industriales
etsii

Escuela Técnica Superior de Ingeniería Industrial

Universidad Politécnica de Cartagena

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Escuela Técnica Superior de Ingeniería Industrial

SISTEMA ELECTRÓNICO DE BAJO COSTE PARA LA MONITORIZACIÓN DE VARIABLES AGRONÓMICAS EN UN SEMILLERO

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA ELECTRÓNICA INDISTRIAL Y AUTOMÁTICA

Autor: **Pedro Mayordomo Molina**
Director: Juan Suardíaz Muro

Cartagena, a 15 de marzo del 2018

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INDUSTRIAL

DEPARTAMENTO DE TECNOLOGIA ELECTRÓNICA

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA ELECTRÓNICA INDISTRIAL Y AUTOMÁTICA

"SISTEMA ELECTRÓNICO DE BAJO COSTE PARA LA MONITORIZACIÓN DE VARIABLES AGRONÓMICAS EN UN SEMILLERO"

AUTOR: **Pedro Mayordomo Molina**
DIRECTOR: **D. Juan Suardíaz Muro**

CURSO 2017-2018

Este presente trabajo está dedicado a mi familia por su apoyo y esfuerzo durante todos estos años.

Índice de Contenido

Índice de Figuras

Índice de Tablas

CAPÍTULO 1

1 INTRODUCCIÓN

1.1 INTRODUCCIÓN

El diccionario de la RAE define la agricultura como “el arte de cultivar la tierra”. La agricultura se enfrenta al enorme reto de alimentar a una población en aumento, se estima que para 2050 la población mundial alcanzara los 9.200 millones de personas. Para lograr abastecer a la población ha sido necesario incorporar la tecnología al sector. Se invierten millones en I+D+I para lograr aunar la tecnología y la agricultura logrando obtener un mayor rendimiento y respeto por el medioambiente.

Uno de los indicadores que ha caracterizado la evolución a lo largo de la historia de la tecnología aplicada a la agricultura, ha sido el desarrollo de la tecnología militar. En la mayoría de los casos ambas tecnologías han ido de la mano, por ejemplo los Sistemas de Navegación Global por Satélite (Global Navigation Satellite System , GNSS) y los Sensores Remotos (Remote Sensors , RS) usados para la localización y control remoto, es un claro ejemplo de tecnologías creadas para la industria militar , que con el tiempo han encontrado un hueco en el sector agrario de gran importancia.

La “Agricultura de Precisión” (AP) no es ni más ni menos que la interrupción de Las llamadas “Tecnologías de la Información y la Comunicación” (TIC) en la agricultura, logrando la integración de la tecnología en la Producción agraria.

Las características a destacar de la AP son sus sistemas de posicionamiento, con implementación de GNSS, para facilitar la precisión y captadores de adquisición de información.

La industria de la agricultura implementa la AP, al igual que cualquier negocio, buscando maximizar el rendimiento, rentabilizar la producción, disminuir los costes, además de mantener una agricultura sostenible procurando llegar a la máxima eficiencia en el uso de recursos

naturales como agua, suelo, energía. En definitiva, se busca duplicar el rendimiento de la plantación sin duplicar en recursos.

Uno de los problemas a solventar es la agricultura de regadío que está siendo presionada en Europa, para obtener una mayor gestión de sus recursos hídricos, con el fin de aumentar la productividad del agua utilizada.

Dado el alto nivel de consumo de agua en España en las zonas de regadío, se hace necesario proponer una solución para la monitorización de la humedad del terreno y la climatología, para poder desarrollar un riego de precisión. La implementación de la tecnología para la gestión del agua tiene grandes beneficios para el uso sostenible del agua, gracias al control de las necesidades de la planta el usuario puede saber la aportación de agua necesaria en cada momento.

Tras la premisa de "renovarse o morir" aplicada al sector agrícola, debido a la necesidad de abastecer a una población y con el hándicap de una crisis mundial, es de vital importancia la integración de la tecnología.

En este proyecto buscamos realizar un sistema de bajo coste, con el cual el usuario podrá tener monitorizado todas las variables ambientales de su cultivo, el cual dispondrá de un seguimiento por control remoto, además de tener una base de datos para observar como fluctúan esas variables a lo largo de periodo de plantación de su cultivo. Una alta robustez y calidad comparable a los sistemas de monitorización disponibles en el mercado, pero con un coste menor.

1.2 OBJETIVOS DEL PROYECTO

El objetivo del presente proyecto es el desarrollo de un sistema de bajo coste que permita la monitorización de variables agronómicas ambientales en un semillero agrícola. En concreto, el proyecto envuelve las siguientes variables a monitorizar:

- Temperatura y humedad del sustrato
- Temperatura y humedad del ambiente
- Radiación solar

Otro objetivo secundario será, además, el desarrollo de un servidor para la recepción de datos y almacenamiento de variables en una base de datos, con posibilidad de un acceso remoto a ellos (presentados mediante interfaz visual diseñada a tal efecto) para poder reaccionar de manera inmediata ante las necesidades de la plantación.

1.3 MOTIVACIONES DEL PROYECTO

En los últimos años se ha desarrollado el nivel de automatización en la agricultura ya sea a nivel explotación o del hogar para cultivos del hogar.

Por esta razón las grandes empresas del sector ofrecen en el mercado sus productos compactos a precios demasiados elevados, además ofrecen la posibilidad de escalabilidad, pero deberán ser dispositivos o sensores de su misma línea.

Una de las principales motivaciones de este proyecto es poder contribuir al desarrollo de un sistema de bajo coste, el cual sea capaz de supervisar y monitorizar las variables que se desean controlar a través de internet, además de subir los datos recibidos a internet formando parte de la IOT(Internet of Things) , concepto que será revisado posteriormente.

Por supuesto este proyecto tendrá una alta escalabilidad y flexibilidad, se realizará un código sencillo de implementar a la hora de aumentar la cantidad de nodos o añadir nuevos sensores.

1.4 ESCENARIO PLANTEADO

Para poder desarrollar el proyecto, es necesario una descripción funcional del proceso que se requiere, con la cual es posible marcar unas pautas para llevar a cabo una solución del problema el cual se quiere solucionar.

En la siguiente ilustración se encuentra el emplazamiento ofrecido por la UPCT

Ilustración 1:Emplazamiento

Se puede apreciar una pequeña plantación compuesta por una matriz de tres filas por seis columnas. Se plantea una situación en la cual en cada fila se dispondrá de distintas plantaciones o la misma plantación, pero con sustratos diferentes.

Ante esta situación es necesario conseguir recolectar los datos de temperatura, humedad, incidencia solar que se estimen necesarios para poder tener una información detallada de la plantación.

Para ello se tomarán tres nodos, uno colocado en cada fila por lo general en el sitio más representativo para poder extrapolar y hacer una estimación de todo el conjunto, lo más lógico será colocar los nodos en 3º o 4º puesto de cada fila.

Cada nodo será el encargado de recolectar los parámetros necesarios para tener un estudio completo de cada fila, estos nodos le enviaran la información a un sumidero.

En el desarrollo del trabajo se expone cómo se resuelve el problema planteado, explicando los pasos seguidos además de las herramientas utilizadas

CAPÍTULO 2

2 ESTADO DEL ARTE

2.1 INTRODUCIÓN

Para poder hablar del vertiginoso desarrollo de las redes inalámbricas y como han influido drásticamente en nuestra sociedad, se debe de hacer un breve recorrido en la historia de las comunicaciones.

La primera y obligada parada en nuestro recorrido es por el físico Escoces James Clerk Maxwell, conocido por haber desarrollado la teoría del electromagnetismo clásica.

Las ecuaciones de Maxwell fueron demostradas experimentalmente por el físico alemán Heinrich Rudolf *Hertz*. Pudo demostrar que las ondas electromagnéticas se pueden propagar en el aire y el vacío. Como reconocimiento a este descubrimiento, la unidad de medida de la frecuencia recibe su nombre, Hertz.

Los artículos de Hertz dieron paso al gran desarrollo de las telecomunicaciones, como el ruso Guglielmo Marconi quien fue uno de los impulsores de la radiotransmisión a distancia. El llevo a cabo el primer uso práctico y comercial de la radio.

Pero el despegue definitivo llego de la mano de Lee DeForest en 1906, invento el tríodo. Lo cual suponía poder modular y amplificar señales de radio al mismo tiempo, esto fue toda una revolución tecnológica. También cabe destacar que el nombre de “Radio” se lo debemos a él, puesto que fue el primero en llamarla así.

Comenzaron a darse las primeras aplicaciones prácticas, como el uso por la policía de Detroit en 1921, primer cuerpo de seguridad en utilizar radiocomunicación. La primera radiodifusión con fines comerciales fue en Buenos Aires.

Las primeras comunicaciones operaban en bandas de 2Mhz, pero el aumento de la demanda y de la tecnología permitió el uso de frecuencias mayores [1].

2.2 ONDAS DE RADIO

En este apartado se explican las ondas de radio, puesto que son las que son parte importarte del proyecto para la comunicación entre nodos y el sumidero.

Las ondas de radio son una pequeña porción del denominado espectro electromagnético. Como se puede ver en la siguiente ilustración.

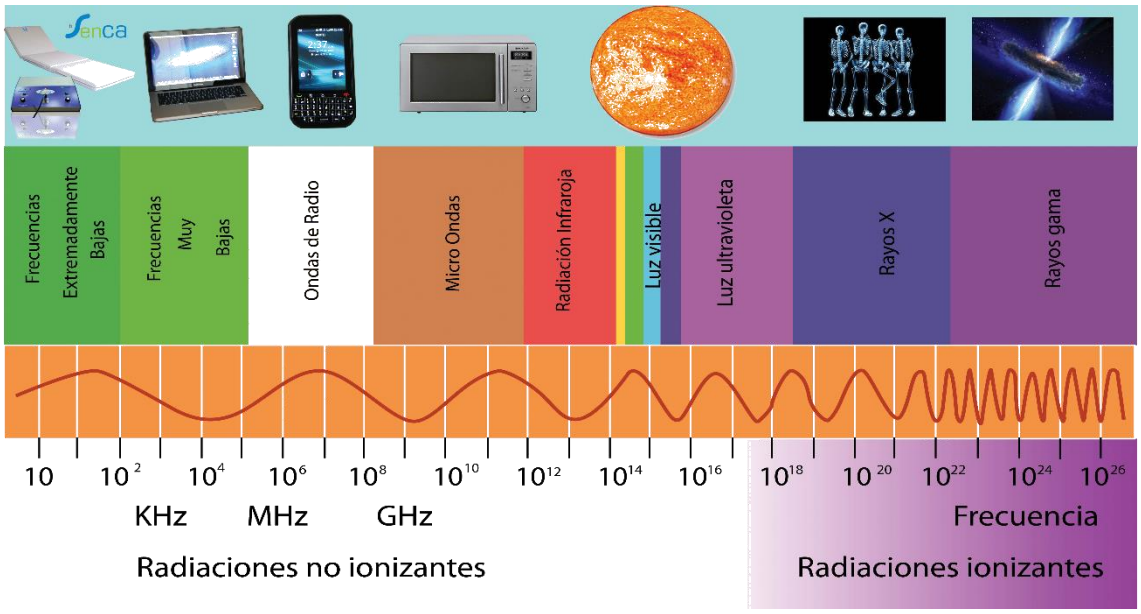

Ilustración 2:Espectro electromagnético

Dentro del espectro electromagnético, es posible diferenciar varias bandas. Recorriendo el espectro desde las bajas frecuencias, las cuales tienen la mayor longitud de onda, hasta los rayos gamma la frecuencia más alta, por lo tanto, la menor longitud de onda.

