Software Engineering
Course Code: DA216A

EasySpeech

Project report

Team name: PMJ

Team members: Pontus Olsson Martin Eriksson Jonas Karlsson

Date: 2015-05-21

Software Engineering
Course Code: DA216A

Innehåll

Software Engineering
Course Code: DA216A

1. Introduction and background

This project will cover a small area of voice recognition using an Arduino board called atmega328p (ATMEL, 2015), a mobile phone with a voice recognition software application that will compare your command with a local database, and if it exists within the database then it will increment the counter for that specific command by one. Depending on the command given to the database an "Id" will be sent to the Arduino via Bluetooth telling the atmega328p which light that shall be lit. This is only the first step towards bigger ambitions like creating a robot in medicine that is fully automatic and capable of understanding your problems through voice recognition. The project only represents a small portion of the voice recognition area, this is because voice recognition is still in development and is a very big area of expertise. Background knowledge for this project are the scientific articles that were researched in this area (1R.A.Ramlee, 2013), (Saume, 2012) and (Zhong, et al., 2014). Further background knowledge for this project has been lectures in embedded systems and lectures in programming courses.

2. Aim and purpose

The purpose of this project is to create a functional voice recognition application that will record your voice, return it in text, compare it to a local database with stored commands and give you the desired result.

2.1 Research questions

- How can we make voice recognition a more reliable input?
- How can voice recognition be used to improve the quality of life?

2.2 Limitation

One limitation of this project is time. This is because the project was only given one month of work. Second limitation to this project is the budget, this is because the time limit was so short and not enough resources were provided to make something bigger and more advanced. The last limitation for this project is the Arduino board, the one used in this project (ATMEL, 2015), is not meant to be used with voice recognition, there is an Arduino that is specified to be used with voice recognition but this was not available for this project (HMC, 2015).

3. Method

3.1 Literature review

- In (1R.A.Ramlee, 2013), a low-budget implementation of an automated home is performed. It exists of two different GUIs (one for PC/Laptop and one for Mobile phone). One interesting aspect of this project is the precautions used in case of malfunctioning hardware, for example; if the Windows GUI (PC/Laptop) does not work properly, the Android GUI (Mobile phone) can communicate directly with the microcontroller instead of going through the Windows GUI. This gives one ideas of different kinds of implementations. For example; instead of only having Bluetooth connection to the microcontroller, it is possible to activate WIFI connection when out of range for Bluetooth. The main difference of this project and the one being conducted is the usage of voice recognition, their future work was the implementation of voice recognition and therefor one might say that Easy Speech is an evolution of this project.

- The article (Zhong, et al., 2014) discuss about how voice controlled systems can help blind or disabled people when it comes to doing something simple like calling someone or even search for something on Google. The article mentions a program that runs in the background and always listens to your voice, then executes the commands given to it. This is a solution to everyday life for people that need a fast and easy way to interact with their phone, when they might not have the possibility or time to reach it. The discussion in this article (Zhong, et al., 2014) is what this project is all about, Easy Speech is a project that will make things easier in normal life, no matter if it is your phone, car or even your house. Therefor Easy Speech is an implementation of this article (Zhong, et al., 2014).

- In (Saume, 2012) there are discussions about voice recognition using android devices, and how it works. They discuss if voice recognition has come to the point that it is a reliable source of input, when it comes to input commands. They conclude that is it not developed enough for being a reliable input source, it often gets the words wrong or do not understand the words spoken. They tried the most popular applications in Android like Google word to text and they noticed a lot of problems with different letters and words. The resemblance in this project and Easy Speech is that we are both using voice to text, but Easy Speech is the next step of voice recognition were it has been implemented to hardware.

When looking for articles for this project, the databases on HKR home page (HKR, 2015), were used and all searching was filtered with the words “voice recognition” and “Android”. About 20 000 articles were found using these filter words. The articles that were reviewed were the top 20 searched ones, these articles were then reviewed and the articles used were the one most relevant to the project. How relevant the articles were to the project was decided by what the article was about, and how similar the articles were to this project. This was determined by if voice recognition were used in some way, or android were used and also about Arduino. The articles that were discarded were either not on the top 20, or not relative to the project.

The conclusion drawn from the articles is that voice recognition is very interesting for scientist interested in sound manipulation and other aspects of sound implementations. Voice recognition is a very young technology that has not yet been mastered, because of the lack of research in the area and there have not been a need for it until now.

3.2 Benchmarking

The result of this project is only a prototype that is not ready for the market. This project in voice recognition is only the start of something that could be used in real life for example: Smart houses, medicine, cars and mobile phones.

4. Results

4.1 Expected results

The expected result from this project is there will be a functional and good working program, which will work well with voice recognition. It will also be able to turn on the right light depending on the given command, and store every given command in the database.

4.2 Results

In this project Extreme Programming was used, between the iterations we used story cards and task cards. The story cards were used to know how far the project had gone, and what was left to do. The task cards were also used between iterations but they were individual between the members in the project, they were tasks for the members to be complete for the next iteration. The testing done in the project was Black box - White box testing, this is done by testing while programming. Instead of doing a complete code for testing, there were test between every code done to see if it worked. In this project GitHub was used as its revision control program. This was a very good combination with the Black box – White box testing, between every successful test the project would sync with GitHub and saved (PMJ, 2015).

The problems that were encountered in this project were:

- Learning the new programs: Android studio, Arduino playground.
- Programming and implementation to the application: Database, application functions, implementing voice to text.
- The connection between the Android app and the Bluetooth unit.
- Sending commands from the Android app to the Arduino.
- Receive commands in Arduino and make the desired output.

