Creating a Bluetooth server running OBject EXhange (OBEX) protocol

Introduction

Origin of OBEX

The OBject EXhange (OBEX) protocol was originally designed to allow file transfer over infrared links, these are using light pulses in the infrared spectrum region.

Current use of OBEX

In recent times OBEX has been seen as a way to transfer files to mobile phones without the need for physical connection. Bluetooth has been adopted by many devices, however many devices Only support the OBEX Push operation. This means you can only send a file to the phone.

Profile hierarchy

The base of the profile hierarchy is the generic access profile (GAP)

Generic OBject EXchange Profile

The underlying profile device in JME (TM) is referenced as "btgeop". There are three profiles

Object Push profile

This is standard on all Nokia Bluetooth mobile phones and some other mobile phones.

Synchronization profile

To use the format you must use the WBMXL format.

Nokia Developer has a thread on this feature SyncML, OBEX over Bluetooth

Change Document for SyncML Synchronization Protocol Synchronization document

File Transfer profile

The phone can use a program to transfer data to and from the phone. There are two types programming language available Symbian 60 and JME (tm) Java language.

UUID codes for use in OBEX

UUID are in hexadecimal setting the value to false uses the full 128 bit code. The use of the true value means that the standard values can be used

UUID uuid = new UUID("1106", true); /* standard for OBEX PUSH */

UUID uuid = new UUID("9106", true); /* chosen value for the code Server */

Libraries

These programming languages talk to the bluetooth devices using supplied libraries. The JME libraries come from Java (TM) Standard Routines JSR82 applies to bluetooth

The file transfer profile allows the file transfer, in JME on some Nokia phones only part of OBEX is implemented. This means extra code must be added so we add Avetana Bluetooth

This is compiled using Sun Wireless Toolkit 2 or using Netbeans Mobility Toolkit

The output file is sent via Nokia PC Suite to a Series 40 or S60 phone.

(C) Copyright Nokia 2014. All rights reserved.

Obex code

Client Code

```
/* FTclient.jar */
```

```
package phone;
/*original imports */
import java.io.*;
import java.util.*;
import javax.microedition.io.*;
/* switch of J2se netbeans dev kit */
import javax.bluetooth.*;
// import javax.obex.*;

/* add the de.aventna obex */
import de.avetana.bluetooth.obex.OBEXConnection;
import de.avetana.bluetooth.obex.HeaderSetImpl;
import de.avetana.bluetooth.obex.CommandHandler;
import de.avetana.bluetooth.obex.MD5;
import de.avetana.bluetooth.obex.OperationImpl;
import de.avetana.bluetooth.obex.SessionNotifierImpl;
import de.avetana.javax.obex.Authenticator;
import de.avetana.javax.obex.ClientSession;
import de.avetana.javax.obex.HeaderSet;
import de.avetana.javax.obex.Operation;
import de.avetana.javax.obex.PasswordAuthentication;
import de.avetana.javax.obex.ResponseCodes;
import de.avetana.javax.obex.ServerRequestHandler;
import de.avetana.javax.obex.ServiceRegistrationException;
import de.avetana.javax.obex.SessionNotifier;


public class FTClient implements DiscoveryListener {
   LocalDevice local = null;
   DiscoveryAgent agent = null;
   int[] attrSet = null;
   RemoteDevice btDev = null;
   String serviceURL = null;
   ClientSession con = null;
   HeaderSet hdr = null;

    public FTClient() throws BluetoothStateException{

       // initialize the stack, if needed
   local = LocalDevice.getLocalDevice();
       agent = local.getDiscoveryAgent();
       agent.startInquiry(DiscoveryAgent.GIAC, this);
    }

    public void deviceDiscovered(RemoteDevice btDevice,DeviceClass cod){
       btDev = btDevice;
       System.out.println("Device discovered " +
         btDevice.getBluetoothAddress());
    }
```