BANDA	**LONGITUD DE ONDA (M)**	**FRECUENCIA(HZ)**
Rayos gamma	< 10x10−12m	> 30,0x1018Hz
Rayos X	< 10x10−9m	> 30,0x1015Hz
Ultravioleta extremo	< 200x10−9m	> 1,5x1015Hz
Ultravioleta cercano	< 380x10−9m	> 7,89x1014Hz

Luz Visible	< 780x10−9m	> 384x1012Hz
Infrarrojo cercano	< 2,5x10−6m	> 120x1012Hz
Infrarrojo medio	< 50x10−6m	> 6,00x1012Hz
Infrarrojolejano/submilimétrico	< 1x10−3m	> 300x109Hz
Microondas	< 10−2m	> 3x108Hzn. 1
Ultra Alta Frecuencia -Radio	< 1 m	> 300x106Hz
Muy Alta Frecuencia -Radio	< 10 m	> 30x106Hz
Onda Corta - Radio	< 180 m	> 1,7x106Hz
Onda Media - Radio	< 650 m	> 650x103Hz
Onda Larga - Radio	< 10x103m	> 30x103Hz
Muy Baja Frecuencia -Radio	> 10x103m	< 30x103Hz

Tabla 1:Rangos Espectro electromagnético

El espectro electromagnético donde se encuentra la Radios, es bastante amplio, está situado desde los 3kHz hasta 300 GHz .

En este espectro como se ve reflejado en la tabla 1 la radio puede subdividirse en diferentes Bandas.

En este proyecto se ha trabajado con la banda de frecuencias ultra altas UHF (Ultra High Frequencies), las cuales abarcan de 300 a 3000Mhz. Posteriormente se explican los Módulos NRF 24L01 los cuales son utilizados para la comunicación entre nodos.

I. ATENUACIONES EN EL ESPACIO

Debido a la propagación en el espacio libre las ondas electromagnéticas sufren una atenuación por el espacio libre o por su condición que les hace estar expuestas a fenómenos los cuales puede ser capaces de alterar la forma de onda. Esto lleva consigo el requisito de una alta potencia en comparación con aquellas cuales su medio de trasmisión es un cable físico.

El hecho de la atenuación por la propagación se puede explicar de la siguiente forma.

Si se imagina que la dirección de propagación de las ondas es uniformes en todas las direcciones (fuente isotrópica), como se puede observar en la siguiente ilustración.

Ilustración 3:Propagación de fuente isotrópica

En este escenario la densidad de potencial no cambiaría con la distancia, será igual para el caso de X=0,5m y en X=1,5m. Se trataría de un caso ideal.

Pero el caso real es distinto, las ondas se trasmiten en forma esférica. Un ejemplo que pude servir para explicar cómo se expande la onda es el efecto que se genera al tirar una piedra sobre el agua, la expansión de la forma de onda, aunque esta sea bidimensional.

Se podría decir que sirve para explicar cómo se dispersa la onda electromagnética de nuestra fuente, como se puede observar en la siguiente ilustración.

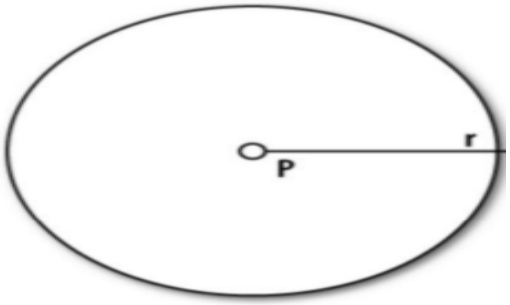

Ilustración 4:Forma de onda de fuente real

Para tratar de explicar este caso como el caso anterior, se supone que se toma un plano de 1 m^2 y es cortado en X=0,5 y posteriormente en X=1,5m, la densidad de potencial en este caso disminuirá con forme aumentamos la distancia con respecto al origen.

$$P_r = \frac{G_1 \times G_2 \times \lambda^2 \times P_t}{(4\pi)^2 \times \mathrm{d}^2}$$

La ecuación anterior muestra la atenuación en el espacio libre, es uno de los parámetros más importantes en las antenas este es medido en decibelios.

- G1 y G2 son las ganancias de las antenas
- P la potencia del emisor
- Landa (λ), longitud de onda

- d, la distancia entre emisor y receptor

De esta fórmula, esencial en telecomunicaciones para calcular la atenuación se puede deducir que cada vez que se doble la distancia, la potencia deberá ser aumentada por cuatro para obtener una compensación a la atenuación de la señal producida.

2.3 REDES DE SENSORES INALAMBRICAS

I. INTRODUCIÓN

Las redes inalámbricas son aquellas que tienen una comunicación sin utilizar un medio físico, comunicadas por medio de ondas electromagnéticas, dejando a un lado la conexión de cables.

El avance de las comunicaciones ha ayudado a realizar la conectividad entre sensores, los cuales son indispensables debido a que son los que nos ayudan a recolectar la información del medio físico a monitorizar y/o controlar.

El campo de aplicación de las redes de sensores inalámbricas, WSN (Wireless Sensors Network) se ha extendido considerablemente hacia la monitorización y adquisición de datos en distintos entornos tales como la industria, domestica, agricultura, seguridad.

Sus posibilidades de interconexión le otorgan un enorme potencial para poder hacer proyectos con alto nivel de escalabilidad y flexibilidad.

Actualmente los dispositivos están avanzando en todos los campos, cada vez tienen una mayor capacidad de comunicación, pueden llegar a cualquier localización, su consumo es mucho menor lo cual les aporta una mayor autonomía.

En resumen, las WSNs serán conjunto de nodos sensores que trabajan con un mismo fin de recopilar datos del entorno que les rodea, este concepto está fuertemente ligado al Internet de las cosas, como se muestra más adelante.

II. VENTAJAS

Las principales ventajas que presentan las redes inalámbricas con respecto a las redes cableadas son las siguientes.

- **Rapidez de despliegue**

El proceso de instalación de una red es rápido, en ocasiones donde no es posible la instalación de cables o el tiempo será bastante grande debido a la infraestructura a diseñar y construir, lo más sencillo es realizar una comunicación inalámbrica, nos ayuda a reducir tiempo y costes.

- **Flexibilidad**

Las redes inalámbricas ofrecen una gran flexibilidad, puesto que la reconfiguración de la red es muy sencilla. El único proceso es mover los nodos a un lugar dentro del rango de comunicación.

- **Reducir costes**

Gracias a la tendencia del mercado actual de centrarse en las redes inalámbricas esto influye en abaratar los costes debido a la competencia de mercado.

También cabe destacar que en ocasiones el precio de hacer un proyecto con muchos metros de cable y los costes de infraestructura o los cambios frecuentes en la topología conlleva un incremento del coste final.

- **Escalabilidad**

Es una de las grandes ventajas, la incorporación de nuevos elementos a la red es bastante sencillo.

III. DESVENTAJAS

Por todas las ventajas expuestas anteriormente se ha originado la expansión rápida de las redes inalámbricas, pero también tiene algunas desventajas.

- **Seguridad**

Uno de los miedos de esta tecnología es la vulnerabilidad que tienes al comunicar en un espacio abierto, todos los paquetes fluyen en el aire. Por lo que se hace necesario la utilización de algoritmos de cifrado de mensajes e impedir a los atacantes esnifar el tráfico en texto plano. Cualquiera que esté en el rango de cobertura de tu señal puede ver los paquetes de datos.

- **Disponibilidad**

la disponibilidad del canal de comunicación es un punto en contra en la comunicación de redes inalámbricas, no tiene el mismo porcentaje que una línea de comunicación física dedicada a ese canal, en ocasiones depende de la banda en la cual estas comunicando pueden a ver dos emisiones de datos en un mismo momento y producirse una interferencia de señales que ocasiona la perdida de datos.

También puede verse afectado por la climatología, u obstáculos en el medio. Para procesos en los cuales se necesita una disponibilidad muy alta no es admisible una comunicacional inalámbrica, se utilizan línea cableada dedicada al proceso.

- **Impacto en la salud**

Desde comienzo del cambio tecnología y evolución de las comunicaciones inalámbricas, siempre se ha hablado de los riesgos asociados a la radiación electromagnéticas que afectan a los seres humanos.

De momento no se ha podido demostrar nada con respecto al efecto de la radiofrecuencia sobre las personas [2].

IV. TIPO DE REDES INALAMBRICAS

Debido a la multitud de dispositivos que han sido desarrollados para la comunicación inalámbrica, se ha dado la necesidad de realizar una clasificación de los mismos.

La clasificación por la que se ha optado es por el tipo de alcance que tienen los dispositivos, es decir la extensión de área la cual pueden abarcar.

En función del alcance es posible dividir las redes en cuatro áreas:

- **Redes de área personal**

Las redes de área personal, WPAN (Wireless Personal Area Network), están orientadas al radio de acción que pueda tener una persona, un bajo alcance. Están destinados a la sustitución de los cables.

Un uso típico podría ser en una oficina, en la cual sustituyes el teclado, mouse, impresora ...Por dispositivos inalámbricos, se suele utilizar Bluetooth, el cual opera en la banda de 2,4Ghz con un corto alcance.

Ilustración 5:Red de área personal, WPAN (Wireless Personal Area Network)

- **Redes de área local**

Las redes de área local inalámbricas, WLAN (Wireless Local Area Network) es la alternativa a las redes LAN. Conseguimos eliminar los cables de conexión a impresoras, PC, etc.

Las redes WLAN incluyen tecnologías que operan en las bandas sin licencia, ISM (Industrial, Scientific and Medical), son bandas reservadas para el uso no comercial están en 2,4GHz y 5GHz,

estas bandas han sido popularizadas, algunas de las más conocidas son para WLAN (WIFI) y WPAN (Bluetooth).

El estándar IEE 802.11 define las características de una red WLAN.

Este estándar define los dos niveles inferiores del modelo OSI, es decir la capa física y capa de enlace de datos. Dentro de este estándar se puede destacar las siguientes modificaciones.

-802.11a Proporciona un mayor ancho de banda, admitiendo un máximo de 54Mbps en la banda de 5GHz. Usa modulación OFDM. Tiene 12 canales sin solapa, 8 para red inalámbrica y 4 para conexiones punto a punto.

-802.11b Es el estándar más utilizado en la actualidad. Hasta 11Mbps de datos brutos en la banda ISM de 2, 4GHz.tiene un alcance de 300 metros en espacio abierto, utiliza el rango de frecuencias de 2,4GHz.

