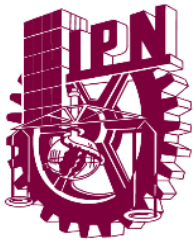

Instituto Politécnico Nacional

Escuela Superior de Cómputo

❖ **INTEGRANTES DEL EQUIPO:**

- Barrera Perez Carlos Tonatihu
- López Higuera Antonio
- Monroy Martos Elioth

❖ **UNIDAD DE APRENDIZAJE:** Application Development for Mobile Devices

❖ **PROFESOR:** Alfredo Sigfrido Cifuentes Alvarez.

❖ **PROYECTO RECUPERACIÓN:** Aplicación Android-Arduino con comunicación Bluetooth, para la extracción de información de diversos sensores.

❖ **FECHA**: 1 de abril de 2019.

❖ **Versión del reporte:** 1.

❖ **GRUPO:** 3CM8.

Índice

Índice **2**

Objetivo **3**

Objetivos específicos 3

Conceptos **3**

Desarrollo **4**

Descripción 4

Programación de la aplicación Android 4

Desarrollo del circuito y programación de Arduino 4

Software y hardware utilizados 4

Diagramas 6

Código fuente 7

Pruebas 19

Conclusiones **20**

Dificultades encontradas 20

Posibles aplicaciones 20

Conclusiones individuales 20

López Higuera Antonio 20

Monroy Martos Elioth 20

Bibliografía **21**

Objetivo

Realizar la medición de los valores arrojados por 4 sensores diferentes (temperatura, voltaje, magnetismo y tacómetro) mediante el uso de Arduino, y mostrar los valores obtenido en una aplicación móvil Android mediante la transferencia de datos vía Bluetooth.

Objetivos específicos

- Medir la temperatura ambiente usando un LM35.
- Medir el magnetismo de un imán usando un 49E.
- Medir el voltaje de salida de un potenciómetro.
- Medir las revoluciones por minuto (rpm) de una hélice.
- Comunicar un arduino con una aplicación Android.

Conceptos

Para el entendimiento de esta proyecto es necesario conocer los siguientes conceptos los cuales fueron clave durante el desarrollo de esta aplicación.

- **Bluetooth**. Bluetooth es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) creado por Bluetooth Special Interest Group, Inc. que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2.4 GHz**.** [3]
- **Arduino**. Es una plataforma de creación de electrónica de código abierto, la cual está basada en hardware y software libre. Esta plataforma permite crear diferentes tipos de microordenadores de una sola placa a los que la comunidad de creadores puede darles diferentes tipos de uso. [4]
- **Java.** Es un lenguaje de programación orientado a objetos, compilado e interpretado por su máquina virtual, Java puede ser utilizado en prácticamente cualquier tipo de proyecto debido a su portabilidad.[2]
- **Android**. Es un sistema operativo para dispositivos móviles, basado en el núcleo de linux, diseñado principalmente para dispositivos táctiles.[5]
- **Activity Android**. Una actividades es un punto de entrada para la interacción con el usuario. Representa una pantalla con una interfaz de usuario.[5]
- **Android Manifest**. Es un archivo que describe la información esencial de la aplicación a las herramientas de construcción de Android, al sistema operativo y a Google Play. En esta se describe el nombre del paquete, los componentes de la aplicación (incluye todas las actividades, servicios,

content provides, entre otros), los permisos que requiere la aplicación, entre otras declaraciones esenciales. [5]

- **Material Design.** Es un lenguaje visual que sintetiza los principios clásicos del buen diseño con la innovación de la tecnología y ciencia.[6]

Desarrollo

Descripción

El proyecto debe medir la temperatura ambiente, el voltaje de salida de un potenciómetro, el magnetismo de un imán, y las revoluciones por minuto de una hélice. A su vez, estas mediciones deben ser leídas con Arduino y enviadas vía Bluetooth a una aplicación móvil hecha en Android. El proyecto por ende, se dividió en dos partes, la primera el desarrollo de la aplicación móvil, y la segunda, el desarrollo del circuito y la programación del Arduino.

Programación de la aplicación Android

Para la programación del mismo, se usó como ejemplo, los programas proporcionados por el profesor, y se realizaron las debidas modificaciones, como el que la aplicación permite la comunicación bidireccional, tanto que reciba datos de entrada enviados por el Arduino, como que pudiera enviar datos hacia el mismo.
Además se hizo uso de un Recycler view para mostrar la información de los sensores.