(C) Copyright Nokia 2014. All rights reserved.

```
    public void servicesDiscovered(int transID, ServiceRecord[] servRecord){
       System.out.println("Discovered a service ....");
       for(int i =0; i < servRecord.length; i++){
          serviceURL =
         servRecord[i].getConnectionURL(ServiceRecord.NOAUTHENTICATE_NOENCRYPT,
            true);
          System.out.println("The service URL is " + serviceURL);
       }
    }

    public void serviceSearchCompleted(int transID, int respCode){
       System.out.println("Service search completed ........... ");
       System.out.println("Opening a connection with the server ....");
       try{
         con = (ClientSession)Connector.open(serviceURL);
         hdr = con.connect(hdr);
         System.out.println("Response code of the server after connect..." +
           hdr.getResponseCode());
           //Sending a request to server for file Hello.txt
         hdr = con.createHeaderSet();
         hdr.setHeader(HeaderSet.TYPE,"text/vCard");
         hdr.setHeader(HeaderSet.NAME,"Hello.txt");
         Operation op = con.get(hdr);

         //The server is now sending the file
         InputStream in = op.openInputStream();

         // Writing the file from server to local file system.
     StreamConnection filestream =
           (StreamConnection)Connector.open("file://name=HelloFile.txt;mode=w");
     OutputStream out = filestream.openOutputStream();

           //read and write the data
           int data = in.read();
           while(data != -1){
              out.write((byte)data);
              data = in.read();
           }

           // send the DISCONNECT Operation
           //con.disconnect();

           // cleanup
       op.close();
           in.close();
           out.close();
      }
      catch(IOException e){
       System.out.println(e.getMessage());
      }
    }
    public void inquiryCompleted(int discType){
       System.out.println("Inquiry completed ... ");
       UUID[] uuids = new UUID[1];
       uuids[0] = new UUID("1106",true);
```

 (C) Copyright Nokia 2014. All rights reserved.

```
        try{
           if(btDev == null){
         System.out.println("No device has been discovered, " +
              "hence not worth proceeding exiting .... ");
         System.exit(1);
         }
         System.out.println("Now searching for services ........ ");
         agent.searchServices(attrSet, uuids, btDev, this);
      }
      catch(BluetoothStateException e) {System.out.println(e.getMessage());}
     }

     public static void main(String args[]) throws IOException {
        FTClient client = new FTClient();
     }
 }
```

Programming in OBEX

Required Software

The Microsoft Windows XP Service Pack 2 (TM) bluetooth supported devices

see the list of Windows XP Service Pack 2 devices

Netbeans IDE package will install the Netbeans IDE then add the Netbeans Mobility Toolkit

The runtime Java (TM) version 2

The BlueCove XP installation kit.

The binary and tutorial on setting up BlueCove here

The Java (TM) Standard Edition (J2SE) Server code

```
/* J2SEbtSPPlinkServer.jar */
/* j2se compile and run under netbeans */
/* uses Bluecove XP */
/* and uses de.avetana package*/
//
package J2SEbtSPPlinkServer;
//import bartelo.javax.microedition.io.HttpConnection;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import javax.microedition.io.*;
import java.io.*;
import javax.bluetooth.*; //comes from bluecove
//import javax.obex.*;
// uses the de.avetana OBEX library
import de.avetana.obexsolo.OBEXConnector;
import de.avetana.javax.obex.HeaderSet;
import de.avetana.javax.obex.Operation;
import de.avetana.javax.obex.ResponseCodes;
import de.avetana.javax.obex.ServerRequestHandler;
import de.avetana.javax.obex.SessionNotifier;


public class J2SEbtSPPlinkServer implements ActionListener, Runnable{
```

 (C) Copyright Nokia 2014. All rights reserved.

```
     // Bluetooth singleton object
       LocalDevice device;
       DiscoveryAgent agent;
       String HTBTurl = null;
       Boolean mServerState = false; // stop is default state

       Thread mServer = null;
       String msgOut = "srv out msg";
       String msgIn = "no msg rcv";
       StreamConnectionNotifier btServerNotifier     ;
       UUID uuid = new UUID("9106", true);
       JLabel spacerlabel = new JLabel(" ");
 JButton startButton = new JButton("Start Server");
 JTextArea textarea = new JTextArea("",20, 40);
       JButton endButton = new JButton("End Server");

  public J2SEbtSPPlinkServer(){

  //Give it the Java look and feel
  JFrame.setDefaultLookAndFeelDecorated(true);

  JFrame frame = new JFrame("FileServer ");
  frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

  JScrollPane scrollPane = new JScrollPane(textarea);
  textarea.setEditable(false);

  Container cp = frame.getContentPane();
  cp.setLayout(new BoxLayout(cp, BoxLayout.Y_AXIS));

  startButton.setAlignmentX(Component.CENTER_ALIGNMENT);
  startButton.addActionListener(this);
               cp.add(startButton);

  endButton.setAlignmentX(Component.CENTER_ALIGNMENT);
  endButton.addActionListener(this);
               cp.add(endButton);


  spacerlabel.setAlignmentX(Component.CENTER_ALIGNMENT);
  cp.add(spacerlabel);

  scrollPane.setAlignmentX(Component.CENTER_ALIGNMENT);
  cp.add(scrollPane);

  frame.pack();
  frame.setVisible(true);

  updateStatus("[server:] FileServer Application started");
  updateStatus("[server:] Press the \"Start Server\" button to await for client
devices");


 }

       private void startServer() {
           if (mServer !=null)
```


