

.....

---

# Programación para dispositivos móviles con .NET CF



## **Comunicaciones por Bluetooth**

---

## Objetivos

- Desarrollar aplicaciones que comuniquen vía Bluetooth;

## Material a utilizar

- Visual Studio 2005 + Pocket PC 2005 SDK;
- Material didáctico-multimedia que acompaña a este documento y que ya habréis descargado de poliformaT;
- Apuntes Tema 5 de ADM.

## Estudio de mercado

### Soporte Bluetooth en .NET

Lógicamente lo primero que debemos averiguar es, puesto que estamos trabajando con .NET, si esta plataforma ya da algún soporte para Bluetooth. La respuesta es que no, .NET no ofrece ningún mecanismo con el que, directamente, podamos descubrir dispositivos o servicios Bluetooth en los alrededores. Por tanto, en caso de querer dicha funcionalidad deberemos recurrir a librerías externas.

### Soporte externo : Uso de librerías

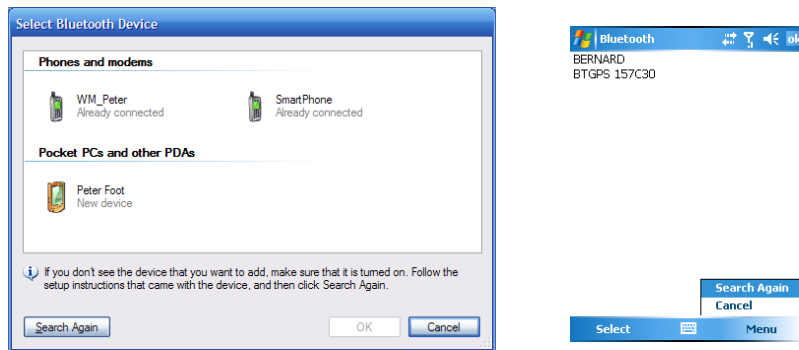
Como ya se ha comentado, si deseamos hacer uso de funciones específicas de Bluetooth, como es, el poder descubrir otros dispositivos Bluetooth que se encuentren en los alrededores, vamos a tener que hacer uso de alguna librería externa que ofrezca alguna API con dicha funcionalidad.

### 32Feet.NET

En nuestro caso hemos usado el paquete 32Feet.NET de InTheHand (<http://32feet.net>). 32Feet.NET es una colección de librerías para .NET que ofrece soporte para los protocolos de Bluetooth, IrDA y Object Exchange (OBEX), tanto para PCs como para dispositivos móviles y sistemas empujados.

¿Qué ofrece? 32Feet.NET ofrece una API muy sencilla de usar que nos va a permitir:

- Acceder a la radio Bluetooth (BluetoothRadio): Cambiar su estado a encendido, apagado o detectable. Detectar si la pila Bluetooth de Microsoft está instalada.
- Descubrir dispositivos Bluetooth cercanos (BluetoothClient.DiscoverDevices()): De hecho directamente ofrece un formulario (SelectBluetoothDeviceDialog) en donde podremos seleccionar uno de los dispositivos disponibles en ese momento.
- Conectarnos a un servicio Bluetooth (BluetoothClient.Connect()), y a partir de ahí obtener un stream y trabajar como un flujo de datos corriente.
- Registrar nuestros propios servicios Bluetooth (BluetoothListener.Start()) y empezar a atender a los clientes (BluetoothListener.AcceptBluetoothClient()) .
- Enviar y recibir objetos (ObexWebRequest y ObexListener) al igual que haríamos con el HTTP.



## High – Point

Ésta es otra librería con la que poder acceder a las funciones específicas de Bluetooth y la cual consta de las siguientes características:

- Puede encontrarse en la siguiente dirección: <http://www.high-point.com>
- No es gratuita. Su precio oscila entre los 30\$ y los 750\$. Sin embargo puede bajarse una versión de evaluación, idéntica a la de pago, salvo porque aparece un mensaje advirtiéndolo que se trata de una versión prueba.
- Solo es compatible con la pila Widcomm y únicamente para dispositivos móviles (PocketPC, PocketPC2002, PocketPC 2003 / Windows Mobile con soporte para .Net v1.1, y Windows Mobile 5 con soporte para .Net v2.0).
- El descubrimiento de Dispositivos Bluetooth se gestiona con eventos.