All these problems were solved by implementing code from tutorials and complete code, Black box – White box testing and pair programming. Tutorials and complete code were searched, to be able to learn the programs and to manage the implementations of the database, application functions and voice to text. The tutorials were searched were to learn the different programs, and learn how to manage the send and receive commands from the application to the Arduino board. Using all this knowledge this project was able to achieve its goals and make a functional program, and create all the functions needed for this project to work. After a lot of research the Bluetooth connection between the Arduino and the application was achieved, using all the tutorials and code found. Sending and receiving the commands to the Arduino was a big problem in this project, this was the problem researched the most and it showed out to be the physical connection that was wrong.

4.3 Velocity

The average velocity for this project have been 0.8, this have been calculated using the velocity's gained from each iteration, by dividing the actual worked hours with the planned hours for each iteration. For more detailed information regarding the velocity for each iteration see *figures (1) , (2) , (3) , (4) , (5)* in the appendix.

5. Social and ethical aspects

Voice recognition can be wrong if you implement it somewhere and save personal data without the authorization from the individual in question. Even if you have authorization from the user to save personal data, it needs to be done in a good and secure way. It would also be an ethical problem if this would be mass produced, it would not be good for the environment. There is also a good side to the environment, this is because you could optimize the electronics in the house so a lot of electricity would be saved. This leads to greater life length on the equipment and less environment pollution from making more electronics.

6. Discussion and conclusion

6.1 Discussion

In the start of this project all members were declared different areas to be their own, and that they should focus on in this project. These areas were: Android programming, Database, Arduino and Bluetooth. All these areas were researched and implemented in this project. This project has processed in a good rate with all the tasks, and has always been on schedule with the customers demands and wishes. Overall this project have been running smoothly over this past month and all its members have committed their all for this project to make it as good as possible.

6.2 Conclusion

Conclusions drawn from this project is that you can easily improve your normal life with voice recognition, by implementing it to your house, car or phone. Making it easier to do normal days activates like turning off all lights in your house with one command, or calling your mother from your car by just telling it to. About voice recognition not being a good input is true, due to the voice recognition in this project have difficulties recognizing exactly what have been spoken, but gets it right for about 75% of the time.

7. Suggestions for further work

Things that could be done further after this project is to develop the app more, insert more functions to make it more universal, use more advanced hardware to be able to implement it to for example the smart house concept, example (Gavazzi, 2015). The areas that were not implemented in this project were to advance to be done in time, and the project did not have enough resources to be able to create the functions needed, and get the hardware needed. The areas that should have been implemented were voice recognition, voice feedback from a robot with different feedbacks depending on the person and the command given. There were also supposed to be a functional robot, that would be able to respond to different people depending on what they said to the robot and who it was.

8. References

- 1R.A.Ramlee, 2. 3. 4. 5. 6. 7. 8. S., 2013. Bluetooth Remote Home Automation System Using Android Application. *Bluetooth Remote Home Automation System Using Android Application*, 1(1), p. 5.

- ATMEL, 2015. ATmega48A/PA/88A/PA/168A/PA/328/P. Volym 1, p. 650.

- Gavazzi, C., 2015. *Smart-House.* [Online]
  Available at: http://www.smart-house.nu/
  [Använd 25 05 2015].

- HKR, 2015. *www.HKR.se.* [Online]
  Available at: http://www.hkr.se/sv/lrc/biblioteket/databaser/
  [Använd 17 05 2015].

- HMC, 2015. HM2007. *HM2007 Datasheet,* Volym 1, p. 21.

- PMJ, 2015. *GitHub.* [Online]
  Available at: https://github.com/Pontus92/JAMP
  [Använd 25 05 2015].

- Saume, D. L. &. C., 2012. En undersökning av röststyrning för Android-enheter. Volym 1, p. 35.

- Zhong, Y. o.a., 2014. JustSpeak: Enabling Universal Voice Control on Android. p. 4.

9. Appendices and enclosures

9.1 Arduino Code

```
Author: Jonas Karlsson Pontus Olsson Martin Eriksson
int command;


int
led1 = 2, //Connect LED 1 To Pin #2 Head
led2 = 3, //Connect LED 2 To Pin #3 Right Arm
led3 = 4, //Connect LED 3 To Pin #4 Right Leg
led4 = 5, //Connect LED 4 To Pin #5 Left Leg
led5 = 6; //Connect LED 5 To Pin #6 Left Arm


//-------------------------Call A Function------------------------------//
void allon() {
  digitalWrite(led1, HIGH);
  digitalWrite(led2, HIGH);
```

```
    digitalWrite(led3, HIGH);
    digitalWrite(led4, HIGH);
    digitalWrite(led5, HIGH);
  }
  void alloff() {
    digitalWrite(led1, LOW);
    digitalWrite(led2, LOW);
    digitalWrite(led3, LOW);
    digitalWrite(led4, LOW);
    digitalWrite(led5, LOW);
  }
  //-------------------------------------------------------------------------//
  void setup() {
    Serial.begin(9600);
    pinMode(led1, OUTPUT);
    pinMode(led2, OUTPUT);
    pinMode(led3, OUTPUT);
    pinMode(led4, OUTPUT);
    pinMode(led5, OUTPUT);
    digitalWrite(led1, LOW);
    digitalWrite(led2, LOW);
    digitalWrite(led3, LOW);
    digitalWrite(led4, LOW);
    digitalWrite(led5, LOW);
  }
  //-------------------------------------------------------------------------//
  void loop() {
    Serial.flush();
    if(Serial.available()) { //Check if there is an available byte to read
      while(1){
        delay(10);
        command = Serial.read(); //Conduct a serial read
        break;
        }

    }
    if(command == '1'){
      allon();
    }
    else if(command == '2'){
      alloff();
    }
    else if(command == '3'){
      if (digitalRead(led1) == LOW) {
          digitalWrite(led1, HIGH);
        }
        else{
          digitalWrite(led1, LOW);
        }
    }
    else if(command == '4'){
      if (digitalRead(led2) == LOW) {
          digitalWrite(led2, HIGH);
        }
        else{
          digitalWrite(led2, LOW);
        }
    }
    else if(command == '5'){
      if (digitalRead(led3) == LOW) {
          digitalWrite(led3, HIGH);
        }
        else{
          digitalWrite(led3, LOW);
        }
    }
    else if(command == '6'){
      if (digitalRead(led4) == LOW) {
          digitalWrite(led4, HIGH);
        }
```

```
         else{
           digitalWrite(led4, LOW);
         }
     }
     else if(command == '7'){
       if (digitalRead(led5) == LOW) {
           digitalWrite(led5, HIGH);
         }
         else{
           digitalWrite(led5, LOW);
         }
     }
       command = 0; //Reset the variable after initiating
   }
```

