Twitduino an Arduino RFID Twitter implementation for people with communication disabilities

René Alejandro Lobo Quintero

Abstract—in this paper the creation of a hardware-software interface that creates a new communication form for people with disabilities. The idea of the project is to design and implement a pervasive system that using Arduino and RFID will allow people with disabilities to send pre-loaded tweets just by approximating a card with an rfid-tag to the receiver , this tweets can contain any kind of message for example: asking for help, make request, communicate moods and interests and so on.
In the future the system could be connected with other services or devices creating a large set of possibilities

***Index Terms*—Arduino, RFID, Pervasive Systems, Twitter**

I. INTRODUCTION

NOWADAYS technology is taking a huge protagonism in the society for example helping the communication between people who are geographically separated using several new kinds of techniques. Some of them can be used to help people with communication or language disabilities, facilitating them this process using software and hardware specially designed for them in order to be easy to use, fast, reliable and pervasive.

The Arduino Hardware Platform is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators.

In our case we will use the Arduino to recognize inputs from the user using the RFID technology in order to send messages over the internet with the minimal effort for the user

The paper will be divide into a problem explanation, a hardware design process and a software design process finishing with some conclusions and future work

II. PROBLEM

People with communication disabilities need easy, reliable and cheap ways to interact with the technology and with the people surrounding them. Normally they have to make several actions or use complicated software tools just to send one message to report their status ask for help or tell how they feel. The use of new technologies and the integration of software and hardware into pervasive systems can help them to overcome this problem.

The idea of sending a custom message just with one action by using special but low cost hardware can be a great way to attack this problem allowing the user to communicate simple ideas with a very small effort and will be described in the following sections.

III. SYSTEM DESCRIPTION

The purpose of the application is done by integrating three technologies: RFID, Arduino and Twitter.

The key idea is to send a predesigned message by twitter just by approaching a RFID-Tag to the correspondent RFID sensor.

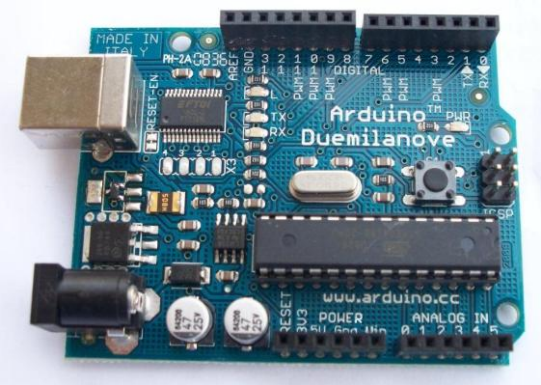
This implementation works by a reader sending out a signal that powers a “tag”, the RFID tag then responds with a unique 12 digit hexadecimal serial number. Because each tag sends back a unique number, it can be used to assign a messages to each tag.

the following sections will illustrate the hardware and software development created for this application

IV. HARDWARE DEVELOPMENT

In order to develop the solution the following hardware was used:

Arduino duemilanove:



The Arduino Duemilanove[2] is a micro-controller board based on the ATmega328. It has 14 digital input / output pins (which 6 can be used as PWM2 outputs), 6 analog inputs, a 16Mhz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

The Arduino Duemilanove can be powered via the USB connection or with an external power supply. The power source is selected automatically. Each of the 14 digital pins on the Duemilanove can be used as input or output using the functions: pinMode(), digitalWrite(), digitalRead()

RFID ID-2 Module

The RFID ID-2 Module is a very simple to use RFID reader module produced by ID Innovations and sold by sparkfun web site[1] with the following features

- 5V supply
- 125kHz read frequency
- EM4001 64-bit RFID tag compatible
- 9600bps TTL and RS232 output
- Magnetic stripe emulation output
- 100mm read range

The module has a limitation in the distance between the module and the RFID-Tag but is not a problem due to the nature of the application where the user will be located near the module.

Also the ID-2 module doesn't have an internal antenna that is needed to recognize the RFID-Tag but has the connection pins prepared to attach any antenna that satisfy the requirements of the recommended Inductance that is 1.08mH to be used with an internal tuning capacitor of 1n5