## Franson

Otra de las librerías que actualmente podemos encontrar en el mercado es la BlueTools de Franson (<http://franson.com/bluetools/index.asp>). De hecho esta puede considerarse una de las mejores, puesto que:

- Es compatible tanto para la pila de Widcomm como para la de Microsoft.
- Es válida tanto para Windows como para Windows CE y Windows Mobile.
- Además de ofrecer todas las funcionalidades propias de Bluetooth, también da soporte a OBEX y SMSs.

¿La pega? Lógicamente es de pago. Su precio oscila entre 129\$ y 199\$. Aunque se puede adquirir una licencia de evaluación que nos permite usarla durante 14 días.

## Cambiar la pila Bluetooth

La pila Bluetooth que usa nuestro dispositivo es dependiente del hardware y por lo tanto, no siempre va a ser posible cambiar entre una y otra (especialmente si tenemos la pila de Microsoft y queremos pasar a la de Widcomm).

Sin embargo hay casos en los que si que vamos a poder, y nos referimos al caso de querer pasar de la pila de Widcomm a la de Microsoft, si se tiene instalado el Windows Xp sp2 o posterior. Esto es así porque la pila de Microsoft ya viene integrada a partir de Service Pack 2. De esta forma, lo que debemos hacer para realizar el cambio es, desinstalar todos los drivers de nuestro dispositivo Bluetooth que lo relacionan con la pila de Widcomm. Una vez desinstalados, dejaremos que el sistema operativo gestione este dispositivo, por lo que a partir de ese momento pasará a ser usuario de la pila de Microsoft.

## Ejemplos prácticos

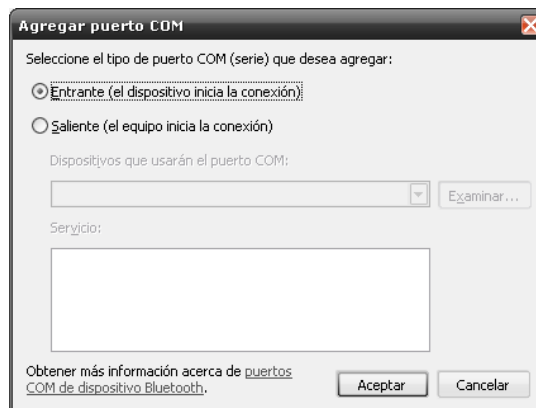
### EJEMPLOS PRÁCTICOS

#### Conexión Bluetooth PDA-PC mediante puertos COM

El hecho de que .NET no ofrezca ningún soporte para Bluetooth no es impedimento para que nuestras aplicaciones no puedan comunicarse. Esto es así ya que en la mayoría de los casos, o bien el sistema operativo, o bien las propias aplicaciones que vienen junto con los dispositivos bluetooth, nos van a permitir realizar la búsqueda de dispositivos y el establecimiento de puertos COM. Por tanto nuestra aplicación no tiene más que escribir los datos a transmitir directamente sobre esos puertos COM.

Dicho esto, lo primero que vamos a hacer es explicar cómo realizar la búsqueda, asociación de dispositivos y establecimiento de la conexión. Este proceso se realiza en tres pasos:

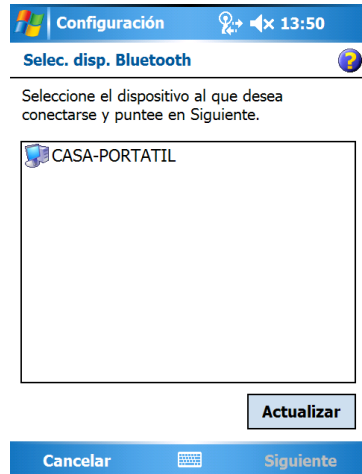
- 1) Crear y abrir un puerto de entrada. Esto viene a ser como publicar un servicio de Bluetooth. En nuestro caso este punto se va a hacer en un PC.
  - Abrimos la aplicación de Bluetooth, pestaña “Puertos COM”, Agregar, puerto de entrada.