9.2 Java Code

9.2.1 Main Activity

```
package com.example.martin.speechtotext;
//Authors: Jonas Karlsson, Martin Eriksson, Pontus Olsson

import java.util.ArrayList;
import java.util.Locale;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.graphics.Color;
import android.os.Bundle;
import android.speech.RecognizerIntent;
import android.util.Log;
import android.view.Gravity;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Switch;
import android.widget.Toast;
import android.database.sqlite.SQLiteDatabase;

public class MainActivity extends Activity {

    protected static final int RESULT_SPEECH = 1;

    private ImageButton btnSpeak;

    private Switch head;
    private Switch leftArm;
    private Switch rightArm;
    private Switch leftLeg;
    private Switch rightLeg;

    private TextView txtText;

    private Button blueBtn;

    private BluetoothArduino mBlue;
    private Intent btActivity;
    private SQLiteDatabase db;

    private int lOn = 0;
    private int lOff = 0;
    private int h = 0;
    private int ra = 0;
    private int la = 0;
    private int rl = 0;
    private int ll = 0;
    private int i = 1;

    private boolean headIsChecked = false;
    private boolean rightArmIsChecked = false;
    private boolean leftArmIsChecked = false;
    private boolean rightLegIsChecked = false;
    private boolean leftLegIsChecked = false;

    private String device;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        head = (Switch) findViewById(R.id.headSwitch);
        leftArm = (Switch) findViewById(R.id.leftArmSwitch);
        rightArm = (Switch) findViewById(R.id.rightArmSwitch);
        leftLeg = (Switch) findViewById(R.id.leftLegSwitch);
        rightLeg = (Switch) findViewById(R.id.rightLegSwitch);

        txtText = (TextView) findViewById(R.id.textView);

        blueBtn = (Button) findViewById(R.id.bluetoothButton);

        db = openOrCreateDatabase("SpeechToText.db", Context.MODE_PRIVATE, null);
        db.execSQL("CREATE TABLE IF NOT EXISTS bodyparts(id integer," +
                "bodypart VARCHAR,used integer);");

        btActivity = new Intent(this, Main_Bluetooth.class);

        blueBtn.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                startActivityForResult(btActivity, 2);
            }
        });

        btnSpeak = (ImageButton) findViewById(R.id.btnSpeak);

        btnSpeak.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {

                Intent intent = new Intent(
                        RecognizerIntent.ACTION_RECOGNIZE_SPEECH);

                intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
                        RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
                intent.putExtra(RecognizerIntent.EXTRA_PROMPT, "Choose Command");
                intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE, Locale.ENGLISH);

                try {
                    startActivityForResult(intent, RESULT_SPEECH);
                } catch (ActivityNotFoundException a) {
                    Toast t = Toast.makeText(getApplicationContext(),
                            "Ops! Your device doesn't support Speech to Text",
                            Toast.LENGTH_SHORT);
                    t.show();
                }
            }
        });
    }

    @Override
    public void onSaveInstanceState(Bundle savedInstanceState) {
        super.onSaveInstanceState(savedInstanceState);
        savedInstanceState.putBoolean("Head", head.isChecked());
        savedInstanceState.putBoolean("Right Arm", rightArm.isChecked());
        savedInstanceState.putBoolean("Left Arm", leftArm.isChecked());
        savedInstanceState.putBoolean("Right Leg", rightLeg.isChecked());
        savedInstanceState.putBoolean("Left Leg", leftLeg.isChecked());
    }

    @Override
    protected void onRestoreInstanceState(Bundle savedInstanceState) {
        super.onRestoreInstanceState(savedInstanceState);
        headIsChecked = savedInstanceState.getBoolean("Head");
        rightArmIsChecked = savedInstanceState.getBoolean("Right Arm");
        leftArmIsChecked = savedInstanceState.getBoolean("Left Arm");
        rightLegIsChecked = savedInstanceState.getBoolean("Right Leg");
        leftLegIsChecked = savedInstanceState.getBoolean("Left Leg");
    }

    protected void onResume(){
        super.onResume();
        head.setChecked(headIsChecked);
        rightArm.setChecked(rightArmIsChecked);
        leftArm.setChecked(leftArmIsChecked);
        rightLeg.setChecked(rightLegIsChecked);
        leftLeg.setChecked(leftLegIsChecked);
    }

    public void onStart(){
        super.onStart();
        db.execSQL("INSERT INTO bodyparts VALUES('"+1+
                "', '"+"lights on"+"', '"+lOn+"');");
        db.execSQL("INSERT INTO bodyparts VALUES('"+2+
                "', '"+"lights off"+"', '"+lOff+"');");
        db.execSQL("INSERT INTO bodyparts VALUES('"+3+
                "', '"+"head"+"', '"+h+"');");
        db.execSQL("INSERT INTO bodyparts VALUES('"+4+
                "', '"+"left arm"+"', '"+la+"');");
        db.execSQL("INSERT INTO bodyparts VALUES('"+5+
                "', '"+"right arm"+"', '"+ra+"');");
        db.execSQL("INSERT INTO bodyparts VALUES('"+6+
                "', '"+"left leg"+"', '"+ll+"');");
        db.execSQL("INSERT INTO bodyparts VALUES('"+7+
                "', '"+"right leg"+"', '"+rl+"');");
    }


    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            showDatabase();
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
    public void btDevice(String device){
        txtText.setText("");
        mBlue = new BluetoothArduino(device);

        final ViewGroup layout = (ViewGroup) blueBtn.getParent();
        final TextView btStatus = new TextView(this);
        final Button dcButton = new Button(this);

        if(mBlue.Connect()) {
            if (layout != null) {
                layout.removeView(blueBtn);
            }
            LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(
                    LinearLayout.LayoutParams.WRAP_CONTENT,
                    LinearLayout.LayoutParams.WRAP_CONTENT);
            lp.gravity = Gravity.CENTER_HORIZONTAL;

            layout.addView(btStatus, lp);
            btStatus.setText("Connected to: " + device);
            btStatus.setTextColor(Color.WHITE);

            layout.addView(dcButton, lp);
            dcButton.setText("Disconnect");
        }else{
            txtText.setText("Error in connecting, try again!");
        }
        dcButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (layout != null) {
                    mBlue.Disconnect("FireFly-11AF");
                    layout.removeView(btStatus);
                    layout.removeView(dcButton);
                    layout.addView(blueBtn);
                }
            }
        });
    }

    @Override
    protected synchronized void onActivityResult(int requestCode, int resultCode, Intent data)
{
        super.onActivityResult(requestCode, resultCode, data);

        device = data.getDataString();

        if(requestCode == 2){
            if(resultCode == RESULT_OK) {
                Log.i("Log", "Device recieved: " + device);
                btDevice(device);
            }
        }else {
            switch (requestCode) {
                case RESULT_SPEECH: {
                    if (resultCode == RESULT_OK && null != data) {