-802.11g Es compatible con el estándar 802.11b, lo que significa que los dispositivos que admiten el estándar 802.11g también pueden funcionar con el 802. 11b.El ancho de banda ofrecido es elevado, trabaja en el rango de frecuencia de 2,4GHz.

Ilustración 6:Red de área local inalámbrica, WLAN (Wireless Local Area Network)

- **Redes inalámbricas de área metropolitana**

Las redes inalámbricas de área metropolitana, WMAN's (Wireless Metropolitan Area Networks) dan cobertura a un área extensa. Con un gran ancho de banda donde consiguen desplegar una gran cantidad de servicios como voz, acceso a Internet, comunicaciones de datos y vídeo.

La tecnología está basada en WIMAX (Worlwide Interoperability for Microwave Access)un estándar de comunicación inalámbrico basado en la norma IEEE 802.16 , es un protocolo parecido a Wi-Fi, pero WIMAX tiene una mayor cobertura y ancho de banda.

Los bucles locales inalámbricos ofrecen una velocidad efectiva de 1 a 10 Mbps, un alcance de 4 a 10 kilómetros. Esta tecnología permite conectar urbanizaciones sin el alto coste de tener que desarrollar una instalación de cables.

Ilustración 7:Redes inalámbricas de área metropolitana, WMAN's (Wireless Metropolitan Area Networks)

- **Redes Inalámbricas de área extensa**

Las redes Inalámbricas de área extensa, WWAN (Wireless Wide Area Network) pueden llegar a alcanzar una cobertura de miles de kilómetros, los teléfonos móviles están conectados a estas redes.

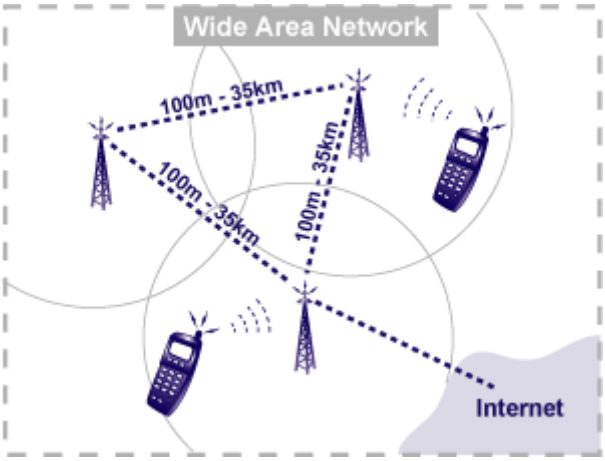

Ilustración 8:Redes Inalámbricas de área extensa, WWAN (Wireless Wide Area Network)

Estándares como AMPS (Advanced Mobile Phone System), GSM (Global System for Mobile Communications), D-AMPS (Digital-AMPS) o UMTS (Universal Mobile Telecommunications System) pertenecen a este conjunto de redes.

En la siguiente ilustración es posible ver el rango de distancias en la cual entre cada clasificación y los distintos protocolos utilizados

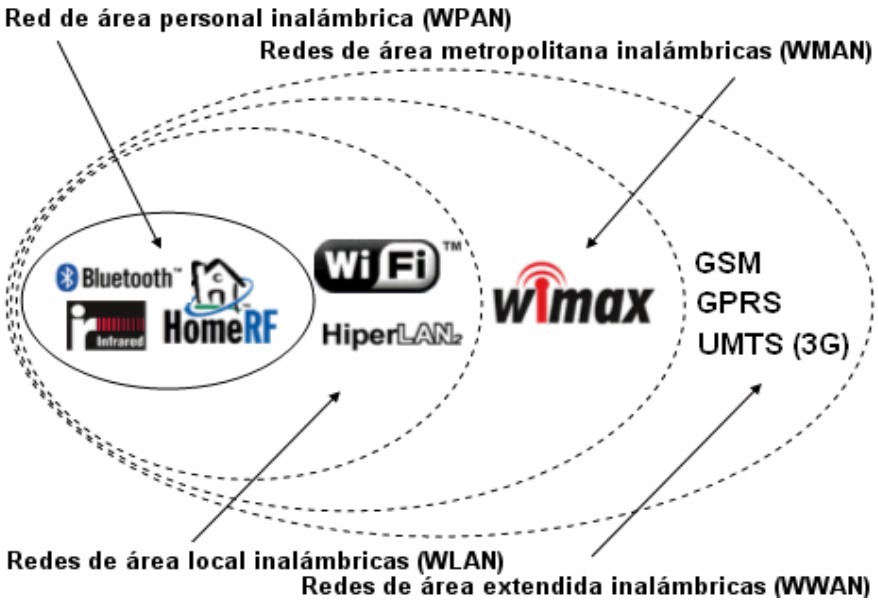

Ilustración 9:Protocolos de comunicación de redes inalámbricas

V. MODOS DE FUNCIONAMIENTO

Según la configuración utilizada para formar la red se pueden calificar en los siguientes tipos:

- **Modo Ad-Hoc**

No requiere un punto de Acceso (AP), las estaciones se comunican directamente entre sí, esto hace que la cobertura este limitado por el alcance que tengan cada estación entre sí.

Ya que son redes descentralizadas, no se utilizan nodos centrales. Son redes con una alta escalabilidad.

- **Modo Infraestructura**

En esta estructura se encuentra como mínimo un punto de acceso, para la comunicación entre estaciones deben establecer comunicación con AP primeramente.

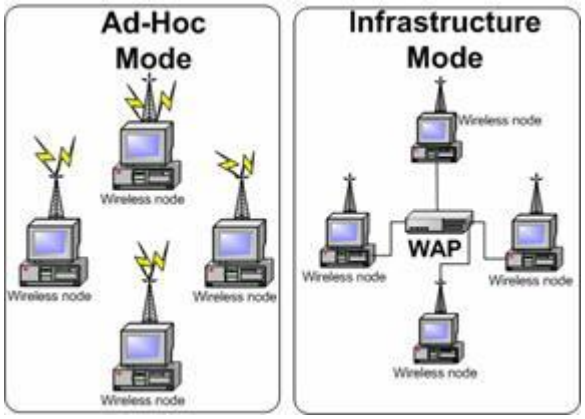

Ilustración 10:Configuración utilizada en redes inalámbricas

2.4 IOT

I. INTRODUCION

El término “Internet de las cosas”, IOT (Internet Of Things), es una expresión que cada vez está teniendo más importancia en el mundo de la industria ,el avance de la tecnología está dando la oportunidad de realizar la interconexión de las “cosas” a internet.

Esta terminología comenzó a utilizarse cuando los dispositivos conectados a internet fueron comenzando a ser mayor que el número de personas [3].

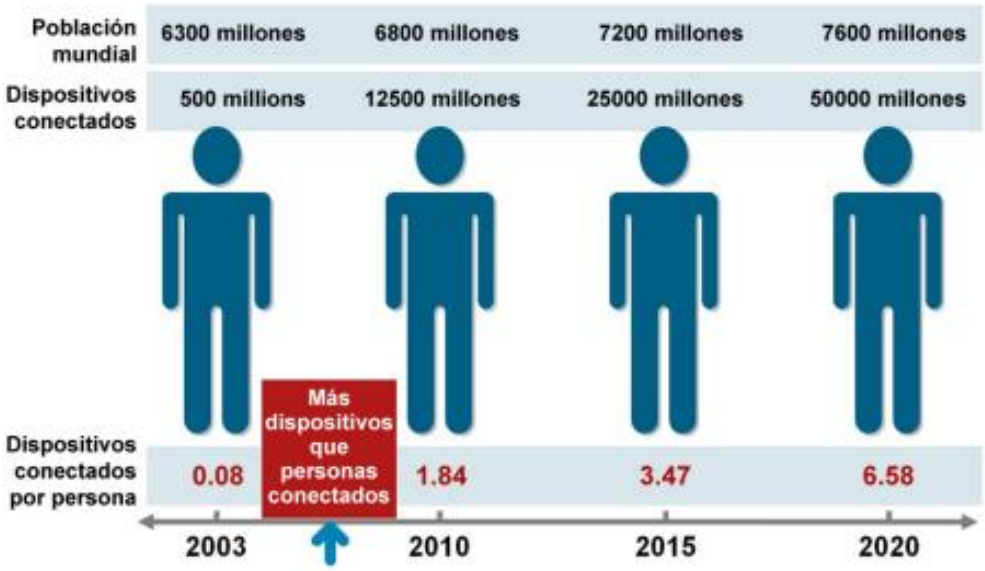

Ilustración 11:Evolucion IOT (Fuente:IBSG de Cisco,Abril de 2011)

Aunque esta gráfica no es del todo verídica, ya que se basa en la totalidad de la población mundial, la cual gran parte no está conectada a internet. En cuyo caso el valor de dispositivos conectados por persona aumentara drásticamente si se toma el número de personas conectadas a Internet.

Esto se debe al crecimiento explosivo de los PC portátiles, Tablet, Smartphone … Todos ellos son dispositivos conectados a Internet.

Pero hablar del IOT, lleva a pensar en todos los objetos cotidianos conectados a Internet, como la ropa, televisión, los utensilios de cocina, gachet, …todo comunicado a través de la red.

No dejan de salir proyectos que anteriormente podrían pensarse como futurísticos, por ejemplo, una tarea cotidiana como ir al supermercado podría quedar en el olvido, el frigorífico podría ser capaz de gestionarlo todo por ti, enviarte correos electrónicos de aviso, controlar caducidad de la comida … También proyectos ya implementados, como en domótica, la cual permite a muchas

familias controlar y monitorizar remotamente, la calefacción, cortinas, apagar y encender luces, etc.

Según las previsiones de millones de dispositivos conectados a Internet está haciendo que las grandes marcas de la industria piensen en el futuro y aprovechen esta trasformación para desarrollar productos orientados al Internet de las cosas.

Los elementos conectados cada vez son más, la complejidad radica en la infraestructura de gestión de esos datos, puesto que cada vez son mayores. Además de la complejidad de la seguridad de los datos.

II. DESARROLLO DE LA IOT

Se puede decir que el IOT ya es una realidad, con sus grandes ventajas y desventajas por su puesto. Nos lleva en una dirección hacia aplicaciones revolucionarias las cuales pretende mejorar drásticamente nuestra manera de vivir, trabajar, aprender y entretenernos.

Hay que ser consciente de la cantidad de datos que se están ingresando a la red constantemente, toda esta información tiene que ser procesada y digerida para poder obtener una información útil. Esta y otra información ayuda a generar un conocimiento. Con el conjunto de conocimientos se puede sentar las bases para generar una gran sabiduría.

Ilustración 12:Pirámide de conocimiento (Fuente:Cisco IBSG, abril de 2001)

Como se puede observar en la ilustración anterior, a mayor cantidad de datos mayor cantidad de conocimiento es posible obtener. El IOT apunta en esta dirección, en la cual la cantidad de datos generada aumente drásticamente. Esto nos ayudara a tener un mayor conocimiento sobre lo que nos rodea.