Desarrollo del circuito y programación de Arduino

En primer instancia, se eligió el dispositivo que le daría la capacidad de comunicación inalámbrica al Arduino Uno, en este caso, el dispositivo elegido, fue el módulo Bluetooth HC-05, debido a que era un dispositivo con el cual habíamos trabajado anteriormente. Además, de que cumple con el objetivo de permitir una comunicación inalámbrica entre Arduino y teléfono.
Posteriormente, se ensambló el circuito con el cual se pudiera realizar la medición de los valores arrojados por los distintos sensores, para realizar esto nos apoyamos de las hojas de datos de cada uno.[1]

Software y hardware utilizados

A continuación, se presenta una lista del software y hardware usado para la elaboración del proyecto:

- **Android Studio 3.4.** Android IDE para el desarrollo de aplicaciones para el sistema operativo Android.

- **Arduino UNO.** Es una placa con un microcontrolador que permite su programación para el desarrollo de aplicaciones electrónicas.
- **Arduino IDE**. Entorno de desarrollo para arduino que permite crear programas y además permite cargar los mismos a la placa de Arduino.
- **Modulo Bluetooth *HC-05.*** Es el módulo que se utilizó en la placa arduino para la comunicación entre dicho microcontrolador y un teléfono celular, debido a que es un módulo que se ajustaba a las necesidades del proyecto, ya que no es costoso, y su uso es sencillo.
- **Teléfono con Android 5+.** Usado para realizar la pruebas correspondientes.
- **Sensor LM35.** Sensor analógico que permite medir la temperatura ambiente.
- **Sensor Hall radiométrico 49E.** Sensor analógico (no latch) que permite medir el magnetismo de un objeto.
- **Potenciómetro.** Es un elemento que actúa como una resistencia variable.
- **Sensor infrarrojo.** También conocido como sensor de proximidad, tiene una salida digital la cual indica si tiene algún objeto enfrente del mismo.

Diagramas

El diagrama general del circuito del sistema, es el que se muestra en la Figura 1.

