

Sistema para visualización en la nube de la polución de la ciudad a través de Drones

Ing. Andres Julian Moreno Moreno

Universidad Distrital Francisco José de Caldas

6 de julio de 2016

Proyecto Final - Programación de Aplicaciones para Internet y la Nube



Índice

- 1 Descripción del Proyecto
- 2 Sensor de Polución CO
- 3 Hardware, programación y montaje
- 4 Drone o Vehículo Aéreo No Tripulado VANT
- 5 Sistema de comunicaciones Digimesh
- 6 Estación Terrena y Processing
- 7 Plataforma en la nube
- 8 Análisis Dinámico de Datos
- 9 Conclusiones

Proyecto Final

Descripción del Proyecto

Objetivo General

Visualizar a través de un servicio web, datos tomados en "tiempo real" de polución desde un Drone, basado en un sistema de comunicaciones wireless y conectado a una estación terrena.

Arquitectura del sistema



Figura: Diagrama general de la solución planteada

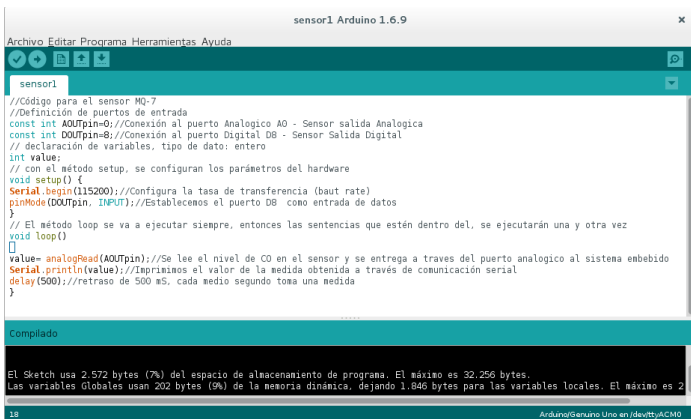
Sensor de Polución Monóxido de Carbono CO

De acuerdo a los niveles actuales de contaminación y acumulación de gases de combustibles para el desarrollo del proyecto se usa un sensor de monóxido de carbono, referencia MQ-7.



Figura: Sensor CO

Arduino UNO, programación y montaje



```
sensor1 Arduino 1.6.9
Archivo Editar Programa Herramientas Ayuda
sensor1
//Código para el sensor MQ-7
//Definición de puertos de entrada
const int AOUTpin=0;//Conexión al puerto Analógico A0 - Sensor salida Analógica
const int DOUTpin=8;//Conexión al puerto Digital D8 - Sensor Salida Digital
// declaración de variables, tipo de dato: entero
int value;
// con el método setup, se configuran los parámetros del hardware
void setup() {
  Serial.begin(115200);//Configura la tasa de transferencia (baud rate)
  pinMode(DOUTpin, INPUT);//Establecemos el puerto D8 como entrada de datos
}
// El método loop se va a ejecutar siempre, entonces las sentencias que estén dentro del, se ejecutarán una y otra vez
void loop()
{
  value= analogRead(AOUTpin);//Se lee el nivel de CO en el sensor y se entrega a través del puerto analógico al sistema embebido
  Serial.println(value);//Imprimimos el valor de la medida obtenida a través de comunicación serial
  delay(500);//retraso de 500 ms, cada medio segundo toma una medida
}

Compilado

El Sketch usa 2.572 bytes (7%) del espacio de almacenamiento de programa. El máximo es 32.256 bytes.
Las variables Globales usan 202 bytes (9%) de la memoria dinámica, dejando 1.846 bytes para las variables locales. El máximo es 2

18 Arduino/Genuino Uno en /dev/ttyACM0
```