Las comunicaciones inalámbricas como Bluetooh , Zigbee,WI-FI, módulos de comunicación RF son los encargados de la interconexión de los nodos sensores , los cuales recolectan la información del entorno , estos nodos son los generadores de los datos los cuales forman parte importante del IOTs.

Analizando el término “Internet Of Things”, los objetos (things) son los sensores o actuadores que están en constante interacción con el entorno. Internet permite manejar los datos a través de diferentes interfaces.

III. LIMITACIONES DEL IOT

IOT tiene unas cuantas barreras que superar antes de comenzar su gran expansión:

- **Implementación de IPv6**

Unos de los grandes problemas de la gran ambición de IOT, son las direcciones de IPV4 en el mundo, no habría suficientes direcciones para todos los dispositivos.

Con el protocolo IPv4 se ha logrado explotar todos los dispositivos de usuarios de internet a través de PC, Notebook, además, de los Smartphone y Tablet.

Pero en este protocolo no tiene cavidad la explotación del Transporte, o los bienes de consumo como los electrodomésticos.

Con el protocolo IPv6 se solventa este problema puesto que dispone de 340 sextillones (3,4 x 10^{38}) de direcciones IP. En cambio, IPv4 solamente tiene capacidad para 4300 millones que parecían suficientes en la década de los 70. Hoy en día no se puede comprar más bloques de IPv4, a no ser que lo consigas a tu proveedor de internet, por lo tanto, ya se han comenzado la implementación de IPv6, compañías como YouTube o Google, ofreciendo sus servicios con ambos protocolos

- **La energía para alimentar los sensores**

Para conseguir lleva el IOT a su máximo potencia se deberá conseguir que los dispositivos conectados puedan ser autosuficientes, si no se debería estar cambiando constantemente la batería de los dispositivos.

- **La seguridad**

Uno de los mayores miedos de estar todo conectado a Internet es la vulnerabilidad a la cual estará sometida la gente, el peligro de no tener privacidad.

IV. TENDENCIAS DEL IOT

El crecimiento de IOT trasformara por completo el panorama actual, genera nuevos productos y servicios valiosos. Existirán millones de dispositivos conectados que generarán una enorme cantidad de datos.

Con tal volumen de datos, aparece el término “Big Data”, se trata de la gestión y análisis de esta gran cantidad de datos. Este concepto engloba la infraestructura, tecnología y servicios.

Los datos generados por Facebook, YouTube, un e-mail, facturas, sensores…son datos que por si mismo no tienen ninguna relación, estos datos son almacenados en con un formato especifico en una base de datos.

Gracias a los modelos analíticos que se están implementando a partir de la información recogida por los sensores, se puede entender el comportamiento del cliente y ofrecerle en cada momento el servicio que necesita.

En un futuro se deberá rediseñar los tradicionales sistemas operativos como Windows e iOS , puesto que no están diseñados para aplicaciones IOT , consumen demasiada energía ,además , necesitaran procesadores más rápidos para procesar las grandes tasas de datos .

Los ámbitos de aplicación de IOT cubren prácticamente todas las facetas de la vida diaria, no hay lugar en el cual no se pueda implantar el uso de esta tecnología o modo de pensar.

Los grandes ámbitos en los cuales está siendo desarrollado podría ser el medioambiente, la industrial, las ciudades, hogares, objetos de ocio o personales.

Está claro que queda mucho camino por recorrer, pero el fin es conseguir facilitar la vida a las personas y poder crear un mundo más sostenible conociendo en todo momento que está sucediendo [4].

CAPÍTULO 3

3 REQUISITOS DEL SISTEMA

3.1 INTRODUCCIÓN

En este proyecto se quiere conseguir un sistema que pueda monitorizar y controlar las variables de un invernadero. Gracias a la capacidad de controlar los parámetros recolectados se le otorgara una cierta inteligencia.

El objetivo es poder crear un sistema que tenga una alta capacidad de escalabilidad y un uso sencillo.

Para ello se debe de analizar el mercado, realizando una investigación previa de los productos que se ofertan hoy en día y posteriormente plantear una tabla con las funcionalidades que debe integrar nuestro proyecto.

3.2 PRODUCTOS SIMILARES

Para poder comenzar el desarrollo se debe hacer una investigación del mercado actual previamente, se puede encontrar una gran variedad de soluciones para realizar una monitorización de variables de temperatura, humedad,etc.

Los rangos de precios son muy variados dependiendo de la necesidad del cliente, debido al entorno en el cual va a ser usado el sistema se puede encontrar con una mayor robustez o por ejemplo necesidades de medir variables muy específicas para procesos delicados como podría ser medir el nivel de CO_2, estos dispositivos incrementan el coste final del producto.

A continuación, se muestran algunos de los sistemas que son posibles encontrar en el mercado, para poder tomar conciencia del panorama actual.

I. OPEN GARDEN SHIELD

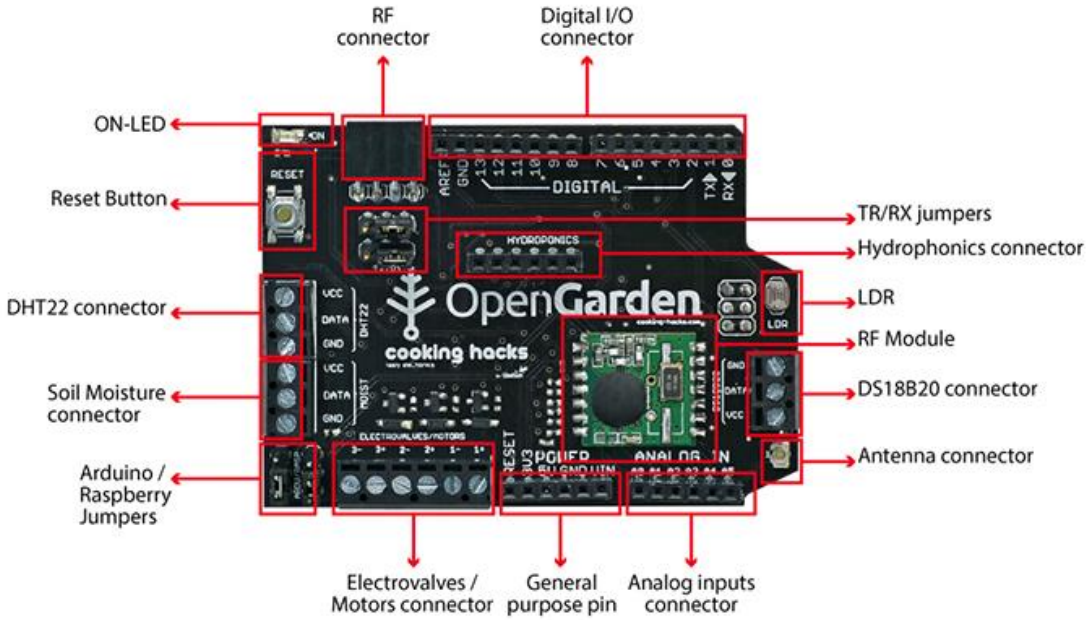

Ilustración 13:Open Garden Shield

Open Garden es una plataforma de código abierto, utilizada para controlar las plantas de interior o aire libre de forma remota, shield desarrollado por la empresa Cooking-Hacks .

Se venden por kit, dependiendo el uso el cual se le vaya a dar, de interior, aire libre o hidroponía, los distintos kits van acompañados de actuadores o sensores correspondientes al kit comprado.

Los Nodos serán los encargados de enviar la información hacia un GATAWEY que mostrara los datos por puerto serial o cargara los datos a un servidor web si se dispone de modulo wifi, mediante uso de la interfaz inalámbrica disponible.

Los códigos son dispuestos por el desarrollador, los cuales son actualizados, además de tener disponibles, tutoriales para el uso de las librerías y uso de las funciones para tener una sencilla programación.

El precio para conseguir un kit el cual incluya un nodo y el Gateway es de 199€, a partir de aquí la expansión por cada nodo 105€, cada kit viene con sus transmisores y receptores correspondientes además de los sensores y actuadores.

Este precio será incrementado al incluir en las placas Arduino, para cada nodo y el GATEWEY [5].

Para poder dar la una solución con Open Garden Shield similar al proyecto que se está desarrollando sería necesario tener 3 nodos y un Gateway , además de las placas de Arduino . El coste resultante será el siguiente

Precio:

Coste de Open Garden Shield		
Descripción	**Coste**	**Cantidad**
kit con nodo y Gateway	99€	1
Placa Nodo	105€	2
Placas Arduino Uno	20€	4
Total	389€	

Tabla 2:Coste Open Garden Shield

II. PARROT FLOWER POWER

Este accesorio directamente es conectado al substrato que se desea monitorizar, desarrollado por la compañía Francesa Parrot.

Flower Power permite conocer el estado de la planta, la sincronización es mediante Bluetooth usando una aplicación específica para el producto, la cual graficara la información recogida como el porcentaje de humedad, la calidad del abono, la temperatura ambiente…

Parrot ofrece una biblioteca con más de 7000 plantas, la cual permitirá buscar la ficha de tu planta y poder optimizar su cultivo.

Ilustración 14:Parrot flower power

El precio en el mercado es de unos 54 € [6].

Precio:

Para poder comparar el precio con nuestro proyecto se incluyen tres dispositivos Flower Power, para poder recopilar los datos de tres sustratos diferentes.

Coste de Parrot Flower Power		
Descripción	**Coste**	**Cantidad**
Dispositivo Flower Power	99€	3
Total	297€	

Tabla 3:Coste Parrot Flower Power

Como se puede observar en los dos casos anteriores a pesar de ser soluciones de bajo coste, tratar de dar una solución eficiente con estos productos en ambos casos el coste final asciende por encima de los 290€ y en el caso de Parrot Flower no tendríamos flexibilidad de poder escalar nuestro proyecto, en cambio con los shields de Open Garden si es posible modificar el código o mejorar la calidad de los sensores, aunque incrementaría el precio.

III. WASPMOTE

Este producto es un proyecto ya desarrollado, profesional y comercializado por grandes empresas. Dispositivo diseñado para la creación de redes inalámbricas de sensores distribuidas. Desarrollados por la empresa española Libelium.

Estos dispositivos otorgan una alta modularidad, permiten la instalación de una gran cantidad de sensores de diversas clases, además de el acoplamiento de distintos métodos de comunicación, apostando fuertemente por transceptores de radio xBee de la empresa Digi.