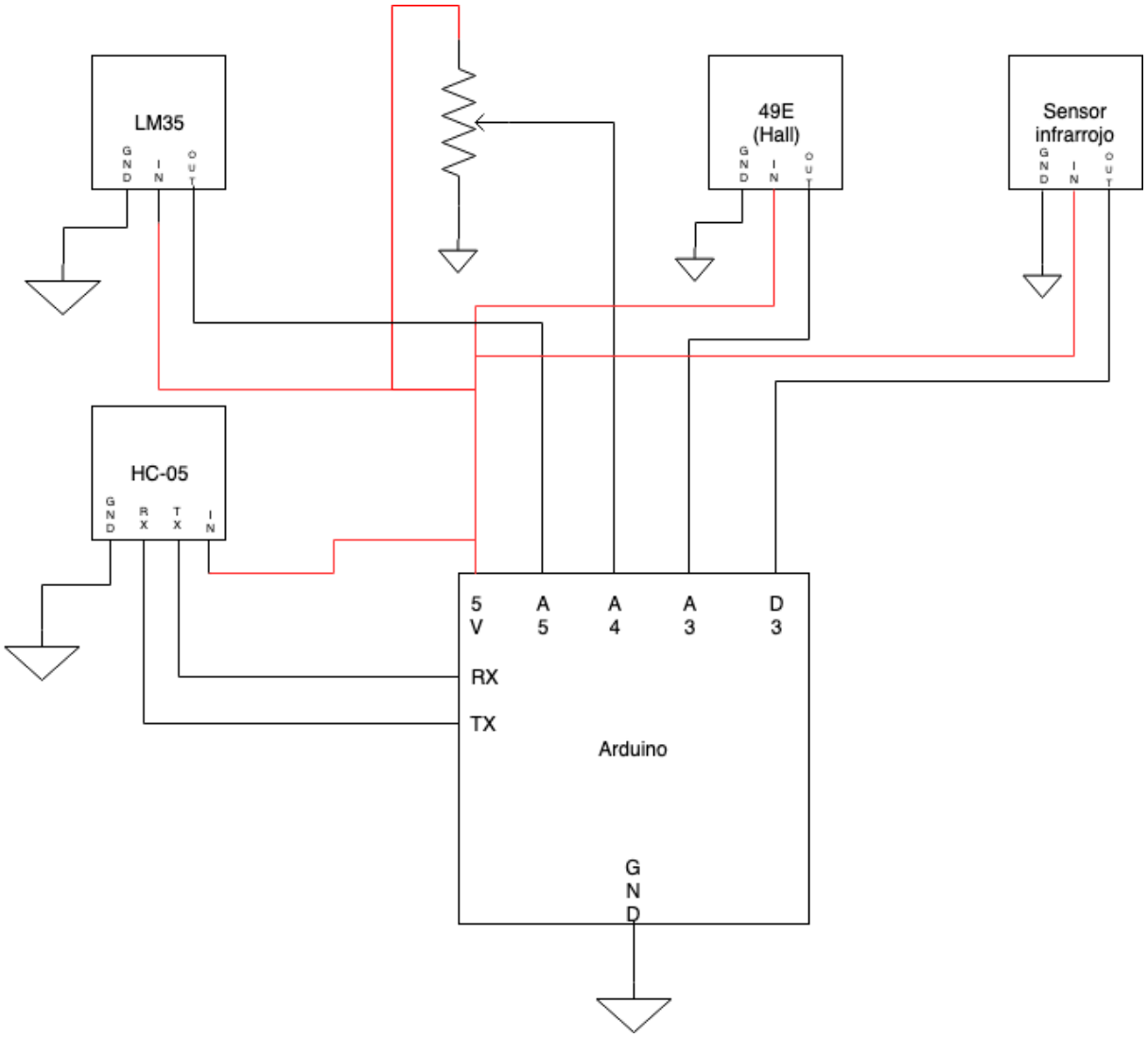

Figura 1. Diagrama general del circuito.

Una vez que se tuvo el diseño del circuito y los materiales mencionados anteriormente, se procedió al ensamblado del mismo. El resultado se puede observar en la Figura 2.