                        ArrayList<String> text = data.
                                getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);

                        txtText.setText(text.get(0));

                        if (txtText.getText().equals("lights on")) {
                            i = 1;
                            lOn = lOn + 1;
                            db.execSQL("UPDATE  bodyparts  SET  bodypart='"  +  "lights  on"  +
"',used='" + lOn +
                                    "' WHERE id='" + i + "'");

                            head.setChecked(true);
                            leftArm.setChecked(true);
                            rightArm.setChecked(true);
                            leftLeg.setChecked(true);
                            rightLeg.setChecked(true);

                            headIsChecked = true;
                            rightArmIsChecked = true;
                            leftArmIsChecked = true;
                            rightLegIsChecked = true;
                            leftLegIsChecked = true;

                            mBlue.sendMessage("1");
                        } else if (txtText.getText().equals("lights off")) {
                            i = 2;
                            lOff = lOff + 1;
                            db.execSQL("UPDATE  bodyparts  SET  bodypart='"  +  "lights  off"  +
"',used='" + lOff +
                                    "' WHERE id='" + i + "'");

                            head.setChecked(false);
                            leftArm.setChecked(false);
                            rightArm.setChecked(false);
                            leftLeg.setChecked(false);
                            rightLeg.setChecked(false);

                            headIsChecked = false;
                            rightArmIsChecked = false;
                            leftArmIsChecked = false;
                            rightLegIsChecked = false;
                            leftLegIsChecked = false;

                            mBlue.sendMessage("2");
                        } else if (txtText.getText().equals("head")) {
                            i = 3;
                            if(!head.isChecked()) {
                                h = h + 1;
                                db.execSQL("UPDATE  bodyparts  SET  bodypart='"  +  "head"  +
"',used='" + h +
                                        "' WHERE id='" + i + "'");
                            }
                            head.toggle();
                            if(headIsChecked == false){
                                headIsChecked = true;
                            }else{
                                headIsChecked = false;
                            }

                            mBlue.sendMessage("3");
                        } else if (txtText.getText().equals("left arm")) {
                            i = 4;
                            if(!leftArm.isChecked()) {
                                la = la + 1;
                                db.execSQL("UPDATE  bodyparts  SET  bodypart='"  +  "left  arm"  +
"',used='" + la +
                                        "' WHERE id='" + i + "'");
                            }
                            leftArm.toggle();
                            if(leftArmIsChecked == false){
                                leftArmIsChecked = true;
                            }else{
                                leftArmIsChecked = false;
                            }

                            mBlue.sendMessage("7");
                        } else if (txtText.getText().equals("right arm")) {
                            i = 5;
                            if(!rightArm.isChecked()) {
                                ra = ra + 1;
                                db.execSQL("UPDATE  bodyparts  SET  bodypart='"  +  "right  arm"  +
"',used='" + ra +
                                        "' WHERE id='" + i + "'");
                            }
                            rightArm.toggle();
                            if(rightArmIsChecked == false){
                                rightArmIsChecked = true;
                            }else{
                                rightArmIsChecked = false;
                            }

                            mBlue.sendMessage("4");
                        } else if (txtText.getText().equals("left leg")) {
                            i = 6;
                            if(!leftLeg.isChecked()) {
                                ll = ll + 1;
                                db.execSQL("UPDATE  bodyparts  SET  bodypart='"  +  "left  leg"  +
"',used='" + ll +
                                        "' WHERE id='" + i + "'");
                            }
                            leftLeg.toggle();
                            if(leftLegIsChecked == false){
                                leftLegIsChecked = true;
                            }else{
                                leftLegIsChecked = false;
                            }

                            mBlue.sendMessage("6");
                        } else if (txtText.getText().equals("right leg")) {
                            i = 7;
                            if(!rightLeg.isChecked()) {
                                rl = rl + 1;
                                db.execSQL("UPDATE  bodyparts  SET  bodypart='"  +  "right  leg"  +
"',used='" + rl +
                                        "' WHERE id='" + i + "'");
                            }
                            rightLeg.toggle();
                            if(rightLegIsChecked == false){
                                rightLegIsChecked = true;
                            }else{
                                rightLegIsChecked = false;
                            }

                            mBlue.sendMessage("5");
                        }
                    }
                    break;
                }
            }
        }
    }

    @Override
    protected void onStop(){
        super.onStop();
        try {
            db.delete("bodyparts", null, null);
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        try {
            if(!mBlue.Connect()) {
                mBlue.Connect();
                mBlue.sendMessage("2");
                mBlue.Disconnect("FireFly-11AF");
            }else{
                mBlue.sendMessage("2");
```

```
                mBlue.Disconnect("FireFly-11AF");
            }
            db.delete("bodyparts", null, null);
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }

    public void showDatabase(){
        Cursor c=db.rawQuery("SELECT * FROM bodyparts", null);
        if(c.getCount()==0)
        {
            showMessage("Error", "No records found");
            return;
        }
        StringBuffer buffer=new StringBuffer();
        while(c.moveToNext())
        {
            buffer.append("Id: "+c.getString(0)+"\n");
            buffer.append("Bodypart: "+c.getString(1)+"\n");
            buffer.append("Used: "+c.getString(2)+"\n\n");
        }
        showMessage("Bodypart Details", buffer.toString());
    }

    public void showMessage(String title,String message)
    {
        AlertDialog.Builder builder=new AlertDialog.Builder(this);
        builder.setCancelable(true);
        builder.setTitle(title);
        builder.setMessage(message);
        builder.show();
    }
}
```

