Facultad de Ingeniería
Escuela de Informática y Telecomunicaciones

Proyecto TICS – Garduino

Documento: Descripción del producto

Profesor: Jorge Eliott - Integrantes: Diego Farías, Benjamín Morales, Cristóbal Urra

Índice

Descripción del producto

El producto consiste en un invernadero con las siguientes funciones automatizadas:

- Control de riego
- Iluminación
- Ventilación
- Monitoreo del Ambiente

Las susodichas funciones se encuentran automatizadas mediante la programación de una placa Arduino UNO, el objetivo del producto desde simular un ambiente adecuado para plantas cuyas posibilidades de crecimiento y supervivencia se encuentran reducida en ambientes inadecuados u hostiles, a facilitar el cuidado de la planta mediante sus funciones automatizadas para usuarios comunes y en el campo de botánica.

Mientras el invernadero se encuentra en funcionamiento los datos captados por los sensores para el control del ambiente sometido en la planta, son enviados por internet a un servidor del servicio para realizar un monitoreo para su revisión en caso de algún problema o cambio.

Para efectuar el cuidado y la simulación la placa Arduino deberá hacer uso de los siguientes componentes:

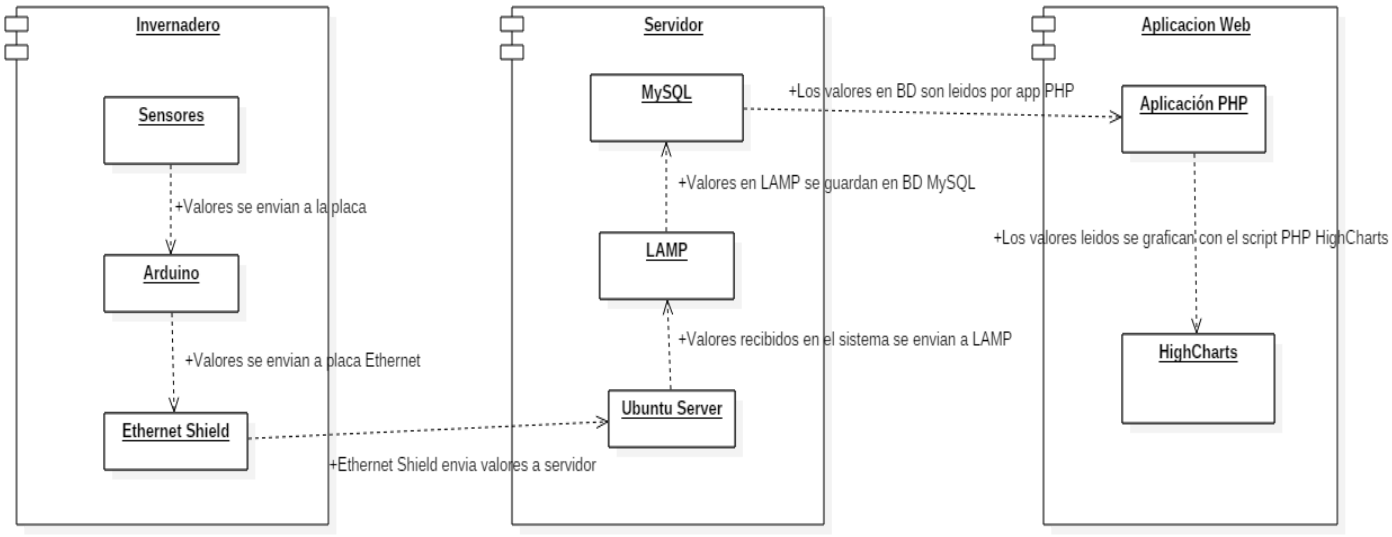
- Relés.
- Sensor de Humedad y Temperatura (Módulo DHT11).
- Sensor de Humedad de suelo.
- Luminaria.
- Ventilador.
- Bomba de agua.
- Transformador.
- Manguera.
- Armazón/estructura.
- Ethernet Shield

Estos componentes se incorporan de forma fija a la zona interior de un armazón hecho de madera con forma de prisma rectangular que cubre la planta (anteriormente colocada en una maceta adecuada), cuyas dimensiones dependen del tamaño de la planta a colocar.