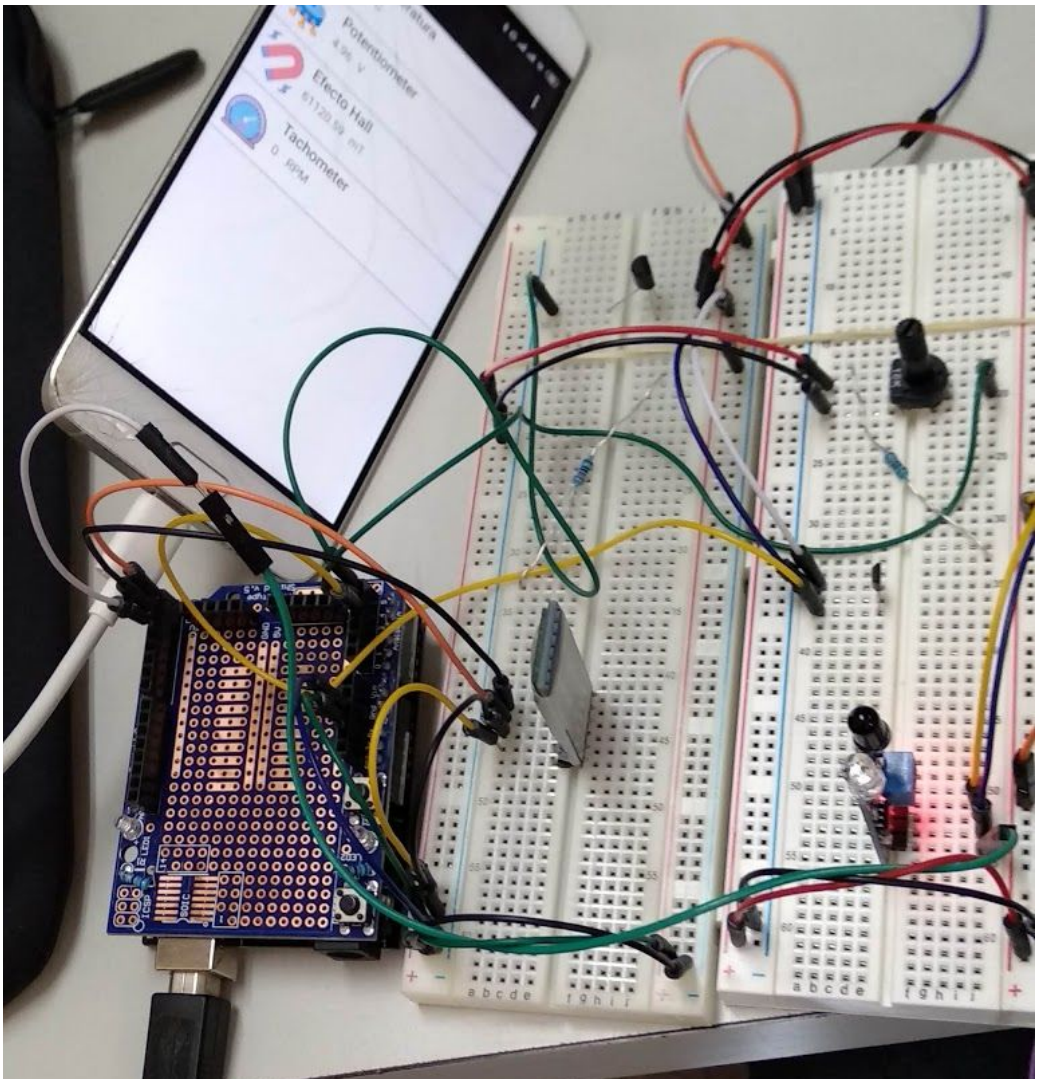

Figura 2. Ensamblado completo del circuito.

Código fuente

A continuación, se presenta el código elaborado para el desarrollo del proyecto.
AndroidManifest.xml (Para solicitar permisos de Bluetooth al dispositivo):

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.devicelist" >
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.devicelist.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
```

```
            <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
    </activity>
    <activity
        android:name="com.example.devicelist.ledControl"
        android:label="@string/title_activity_led_control" >
    </activity>
</application>
</manifest>
```

MainActivity.java (Clase desde la cual se selecciona el dispositivo bluetooth):

```
package com.example.devicelist;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.widget.TextView;
import android.widget.Toast;
import java.util.ArrayList;
import java.util.Set;
public class MainActivity extends AppCompatActivity
{
    //widgets
    Button btnPaired;
    ListView devicelist;
    //Bluetooth
    private BluetoothAdapter myBluetooth = null;
    private Set<BluetoothDevice> pairedDevices;
    public static String EXTRA_ADDRESS = "device_address";
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //Calling widgets
        btnPaired = (Button)findViewById(R.id.button);
        devicelist = (ListView)findViewById(R.id.listView);

        //if the device has bluetooth
        myBluetooth = BluetoothAdapter.getDefaultAdapter();

        if(myBluetooth == null)
        {
            //Show a mensag. that the device has no bluetooth adapter
            Toast.makeText(getApplicationContext(), "Bluetooth Device Not
Available", Toast.LENGTH_LONG).show();
```

```
            //finish apk
            finish();
        }
        else if(!myBluetooth.isEnabled())
        {
            //Ask to the user turn the bluetooth on
            Intent turnBTon = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(turnBTon,1);
        }
        btnPaired.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v)
            {
                pairedDevicesList();
            }
        });
    }
    private void pairedDevicesList()
    {
        pairedDevices = myBluetooth.getBondedDevices();
        ArrayList list = new ArrayList();

        if (pairedDevices.size()>0)
        {
            for(BluetoothDevice bt : pairedDevices)
            {
                list.add(bt.getName() + "\n" + bt.getAddress()); //Get the
device's name and the address
            }
        }
        else
        {
            Toast.makeText(getApplicationContext(), "No Paired Bluetooth
Devices Found.", Toast.LENGTH_LONG).show();
        }
        final ArrayAdapter adapter = new
ArrayAdapter(this,android.R.layout.simple_list_item_1, list);
        devicelist.setAdapter(adapter);
        devicelist.setOnItemClickListener(myListClickListener); //Method
called when the device from the list is clicked    }
    private AdapterView.OnItemClickListener myListClickListener = new
AdapterView.OnItemClickListener()
    {
        public void onItemClick (AdapterView<?> av, View v, int arg2, long
arg3)
        {
            // Get the device MAC address, the last 17 chars in the View
            String info = ((TextView) v).getText().toString();
            String address = info.substring(info.length() - 17);

            // Make an intent to start next activity.
            Intent i = new Intent(MainActivity.this, ledControl.class);

            //Change the activity.
            i.putExtra(EXTRA_ADDRESS, address); //this will be received at
ledControl (class) Activity
```