KCA Keylock Antenna B-1080J

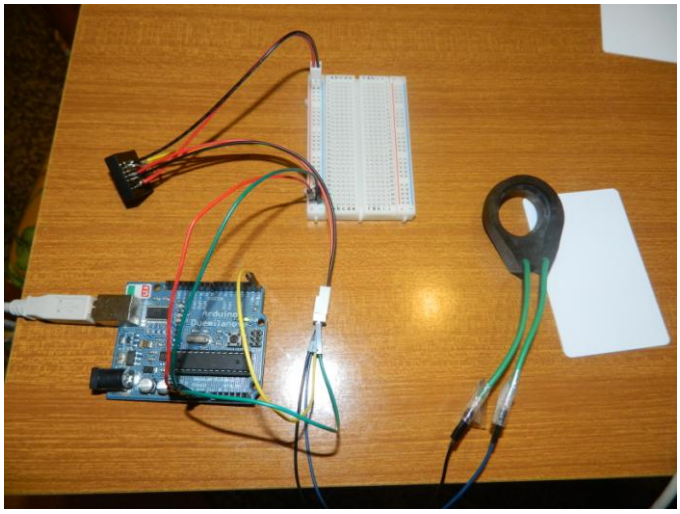
This antenna is produced by Premo Group and designed to emit a RF signal and identify the correspondent RFID-Tag associated and it is build with top materials and requirements[3]

characteristics:

- High stability in temperature(-40°C to +85°C)
- 1.08mH inductance
- LF transponder Emitter/transmitter antenna

its attached to the ID-2 module by two simple wires that can be as long as its desired so it could be used for a very range of application

This components are assembled following the instructions in the datasheets obtaining the final hardware platform:



V. SOFTWARE DEVELOPMENT

The project was made using the following software

Arduino Software, is an open-source environment that makes easy to write code and upload it to the i/o board. It runs on Windows, Mac OS X, and Linux.
The environment is written in Java and based on Processing, avrgcc, and other open source software. [4]

Processing Software: is an open source programming
language and environment for people who want to create images, animations, and interactions. It runs on Windows, Mac OS X, and Linux and allows interaction with the Arduino microcontrollers. [5]

Twitter4J Library: Twitter4J is an unofficial Java library for the Twitter API . that allows to read and post to twitter in a easy and secure way Twitter4J, can be easily integrated with Java/Processing applications. [6]

Design

The Arduino software reads the signal registered by the antenna using the RX input PIN using the Serial.read() function. This signal contains the 12 digits that identify the RFID-Tag in hexadecimal coding and checks for the stop bytes to stop reading the signal.

After an Ascii/Hex conversion is performed in order to transform the signal into a easier to transfer representation

Finally the signal is prepared to be sent. The program created with the Arduino software is uploaded to the Arduino Hardware. Each time it is connected will run it without any kind of intervention.

As the Arduino Platform can't be communicated with the internet in a easy way we have to use the Processing software which reads the string sent by the Arduino for the serial port using the following instructions:

```
Serial myPort;
myPort = new Serial(this, Serial.list()[0], 9600);
myPort.bufferUntil(lf);
void serialEvent(Serial p) {
  inString = p.readString();
}
```

and checks for the identification of the RFID-Tag, if it matches the preloaded one a tweet will be sent with a preloaded messages for each one of the tags.

The twitter Library Twitter4J creates a link between the processing application and the API of twitter, first performing the authentification via the OAuth protocol, to make this possible a developer application must be created first in the twitter developers page https://dev.twitter.com/ to obtain the needed credentials to interact with the API: consumer_key, consumer_secret, oauth_token, oauth_token_secret. Those credentials need to be placed inside the source code

the posting process is made like this:

```
Twitter twitter = new
TwitterFactory().getOAuthAuthorizedInstance
(consumer_key, consumer_secret, new
AccessToken( oauth_token, oauth_token_secret) );
try
{
    Status st = twitter.updateStatus("wiii twiteando
desde arduino rfid tag 1" + " " + second());
}
catch (TwitterException e) {
    println(e.getStatusCode());
}
```

if the process is successful the result is the following:

the number of messages and its content must be preloaded in the source code

VI. Conclusion

The Created platform is capable of sending simple preloaded messages using the twitter API with minimal effort for the user

The use of the Processing Software with the Arduino Platform creates a great set of possibilities for developing interesting and interactive projects

The Use of RFID-Tags is an interesting and powerful identification method that can be used for other kind of applications designed for people with disabilities

VII. Future Work

Even the project is just a prototype the objective proposed is achieved with no faults, but some improvements could be done in order to facilitate the interaction between the user, the program could have the possibility to change the preloaded messages with a graphical interface, also the systems could have a more visible way to indicate that the message has been post, maybe using leds or some sound, finally the system should have more solid connections to avoid the disconnections of some of its components

REFERENCES

[1] RFID module in the Sparkfun website https://www.sparkfun.com/products/8419?