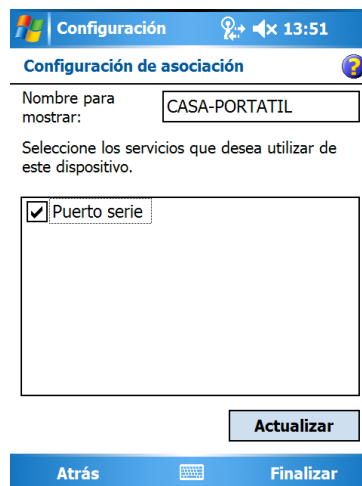
- En este momento tendremos asignado un puerto de entrada, en nuestro caso el COM4. Sin embargo hace falta abrirlo para que el dispositivo del otro extremo lo

detecte. Para ello DEBEMOS USAR la aplicación de ejemplo del PC que se muestra más adelante.

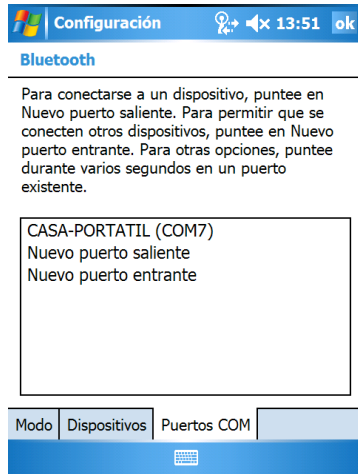
- 2) Establecimiento de una asociación entre dispositivos. En nuestro caso la asociación se va a hacer entre una PDA y un PC con Bluetooth. Además el inicio de la asociación se ha hecho desde la PDA. De querer hacerlo desde el PC el procedimiento sería muy similar.
  - Abrimos la aplicación de Bluetooth, pestaña “Dispositivos”, Nueva asociación.
  - Elegimos el dispositivo con el que vamos a asociarnos.



- Introducimos una clave de paso y la confirmamos en el otro extremo.
- Si todo ha ido bien, en este punto debería aparecer que el PC, en nuestro caso, ofrece como servicio el Puerto serie. Lo seleccionamos (nos aparece un mensaje en la aplicación de ejemplo del PC con la cadena “CLIENT”).



- 3) Crear el puerto de salida. En nuestro caso el puerto de salida estará en la PDA.
  - En la aplicación de Bluetooth, pestaña “Puertos COM”, Nuevo puerto saliente.
  - Elegimos la asociación que acabamos de crear y COM7



Si todo ha ido bien, en este punto tendremos:

- Un puerto de entrada abierto y la aplicación escuchando por él, en el PC.
- Una asociación entre la PDA y el PC.
- Un puerto de salida creado en la PDA.

Por tanto solo hace falta abrir el puerto de salida de la PDA y enviar los datos que queramos. Para ello podemos usar la aplicación de ejemplo de la PDA que se muestra a continuación.

**\* Código de la PDA (puerto saliente)**

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.IO.Ports;

namespace EjemploCOMCE
{
    public partial class Form1 : Form
    {
        SerialPort sp;

        TextBox tbPuerto, tbCadena;
        Button bEnviar;

        public Form1()
        {
            InitializeComponent();
            GeneraComponentes();
        }

        //creamos todos los componentes que vamos a necesitar
        private void GeneraComponentes()
        {
            //el puerto serie
            sp = new SerialPort();

            //el TextBox con el nombre del puerto
            tbPuerto = new TextBox();
            tbPuerto.Text = "COM1";
            tbPuerto.Size = new Size(Width - 20, 20);
            tbPuerto.Location = new Point(Width / 2 - tbPuerto.Width /
2, 10);
            this.Controls.Add(tbPuerto);

            //el TextBox con la cadena a enviar
            tbCadena = new TextBox();
            tbCadena.Size = new Size(Width - 20, 20);
            tbCadena.Location = new Point(Width / 2 - tbCadena.Width / 2,
tbPuerto.Height + 20);
            this.Controls.Add(tbCadena);

            //el boton de enviar
            bEnviar = new Button();
            bEnviar.Text = "Enviar Cadena";
            bEnviar.Size = new Size(150, 40);
            bEnviar.Location = new Point(Width / 2 - bEnviar.Width / 2,
tbCadena.Location.Y + 50);
            bEnviar.Click += new EventHandler(bEnviar_Click);
            this.Controls.Add(bEnviar);
        }

        .
        .
        .
    }
}
```

```

.
.
.