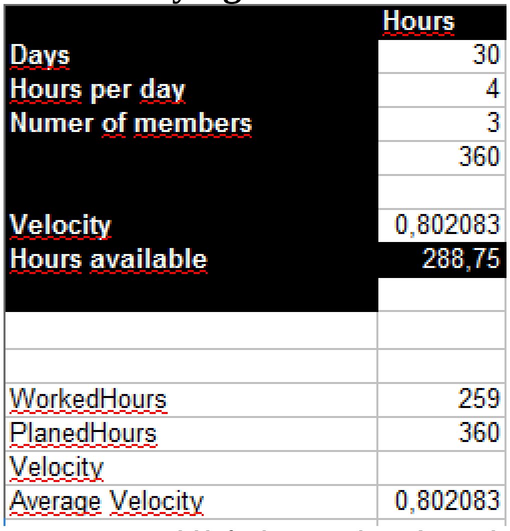
9.2.2 Main Bluetooth

```
package com.example.martin.speechtotext;
//Authors: Jonas Karlsson, Martin Eriksson, Pontus Olsson

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.AdapterView.OnItemClickListener;
import java.util.ArrayList;

public class Main_Bluetooth extends Activity {

    private static final int REQUEST_ENABLE_BT = 1;

    ListView listDevicesFound;
    Button btnScanDevice;
    TextView stateBluetooth;
    BluetoothAdapter bluetoothAdapter;
    ArrayAdapter<String> btArrayAdapter;
    ListItemClicked listItemClicked;
    ArrayList<BluetoothDevice> arrayListBluetoothDevices = null;
    BluetoothDevice bdDevice;
    ArrayList<String> arrayListNameAndAddress = null;
    String deviceString;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main__bluetooth);

        btnScanDevice = (Button)findViewById(R.id.scandevice);

        stateBluetooth = (TextView)findViewById(R.id.bluetoothstate);
        bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

        arrayListBluetoothDevices = new ArrayList<>();
        arrayListNameAndAddress = new ArrayList<>();
        listItemClicked = new ListItemClicked();

        listDevicesFound = (ListView)findViewById(R.id.devicesfound);
        btArrayAdapter           =           new           ArrayAdapter<>(Main_Bluetooth.this,
android.R.layout.simple_list_item_1);
        listDevicesFound.setAdapter(btArrayAdapter);

        btArrayAdapter.notifyDataSetChanged();

        CheckBlueToothState();

        btnScanDevice.setOnClickListener(btnScanDeviceOnClickListener);

        registerReceiver(ActionFoundReceiver,
                new IntentFilter(BluetoothDevice.ACTION_FOUND));
    }

    @Override
    protected void onStart() {
        // TODO Auto-generated method stub
        super.onStart();
        listDevicesFound.setOnItemClickListener(listItemClicked);
    }

    @Override
    protected void onDestroy() {
        // TODO Auto-generated method stub
        super.onDestroy();
        unregisterReceiver(ActionFoundReceiver);
        finish();
    }

    private void CheckBlueToothState(){
        if (bluetoothAdapter == null){
            stateBluetooth.setText("Bluetooth NOT support");
        }else{
            if (bluetoothAdapter.isEnabled()){
                if(bluetoothAdapter.isDiscovering()){
                    stateBluetooth.setText("Bluetooth   is   currently   in   device   discovery
process.");
                }else{
                    stateBluetooth.setText("Bluetooth is Enabled.");
                    btnScanDevice.setEnabled(true);
                }
            }else{
                stateBluetooth.setText("Bluetooth is NOT Enabled!");
                Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
            }
        }
    }

    private Button.OnClickListener btnScanDeviceOnClickListener
            = new Button.OnClickListener(){

        @Override
        public void onClick(View arg0) {
            // TODO Auto-generated method stub
            btArrayAdapter.clear();
            bluetoothAdapter.cancelDiscovery();
            bluetoothAdapter.startDiscovery();
        }};

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        // TODO Auto-generated method stub
        if(requestCode == REQUEST_ENABLE_BT){
            CheckBlueToothState();
        }
    }

    private final BroadcastReceiver ActionFoundReceiver = new BroadcastReceiver(){

        @Override
        public void onReceive(Context context, Intent intent) {
            // TODO Auto-generated method stub
            String action = intent.getAction();
            if(BluetoothDevice.ACTION_FOUND.equals(action)) {
                BluetoothDevice                            device                              =
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
                btArrayAdapter.add(device.getName() + "\n" + device.getAddress());
                arrayListBluetoothDevices.add(device);
                arrayListNameAndAddress.add(device.getName() + " " + device.getAddress());
                btArrayAdapter.notifyDataSetChanged();

            }
        }};

    class ListItemClicked implements OnItemClickListener
    {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            // TODO Auto-generated method stub
            bdDevice = arrayListBluetoothDevices.get(position);
            deviceString = arrayListNameAndAddress.get(position);

            String[] nameAndAddress = deviceString.split(" ");

            bdDevice.createBond();

            Log.i("Log",  "The  device  :  "  +  nameAndAddress[0]  +  "  ["  +  nameAndAddress[1]  +
"]");
            Intent _result = new Intent();
            _result.setData(Uri.parse(nameAndAddress[0]));
            setResult(RESULT_OK, _result);
            finish();
        }
    }
}
```