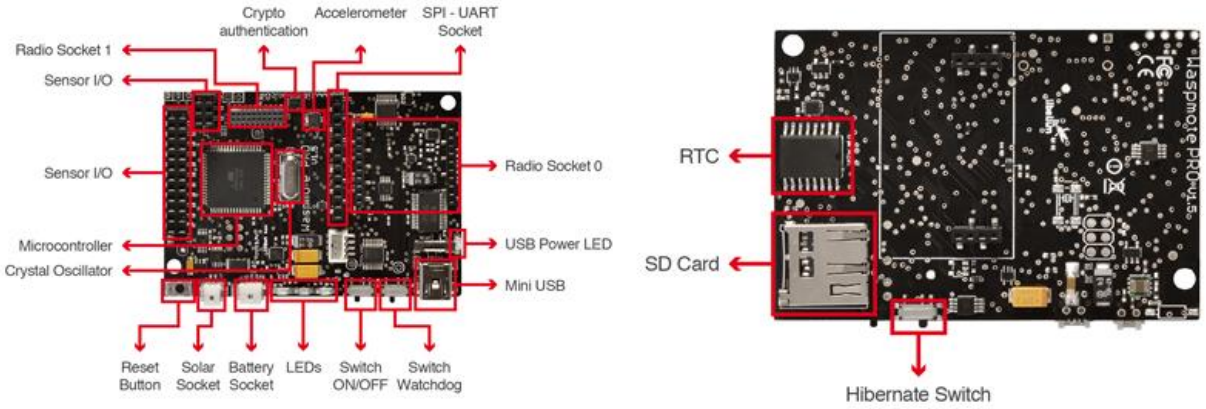

Ilustración 15: Parte Superior /inferior Waspmote y Socket disponibles

Como se ha comentado anteriormente Waspmote se basa en una arquitectura modular, el usuario final puede integrar los elementos en función de las necesidades requeridas. Se puede dividir estos módulos en cinco bloques generales:

- Módulos ZigBee/802.15.4.
- Módulo GSM/GPRS.
- Módulo GPS.
- Módulos Sensoriales.
- Módulo de almacenamiento: SD Memory Card.

Puntos interesantes que destacar de Waspmote:

- Medición en tiempo Real, gracias a RTC integrada en Waspmote.
- API que facilita la programación de las aplicaciones, cada sensor lleva su propia función. API desarrollado en C/C++.
- Sencillo de desplegar en campo y diferentes topologías de red gracias a los protocolos de Zigbee.
- Libelium aporta una enorme cantidad de información, manuales, ejemplos… para ayudar al cliente a desarrollar la red de Waspmote.

Los Waspmote están pensados para trabajar en campo, programados para la recolección de datos y enviados a un Gateway. Libelium ofrece en este momento en el mercado dos tipo de Gateway :

- Waspmote Gateway:

Gateway es la puerta de acceso, puente entre los datos de la red sensorial y el equipo receptor, donde se realizará la monitorización de los mismos.

Waspmote Gateway es un módulo receptor de datos, integra in zocalo xBee y un conversor FTDI(emula puerto serie , convierte de RS232-USB).

conectándolo en el ordenador y trabajando con API de Libelium se podrá recibir los datos por puerto serie.

Ilustración 16: Waspmote Gateway

- Meshlium
  En ,grosso modo, Meshlium es el servidor de desarrollado por Libelium tiene la capacidad de realizar las siguientes funciones :
  - Recibir los datos enviados desde Waspmote, mediante diferentes medios de trasmisión como XBee / LoRa / WiFi / 3G / GPRS.
  - Analiza las tramas.
  - Almacena en una base de datos local.
  - Almacena en base de datos Externa.
  - Interfaz Web propia donde se pude controlar la lista de sensores, filtrar database para búsqueda de datos , localización de waspmote en campo , estado de Waspmote…
  - Permite el acceso desde el exterior de forma segura.

En definitiva, cumple con todas las características que te puede llegar a dar un servidor, además de tener un perfecto acabado en todos los sentidos.

Ilustración 17:Libelium Meshlium

Con Waspmote no es posible realizar una comparación de precios, su producto esta desarrollado listo para la creación de la red y su instalación. Este producto es interesante por el carácter orientativo que aporta a este proyecto, no tiene sentido la comparativa de los precios.

Simplemente el precio del servidor supera los 15000€, el añadir cada nodo "Waspmote "sin contar con los sensores y comunicación mediante RF el precio de cada nodo es de 180€.Los sensores utilizados podrán ser los mismo que se ha utilizado en el proyecto actual.El importe final supera los 2000€.

Tras este análisis de mercado es posible ver hacia donde se está orientando los productos sacados al mercado en este sector, se podría seguir nombrando productos como shield para Arduino, placas de desarrollo, productos específicos para la monitorización, pero los productos de bajo coste siguen la misma dirección de los anteriores productos.

Con esta búsqueda se ha podido obtener ideas de la situación del mercado actual y poder conseguir una solución a nuestro proyecto, consiguiendo superar los requisitos impuestos, un producto flexible y económico.

3.3 COMPETENCIAS DEL PROYECTO

Para la realización del proyecto es necesario tener una planificación, unos procedimientos y sobre todo un propósito el cual alcanzar y determinar unos objetivos. Estos anteriores puntos son de vital importancia, hay todos unos procedimientos o "buenas tareas" los cuales realizar antes de comenzar un proyecto, ayudan a organizar de cara a la realización de un proyecto.

Una buena forma de comenzar es realizarse una serie cuestiones, esto no hace referencia a una planificación, es más una forma de anticiparse a sucesos, descartar propuestas incluso llegar a descartar el proyecto por no ser viable.

CUESTIONES BASICAS EN UN PROYECTO	
¿QUE?	La naturaleza del proyecto como ya sea comentado anteriormente reside en la realización de un sistema para solventar la necesidad de monitorizar y controlar un sistema de riego.
¿POR QUE?	Hay diferentes sistemas en el mercado los cuales pueden cubrir nuestra necesidad de poder controlar nuestro sistema de regadío, pero estas tienen varios puntos desfavorables: • Una vez que compramos algún sistema de alguna marca digamos que estaríamos ligados a ellos, no ofreciéndonos una oportunidad de escalamiento de nuestro sistema o modificaciones. • También se debe de ver desde un punto de vista económico, estos sistemas tienen un alto coste.
¿PARA QUE?	El objetivo final es ofrecer un sistema de calidad a un bajo precio, además de ofrecer una alta estabilidad y flexibilidad y no olvidando la seguridad.
¿CUÁNDO?	También es necesario pronosticar el tiempo que se dispone para hacer este proyecto. Se debería tener un prototipo antes de la llegada del mes de Julio de 2016. Eso nos deja un intervalo de 7meses desde el comienzo del mismo.
¿DONDE?	La ubicación física del proyecto será un huerto el cual es propiedad de la UPCT(Universidad Politécnica de Cartagenera).
¿CON QUE?	Para comienzo de este proyecto se pudo reutilizar algunos dispositivos hardware de un proyecto anterior realizado por Alex Sánchez García

CUESTIONES BASICAS EN UN PROYECTO	
	(desarrollo de un sistema electrónico para la monitorización de variables ambientales en huertos urbanos) De este provecho se ha podido recuperar los siguientes componentes: - 3 Arduino s UNO - 4 módulos RF - 1 DTH22 - 2 sensores de irradiación - Reloj en tiempo real

Tabla 4:Cuestiones básicas de un proyecto

A partir de este estudio y preguntarnos cuales serán nuestras necesidades es posible definir las partes que tendrá nuestro proyecto.

El proyecto es dividido en dos grandes bloques:

- **Hardware:** Esta es una parte fundamental, será donde se encuentran todos los dispositivos, sensores, actuadores y microcontroladores seleccionados.

- **Software:** Sera la parte encargada de que el usuario pueda interactuar con el sistema.

Para la selección de los componentes a utilizar se debe tener en cuenta una serie de requisitos mínimos que han de cumplirse. Los cuáles serán resumidos en la siguiente tabla.

Nº	Función
1	Mediciones de temperatura del sustrato en cada nodo
2	Mediciones de temperatura del ambiente en cada nodo
3	Mediciones de humedad del sustrato en cada nodo
4	Mediciones de humedad del ambiente en cada nodo
5	Medición de radiación Solar
6	Control de Válvulas

7	Control remoto mediante interfaz web
8	Almacenamiento de variables en base de datos
9	Visualizador de gráficas de los datos registrados

Tabla 5:Requisitos mínimos que han de cumplirse

CAPÍTULO 4

4 HARDWARE EMPLEADO

A continuación, se describen las principales características de los elementos Hardware utilizado para la elaboración de este proyecto. En la carpeta de anexos se incluyen los datasheet con descripciones más detalladas de cada uno de ellos.

4.1 ARDUINO

En este apartado se mostrarán los microcontroladores de Arduino seleccionados para este proyecto, y serán comparados con las diferentes placas ofrecidas por Arduino.

I. UNO

Arduino UNO es la placa más conocida de la plataforma Arduino. En parte su éxito se debe a su reducido coste. Es el modelo de referencia para la compañía, además tiene compatibilidad con la mayoría de shields disponibles en el mercado.

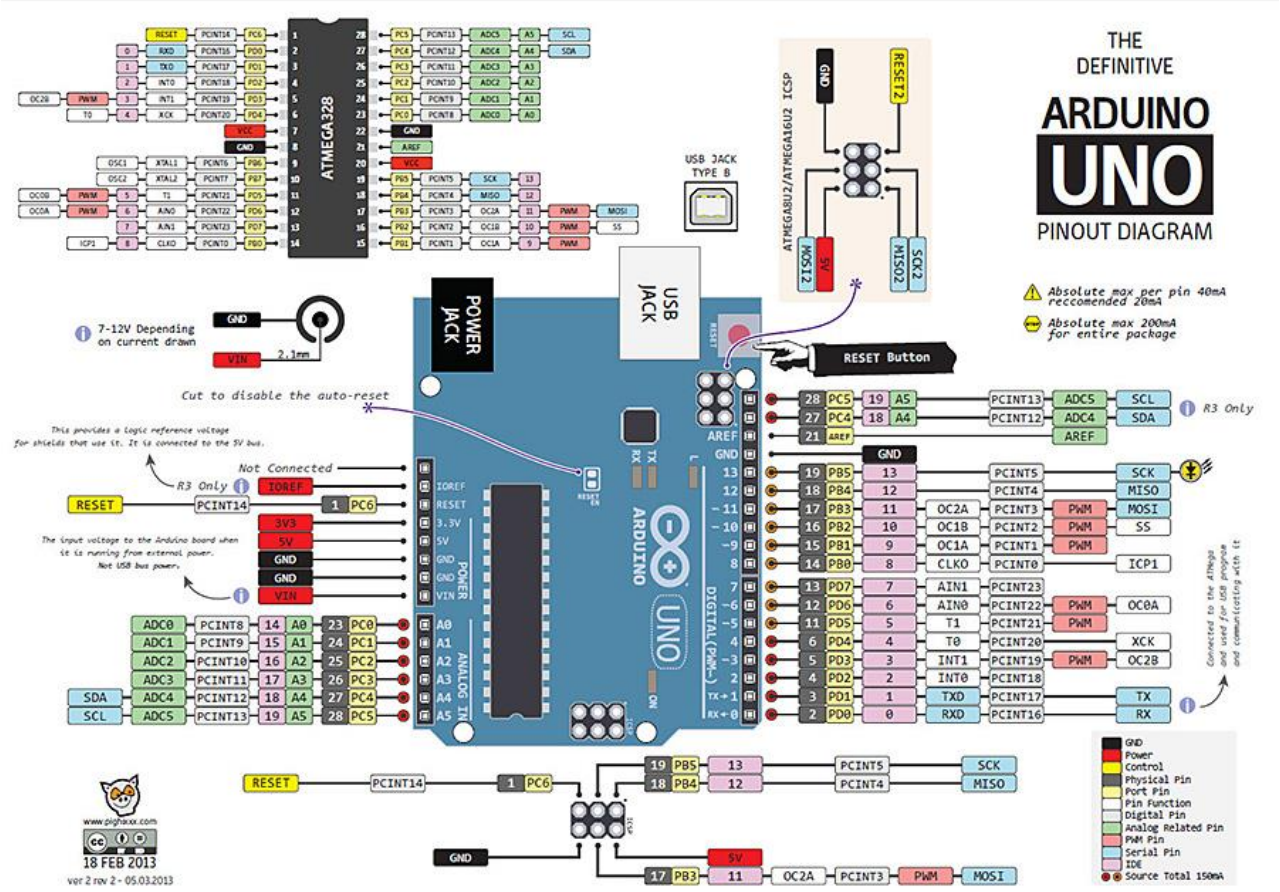

Ilustración 18:Pinout Arduino Un

Sus características principales son:

