like a webcam, allowing to use it in may programs like: Skype, MSN Messenger ...
- The connection between the smartphone and the computer can be in different ways: Wi-Fi, Bluetooth, GPRS, 3G ...
- Compatible with the most modern smartphones and PDA, it is compatible with HTC Touch 3G.

The procedure to use this software is downloading from the website's link (15) and installs it in the smartphone through ActiveSync. After that, we have to synchronize smartphone and computer and then it is possible to watch the real time video in the computer. This software is a good choice, but the downloaded version is only trial, so it is necessary to pay for use this software.

5 Serial Port communication

The second goal of the project has been to control the airship by reading and sending information from the serial channel. This goal has not been reached, because of the first goal has not been finished. However, until the smartphone arrived, there was time to think about possibilities to get this goal.

The first option was realize the communication using the infrared port. There was the possibility to use a serial port communication only if the smartphone had incorporated an Infrared port like a Serial port. Next, it is showed an application code, which has an easy working procedure. Firstly, initialize the Infra Red port, after that, it opens the port and configures the desired properties in the communication. Once the properties are set up, it sends information to the receiver and finally the port is closed.

```
private void menuItem1_Click(object sender, EventArgs e)
{
    string text;

    if (SerialPort.IsOpen)
    {
        SerialPort.Close();
    }
    SerialPort.Open(); //open the port

    //Set the appropiate properties
    SerialPort.BaudRate =9600;
    SerialPort.DataBits = 8;
    SerialPort.Parity = Parity.None;
    SerialPort.PortName = "COM2"; //IR device
    SerialPort.StopBits = StopBits.One;
    SerialPort.Handshake = Handshake.None;

    SerialPort.Write("text"); //send data to the receiver

    SerialPort.Close(); //close the port

}
private void exit_app_Click(object sender, EventArgs e)
{
    this.Close();
}
```

However, the last smartphones are made without Infrared port, so it was necessary to study a second alternative.

The majority of smartphones are provided with a mini USB connection, so, the second possibility was to study the method to use this connection like a serial port connection and a possible solution was found and it is explained in the following paragraphs:

- Firstly, an adapter from mini usb to usb would be necessary. There are different models in the market. An example could be a mini usb male to an usb female adapter; it is showed in the **Figure 5.1**.

Figure 5.1 – Mini usb to usb adapter

- After that, the device selected could be an usb to serial port converter, which is showed in the **Figure 5.2**. Once it is connected, it could be used like a serial port. The manufacturer would incorporate some drivers to be installed and after that it could be used like a serial port, therefore, a similar code to create the communication with the airship electronics, like showed in the Infra Red solution, could be useful.

Figure 5.2 – Usb to serial port converter

After this study, this solution can be neither useful, because of this kind of usb to serial port converter are not supported by smartphones, although physically it is possible, this solution won't be able to work fine.

Finally, it seems that the best solution would be use the smartphone like a USB device and on the side of the electronics would be a USB host, which would incorporate a driver or some protocol to allow the communication between smartphone and electronics.

This study is just to open the way to find the best solution to the communication between the smartphone and the airship electronics.

6 Future improvements

In this section are explained the future possible improvements to realize in the Airship Electronics project. The project goals are not finished, however there is a researching which allows continuing working in the project and improving it following some steps:

- Continue the researching in Direct Show for Windows Mobile, because of it seems that is a way to get the video data from the smartphone.
- Another alternative could be to look for future version of VLC media Player, due to they are working in a version to be implemented in smartphones, this version would be free source code, so would be possible to stream real time video from the smartphone to a computer which would use the VLC media player like a receiver.
- Continue the researching realized for the communication between the smartphone and the airship electronics, try to use the proposed methods or find possible new ways.

7 Conclusions

After the project realized, it is necessary to extract some conclusions, which can help to analyse the good or bad things done in the project, as well as, to improve the realization of future projects.

Next, are showed the main extracted conclusions:

- In spite of the goals of the project have not been reached, the researching work done has help me to improve knowledge, like for instance, the smartphone application development, the use and the functionality of streamers.
- The possibility to work in a real project which is being developed in the RDC department has been more interesting, like for instance, the possibility to realize a simple project, just to get the credits and the mark. I think it is a good way to motivate future students and to help them to approach to the real jobs.
- In general, this experience has been rewarding, on the one hand it has been a good opportunity to improve my English and on the other hand this experience has opened my mind and it has help me to plan new goals in my life.

For all this reasons, I have to be grateful for this opportunity to everybody in the RDC department that have helped me along my project.

8 References

1. **HTC Touch 3G**. [Online]
http://www.htc.com/www/product/touch3g/overview.html.

2. **Wi-fi**. [Online]
http://searchmobilecomputing.techtarget.com/sDefinition/0,,sid40_gci838865,00.html.

3. **GPRS**. [Online]
http://searchmobilecomputing.techtarget.com/sDefinition/0,,sid40_gci213689,00.html.

4. **Bluetooth**. [Online]
http://www.mariosalexandrou.com/definition/bluetooth.asp.

5. **User Datagram Protocol**. [Online] http://tools.ietf.org/html/rfc768.

6. **Transmission Control Protocol**. [Online] http://tools.ietf.org/html/rfc793.

7. **Live555 Streaming Media**. [Online] http://www.live555.com/liveMedia/.