9.2.3 Bluetooth Arduino

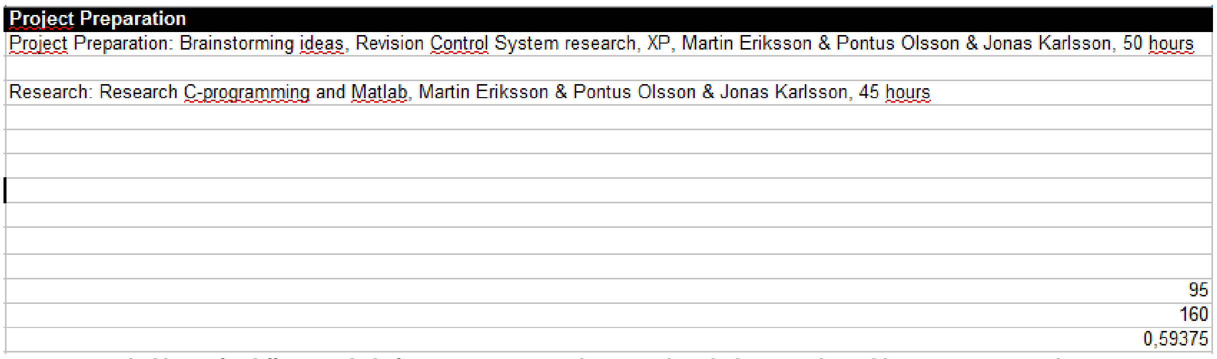
```
package com.example.martin.speechtotext;
//Authors: Jonas Karlsson, Martin Eriksson, Pontus Olsson

import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.util.Log;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.List;
import java.util.Set;
import java.util.UUID;

public class BluetoothArduino extends Thread {
    private BluetoothAdapter mBlueAdapter = null;
    private BluetoothSocket mBlueSocket = null;
    private BluetoothDevice mBlueRobo = null;

    OutputStream mOut;
    InputStream mIn;
    private boolean robotFound = false;
    private boolean connected = false;
    private String robotName;
    private List<String> mMessages = new ArrayList<String>();
    private String TAG = "BluetoothConnector";

    public BluetoothArduino(String Name){
        try {
            for(int i = 0; i < 2048; i++){
                mMessages.add("");
            }
            robotName = Name;
            mBlueAdapter = BluetoothAdapter.getDefaultAdapter();
            if (mBlueAdapter == null) {
                LogError("\t\t[#]Phone does not support bluetooth!!");
                return;
            }
            if (!isBluetoothEnabled()) {
              LogError("[#]Bluetooth is not activated!!");
            }

            Set<BluetoothDevice> paired = mBlueAdapter.getBondedDevices();
            if (paired.size() > 0) {
                for (BluetoothDevice d : paired) {
                    if (d.getName().equals(robotName)) {
                        mBlueRobo = d;
                        robotFound = true;
                        break;
                    }
                }
            }

            if (!robotFound)
                LogError("\t\t[#]There is no device paired!!");

        }catch (Exception e){
            LogError("\t\t[#]Error creating Bluetooth! : " + e.getMessage());
        }

    }

    public boolean isBluetoothEnabled(){
      return mBlueAdapter.isEnabled();
    }

    public boolean Connect(){
        if(!robotFound)
            return false;
        try{
            LogMessage("\t\tConnecting to the robot...");

            UUID uuid = UUID.fromString("00001101-0000-1000-8000-00805f9b34fb");
            mBlueSocket = mBlueRobo.createRfcommSocketToServiceRecord(uuid);
            if(!mBlueSocket.isConnected()) {
                Thread.sleep(300);
                mBlueSocket.connect();
            }
            mOut = mBlueSocket.getOutputStream();
            mIn = mBlueSocket.getInputStream();
            connected = true;
            this.start();
            LogMessage("\t\t\t" + mBlueAdapter.getName());
            LogMessage("\t\tOk!!");
            return true;

        }catch (Exception e){
            LogError("\t\t[#]Error while connecting: " + e.getMessage());
            return false;
        }
    }

    public void sendMessage(String msg){
        try {
            if(connected) {
                mOut.write(msg.getBytes());
            }
        } catch (IOException e){
            LogError("->[#]Error while sending message: " + e.getMessage());
        }
    }

    private void LogMessage(String msg){
      Log.d(TAG, msg);
    }

    private void LogError(String msg){
      Log.e(TAG, msg);
    }

    public void Disconnect(String name){
        try {
            mIn.close();
            mOut.close();
            mBlueSocket.close();
            Log.i("DC", "Disconnected from: " + name);
        }catch (Exception e){
            Log.i("DC", "Couldn't disconnect from: " + name);
        }
    }

}
```