//cuando se pincha en enviar
private void bEnviar_Click(object sender, EventArgs e)
{
    try
    {
        sp.Close(); //cerramos el puerto para que no de
errores
        sp.PortName = tbPuerto.Text; //cambiamos el puerto
        sp.Open(); //abrimos en puerto
        sp.WriteLine(tbCadena.Text); //enviaos los datos
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
}
}
}

```

En este ejemplo se ve cómo hacer uso del componente SerialPort para enviar datos mediante el método WriteLine. Este componente lo ofrece .NET a través de la librería del sistema System.IO.Ports. Puesto que para minimizar el código no se ha creado ninguna etiqueta, cabe aclarar que la aplicación dispone de dos cuadros de texto. En el primero podremos introducir el puerto por donde transmitiremos los datos, mientras que en el segundo introduciremos la cadena de texto a enviar.

**\* Código del PC (puerto entrante)**

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.IO.Ports;

namespace EjemploCOM
{
    public partial class Form1 : Form
    {
        SerialPort sp;

        TextBox tbPuerto;
        Button bAbrirPuerto;

        public Form1()
        {
            InitializeComponent();
            GeneraComponentes();
        }

        //creamos todos los componentes que vamos a necesitar
        private void GeneraComponentes()
        {
            //el puerto serie
            sp = new SerialPort();
            sp.DataReceived += new
            SerialDataReceivedEventHandler(sp_DataReceived);

            //el TextBox con el nombre del puerto
            tbPuerto = new TextBox();
            tbPuerto.Text = "COM1";
            tbPuerto.Size = new Size(Width - 20, 20);
            tbPuerto.Location = new Point(Width / 2 - tbPuerto.Width /
2, 10);
            this.Controls.Add(tbPuerto);

            //el boton para abrir el puerto
            bAbrirPuerto = new Button();
            bAbrirPuerto.Text = "Abrir Puerto";
            bAbrirPuerto.Size = new Size(150, 20);
            bAbrirPuerto.Location = new Point(Width / 2 - bAbrirPuerto.Width /
2, tbPuerto.Location.Y + 30);
            bAbrirPuerto.Click += new
EventHandler(bAbrirPuerto_Click);
            this.Controls.Add(bAbrirPuerto);

            //establecemos el tamaño del "Form"
            this.Height = bAbrirPuerto.Location.Y + 60;
        }
    }
}

.
.
.
```

```

.
.
.

//cuando se pincha en enviar
private void bAbrirPuerto_Click(object sender, EventArgs e)
{
    try
    {
        errores
        sp.Close(); //cerramos el puerto para que no de

        sp.PortName = tbPuerto.Text; //cambiamos el puerto
        sp.Open(); //abrimos en puerto
        MessageBox.Show("Puerto abierto");
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
}

private void sp_DataReceived(object sender, SerialDataReceivedEventArgs
e)
{
    + ">");
    MessageBox.Show("Mensaje recibido: <" + sp.ReadExisting()
}
}
,

```

En este ejemplo vemos como hacer uso del SerialPort para leer datos. Esto se hace capturando el evento DataReceived de este componente. Este ejemplo es el gemelo del anterior, de hecho nada impide intercambiar el código de ambos y que la PDA pase a ser la publicadora del servicio, y el PC el consumidor.

## Conexión Bluetooth PDA-PC usando la librería 32Feet.NET

Aquí vamos a hacer un pequeño ejemplo donde transmitiremos un fichero desde la PDA al PC mediante una conexión Bluetooth y haciendo uso de la librería 32Feet.NET