[2] ARDUINO duemilanove *http://arduino.cc/en/Main/arduinoBoardDuemilanove*

[3] KCA-B-1080J http://www.alldatasheet.com/datasheet-pdf/pdf/295662/PREMO/KCA-B-1080J.html

[4] http://arduino.cc/en/Main/Software

[5] http://processing.org/download/

[6] http://twitter4j.org/en/index.html

[7] http://playground.arduino.cc/Code/ID12

Source Code
ARDUINO

```
void setup() {
 Serial.begin(9600);                    // connect to
the serial port
}

void loop () {
 byte i = 0;
 byte val = 0;
 byte code[6];
 byte checksum = 0;
 byte bytesread = 0;
 byte tempbyte = 0;

 if(Serial.available() > 0) {
   if((val = Serial.read()) == 2) {              // check
for header
     bytesread = 0;
     while (bytesread < 12) {                    // read 10
digit code + 2 digit checksum
       if( Serial.available() > 0) {
         val = Serial.read();
         if((val == 0x0D)||(val == 0x0A)||(val ==
0x03)||(val == 0x02)) { // if header or stop bytes
before the 10 digit reading
           break;                       // stop reading
         }

         // Do Ascii/Hex conversion:
         if ((val >= '0') && (val <= '9')) {
           val = val - '0';
         } else if ((val >= 'A') && (val <= 'F')) {
           val = 10 + val - 'A';
         }

         // Every two hex-digits, add byte to code:
        if (bytesread & 1 == 1) {
           // make some space for this hex-digit by
           // shifting the previous hex-digit with 4 bits to
the left:
           code[bytesread >> 1] = (val | (tempbyte <<
4));
         }
         else {
           tempbyte = val;                    // Store the
first hex digit first...
         };

         bytesread++;                        // ready to
read next digit
       }
     }

     // Output to Serial:

     if (bytesread == 12) {                  // if 12 digit
read is complete
     //  Serial.print("5-byte code: ");
       for (i=0; i<5; i++) {
         if (code[i] < 16) Serial.print("0");
         Serial.print(code[i], HEX);
         Serial.print(" ");
       }
       Serial.println();


     }

     bytesread = 0;
   }
 }
}
```

PROCESSING

```
import twitter4j.conf.*;
import twitter4j.internal.async.*;
import twitter4j.internal.org.json.*;
import twitter4j.internal.logging.*;
import twitter4j.http.*;
import twitter4j.api.*;
import twitter4j.util.*;
import twitter4j.internal.http.*;
import twitter4j.*;

import processing.serial.*;
Serial myPort;    // The serial port
PFont myFont;     // The display font
String inString=" ";  // Input string from serial port
int lf = 10;      // ASCII linefeed

String msg = "twitter4j desde processing";

//copy and paste these from your application in dev.twitter.com
String consumer_key = "vzFv3kREeRnro3xGW4kEXg";
String consumer_secret = "Qs96RD6R1bOf5RQHPuysyNsu4Ea0dAR61HHdsO0Ns";
String oauth_token = "22927286-6u34lNmrxEv4hCAWXMoBAPVuG92BBxl56LY4yIM";
String oauth_token_secret = "NJVOvJTcVtu2i88JxrZO8mQmZUMPIPZ1BQbV4sk";
color bgcolor = color(255);
long timer;

void setup() {
  size(400, 200);
  background(0);
  // List all the available serial ports:
  println(Serial.list());
  // I know that the first port in the serial list on my computer
  //is the one being used
  myPort = new Serial(this, Serial.list()[0], 9600);
  myPort.bufferUntil(lf);
}
void serialEvent(Serial p) {
  inString = p.readString();
}
void draw() {
  if (inString.contains("67 00 73 CA 0D ")) {
      background(255,204,0);
    println(inString);
    text(inString, 30, 30);
     text("RFID-Tag Detected", 30, 30);
   delay(1000);
    text(inString, 30, 30);
      Twitter twitter = new TwitterFactory().getOAuthAuthorizedInstance (consumer_key, consumer_secret, new AccessToken( oauth_token, oauth_token_secret) );
    try {
     Status st = twitter.updateStatus("wiii twiteando desde arduino rfid tag 1" + " " + second());
      println("Successfully updated the status to [" + st.getText() + "].");

      timer = millis();
    }
    catch (TwitterException e) {
      println(e.getStatusCode());
    }
    inString=" ";
    delay(1000);
    }
  if (inString.contains("67 00 73 F7 52 ")) {

    text(inString, 30, 30);
    text("RFID-Tag Detected", 30, 30);
     background(0,255,0);
    println(inString);
    delay(1000);
    text(inString, 30, 30);

     Twitter twitter = new TwitterFactory().getOAuthAuthorizedInstance (consumer_key, consumer_secret, new AccessToken( oauth_token, oauth_token_secret) );
    try {
     Status st = twitter.updateStatus("yeahhh twiteando desde arduino rfid tag 2" + " " + second());
      println("Successfully updated the status to [" + st.getText() + "].");
     timer = millis();
    }
    catch (TwitterException e) {
      println(e.getStatusCode());
    }
    inString=" ";
    delay(1000);

  }
  else {
    }
```