9.3 Velocity figures



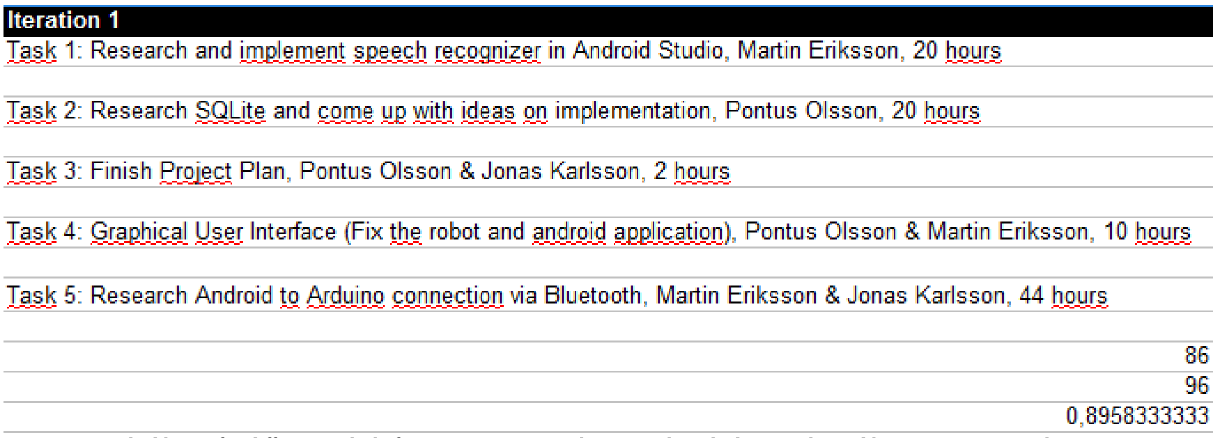
	Hours
Days	30
Hours per day	4
Numer of members	3
	360
Velocity	0,802083
Hours available	288,75
WorkedHours	259
PlanedHours	360
Velocity	
Average Velocity	0,802083

Figure 1: Days available for the project, hours/day, number of members, average velocity for the project and hours available.



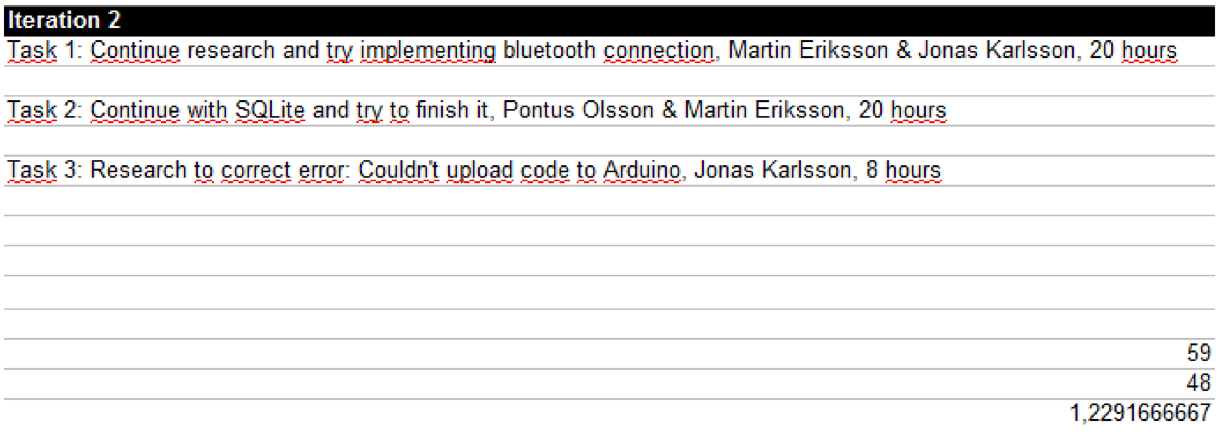
Project Preparation
Project Preparation: Brainstorming ideas, Revision Control System research, XP, Martin Eriksson & Pontus Olsson & Jonas Karlsson, 50 hours
Research: Research C-programming and Matlab, Martin Eriksson & Pontus Olsson & Jonas Karlsson, 45 hours
95
160
0,59375

Figure 2: Worked hours for different tasks before iteration 1, 95 = hours total worked, 160 = planned hours, 0,59375 = velocity.