Microcontrolador	ATmega328
Voltaje de operación	5V
Voltaje de entrada (Recomendado)	7-12V
Voltaje de entrada (Límite)	6-20V
Digital I/O	14
PWM Digital I/O	6
Entradas analógicas	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Memoria Flash	32 KB (0,5 KB ocupados por el bootloader)
SRAM	2 KB

EEPROM	1 KB
Frecuencia de reloj	16 MHz
Largo	68.6 mm
Ancho	53.4 mm
Peso	25 g

Ilustración 19:Resumen características Arduino UNO

II. MEGA

Arduino Mega 2560 placa basada en el microprocesador Atmega 2560, ofrece una mayor memoria RAM, capacidad y pines que Arduino UNO.

También nos ofrece la cualidad de ser compatible con la mayoría de los shields diseñados para el Arduino Uno.

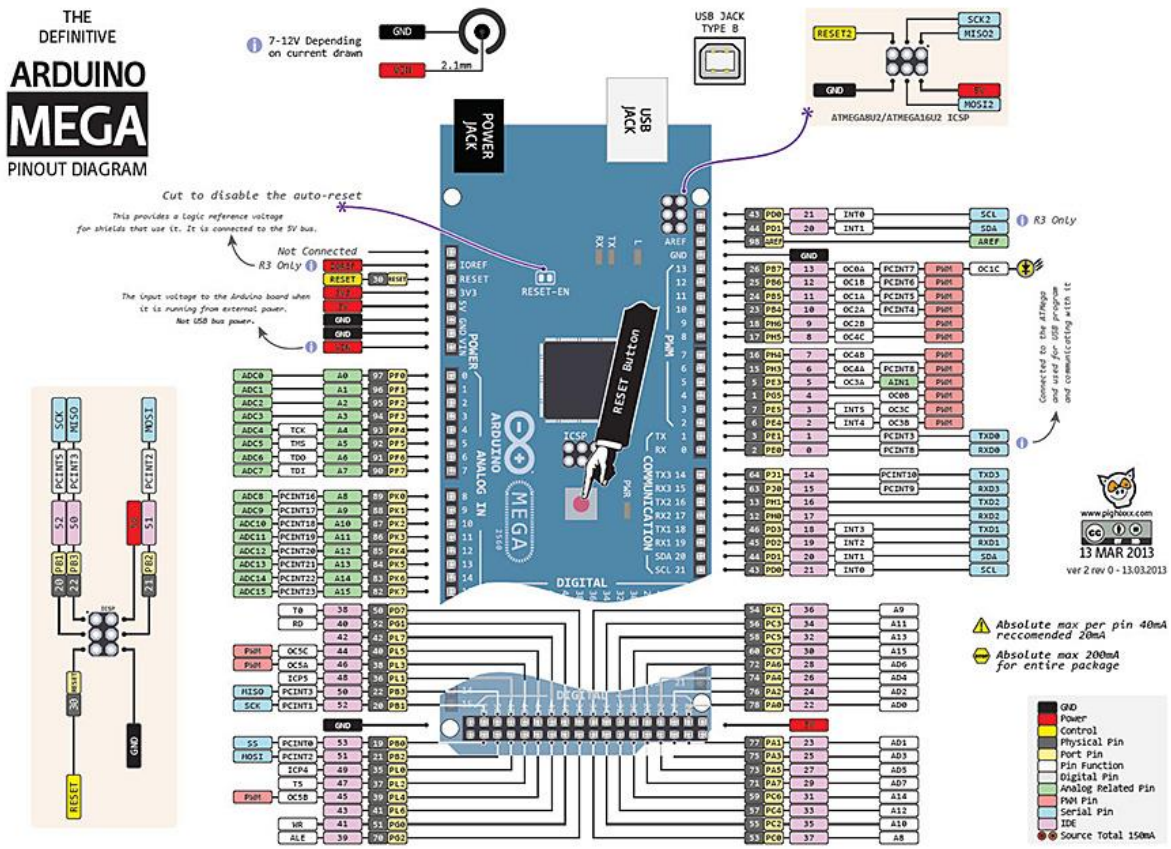

Ilustración 20:Pinout Arduino Mega

Las características de Arduino MEGA 2560 son:

Microcontrolador	ATmega2560
Voltaje de operación	5V
Voltaje de entrada recomendado	7-12V
Voltaje de entrada (límites)	6-20V
Pines digitales E/S	54 (15 PWM disponibles)
Pines de entrada analógica	16
Consumo por pin E/S	20mA
Consumo del pin 3.3V	50mA
Memoria flash	256kB (8kB empleados por el bootloader)
SRAM	8kB
EEPROM	4kB
Frecuencia de reloj	16MHz
Largo	101.52mm
Ancho	53.3mm
Peso	37g

Ilustración 21:Resumen características Arduino Mega Rev3

A continuación, se muestra una tabla con la comparación de todos los Arduino s que estuvieron contemplados en este proyecto:

	Voltaje DeOperacion	***Voltaje De Alimentación***	***Flash [KB]***	***SRAM [KB]***	***I/O digitales***	***Pines Analógicos (I/O)***	***EEPROM [KB]***	***Compatibilidad Con Shields***	***Precio [€]***
Uno	5v	7-12v	32	2	14/6	6/0	1	Excelente	19.95
Pro Mini	5v	3.35-12v	32	2	14/6	6/0	1	N/A	12

Leonardo	5v	7-12v	32	2.5	20/7	12/0	1	Decente (diferencias de pines)	18.55
Micro	5v	5v	32	2.5	20/7	12/0	1	N/A	20
Mega 2560	5v	7-12v	256	8	54/15	16/0	4	Buena	34.50
Mega ADK	5v	7-12v	256	8	54/15	16/0	4	Buena	43.90
Due	3.3v	7-12v	512	96	54/12	12/2	4	Mala	38

Ilustración 22:Comparativa características Arduino s

Para este proyecto se ha optado por la elección de Arduino Uno como opción para los nodos, que serán encargados de la recopilación de los datos y enviárselos al sumidero. Como se ha podido observar nos ofrece las entradas necesarias además de dejar espacio para un escalamiento de nuestro proyecto, es decir dejar entradas o salidas reservadas para una futura ampliación, le seguiría Arduino Leonardo, pero este tiene una menor memoria flash, inicialmente 4kB empleados por el bootloader.

Como sumidero, se ha optado por la selección de Arduino MEGA, el cual nos ofrece una gran potencia de cómputo, nos aporta una alta escalabilidad por su número de I/O digitales y analógicas, además de la capacidad de memoria y el bajo precio [8].

4.2 RASPBERRY PI

En este apartado se explica Raspberry Pi, el cual será encargado de almacenar los datos de distintas maneras, almacenándolos en la base de datos temporalmente para mostrarlos gráficamente mediante una interfaz web, en la memoria microSD en formato de TXT , además de subirlos a Dropbox de forma automatizada cada hora .

Raspberry Pi será la encargada de alojar el servidor web, nos da la oportunidad de crear una topología distribuida, en la que Raspberry Pi será el servidor y la parte de monitorización y control recaerá en Arduino.

En el proyecto antecesor a este, mencionado en el punto 1.5, se utilizó un Arduino YUN el cual hacía de sumidero y servidor.

La decisión de cambiar Arduino YUN por una Raspberry Pi y un Arduino MEGA, reside en no sobrecargar el Arduino y además nos ofrecerá una alta escalabilidad, por ejemplo, a la hora de añadir efectos a nuestro server web, además de la versatilidad que nos ofrecerá Raspberry Pi debido a su sistema operativo Debían, conseguimos una mayor seguridad y un entorno agradable puesto que es un ordenador a pequeña escala.

Raspberry Pi es un ordenando del tamaño de una tarjeta de crédito , una placa de computo de bajo costo ,la cual fue desarrollada en Reino Unido por la fundación Raspberry Pi, su objetivo era estimular en los niños el sentimiento por la ciencia de la computación y que comenzaran a aprender programación .En apenas dos años después de su lanzamiento alcanzo el millón de ventas , ha tenido una gran repercusión formando parte de una gran cantidad de proyectos gracias a ser una plataforma de hardware open Source.

A diferencia de Arduino cuenta con un microprocesador en cambio Arduino cuenta con un microcontrolador.

Los elementos básicos del modelo B+ son las siguientes:

- Un procesador BCM 2835 con 512 MB de RAM
- Consumo bajo debido a los reguladores lineales, el consumo es reducido a 0,5 -1 W
- Numero de pines GPIO 40
- Cuatro puertos USB2.0
- Ranura para tarjeta microSD
- Sistema operativo Linux
- Tipo de procesador ARM710
- Velocidad de procesador 700MHz

- Capacidad de memoria RAM 512MB

La fundación Raspberry Pi [9] da soporte para las descargas de las distribuciones para arquitectura ARM.

Ilustración 23:Raspberry Pi modelo B+

Es fácil comprender el éxito de este dispositivo, ofrece una gran variedad de ventajas como un reducido coste, reducido consumo y espacio, gran comunidad de usuarios y desarrollos Open Source , además la capacidad de poder adaptarse a las necesidades de cualquier persona , puesto que es un sistema Linux por lo tanto puedes programar en el lenguaje que más cómodo se sienta el usuario o en diferentes lenguajes para una misma aplicación .

4.3 SENSORES Y ACTUADORES

I. TRANSCEPTOR NRF24L01

Para la comunicación entre los distintos nodos se ha decidido utilizar el módulo de radiofrecuencia NRF 24L01. Aunque el proyecto comenzó con la idea de utilizar los módulos ZigBee, pero con el desarrollo del proyecto se vieron las ventajas de reutilizar los módulos NRF24L01 del antiguo proyecto , por distintas razones :

- Precio razonablemente más bajo que los productos de xbee.
- Un consumo en modo sleep menor.
- Un consumo en modo de transmisión mucho menor.
- NFR 24L01 tiene menor alcance, pero no es un problema los nodos se encuentran en un espacio abierto y el sumidero estará relativamente cerca.