```
            startActivity(i);
        }
    };
    @Override
    public boolean onCreateOptionsMenu(Menu menu)
    {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.menu_device_list, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();
        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item); }}
```

activity_main.xml:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
    android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
tools:context="com.example.devicelist.MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/list_title2"
        android:textAppearance="@style/Base.TextAppearance.AppCompat.Large"
        android:id="@+id/textView"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentBottom="true"
        android:text="@string/list_title" />
    <ListView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:id="@+id/listView"
        android:layout_centerHorizontal="true"
        android:layout_above="@+id/button"
        android:layout_below="@+id/textView" />
</RelativeLayout>
```

ledControl.java (Clase desde la cual se obtiene y muestras las distintas mediciones de los sensores):

```
package com.example.devicelist;
import android.os.Bundle;
import android.provider.Settings;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.Menu;
import android.view.MenuItem;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.view.View;
import android.widget.Button;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;
import android.app.ProgressDialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.os.AsyncTask;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.UUID;
import android.os.Handler;
public class ledControl extends AppCompatActivity {
    Button btnDis;
    SeekBar brightness;
    TextView labelTemperatura, labelPote, labelHall, labelMotor;
    String address = null;
    private ProgressDialog progress;
    BluetoothAdapter myBluetooth = null;
    BluetoothSocket btSocket = null;
    private boolean isBtConnected = false;
    InputStream socketInputStream;
    OutputStream socketOutputStream;
    //SPP UUID. Look for it
    static final UUID myUUID =
UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");
    private ArrayList<Sensor> sensors;
    private RecyclerView recyclerView;
    private SensorAdapter sensorAdapter;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        Intent newint = getIntent();
```

```
        address = newint.getStringExtra(MainActivity.EXTRA_ADDRESS);
//receive the address of the bluetooth device
        //view of the ledControl
        setContentView(R.layout.activity_led_control);
        //Call the class to connect
        new ConnectBT().execute();
        sensors = new ArrayList<Sensor>();
        sensors.add(new Sensor("℃", "Temperatura", "0",
R.drawable.temperature));
        sensors.add(new Sensor("V", "Potentiometer", "0",
R.drawable.potentiometer));
        sensors.add(new Sensor("mT", "Efecto Hall", "0",
R.drawable.magnetic));
        sensors.add(new Sensor("RPM", "Tachometer", "0",
R.drawable.tacometer));
        recyclerView = findViewById(R.id.rv_sensors);
        sensorAdapter = new SensorAdapter(sensors);
        LinearLayoutManager linearLayoutManager = new
LinearLayoutManager(ledControl.this);
        linearLayoutManager.setOrientation(LinearLayoutManager.VERTICAL);
        recyclerView.setLayoutManager(linearLayoutManager);
        recyclerView.setAdapter(sensorAdapter);
    }
    private void Disconnect()
    {
        if (btSocket!=null) //If the btSocket is busy
        {
            try
            {
                btSocket.close(); //close connection
            }
            catch (IOException e)
            { msg("Error");}
        }
        finish(); //return to the first layout

    }
    // fast way to call Toast
    private void msg(String s)
    {
        Toast.makeText(getApplicationContext(),s,Toast.LENGTH_LONG).show();
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.menu_led_control, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        //noinspection SimplifiableIfStatement
        if (id == R.id.action_settings) {
            return true;
        }
        return super.onOptionsItemSelected(item);
```

```
    }
    private class ConnectBT extends AsyncTask<Void, Void, Void>  // UI
thread
    {
        private boolean ConnectSuccess = true; //if it's here, it's almost
connected
        @Override
        protected void onPreExecute()
        {
            progress = ProgressDialog.show(ledControl.this,
"Connecting...", "Please wait!!!");  //show a progress dialog
        }
        @Override
        protected Void doInBackground(Void... devices) //while the
progress dialog is shown, the connection is done in background
        {
            try
            {
                if (btSocket == null || !isBtConnected)
                {
                    myBluetooth =
BluetoothAdapter.getDefaultAdapter();//get the mobile bluetooth device
                    BluetoothDevice dispositivo =
myBluetooth.getRemoteDevice(address);//connects to the device's address
and checks if it's available
                    btSocket =
dispositivo.createInsecureRfcommSocketToServiceRecord(myUUID);//create a
RFCOMM (SPP) connection
                     BluetoothAdapter.getDefaultAdapter().cancelDiscovery();
                    btSocket.connect();//start connection
                }
            }
            catch (IOException e)
            {
                System.out.println("Error al conectarse: ");
                e.printStackTrace();
                ConnectSuccess = false;//if the try failed, you can check
the exception here
            }
            return null;
        }
        @Override
        protected void onPostExecute(Void result) //after the
doInBackground, it checks if everything went fine
        {
            super.onPostExecute(result);
            if (!ConnectSuccess)
            {
                msg("Connection Failed. Is it a SPP Bluetooth? Try
again.");
                finish();
            }
            else
            {
                System.out.println("Me conecte");
                msg("Connected.");
                isBtConnected = true;
```

```
                try{
                    socketOutputStream =  btSocket.getOutputStream();
                    socketInputStream =  btSocket.getInputStream();
                    socketOutputStream.write('1');
                    socketOutputStream.flush();
                }catch (Exception e){
                    e.printStackTrace();
                }
                final Handler handler = new Handler();
                final Runnable [] runnables = new Runnable[1];
                // Define the code block to be executed
                runnables[0]=new Runnable() {
                    @Override
                    public void run() {
                        // Insert custom code here
                        try {
                            String readMessage="";
                            String aux="";
                            String array[];
                            byte[] buffer;
                            int bytes;
                            buffer = new byte[60];
                            bytes = socketInputStream.read(buffer);
                            readMessage = new String(buffer, 0, bytes);
                            readMessage=readMessage.replace("\n","");
                            array=readMessage.split(";");
                            try{
                                sensors.get(0).setType(array[0]);
                                sensors.get(1).setType(array[1]);
                                sensors.get(2).setType(array[2]);
                                sensors.get(3).setType(array[3]);
                                sensorAdapter.notifyDataSetChanged();
                            }catch(Exception e){
                                e.printStackTrace();
                            }
                        } catch (IOException e) {
                            e.printStackTrace();
                        }
                        // Repetir cada segundo
                        handler.postDelayed(runnables[0], 1000);
                    }
                };
                handler.post(runnables[0]);
            }
            progress.dismiss();
        }
    }}
```