El cableado junto a la placa Arduino, se encontrarán fijados en la zona trasera por el lado exterior del armazón, posición que deja el acceso para la conexión a la corriente y a internet.

El artefacto debe ser puesto en una superficie plana con acceso a corriente eléctrica e internet (entrada de enchufe e ethernet); una vez instalado y en operación, mediante el uso de un reloj incorporado en la placa Arduino, por cada segundo tomará datos captados por los sensores (Humedad de ambiente, Humedad de tierra y temperatura), y cada cierto intervalo de tiempo, estos se enviarán vía internet a un servidor que captará los datos junto a la hora el cual fueron registrados, el cual son guardados e incorporados en una gráfica para un análisis realizado por el equipo mediante el monitoreo del artefacto.

Diagrama de componentes



Resultados de pruebas

Los resultados consisten en pruebas realizadas a cada dispositivo que toma valores se ǵ un lo que evalúa (sensores), exponiendolos a distintos escenarios, esperando que entregue de forma práctica un resultado que se espera a un análisis teórico realizado previamente.

-Sensor Humedad de tierra:

Este sensor fue realizado por el equipo mediante la fabricación de dos mallas de cobre, escogimos este material debido a su adecuada conductividad. Para calibrar, se utilizó dos macetas idénticas, se llenaron con la misma tierra y en una maceta incorpora las mallas de cobre, una en una pared contraria a la otra en la maceta, y en otra se incorporaron dos sensores de humedad de tierra digitales (TDR), uno en la parte superior de la maceta y otro en la parte inferior, sacando un promedio de valores según se ingresa agua.

ML	TDR	Cobre
AIRE	1023	1023
0	892	751
50	848	617
100	835	566
150	797	372
200	667	234
250	501	135
300	467	104
350	455	76
400	431	56
450	428	49
500	420	48

La presente gráfica presenta los datos tomados en cada maceta se ǵ un la cantidad de agua que fue introducida hasta quedar la maceta rebalsada, el comportamiento

analizados muestra una reducción del valor a medida que la humedad generada por el agua presente en la tierra aumentaba.

Posteriormente estos datos fueron gráficas y se les aplicó una traslación de función a la gráfica de las mallas de cobre para que se asimilara a los fidedignos datos del TDR.

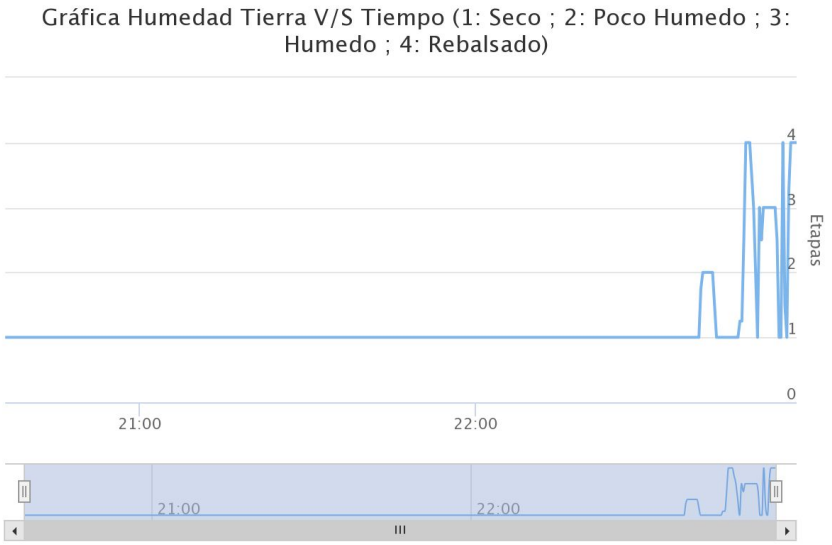
Por último, se aplicaron puntos críticos basado en lo analizado para establecer etapas que el usuario pueda interpretar:

1 : Seco

2: Poco Húmedo

3: Humedo

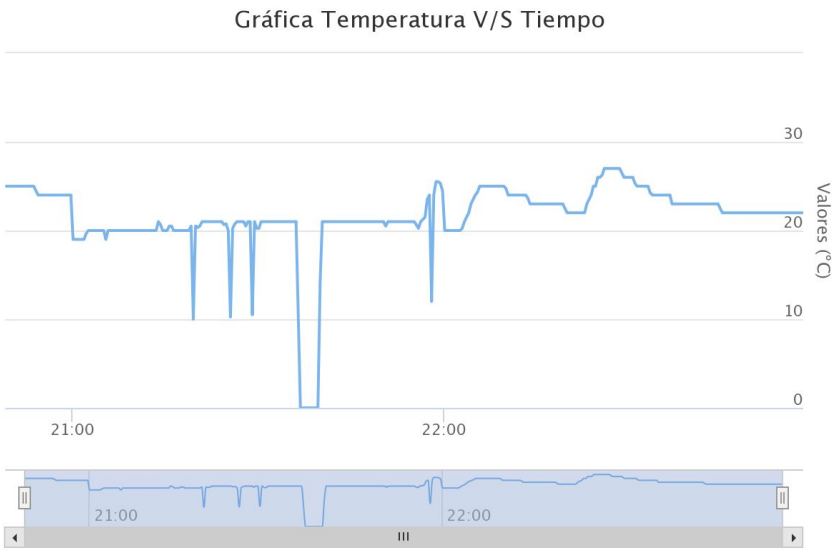
4: Rebalsado



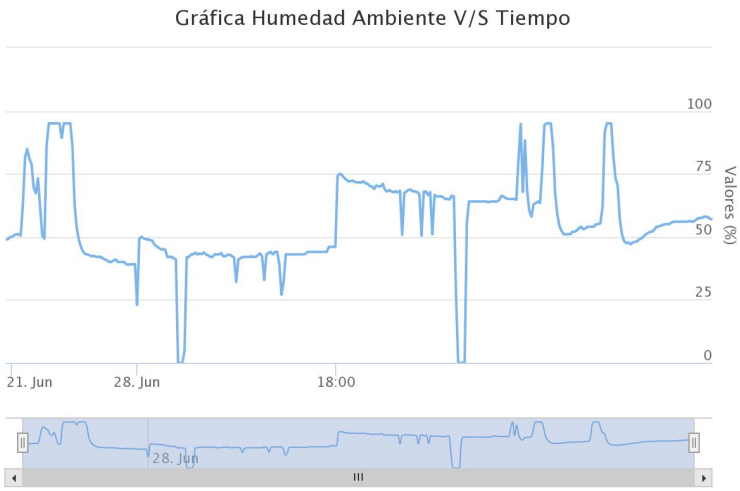
Esta gráfica muestra los datos tomados en tiempo real por las mallas de cobre ya calibradas con los TDR de referencia y los puntos críticos y etapas aplicadas. Pudimos comprobar su funcionamiento pues a medida que ingresamos más líquido o sustancias que provoquen una mayor conductividad, ocurría lo esperado, la gráfica mostraba etapas que indican mayor humedad, mientras que si se deja en el aire el gráfico se mantiene estático en la etapa inicial (seco).

- Sensor Humedad Ambiente y Temperatura (DHT11)

Este sensor digital de por sí garantiza un correcto funcionamiento en la muestra de valores captados, las pruebas realizadas fueron enfocadas a revisar su comportamiento en los gráficos en tiempo real configurados por el equipo.



Esta gráfica muestra la temperatura v/s tiempo captada en tiempo real, como probamos su buen funcionamiento pues al dejar el sensor intacto la gráfica se mantenía en un valor constante, al exponerlo a calor este aumentaba el valor y al desconectarlo o exponerlo a temperaturas bajas , la gráfica disminuía rápidamente.