Son módulos de radiofrecuencia a 2.4Ghz que se basan en el chip de Nordic Semiconductor NRF24L01-PA-LNA. Nordic NRF24L01-PA-LNA integra un completo transceptor RF de 2,4 GHz, un sintetizador de RF y toda la lógica de banda base incluyendo un acelerador de protocolo por hardware Enhanced ShockBurst con una interfaz SPI de alta velocidad para el controlador de la aplicación.

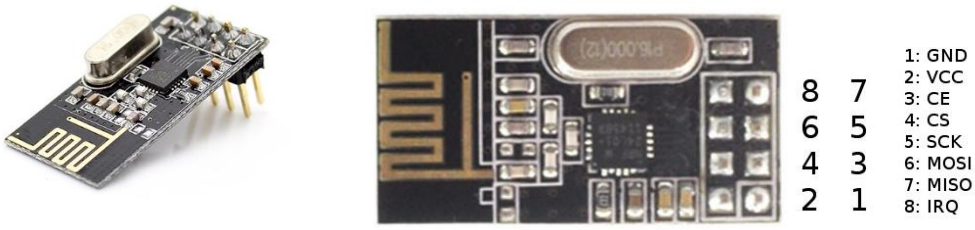

Ilustración 24: Transceptor NRF 24L01 / Pin-out del NRF24L01

El rango de alcance es variable puesto que depende de la situación en la cual este trabajando. Normalmente a estos módulos se le asignan una distancia de alcance de 20 a 25 metros sin obstáculos, disminuyendo drásticamente si se colocan obstáculos de por medio, puesto que la potencia de emisión es muy baja, pero se puede dar el caso de añadir antenas amplificadoras.

Estos módulos incluyen, no solo un adaptador de antena, además llevan integrado un amplificador de señal para potenciar la emisión y recepción de señal, con lo que la distancia útil de transmisión puede alcanzar los 100m.

Ilustración 25:Antena amplificadora NRF24l01

Se comunicarán con Arduino mediante puerto SPI, lo que permitirá una comunicación de alta velocidad entre ambos dispositivos.

Como se ve en el Pin-out del NRF24L0 la comunicación con el puerto serie se realiza a través de los pines MISO y MOSI. Arduino mandará datos a través del MOSI y los recibirá de la radio a través del MISO. Los pines esclavos son los que determinan quién puede usar el puerto SPI en cada momento.

- **CE/CSN:** Son los pines esclavos del dispositivo NRF24L01.Son activos a nivel bajo, lo que significa que a nivel de software se pondrán a cero cada vez que se vaya a trasmitir.

- **MISO/MOSI:** Son los pines de comunicación del SPI por los que se comunican los dispositivos de radio y los Arduitos.

- **SCK:** Es el pin por el que se trasmite el pulso del reloj por el puerto serie, es decir el pin encargado de sincronizar los diferentes dispositivos que compartan el puerto SPI.

- **IRQ:** Es un pin de las radios vinculado a las interrupciones: Dado que no es necesario su uso para este sistema, se dejará sin conectar.

Se dispone de un socket nrf24l01, nos permite colocar el módulo de forma más sencilla, además, de adaptar la tensión de 5V a 3.3V.

Ilustración 26:Socket módulo NRF 24L01

Es Importante conocer conexionado de los pines, puesto que la conexión de las radios a través de estos pines con los Arduinos cambia de un Arduino a otro. Esto se debe a que cada tipo de placa Arduino tiene reservado unos pines específicos para la comunicación SPI.

	ARDUINO UNO	**ARDUINO MEGA**
GND	GND	GND
VCC	5 V	5 V
CE	Variable	Variable
CSN	Variable	Variable
SCK	13	52
MOSI	11	51
MISO	12	50
IRQ	No utilizado	No utilizado

Tabla 6: Comparación Pines Arduino UNO y MEGA para NRF 24L01

Las patillas CE y CSN , son seleccionadas variables puesto que son seleccionadas durante la programación.

Son los denominado pines esclavos .

Las librerías ofrecidas para la comunicación correcta de Arduino con los módulos NRF 24L01.

```
#include <SPI.h>      // Necesaria para comunicación

#include "nRF24L01.h" //librería

#include "RF24.h"       //librería
```

Gracias a la librería no es necesario el control de la comunicación ella se encarga del control SPI [10].

II. RELOJ EN TIEMPO REAL DS1307

Es una placa de diseño compacto, la cual contiene un reloj en tiempo real (RTC), incluye una pila CR1225, la cual es capaz de durar un mínimo de 9 años (típicamente 17 años).

Para acceder al DS1307 debe ser a través del protocolo I2C. Utiliza un oscilador de cristal, para ser lo más preciso posible, aunque es sensible a cambios por la temperatura y los campos magnéticos, dispone de un error de más menos un minuto por mes, aunque es posible rectificar la hora ajustándola manualmente, cargándola al RTC DS1307.

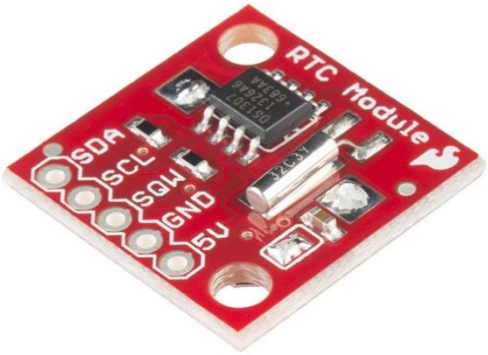

Ilustración 27: Reloj en tiempo real DS1307

Algunas de las características más importantes son las siguientes.

- interfaz I2C (2 hilos)
- Hora: Minutos: Segundos
- Día de la semana , Mes, Fecha - Año
- la compensación del año bisiesto
- Precisa calendario hasta el año 2100
- reserva de la batería incluida
- pin de salida de 1 Hz
- 56 bytes de memoria no volátil disponible para el usuario
- Dimensiones: 20x20mm

Este módulo posee diferentes conexiones para los diferentes microcontroladores conectados:

ARDUINO	**UNO**	**MEGA**
GND	GND	GND
VCC	5V	5V
SDA	A4	SDA
SCL	A5	SCL
SQW	---	----

Tabla 7: Conexiones de módulo DS1307 a Arduino s

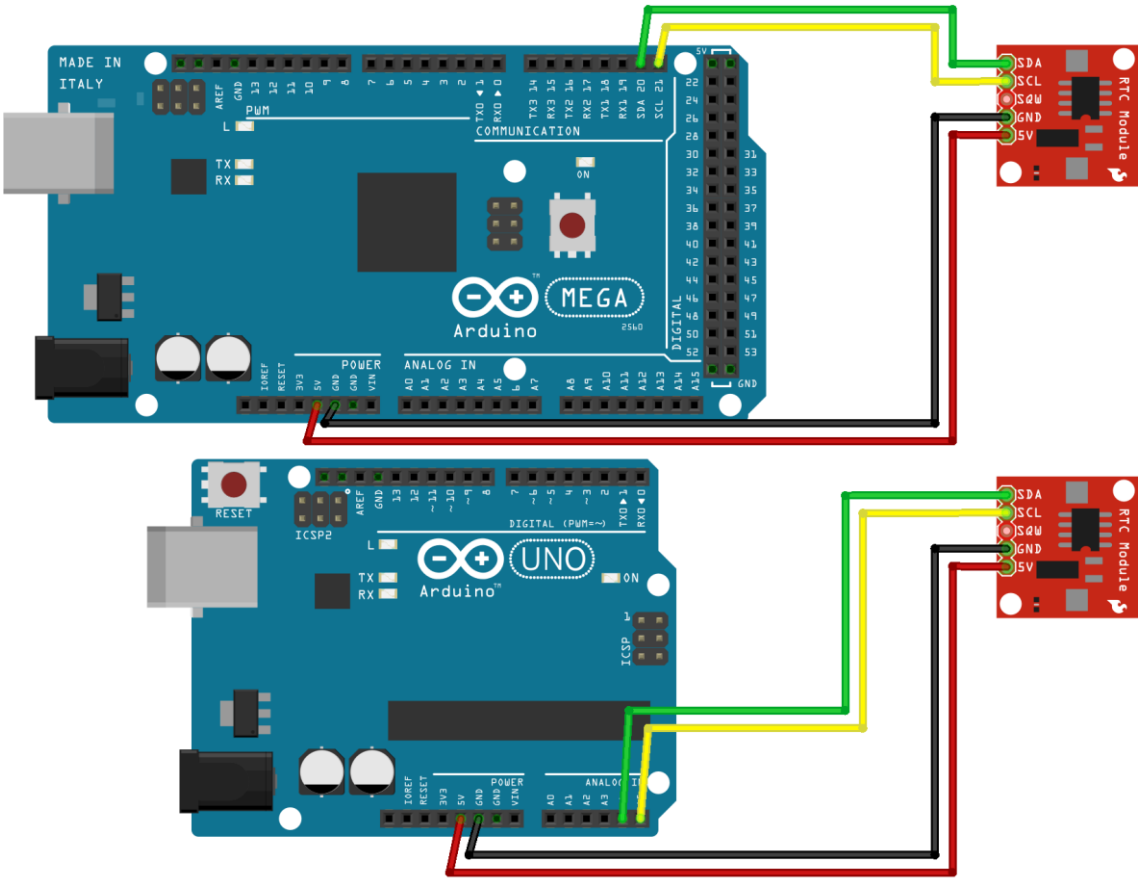

Ilustración 28:Pinout RTC Arduino (MEGA/UNO)

En este caso el RTC se conectará sobre el nodo central que será Arduino MEGA, por lo tanto, será conectado en los pines digitales 20(SDA) y 21(SCL).

III. SENSOR DE TEMPERATURA Y HUMEDAD. DHT22

Es uno de los gadgets para Arduino más conocidos debido a que el mismo dispositivo es capaz de medir dos parámetros, un sensor capacitivo se encarga de medir la Humedad y para la temperatura contiene un termistor. Además, tiene una gran ventaja, la salida es digital, por lo tanto, será conectado a pines digitales.

En el mercado es posible encontrar dos modelos, el hermano pequeño de la familia DHT11, el cual nos ofrece unas peores características técnicas. El DHT22 es el modelo superior, por lo tanto, un precio superior.

Ambos modelos se presentan en un cuerpo de plástico, se pueden diferenciar por el color que les caracteriza, DTH22 (blanco) y DTH11 (azul).

Las principales diferencias son las que nos ayudan a seleccionar como dispositivo a utilizar en este proyecto el sensor DHT22.

Características de cada uno de los sensores:		
Parámetro	DHT11	DHT22
Alimentación	3Vdc ≤ Vcc ≤ 5Vdc	3.3Vdc ≤ Vcc ≤ 6Vdc
Señal de Salida	Digital	Digital
Rango de medida Temperatura	De 0 a 50 °C	De -40°C a 80 °C
Precisión Temperatura	±2 ºC	<±0.5 % °C
Resolución Temperatura	1°C	0.1°C
Rango de medida Humedad	De 20% a 90% RH	De 0 a 100% RH
Precisión Humedad	4% RH	2% RH
Resolución Humedad	1%RH	0.1%RH
Tiempo de respuesta	1s	2s
Tamaño	12 x 15.5 x 5.5mm	14 x 18 x 5.5mm
Precio	1€	3€

Tabla 8:Comparativa DHT22 vs DHT11

Es posible descartar la diferencia en el precio y otra diferencia fundamental como el tiempo de respuesta, puesto que en el proyecto no es una de las grandes prioridades, se antepone la resolución que ofrece el DTH22, ya que no es posible obtener fracciones decimales con el DTH11.