### \* Código de la PDA (parte cliente)

```
using System;
using System.Windows.Forms;
using System.IO;
using InTheHand.Net.Bluetooth;
using InTheHand.Windows.Forms;
using InTheHand.Net.Sockets;

namespace EjemploBluetoothCE
{
    public partial class Form1 : Form
    {
        ListBox lbDispotivosEncontrados;
        Button bBuscarManual;
        Button bBuscarAutomático;
        Button bEnviarArchivo;
        Guid servicio = new Guid("{3EA9F13E-A365-4657-BE6C-81AE4A973B49}");
        BluetoothDeviceInfo[] dispositivosEncontrados;
        SelectBluetoothDeviceDialog sbtd;
        BluetoothDeviceInfo destino;

        public Form1()
        {
            InitializeComponent();
            //comprobamos que soporta la pila de Microsoft
            if (!BluetoothRadio.IsSupported)
                this.Close();
            //activamos el bluetooth
            BluetoothRadio br = BluetoothRadio.PrimaryRadio;
            br.Mode = RadioMode.Discoverable;
            GeneraComponentes();
        }

        //creamos todos los componentes que vamos a necesitar
        private void GeneraComponentes()
        {
            //el ListBox
            lbDispotivosEncontrados = new ListBox();
            lbDispotivosEncontrados.Size = new Size(Width - 40, Height
- 220);
            lbDispotivosEncontrados.Location = new Point(Width / 2 -
            lbDispotivosEncontrados.Width / 2, 10);
            lbDispotivosEncontrados.SelectedIndexChanged += new
            EventHandler(lbDispotivosEncontrados_SelectedIndexChanged);
            this.Controls.Add(lbDispotivosEncontrados);

            //el boton de busqueda manual
            bBuscarManual = new Button();
            bBuscarManual.Text = "Busqueda Manual";
            bBuscarManual.Size = new Size(300, 40);
            bBuscarManual.Location = new Point(Width / 2 - bBuscarManual.Width
            / 2, lbDispotivosEncontrados.Height + 20);
            bBuscarManual.Click += new
            EventHandler(bBuscarManual_Click);
            this.Controls.Add(bBuscarManual);
        }
    }
}
```

```

.
.
.

        //el boton de busqueda automática
        bBuscarAutomático = new Button();
        bBuscarAutomático.Text = "Busqueda Automática";
        bBuscarAutomático.Size = new Size(300, 40);
        bBuscarAutomático.Location = new Point(Width / 2 -
            bBuscarAutomático.Width / 2, bBuscarManual.Location.Y + 50);
        bBuscarAutomático.Click += new
            EventHandler(bBuscarAutomático_Click);
            this.Controls.Add(bBuscarAutomático);

        //el boton de enviar fichero
        bEnviarArchivo = new Button();
        bEnviarArchivo.Text = "Enviar Fichero";
        bEnviarArchivo.Size = new Size(300, 40);
        bEnviarArchivo.Location = new Point(Width / 2 -
            bEnviarArchivo.Width / 2, bBuscarAutomático.Location.Y + 50);
        bEnviarArchivo.Enabled = false;
        bEnviarArchivo.Click += new
            EventHandler(bEnviarArchivo_Click);
            this.Controls.Add(bEnviarArchivo);
    }

    //si seleccionamos un elemento de la lista
private void lbDispositivosEncontrados_SelectedIndexChanged(object
sender, EventArgs e)
    {
        if (lbDispositivosEncontrados.SelectedItem != null)
        {
            destino = dispositivosEncontrados[lbDispositivosEncontrados.
                SelectedIndex];
            bEnviarArchivo.Enabled = true;
        }
    }

    //si pinchamos en hacer la busqueda manual
private void bBuscarManual_Click(object sender, EventArgs e)
    {
        try
        {
            Cursor.Current = Cursors.WaitCursor;
            //hacemos un "Discover"
            BluetoothClient cliente = new BluetoothClient();
            dispositivosEncontrados = cliente.DiscoverDevices(5);

            //llenamos la lista con los dispositivos encontrados
            lbDispositivosEncontrados.Items.Clear();
            foreach (BluetoothDeviceInfo info in
dispositivosEncontrados)

lbDispositivosEncontrados.Items.Add(info.DeviceName);
                Cursor.Current = Cursors.Default;
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error: " + ex.Message);
            }
        }
    }
}

```

```

.
.
.
//si pinchamos en hacer la búsqueda automática
private void bBuscarAutomático_Click(object sender, EventArgs
e)
{
    try
    {
        //mostramos el cuadro de dialogo que busca los
dispositivos
        sbtdd = new SelectBluetoothDeviceDialog();
        sbtdd.ShowAuthenticated = true;
        sbtdd.ShowRemembered = true;
        sbtdd.ShowUnknown = true;
        if (sbtdd.ShowDialog() == DialogResult.OK)
        {
            //obtenemos el dispositivo seleccionado
            if (sbtdd.SelectedDevice != null)
            {
                destino = sbtdd.SelectedDevice;
                bEnviarArchivo.Enabled = true;
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
}

//si pinchamos en enviar archivo
private void bEnviarArchivo_Click(object sender, EventArgs e)
{
    try
    {
        //mostramos un cuadro de dialogo para elegir el
        fichero a enviar
        OpenFileDialog ofd = new OpenFileDialog();
        if (ofd.ShowDialog() == DialogResult.OK)
            EnviaFichero(ofd.FileName);
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
}

.
.