activity_led_control.xml:

```
<android.support.v7.widget.RecyclerView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/rv_sensors"
    android:orientation="vertical"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    tools:context="com.example.devicelist.ledControl">
</android.support.v7.widget.RecyclerView>
```

SensorAdapter.java (Adaptador de Recycler view):

```
package com.example.devicelist;
import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;
import java.util.ArrayList;
public class SensorAdapter extends
RecyclerView.Adapter<SensorAdapter.ViewHolder> {
    private ArrayList<Sensor> sensors;
    public SensorAdapter(ArrayList<Sensor> sensors) {
        this.sensors = sensors;
    }
    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int
i) {
        View view =
LayoutInflater.from(viewGroup.getContext()).inflate(R.layout.sensor_item,
viewGroup, false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolder viewHolder, int i) {
        Sensor sensor= sensors.get(viewHolder.getAdapterPosition());
        viewHolder.img.setImageResource(sensor.getImg());
        viewHolder.title.setText(sensor.getTitle());
        viewHolder.type.setText(sensor.getType());
        viewHolder.value.setText(sensor.getValue());
    }
    @Override
    public int getItemCount() {
        return sensors.size();
    }
    public static class ViewHolder extends RecyclerView.ViewHolder{
        private ImageView img;
        private TextView title;
        private TextView type;
        private TextView value;
        public ViewHolder(@NonNull View itemView) {
            super(itemView);
            img = itemView.findViewById(R.id.iv_img);
            title = itemView.findViewById(R.id.tv_title);
            type = itemView.findViewById(R.id.tv_type);
            value = itemView.findViewById(R.id.tv_value);
        }
    }
```

```
}
```

Sensor.java (POJO que tiene la información de cada sensor):

```
package com.example.devicelist;
public class Sensor {
    String value;
    String title;
    String type;
    int img;
    public Sensor(String value, String title, String type, int img) {
        this.value = value;
        this.title = title;
        this.type = type;
        this.img = img;
    }
    public String getValue() {
        return value;
    }
    public String getTitle() {
        return title;
    }
    public String getType() {
        return type;
    }
    public int getImg() {
        return img;
    }
    public void setValue(String value) {
        this.value = value;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public void setType(String type) {
        this.type = type;
    }
    public void setImg(int img) {
        this.img = img;
    }
}
```

activity_item.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="4dp">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <ImageView
            android:id="@+id/iv_img"
            android:layout_width="70dp"
```

```
            android:layout_height="70dp"
            android:padding="6dp"
            android:src="@mipmap/ic_launcher"/>
        <TextView
            android:id="@+id/tv_title"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/app_name"
            android:paddingLeft="5dp"

android:textAppearance="@style/Base.TextAppearance.AppCompat.Large"
            android:layout_toRightOf="@+id/iv_img"/>
        <TextView
            android:id="@+id/tv_type"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/app_name"
            android:paddingTop="5dp"
            android:paddingLeft="5dp"

android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium"
            android:layout_toRightOf="@+id/iv_img"
            android:layout_below="@+id/tv_title"/>
        <TextView
            android:id="@+id/tv_value"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/app_name"
            android:paddingLeft="10dp"
            android:paddingTop="5dp"

android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium"
            android:layout_toRightOf="@+id/tv_type"
            android:layout_below="@+id/tv_title"/>
    </RelativeLayout>
</android.support.v7.widget.CardView>
```

Código para dispositivo Arduino (Proyecto.ino):