De igual manera con el gráfico de temperatura, el presente gráfico de Humedad ambiente V/S tiempo, aumentaba el valor o se reducía al exponer el sensor a escenarios en los que se esperaba que estos cambiaran su valor a aumentar o disminuir respectivamente, comprobando así el correcto funcionamiento del sensor y los gráficos de tiempo real.

Reflexión

Nosotros como grupo buscamos llegar a una herramienta automatizada, la cual pasó por muchas ideas antes de tener nuestro producto definitivo, la implementación para todo el artefacto fue puesta a prueba en distintas circunstancias, y con lo obtenido pudimos notar que las limitaciones para el proyecto no iban más que en nosotros, ya que las funciones y distintos usos posibles, ya estaban como información en la red. Los resultados de la implementación fueron óptimos en todas sus líneas, siendo datos precisos e interpretables para los fines de nuestro producto, junto con notar el amplio mundo de posibilidades que nos ofrece la tecnología para trabajar con ella.

Nuestros sensores siempre fueron pensados para recopilar información y con ella poder monitorear el estado de nuestro producto, además del cuidado que ofrecemos como servicio, y este objetivo lo cumplimos con artefactos ‘’simples’’, hablando desde el manejo, uso e implementación requerida para nuestro producto, pero juntando todas las funciones que teníamos para poder utilizar, pudimos montar un invernadero con todo lo necesario para el cuidado, por lo que podemos afirmar que por el lado de la tecnología, este proyecto fue necesario para poder tener una base para lo que se quiere llegar a desarrollar y producir más adelante en la Carrera.

El proceso del proyecto estuvo marcado por intentar organizarlo constantemente, tanto por tiempo como por falta de conocimientos a la hora de armar un esquema de trabajo que pudiera llevarse a cabo, suponiendo fechas y quehaceres. Si bien el proyecto estuvo terminado y las fechas fueron cumplidas, nuestro plan de trabajo no fue llevado a cabo con rigurosidad, por lo que es un factor importante a mejorar para seguir adelante, porque en la siguiente ocasión podemos vernos en problemas mayores que podríamos haber prevenido.

El trabajo en equipo fue clave ya que, por el mismo hecho de desorganización a la hora de llevar a cabo las actividades, podíamos suplir las fallas del compañero, o tomar las riendas del proyecto en caso de ser necesario. Como ya hemos dicho, todo el proyecto nos llevó a entender cómo funcionan los plazos, lo necesarios que son y que cumplirlos lleva el

trabajo a una instancia en que está bajo tu control, por lo que un buen equipo de trabajo facilita todo este proceso.

El desarrollo de este proyecto nos llevó a tener que estudiar distintas áreas de la informática, desde la programación del arduino hasta la instalación de circuitos, junto con todo el trabajo de gestión que se debe llevar a cabo para que un trabajo quede acorde a lo que un cliente nos está pidiendo. Todo esto, con los aciertos y errores, nos ayudaron para entender cómo se manejan los proyectos al momento de presentarse como un ingeniero ante un cliente, donde necesitamos mostrar un producto que solucione problemas, que sea funcional y que podamos cumplir los plazos que nos pide.