(C) Copyright Nokia 2014. All rights reserved.

```
                return;
            //start the server and receiver
            mServer = new Thread(this);
            mServer.start();
        }

        private void endServer() {
            if (mServer == null)
                return;
            try {
                mServer.join();
            } catch (Exception ex) {};
            mServer = null;

        }




        public void run(){
  try {
   //UUID uuid = new UUID("1106", true);
                        UUID uuid = new UUID("9106", true);
                        device = LocalDevice.getLocalDevice(); // obtain reference to
singleton
                        device.setDiscoverable(DiscoveryAgent.GIAC); // set Discover mode
to LIAC
                }catch (Exception e)
                           { System.err.println("Cant init set discvover");
                             e.printStackTrace();
                }
   String url = "btspp://localhost:" + uuid +
";name=BTTP;authenticate=false;master=false;encrypt=false";
//Bluecove does not support btgeop on server
//   String url = "btgoep://localhost:" + uuid +
";name=BTTP;authenticate=false;master=false;encrypt=false";

  try{

                  // obtain connection and stream to this service
                    btServerNotifier = (StreamConnectionNotifier) Connector.open( url );
                } catch ( Exception e) {
                    e.printStackTrace();
                }


                while (mServerState )
                  {
                       StreamConnection btConn = null;
                       try {
                          updateStatus("[server:] Now waiting for a client to connect");

                          btConn = btServerNotifier.acceptAndOpen();
                      } catch (IOException ioe) { }
                      if (btConn != null) processConnection(btConn);
                }
```

 (C) Copyright Nokia 2014. All rights reserved.

```
        }

        void processConnection(StreamConnection conn) {
     updateStatus("[server:] A client is now connected");

            try {
                  DataInputStream in = conn.openDataInputStream();

//                   // write data into serial stream
//                   msgIn = in.readUTF();

                char s ='a';

                String h =null;
                try {
                    while (s!='\n'){
                        s = in.readChar();
                        System.out.println(s);
                        h += s;
                    }
                    System.out.println(s); // typing the result
                } catch (IOException ex) {
                    System.out.println("unable to handle incoming data");
                    ex.printStackTrace();
                }
                  msgIn = h;
                  in.close();
                  System.out.print("The receive message is '" + msgIn + "'");

                   Thread.sleep(1000);

                   DataOutputStream out = conn.openDataOutputStream();
                   msgOut = msgIn +" srv reply";
                   System.out.print("Sending the message is '" + msgOut + "'");

                  // write data into serial stream
                  out.writeUTF( msgOut );
                  out.flush();


                   Thread.sleep(1000);

                  // finish, close output stream
                  out.close();

                } catch (Exception e)
                {
                  e.printStackTrace();

                }
                try {
                    conn.close();
        updateStatus("[server:] Finished connection");

                }catch (Exception e ){ }
```

 (C) Copyright Nokia 2014. All rights reserved.

```
        }

  public void actionPerformed(ActionEvent e) {
                if ((e.getActionCommand()).equals("Start Server")    ) {
                    startButton.setEnabled(false);
                    mServerState = true; // set server state started
                    startServer();
                }
                if ((e.getActionCommand()).equals("End Server")    ) {
                    endButton.setEnabled(false);
                    startButton.setEnabled(true);
                    mServerState = false;
                    endServer();

                 }

  }



  public void updateStatus(String message){
   textarea.append("\n" + message);

  }


  public static void main(String[] args) {
   new J2SEbtSPPlinkServer();
  }
 }
```

Modifications to Avetana package

This is original code which replaces code in package provides a working code.

```
/* OBEXConnector.java
 * Created on 20.01.2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */
package de.avetana.obexsolo;

import java.io.IOException;
import java.util.Enumeration;
import javax.bluetooth.DataElement;
import javax.bluetooth.ServiceRecord;
import javax.bluetooth.UUID;
import javax.microedition.io.Connection;
import javax.microedition.io.StreamConnection;
import javax.microedition.io.StreamConnectionNotifier;
import de.avetana.bluetooth.obex.OBEXConnection;
import de.avetana.bluetooth.obex.SessionNotifierImpl;

/**
```