```

```

.
.
.

//enviamos el fichero
private void EnviaFichero(string fichero)
{
    try
    {
        //creamos la conexion entre el cliente y el servidor
        BluetoothClient cliente = new BluetoothClient();
        cliente.Connect(destino.DeviceAddress, servicio);

        //abrimos el fichero
        FileStream streamR = new FileStream(fichero, FileMode.Open,
        FileAccess.Read);
        BinaryReader br = new BinaryReader(streamR);
        StreamWriter sw = new
StreamWriter(cliente.GetStream());

        //enviamos el nombre del archivo
        sw.WriteLine(fichero);

        //Enviamos el fichero
        byte[] data = new byte[2048];
        int count;
        do
        {
            //leemos del fichero
            count = br.Read(data, 0, data.Length);
            //enviamos lo leido por el "Stream"
            if (count > 0)
                sw.WriteLine(Convert.ToBase64String(data, 0,
count));
        } while (count > 0);

        //Cerramos los streams
        sw.Flush();
        sw.Close();
        br.Close();
        streamR.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show("Error: " + ex.Message);
    }
}
}

```

En este ejemplo se ve como descubrir dispositivos Bluetooth de los alrededores invocando el método `DiscoverDevices` o bien usando el formulario que nos proporciona la librería (`SelectBluetoothDeviceDialog`). Además se ve como abrir la conexión entre el cliente y el servidor con `Connect` y como obtener un canal por donde transmitir datos con `GetStream`.

### \* Código del PC (parte servidor)

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.Text;
using System.IO;
using System.Threading;
using InTheHand.Net.Bluetooth;
using InTheHand.Net.Sockets;

namespace EjemploBluetooth
{
    public partial class Form1 : Form
    {
        BluetoothListener btListener;
        Guid servicio = new Guid("{3EA9F13E-A365-4657-BE6C-81AE4A973B49}");

        Button bFinalizar;
        bool salir;

        public Form1()
        {
            InitializeComponent();

            salir = false;
            bFinalizar = new Button();
            bFinalizar.Text = "Finalizar";
            bFinalizar.AutoSize = true;
            bFinalizar.Location = new Point(Width / 2 - bFinalizar.Width / 2,
            Height / 2 - bFinalizar.Height);
            this.Controls.Add(bFinalizar);
            bFinalizar.Click += new EventHandler(bFinalizar_Click);

            //comprobamos que soporta la pila de Microsoft
            if (!BluetoothRadio.IsSupported)
                this.Close();
            //activamos el bluetooth
            BluetoothRadio br = BluetoothRadio.PrimaryRadio;
            br.Mode = RadioMode.Discoverable;
            //arrancamos el servidor Bluetooth con el servicio
            especificado
            btListener = new BluetoothListener(servicio);
            btListener.Start();
            //creamos un hilo encargado de atender a los clientes
            Thread th = new Thread(new ThreadStart(this.hilo));
            th.Start();
        }

        //cuando pulsamos el boton finalizar
        private void bFinalizar_Click(object sender, EventArgs e)
        {
            salir = true;
            btListener.Stop();
            Close();
        }
    }
}
```

```

.
.
public void hilo() //el hilo donde se atienden los clientes
{
    string fichero , buffer;
    int count;
    do
    {
        try
        {
            //esperamos a un nuevo cliente Bluetooth
            BluetoothClient client = btListener.AcceptBluetoothClient();
            //obtenemos un stream del cliente para poder leer los datos
            que nos envíe
            StreamReader streamR = new StreamReader(client.GetStream(),
            Encoding.UTF8);
            //leemos el nombre del fichero
            fichero = streamR.ReadLine();
            if (fichero != "")
            {
                //creamos un nuevo fichero con ese nombre
                FileStream streamW = new FileStream(fichero,
                FileMode.OpenOrCreate, FileAccess.Write);
                BinaryWriter bw = new BinaryWriter(streamW);
                do
                {
                    //leemos una linea enviada por el cliente
                    buffer = streamR.ReadLine();
                    //la procesamos
                    byte[] data =
Convert.FromBase64String(buffer);
                    count = data.Length;
                    //la escribimos en el fichero
                    if (count > 0)
                        bw.Write(data, 0, count);
                    //hasta que detecte el final del fichero
                } while (!streamR.EndOfStream);
                MessageBox.Show("Fichero recibido: " + fichero);
                //cerramos el fichero
                bw.Flush();
                bw.Close();
            }
            //cerramos todas la conexion con el cliente
            streamR.Close();
        }
        catch (Exception ex)
        {
            if (!salir)
                MessageBox.Show("Error: " + ex.Message);
        }
    } while (!salir);
}
}

```

En este ejemplo se ve cómo crear un servidor Bluetooth (BluetoothListener) que publicará un servicio con un GUID específico, y quedará a la espera de que algún cliente Bluetooth decida establecer una comunicación (AcceptBluetoothClient)

## Uso de OBEX PDA-PC con la librería 32Feet.NET

Aprovechando que 32Feet.NET da soporte para OBEX se ha decidido hacer un pequeño ejemplo para ver otra forma de transmitir un fichero entre la PDA y el PC.

### \* Código de la PDA (parte cliente)