Figura: IDE software de programación Arduino

Ensamble de Sensor, Arduino, y modulo Xbee

Foto real del nodo Sensor

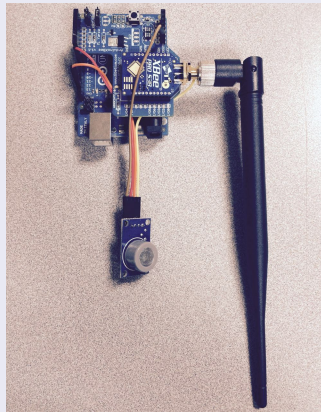


Figura: Montaje Arduino, Sensor y Xbee

Drone o Vehículo Aereo No Tripulado VANT



Figura: Iris 3DRobotics

Sistema de comunicaciones Digimesh

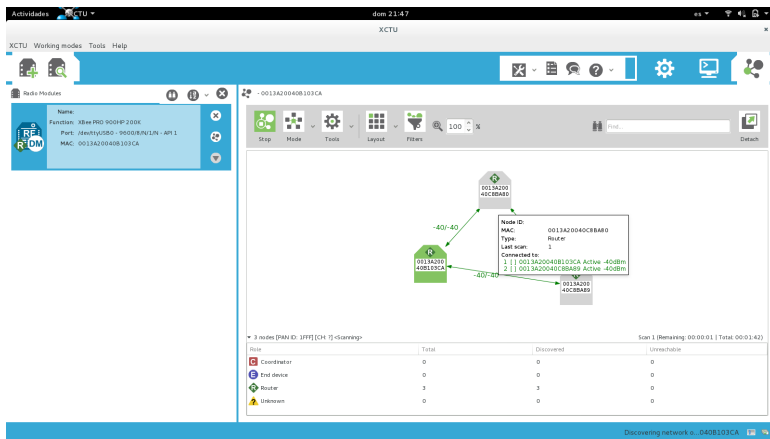
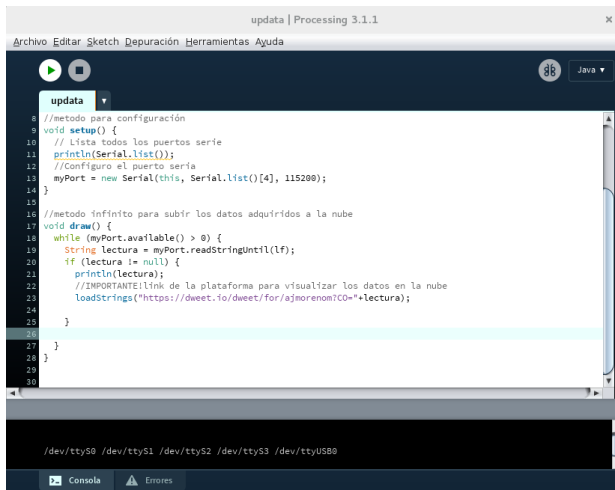


Figura: Software de Digi, configuración XCTU

Estación Terrena y Processing



The screenshot shows the Processing 3.1.1 IDE interface. The title bar reads 'update | Processing 3.1.1'. The menu bar includes 'Archivo', 'Editar', 'Sketch', 'Depuración', 'Herramientas', and 'Ayuda'. The toolbar contains icons for running, stopping, and saving, along with a language dropdown set to 'Java'. The main text area displays the following code:

```
update
8 //metodo para configuración
9 void setup() {
10   // Lista todos los puertos serie
11   println(Serial.list());
12   //Configuro el puerto serie
13   myPort = new Serial(this, Serial.list()[4], 115200);
14 }
15
16 //metodo infinito para subir los datos adquiridos a la nube
17 void draw() {
18   while (myPort.available() > 0) {
19     String lectura = myPort.readStringUntil(lf);
20     if (lectura != null) {
21       println(lectura);
22       //IMPORTANTE! link de la plataforma para visualizar los datos en la nube
23       loadStrings("https://dweet.io/dweet/for/ajmorenom?CO="+lectura);
24     }
25   }
26 }
27
28 }
29
30
```

At the bottom, the console area shows the list of available serial ports: `/dev/ttyS0 /dev/ttyS1 /dev/ttyS2 /dev/ttyS3 /dev/ttyUSB0`. The status bar at the very bottom indicates 'Consola' and 'Errores'.

Figura: Software Processing en ET

Foto real de estación terrena funcionando



Figura: Estación terrena Xbee + Debian + Processing

Plataforma en la nube

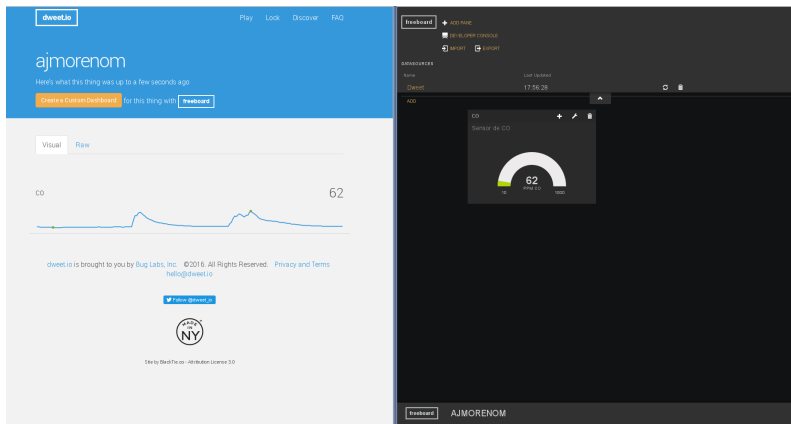


Figura: Resultados visualizados en Internet

Get data

```
library(jsonlite)

## Loading required package: methods

urldew<-"https://dweet.io:443/get/dweets/for/ajmorenom"
urldew<- paste(urldew, sep = "")
dwet<- fromJSON(urldew)
```

Show data

```
dwet

## $this
## [1] "succeeded"
##
## $by
## [1] "getting"
##
## $the
## [1] "dweets"
##
## $with
##           thing                      created CO
## 1 ajmorenom 2016-07-06T14:29:41.996Z 38
## 2 ajmorenom 2016-07-06T14:29:40.986Z 38
## 3 ajmorenom 2016-07-06T14:29:38.993Z 38
## 4 ajmorenom 2016-07-06T14:29:37.993Z 38
```

EDA

```
datos<-dwet$with$content
str(datos)

## 'data.frame': 5 obs. of 1 variable:
## $ CO: int 38 38 38 38 38

summary(datos)

##          CO
## Min.      :38
## 1st Qu.:38
## Median :38
## Mean    :38
## 3rd Qu.:38
## Max.    :38
```

Conclusiones

Se logra realizar el monitoreo en tiempo real del smoke, a través de un drone y una plataforma en la nube, se visualizan las medidas de CO en ppm.

Se logra configurar un sistema integrado capaz de aumentar el alcance del Drone, y entrega las medidas a la estación terrena.

Se logra establecer comunicación desde la estación terrena con el Drone a través de un Drone repetidor de comunicaciones, esto basado en el sistema propuesto basado en topología mesh

El uso del drone facilita realizar este tipo de mediciones.