```
const int pinTemperatura=A5;
const int pinPote=A4;
const int pinHall=A3;
const int pinMotor=2;
float valorTemperatura;
float valorPote;
float valorHall;
float rev=0.0;
long rpm;
int oldtime=0;
int tiempo;
int empiezo=0;
int serial = 0;
char a;
char arreglo[20];
String cadena;
void interrupcion(){
```

```
    rev++;
}
void setup(){
   pinMode(pinTemperatura,INPUT);
   pinMode(pinPote,INPUT);
   pinMode(pinHall,INPUT);
   Serial.begin(9600);
   attachInterrupt(digitalPinToInterrupt(pinMotor),interrupcion,FALLING);
}
void loop(){
    delay(1000);
    //Motor
    detachInterrupt(digitalPinToInterrupt(pinMotor));//quita interrupción
    tiempo=millis()-oldtime;        //encuentra tiempo
    rpm=(rev/tiempo)*60000/3;         //calcula rpm
    oldtime=millis();             //salvamos nuevo tiempo
    rev=0;
    //Temperatura
    valorTemperatura=analogRead(pinTemperatura);
    valorTemperatura=(5.0*valorTemperatura*100.0)/1023.0;//Fahrenheit
    valorTemperatura=(valorTemperatura-32.0)*(5.0/9.0);
    //Potenciometro
    valorPote=analogRead(pinPote);
    valorPote=(5.0*valorPote)/1023.0;
    //Sensor efecto hall
    valorHall=analogRead(pinHall);
    valorHall=(5.0*valorHall*500.0)/1023.0;
    valorHall=valorHall*53.33-133.33;
    if (Serial.available()){
cadena=String((int)valorTemperatura)+";"+String(valorPote)+";"+String(valor
Hall)+";"+String(rpm);
        Serial.println(cadena);
        Serial.flush();
    }
   attachInterrupt(digitalPinToInterrupt(pinMotor),interrupcion,FALLING);
}
```

Pruebas

A continuación se muestran algunas capturas de pantalla tanto del Serial Monitor de Arduino como de la aplicación móvil, en donde se pueden observar los datos medidos.

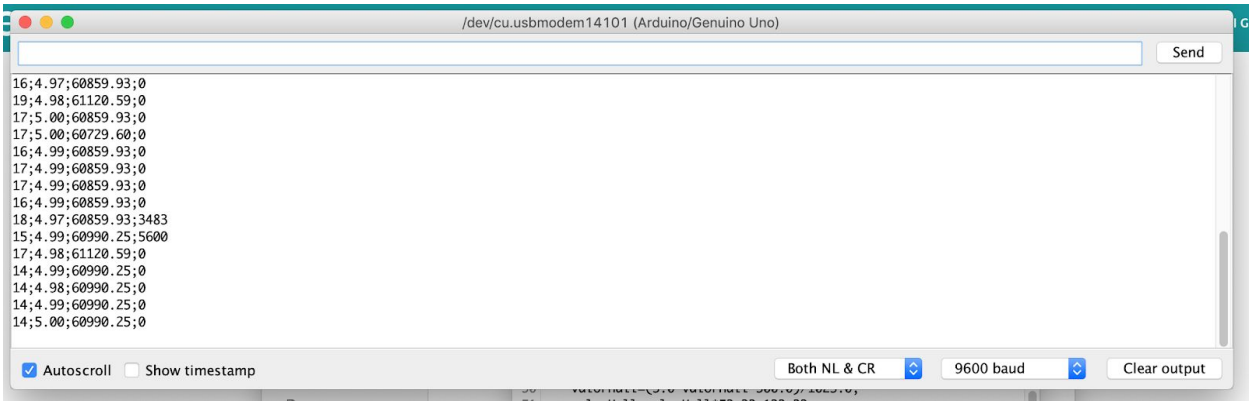

Figura 3. Salida del Arduino Serial Monitor.

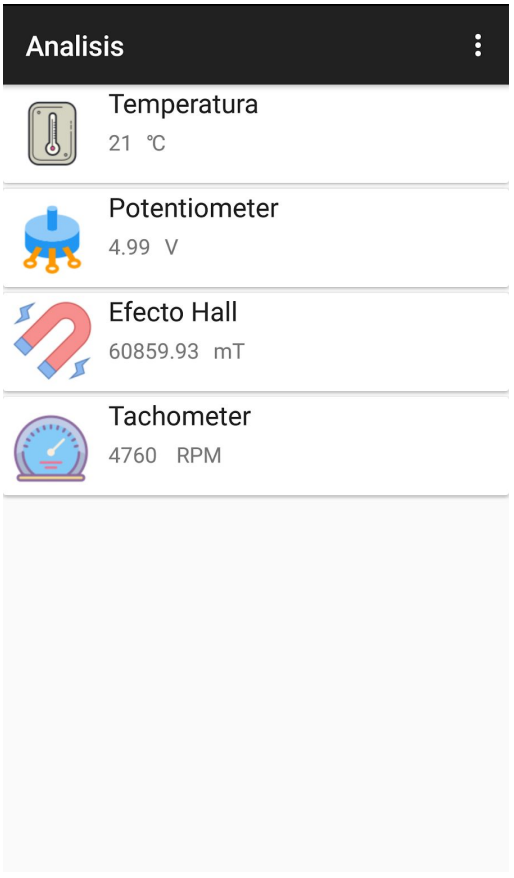

Figura 4. Captura de pantalla de la aplicación móvil Android.

Conclusiones

Dificultades encontradas

La comunicación serial de Arduino puede llegar a ser muy tediosa, en el anterior proyecto se había tenido dificultades con enviar información de la aplicación MIDlet al microcontrolador, sin embargo, esta vez fue al revés. Al enviar información del

microcontrolador al dispositivo Android el canal se llenaba de basura, lo cual generó distorsión en los datos al realizar pruebas, al implementar el método *flush* en la comunicación serial se eliminó el problema y el envío de información funcionó.