(C) Copyright Nokia 2014. All rights reserved.

```
 * @author gmelin
 *
 *
 */
public class OBEXConnector {

 public static Connection open (String url) throws IOException {
  if (!url.startsWith("btgoep://")) throw new IOException ("Only OBEX Connections
supported");
  url = "btspp" + url.substring(6);
        if(url.startsWith("btspp://localhost")) {
          StreamConnectionNotifier notifier = (StreamConnectionNotifier)
javax.microedition.io.Connector.open (url);
         try {

    ServiceRecord srec =
javax.bluetooth.LocalDevice.getLocalDevice().getRecord(notifier);

    //Set the serviceclassID List to OBEX-OBJECT-PUSH
    DataElement serviceClassIDList = srec.getAttributeValue(0x01);
    DataElement newSCList = new DataElement (DataElement.DATSEQ);
    Enumeration v = (Enumeration)serviceClassIDList.getValue();
    while (v.hasMoreElements()) {
     DataElement de = (DataElement)v.nextElement();
     if (!(de.getValue().equals(new UUID (0x1101))))
       newSCList.addElement(de);
    }
    newSCList.addElement(new DataElement(DataElement.UUID, new UUID(0x1105)));
       srec.setAttributeValue(0x01, newSCList);

       //Upate the protocol Descriptor list to contain OBEX
    /* Updating the protocolDescriptor list to change from SPP to OBEX does not seem to
work
       DataElement protocolDescriptorList = srec.getAttributeValue(0x04);
    DataElement newProtDescList = new DataElement (DataElement.DATSEQ);
    v = (Enumeration)protocolDescriptorList.getValue();
    while (v.hasMoreElements()) {
     DataElement de = (DataElement)v.nextElement();
     newProtDescList.addElement(de);
    }

       DataElement obexDescriptor = new DataElement(DataElement.DATSEQ);
       obexDescriptor.addElement(new DataElement(DataElement.UUID, new UUID(0x08)));

       newProtDescList.addElement(obexDescriptor);
       srec.setAttributeValue(0x04, newProtDescList);
       */

       //Update Supported Formats list
        DataElement sfl = new DataElement (DataElement.DATSEQ);
        sfl.addElement(new DataElement (DataElement.U_INT_1, 1));
        sfl.addElement(new DataElement (DataElement.U_INT_1, 2));
        sfl.addElement(new DataElement (DataElement.U_INT_1, 4));
        sfl.addElement(new DataElement (DataElement.U_INT_1, 5));
        sfl.addElement(new DataElement (DataElement.U_INT_1, 6));
        sfl.addElement(new DataElement (DataElement.U_INT_1, 255));
        srec.setAttributeValue(0x0303, sfl);
```

 (C) Copyright Nokia 2014. All rights reserved.

```
        //Update service availability
        srec.setAttributeValue(0x0008, new DataElement (DataElement.U_INT_1, 255));

        //Update Profile Descriptor List
        DataElement profileDescriptorList = new DataElement(DataElement.DATSEQ);
           DataElement profileDescriptor = new DataElement(DataElement.DATSEQ);
           profileDescriptor.addElement(new DataElement(DataElement.UUID, new
UUID(0x1105)));
        profileDescriptor.addElement(new DataElement(DataElement.U_INT_2, 256));
            profileDescriptorList.addElement(profileDescriptor);
            srec.setAttributeValue(0x0009, profileDescriptorList);

            //Add to public browse group
        DataElement elem = new DataElement(DataElement.DATSEQ);
        elem.addElement(new DataElement(DataElement.UUID, new UUID(0x1002)));
       srec.setAttributeValue(0x0005, elem);

//       update the service db:
// wont work talking direct to server without pointers!
//       javax.bluetooth.LocalDevice.getLocalDevice().updateRecord(srec);
          } catch (Exception e) {
     // TODO Auto-generated catch block
     e.printStackTrace();
    }

         return (Connection) new SessionNotifierImpl (notifier);
         }
         else {
         StreamConnection streamCon = (StreamConnection)
javax.microedition.io.Connector.open (url);
           return (Connection)new OBEXConnection (streamCon);
         }
 }

}
```

Setup Windows XP for transfer

The phone needs to be bluetooth "paired" with Windows XP or you will need to enter a security "pin" number every time. This is a security feature built into bluetooth which cannot be overidden. Manual Windows XP transfer is carried out using the bluetooth icon in the system tray.

Appendix

There does not seem to be any bluetooth OBEX server code for Microsoft Visual C++ or .NET, even for Windows Mobile.

Linux does have the support of BlueZ, however since integration to Linux kernel most of the bluetooth applications need updating or rewriting.

References

The J2SEbtSPPlinkServer is based on the published work

Bluetooth boogies, Part 1: File transfer with JSR-82 and OBEX

 (C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.