```
namespace PruebaOBEXCE
{
    public partial class Form1 : Form
    {
        Button bEnviar;

        public Form1()
        {
            InitializeComponent();
            GeneraComponentes();
        }

        //creamos todos los componentes que vamos a necesitar
        private void GeneraComponentes()
        {
            //el boton de enviar
            bEnviar = new Button();
            bEnviar.Text = "Enviar Archivo";
            bEnviar.Size = new Size(300, 40);
            bEnviar.Location = new Point(100, 100);
            bEnviar.Click += new EventHandler(bEnviar_Click);
            this.Controls.Add(bEnviar);
        }

        private void bEnviar_Click(object sender, System.EventArgs e)
        {
            // creamos un cuadro de dialogo para elegir dispositivo
            SelectBluetoothDeviceDialog sbdd = new
            SelectBluetoothDeviceDialog();
            sbdd.ShowAuthenticated = true;
            sbdd.ShowRemembered = true;
            sbdd.ShowUnknown = true;
            if (sbdd.ShowDialog() == DialogResult.OK)
            {
                //cuadro de dialogo para elegir fichero
                OpenFileDialog ofdFileToBeam = new OpenFileDialog();
                if (ofdFileToBeam.ShowDialog() == DialogResult.OK)
                {
                    Cursor.Current = Cursors.WaitCursor;
                    System.Uri uri = new Uri("obex://" +
                    sbdd.SelectedDevice.DeviceAddress.ToString() + "/" +
                    Path.GetFileName(ofdFileToBeam.FileName));
                    ObexWebRequest request = new ObexWebRequest(uri);
                    request.ReadFile(ofdFileToBeam.FileName);

                    ObexWebResponse response =
                    (ObexWebResponse)request.GetResponse();
                    MessageBox.Show(response.StatusCode.ToString());
                    response.Close();
                    Cursor.Current = Cursors.Default;
                }
            }
        }
    }
}
```

### \* Código del PC (parte servidor)

```
using System;
using System.Windows.Forms;
using System.Threading;
using InTheHand.Net;
using InTheHand.Net.Bluetooth;

namespace EjemploOBEX
{
    public partial class Form1 : Form
    {
        private ObexListener obexListener;

        public Form1()
        {
            InitializeComponent();
            //comprobamos que soporta la pila de Microsoft
            if (!BluetoothRadio.IsSupported)
                this.Close();
            //activamos el bluetooth
            BluetoothRadio br = BluetoothRadio.PrimaryRadio;
            br.Mode = RadioMode.Discoverable;
            //arrancamos el servidor OBEX
            obexListener = new ObexListener(ObexTransport.Bluetooth);
            obexListener.Start();
            //arrancamos el hilo
            Thread th = new Thread(new ThreadStart(hilo));
            th.Start();
        }

        public void hilo()
        {
            while (obexListener.IsListening)
            {
                try
                {
                    ObexListenerContext olc =
obexListener.GetContext();
                    ObexListenerRequest olr = olc.Request;
                    string filename =
                    Uri.UnescapeDataString(olr.RawUrl.TrimStart(new char[] {
                    '/' }));
                    olr.WriteFile(System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal) + "\\ " +
                    DateTime.Now.ToString("yyMMddHHmmss") + " " + filename);
                    MessageBox.Show("Recibido fichero <" + filename +
">");
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Error: " + ex.Message);
                    break;
                }
            }
        }
    }
}
```