8. **VLC Media Player**. [Online] http://www.videolan.org/vlc/.

9. **The VideoLAN project**. [Online] http://www.videolan.org/project/.

10. **VideoLAN Streaming Solution**. [Online]
http://www.videolan.org/streaming-features.html.

11. **DirectShow**. [Online] http://msdn.microsoft.com/en-us/library/aa454909.aspx.

12. **DirectShow.NET**. [Online]
http://www.codeproject.com/KB/directx/directxcapture.aspx.

13. **Using a Camera with Windows Mobile**. [Online]
http://www.developer.com/ws/pc/article.php/3621211.

14. **Qik**. [Online] http://qik.com/info/product_highlight.

15. **Ateksoft WebCamera Plus**. [Online]
http://www.ateksoft.com/webcamplus.html.

9 Annexes

In this chapter is showed the whole application code when it is using TCP sockets in the communication. It can be observed the function to activate the camera, the function to press the enter button which starts to record and finally the function to send the video to the client.

9.1 TCP application code

```
/*-------------------------   SERVER   -------------------------*/
        public Form1()
        {
            InitializeComponent();
            Thread th1 = new Thread(new ThreadStart(takepicture));
            Thread th2 = new Thread(new ThreadStart(accept));
            Thread th3 = new Thread(new ThreadStart(sendvideo));

            th1.Start();
            th2.Start();
            th3.Start();

            th1.Join();
            th2.Join();
            th3.Join();
        }

        public void takepicture()
        {
            CameraCaptureDialog cameraCaptureDialog = new
CameraCaptureDialog();
            cameraCaptureDialog.Owner = null;
            cameraCaptureDialog.Title = "Take Video";
            cameraCaptureDialog.Mode = CameraCaptureMode.VideoOnly;
            cameraCaptureDialog.Resolution = new Size(176, 144);
            cameraCaptureDialog.DefaultFileName = @"videos2.mp4";
            cameraCaptureDialog.InitialDirectory = @"\My Documents";
            cameraCaptureDialog.VideoTypes =
CameraCaptureVideoTypes.Standard; //record .wvm files
            cameraCaptureDialog.ShowDialog();
        }

        public void accept()
        {
            Thread.Sleep(2000);
            keybd_event(0x0d, 0, 0x0, 0);
            keybd_event(0x0d, 0, 0x2, 0);
        }

        public void sendvideo()
        {
            Thread.Sleep(10000);
            IPEndPoint ip = new IPEndPoint(IPAddress.Any, 9999);
            Socket socket = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);
```

```
            socket.Bind(ip);
            socket.Listen(10);
            Socket client = socket.Accept();
            IPEndPoint clientep = (IPEndPoint)client.RemoteEndPoint;

            string name = @"\My Documents\videos2.mp4";
            FileStream filenew = new FileStream(name, FileMode.Open,
FileAccess.Read);

            FileInfo finfo = new FileInfo(name);

            string filename = finfo.Name;//send name
            byte[] bytefilename = new byte[filename.Length];
            bytefilename =
Encoding.ASCII.GetBytes(filename.ToCharArray());
            client.Send(bytefilename, SocketFlags.None);

            long filesize = finfo.Length;//send filesize
            byte[] bytefilesize = new byte[filesize];
            bytefilesize = BitConverter.GetBytes(filesize);
            client.Send(bytefilesize, SocketFlags.None);

            int bytesSize = 0;//send data
            byte[] buffer2 = new byte[2048];
            while ((bytesSize = filenew.Read(buffer2, 0,
buffer2.Length)) > 0)
            {
                client.Send(buffer2, SocketFlags.None);
            }
            client.Close();
            socket.Close();
        }

/*-------------------------   CLIENT   -------------------------*/
    public static void Main()
    {
        IPAddress ipad = Dns.GetHostByName("HTC79").AddressList[0];
        EndPoint ip = new IPEndPoint(ipad, 9999);
        Socket server = new Socket(AddressFamily.InterNetwork,
SocketType.Stream, ProtocolType.Tcp);

        try
        {
            server.Connect(ip);
        }
        catch (SocketException e)
        {
            Console.WriteLine("Unable to connect to server.");
            return;
        }

        int bytesize = 0;

        byte[] buffer = new byte[2048];//receive name
        bytesize = server.Receive(buffer, SocketFlags.None);
        byte[] buffer1 = new byte[bytesize];
        Array.Copy(buffer,buffer1,bytesize);
        string filename = Encoding.ASCII.GetString(buffer1, 0,
bytesize);
```

```
        byte[] buffer2 = new byte[2048];//receive filesize
        bytesize = server.Receive(buffer2, SocketFlags.None);
        int filesize = BitConverter.ToInt32(buffer2, 0);

        byte[] buffer3 = new byte[2048];//receive data
        byte[] buffer4 = new byte[filesize];
        int recdata = 0;
        FileStream filerec = new FileStream(@"C:\2" + filename,
FileMode.Create, FileAccess.ReadWrite, FileShare.ReadWrite);

        while((recdata = server.Receive(buffer3, SocketFlags.None))>0)
        {
            filerec.Write(buffer3, 0, recdata);
        }

        server.Shutdown(SocketShutdown.Both);
        server.Close();
    }
```