Posibles aplicaciones

Este proyecto entra dentro del campo de *internet de las cosas* teniendo muchísima aplicación, en especial en robótica y domótica, debido a que al controlar los sensores y recibir información, podríamos agregar otros componentes y accionar al recibir ciertas lecturas, como un *listener*.

Conclusiones individuales

López Higuera Antonio

El desarrollar aplicaciones que requieren interconexión siempre dejan un gran aprendizaje, permiten no sólo enfocarse en los problemas de una plataforma sino tener que buscar a un nivel más grande. Los problemas presentados en cualquier tipo de comunicación siempre estarán, lo cual me ha preparado para poder realizar más aplicaciones del estilo o *software* que requiere comunicación con alguna otra plataforma o servicio.

Monroy Martos Elioth

La elaboración de este proyecto me permitió trabajar de una forma más fuerte con Arduino, ya que el usar más sensores y tener que hacer uso de los pines de entrada analógica del mismo fue algo nuevo para mi, además de que el trabajar con la tecnología Bluetooth en Android fue algo complejo, ya que me tomó varios días lograr implementarlo. Además, la aplicaciones Android son muy usadas por lo cual es importante tener conocimiento sobre como implementarlas, y dado la conectividad del mundo actual, también resulta importante saber como conectar estas aplicaciones con otros dispositivos.

Bibliografía

- [1] Electronica60Norte, *Datasheet bluetooth to serial port module HC05*. 2006. [Consultado: 25- Feb- 2019]
- [2] M. Lozano Ortega and B. Bonev, "Introducción a Android", *Curso de Android y Java para Dispositivos Móviles*, 2010. [Online]. Disponible en: http://www.jtech.ua.es/apuntes/ajdm2010/sesiones/sesion09-apuntes.html. [Consultado: 25- Feb- 2019].

- [3] "Bluetooth", *Wikipedia*, 2019. [Online]. Disponible en: https://es.wikipedia.org/wiki/Bluetooth. [Consultado: 25- Feb- 2019].
- [4] Y. FM, "Qué es Arduino, cómo funciona y qué puedes hacer con uno", *Xataka.com*, 2018. [Online]. Disponible en: https://www.xataka.com/basics/que-arduino-como-funciona-que-puedes-hacer-uno. [Consultado: 25- Feb- 2019].
- [5] “Application Fundamentals”, Android Developer, 2019. [Online]. Disponible en: https://developer.android.com/guide/components/fundamentals. [Consultado: 27-Marzo-2019].
- [6] “Introduction”, Material Design, 2019. [Online]. Disponible en https://material.io/design/introduction/. [Consultado: 27-Marzo-2019].