Código

```
#include <DHT_U.h> //Sublibreria del sensor DHT11
#include <DHT.h> //Libreria de funciones del sensor humedad - temperatura DHT11
#include <Time.h> //Libreria del reloj
#include <Ethernet.h> //Libreria para conexión Ethernet
#include <SPI.h>  //Sublibreria de conexión Ethernet

#define DHTPIN 7 // Definimos el pin digital donde se conecta el sensor (modificable)
#define DHTTYPE DHT11 // Definimos el sensor (Dependiendo del tipo de sensor) (modificable)

DHT dht(DHTPIN, DHTTYPE); // Inicializamos el sensor DHT11

// Configuracion del Ethernet Shield
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFF, 0xEE}; // Direccion MAC
byte ip[] = { 192,168,0,131 }; // Direccion IP del Arduino (Modificable)
byte server[] = { 192,168,0,100}; // Direccion IP del servidor (Modificable)
EthernetClient client;

int moistureSensor = 0; //Declaramos la variable para leer la humedad del suelo
float moisture_val = 0; //Declaramos la variable que se le asignara el valor de la humedad del suelo recibida
const int pinB = 2; //Declaramos el pin digital que se conectara a la bomba de agua (modificable)
const int pinV = 3; //Declaramos el pin digital que se conectara al ventilador (modificable)
const int pinL = A5; //Declaramos el pin analogo que se conectara al rele (modificable)
int verificarA = false; //Verificador el estado actual de riego
int riegoON = false; //Estado de riego
```

```
int aguaN = 0; //Contador de veces de riego
int Nr = 3; //Numero de veces de riego (modificable)
int Ir = 10; //Segundos de intervalos de riego y no riego (modificable)
int m = 0; //variable auxiliar de segundos
int Pt = 25; //Temperatura al cual prender y apagar la ventilacion (modificable)

void setup() {
  Serial.begin(9600); //Abrimos el puerto serial
  dht.begin(); // Comenzamos el sensor DHT
  pinMode(pinB, OUTPUT);  //Definimos pin de la bomba de agua como salida
  pinMode(pinV,  OUTPUT);  //Definimos pin de ventilador como salida
  pinMode(pinL, OUTPUT); //Definimos pin de luz como salida
  Ethernet.begin(mac, ip); // Inicializamos el Ethernet Shield

}

void loop() {
  delay(1000);
  int h = dht.readHumidity(); // Leemos la humedad relativa
  int t = dht.readTemperature(); // Leemos la temperatura en grados centígrados (por defecto)
  moisture_val = humedadsuelo(); //Leemos la humedad del suelo

  // Comprobar si ha habido algún error en la lectura
  if (isnan(h) || isnan(t)) {
    Serial.println("Error obteniendo los datos del sensor DHT11");
    return;
  }
  if (isnan(moisture_val)) {
    Serial.println("Error obteniendo los datos del sensor humedad - suelo");
    return;
  }

  // Imprimir informacion
  Serial.print("Humedad Ambiente: ");
  Serial.print(h);
  Serial.print(" %\t");
  Serial.print("Temperatura Ambiente: ");
  Serial.print(t);
  Serial.print(" °C\t ");
  Serial.print("Humedad Suelo: ");
  Serial.print( moisture_val );
  Serial.print("\t");
  Serial.print("Hora: ");
  Serial.print(hour());
  Serial.print(":");
  Serial.print(minute());
  Serial.print(":");
  Serial.println(second());

```

```
  //Acciones
  if(moisture_val>=892)
    moisture_val=1;
  else if(moisture_val>=501)
    moisture_val=2;
  else if(moisture_val>=420)
    moisture_val=3;
  else
    moisture_val=4;

  //Enviar datos
  enviar(h,"HA");
  enviar(t,"TE");
  enviar(moisture_val,"HS");
  //Iluminacion Ej: prenderLuz(Hora,Minuto,Segundo) -
 apagarLuz(Hora,Minuto,Segundo) (Modificable)

  prenderLuz(0,0,5);
  apagarLuz(0,0,25);

  //Ventilacion
    if(t>Pt){
      digitalWrite(pinV, HIGH);

    }
    if(t<=Pt){
      digitalWrite(pinV, LOW);

       }

     //Riego
     if(riegoON == true)
     {
         if(verificarA==true&&second()==m)
         {
           apagarRiego();
           verificarA=false;
           if(aguaN<Nr){m = tiempoR(m);}
           if(aguaN==Nr)
           {
               riegoON=false;
               terminoRiego();
           }

         }else if(verificarA==false&&second()==m){
           prenderRiego();
           verificarA=true;
           aguaN += 1;
           m = tiempoR(m);

         }
     }