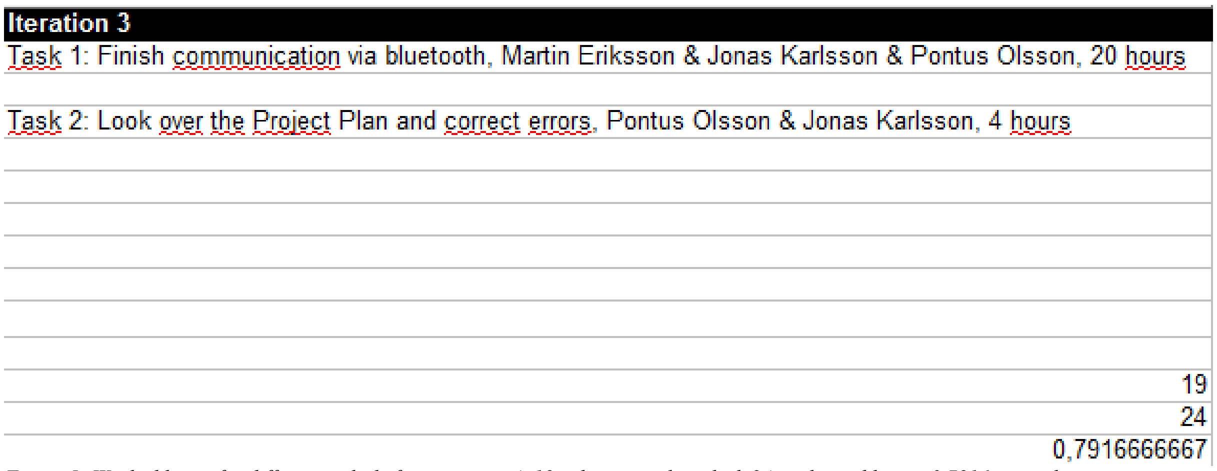
Iteration 1
Task 1: Research and implement speech recognizer in Android Studio, Martin Eriksson, 20 hours
Task 2: Research SQLite and come up with ideas on implementation, Pontus Olsson, 20 hours
Task 3: Finish Project Plan, Pontus Olsson & Jonas Karlsson, 2 hours
Task 4: Graphical User Interface (Fix the robot and android application), Pontus Olsson & Martin Eriksson, 10 hours
Task 5: Research Android to Arduino connection via Bluetooth, Martin Eriksson & Jonas Karlsson, 44 hours
86
96
0,8958333333

Figure 3: Worked hours for different tasks before iteration 2, 86 = hours total worked, 96 = planned hours, 0,89583 = velocity.



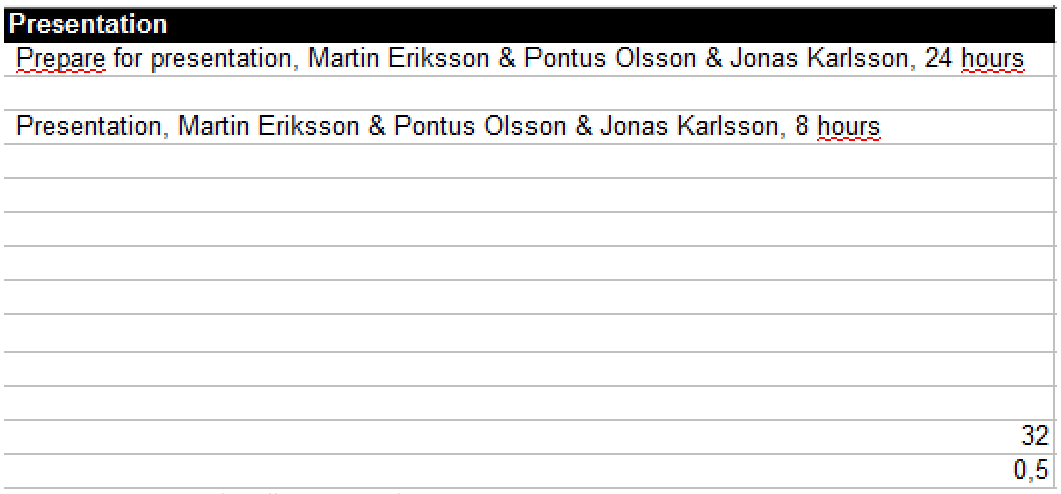
Iteration 2
Task 1: Continue research and try implementing bluetooth connection, Martin Eriksson & Jonas Karlsson, 20 hours
Task 2: Continue with SQLite and try to finish it, Pontus Olsson & Martin Eriksson, 20 hours
Task 3: Research to correct error: Couldn't upload code to Arduino, Jonas Karlsson, 8 hours
59
48
1,2291666667

Figure 4: Worked hours for different tasks before iteration 3, 59 = hours total worked, 48 = planned hours, 1,2291… = velocity.



Iteration 3
Task 1: Finish communication via bluetooth, Martin Eriksson & Jonas Karlsson & Pontus Olsson, 20 hours
Task 2: Look over the Project Plan and correct errors, Pontus Olsson & Jonas Karlsson, 4 hours
19
24
0,7916666667

Figure 5: Worked hours for different tasks before iteration 4, 19 = hours total worked, 24 = planned hours, 0,7916… = velocity.



Presentation
Prepare for presentation, Martin Eriksson & Pontus Olsson & Jonas Karlsson, 24 hours
Presentation, Martin Eriksson & Pontus Olsson & Jonas Karlsson, 8 hours
32
0,5

Figure 6: Worked hours for different tasks before the presentation, 32 = planned hours, 0,5 = velocity (symbolic value).