```

```
126.    //Horas de riego, EJ: horaRiego(Hora,Minuto,Segundo);
   (Modificable)
127.    horaRiego(0,0,2);
128.
129.  }
130.
131.  void prenderLuz(int a, int b, int c) //Funcion para prender luz
132.  {
133.    if(hour()==a&&minute()==b&&second()==c)
134.      digitalWrite(pinL,HIGH);
135.  }
136.
137.  void apagarLuz(int a, int b, int c) //Funcion para apagar Luz
138.  {
139.    if(hour()==a&&minute()==b&&second()==c)
140.      digitalWrite(pinL,LOW);
141.  }
142.
143.  void horaRiego(int a, int b, int c) //Funcion para iniciar un
   proceso de riego
144.  {
145.    if(hour()==a&&minute()==b&&second()==c)
146.    {
147.        prenderRiego();
148.        inicioRiego();
149.    }
150.
151.  }
152.
153.  void inicioRiego() //Funcion para declarar el inicio del riego
154.  {
155.        aguaN += 1;
156.        verificarA  = true;
157.        riegoON = true;
158.        m = second();
159.        m = tiempoR(m);
160.  }
161.
162.  void terminoRiego() //Funcion para declarar el termino del riego
163.  {
164.        aguaN = 0;
165.        verificarA  = false;
166.        riegoON = false;
167.        m = 0;
168.  }
169.
170.  int tiempoR(int m) //Funcion para retornar la siguiente vez que se
   prenda o apague el riego
171.  {
172.      int x = 7;
173.      x += m;
174.      if(x>60)
175.      {
176.          x = x-60;
```

```
177.       }
178.       return x;
179.   }
180.
181.   void prenderRiego() //Funcion quwe acciona la bomba de agua
182.   {
183.     digitalWrite(pinB, HIGH);   // Encender el pin de la bomba de
   agua
184.   }
185.
186.   void apagarRiego() //Funciona que apaga la bomba de agua
187.   {
188.     digitalWrite(pinB, LOW); //Apagar el pin de la bomba de agua
189.   }
190.
191.   void prenderLuz() //Funcion para prender luz
192.   {
193.     digitalWrite(pinL, HIGH); //Encender pin de la luz
194.   }
195.
196.   void apagarLuz() //Funcion para apagar luz
197.   {
198.     digitalWrite(pinL, LOW); //Apagar pin de la luz
199.   }
200.
201.   int humedadsuelo()//Funcion para medir la humedad del suelo
202.   {
203.     int x = 0;
204.     x = analogRead(moistureSensor); //Leer la humedad del suelo
   recibida
205.     if(x<1000&&x>=751)
206.         x+=131;
207.     if(x<751&&x>=566)
208.         x+=269;
209.     if(x<566&&x>=472)
210.         x+=325;
211.     if(x<472&&x>=234)
212.         x+=433;
213.     if(x<234&&x>=135)
214.         x+=366;
215.     if(x<135&&x>=104)
216.         x+=363;
217.     if(x<104&&x>=76)
218.         x+=379;
219.     if(x<76&&x>=56)
220.         x+=375;
221.     if(x<56&&x>=49)
222.         x+=379;
223.     if(x<49)
224.         x+=372;
225.     return x; //Retornar valor
226.   }
227.
```

```
228.  void enviar(int x, String y) // Proceso de envio de muestras al
   servidor
229.  {
230.    Serial.println("Connecting...");
231.    if (client.connect(server, 80)>0) {  // Conexion con el servidor
232.      if(y == "TE")
233.        client.print("GET /garduino/iot.php?valor=");
234.      else if(y == "HA")
235.        client.print("GET /garduino/ioa.php?valor=");
236.      else if(y == "HS")
237.        client.print("GET /garduino/ios.php?valor=");
238.      client.print(x);
239.      client.println(" HTTP/1.0");
240.      client.println("User-Agent: Arduino 1.0");
241.      client.println();
242.      Serial.print("Datos enviados de ");
243.      Serial.print(y);
244.      Serial.println(" enviados");
245.    } else {
246.      Serial.println("Fallo en la conexion");
247.    }
248.    if (!client.connected()) {
249.      Serial.println("Desconectado");
250.    }
251.    client.stop();
252.    client.flush();
253.
254.  }
```