También se tiene en cuenta el rango de trabajo, no se podrían medir temperaturas negativas con DTH11.Por lo tanto se escoge el DTH22 por ser más estable y confiable.

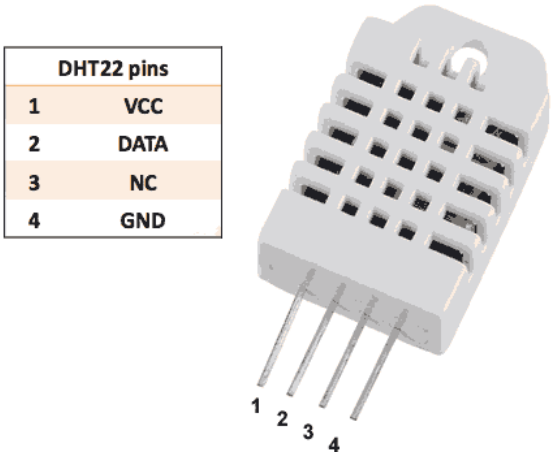

Ilustración 29:Pinout dth22

Del montaje realizado, se puede destacar que el pin 3 no es utilizado, además entre la salida de datos y la alimentación a de colocarse una resistencia pull-up de 10KΩ.

IV. SENSOR DE IRRADIACIÓN SOLAR S1087-01

Fotodiodo utilizado para medir la irradiación solar y poder obtener la longitud de onda la cual está irradiando en nuestro huerto.

El encapsulado ofrece una salida proporcional a la luz que le llega, es un encapsulado cerámico impermeable a la luz.

Ilustración 30:Sensor S1087-01

El espectro de luz que abarca el modelo S1087-01 tiene un rango de 320 a 1100 nm, el rango en el cual nuestra plantación realizaran la fotosíntesis es el espectro visible el cual está comprendido entre mayores de 400nm y menores de 700nm.

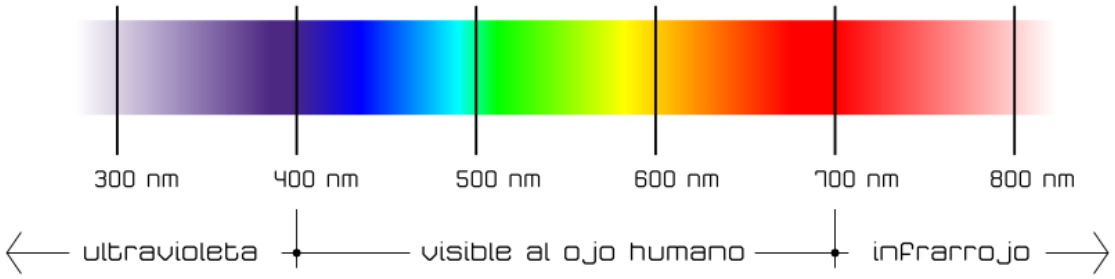

Ilustración 31:Espectro Visible al ojo humano

Usar el sensor es relativamente sencillo, solamente es necesario alimentarlo y realizar un pequeño acondicionamiento. Para poder conocer la longitud de onda se utiliza la siguiente tabla, el sensor nos proporcionara un valor equivalente a la recepción de fotones, tomando la tabla se debe entrar por el eje de ordenadas con el valor obtenido hasta cortar con la recta del sensor S1087-01.

Para el espectro de luz visible se obtienen valores de 0.2 a 0.4 [A/W].

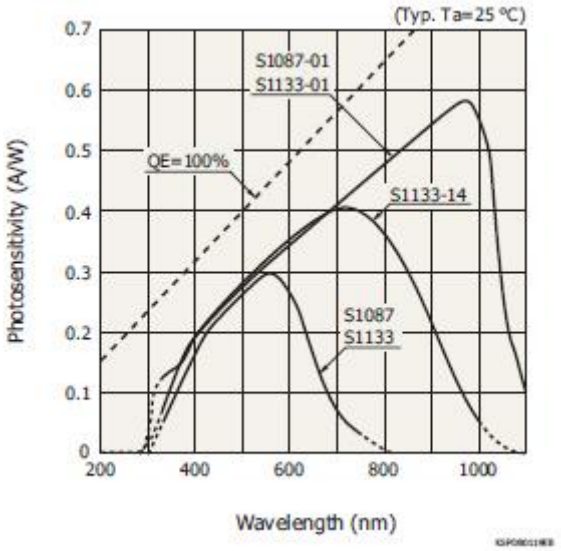

Ilustración 32: S1087/S1087-01 Photosensitivity vs. Wavelength

V. SENSOR DE HUMEDAD Y TEMPERATURA SHT10

Para medir la temperatura y la humedad del sustrato de la planta se opta por un módulo SHT10 capaz de medir ambas variables, fabricado por Sensirion, incluye una carcasa de acero inoxidable porosa que lo protegerá del óxido y condiciones adversas. Esta preparado para ser sumergido al agua , puesto que el agua no pasa solamente pasa el aire para poder leer la humedad, pero es recomendable que no esté en el agua más de una hora.

Para la humedad del suelo se pudo optar por otro tipo de sensores especiales para Arduino, como el módulo YL_69 y para la temperatura el sensor digital DS18D20, pero se decide utilizar el sensor SHT10 por tener unificadas ambas variables y por la precisión que nos ofrece comparándolo con los anteriores módulos, además la humedad del sustrato es la variable más crítica de nuestro proyecto, puesto que será la variable a controlar.

Ilustración 33:Sensor de humedad y temperatura SHT10

Las características de este sensor son las siguientes:

Características SHT10	
Consumo medio	0.15mW
Rango de humedad	0 - 100%RH
Rango de temperatura	-40 - 120°C
Precisión (humedad)	±4.5%RH
Precisión (temperatura)	±0.5°C
Tamaño	49mm x 14mm

Tabla 9:Características SHT10

El conexionado del sensor al microcontrolador se realiza a través de las entradas digitales. El sensor cuenta con cuatro cables VCC, GND, DATA y CLOCK.

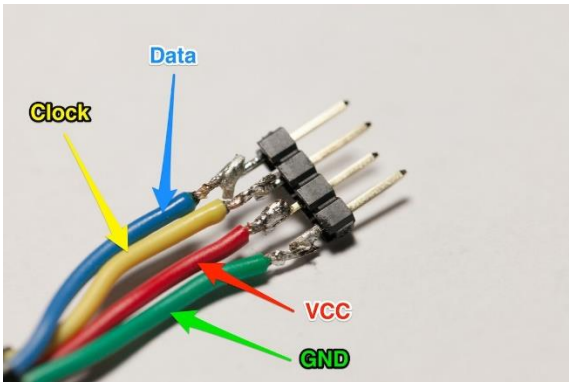

Ilustración 34:Cables sht10

Se recomienda colocar una resistencia pull-up de 10k entre VCC y DATA, es el pequeño acondicionamiento necesario para el correcto funcionamiento el sensor.

VI. MÓDULOS RELÉS

Como actuadores directamente se utilizaran los relés, los cuales serán alimentados directamente por el Arduino MEGA , quien está encargado de realizar el control .Este control estará destinado a controlar la Humedad relativa que hay en el sustrato de la planta ,la cual es posible controlar mediante las válvulas. las cuáles serán alimentadas por los relés.

Se utiliza un módulo de cuatro canales, para dejar al proyecto poder tener capacidad de una ampliación en el futuro, ya sea un nodo más o una luz de emergencia a 220 V … Los relés serán encargados de alimentar las válvulas que se encargarán de llevar el nivel de humedad del sustrato a los niveles requeridos por el cliente, los mismo serán seleccionados por la interfaz web.

Características:

- Voltaje alimentación: +5V
- Voltaje accionamiento: +5V
- Canales opto-acoplados.
- Led de estado en cada canal.
- Consumo por canal: 80mA
- Relés: AC250V 10A ; DC30V 10A
- Dimensiones módulo 4CH: 75.09x54.91mm

Ilustración 35:Módulo relés

La conexión de este dispositivo es muy sencilla, por un lado, simplemente se debe configurar una patilla digital como salida y activarla o desactivarla cuando convenga.

Como se ve en la ilustración anterior, la parte de continúa accionada digitalmente por Arduino la encontramos en la derecha, las conexiones son las siguientes.

PIN	Arduino
GND	GND
VCC	5 V
IN	Salida digitan Arduino MEGA

Tabla 10:Conexionado Relé

Por el otro lado se tiene la conexión de las válvulas las cuales están conectadas a 220V, el relé será el interruptor comandando por Arduino MEGA el cual cortará la alimentación a la Válvula.

CAPÍTULO 5

5 ARQUITECTURA SOFTWARE

5.1 INTRODUCCIÓN

Para llevar a cabo el desarrollo de este proyecto se ha tenido que utilizar diversos softwares, así como diferentes entornos de programación y programas necesarios para la interpretación de los lenguajes utilizados, además de diferentes programas para la realización de algunas etapas del proceso.

A continuación, se explica de forma resumida los distintos programas utilizados en el proyecto.

5.2 IDE ARDUINO

Para el desarrollo de los “sketch “de Arduino se ha utilizado el IDE (Integrated Development Environment) de Arduino. El software es gratuito que puede ser descargado de la página oficial de Arduino [8]. Cabe destacar que se puede utilizar otras herramientas de editor de texto y compiladores de microcontroladores, la decisión de utilizar la IDE propia de Arduino además de ser la oficial el entorno grafico es bastante cómodo y tiene una gran comunidad de desarrollo detrás.

El funcionamiento es bien simple, lo primero es crear el “sketch” en el editor de texto, la estructura de programación será “C++” aunque también acepta “C”. Permite ir compilando para saber si se han cometido errores de sintaxis, lo cual es muy común en cualquier código extenso, siempre se pueden encontrar los típicos errores de punto y coma o un simple paréntesis .

Una vez que se puede asegurar que el código está bien escrito llega el paso deseado por todo programador compilación final y volcado de datos al microcontrolador pero no sin antes seleccionar el tipo de dispositivo y puerto COM el cual está conectada nuestra placa Arduino .

Además, el IDE cuenta con del uso de librerías lo cual facilita la programación. Algunas librerías bastante utilizadas están ya incorporadas como puede ser la utilización de memoria EEPROM, conexión Ethernet, la librería LiquidCrystal… Para este proyecto se ha tenido que añadir varias librerías como la de módulo de radio frecuencia nRF24l01, estas librerías han sido añadidas manualmente en la ruta "C:\Program Files (x86)\Arduino \libraries" estas librerías estarán incluidas en el anexo .

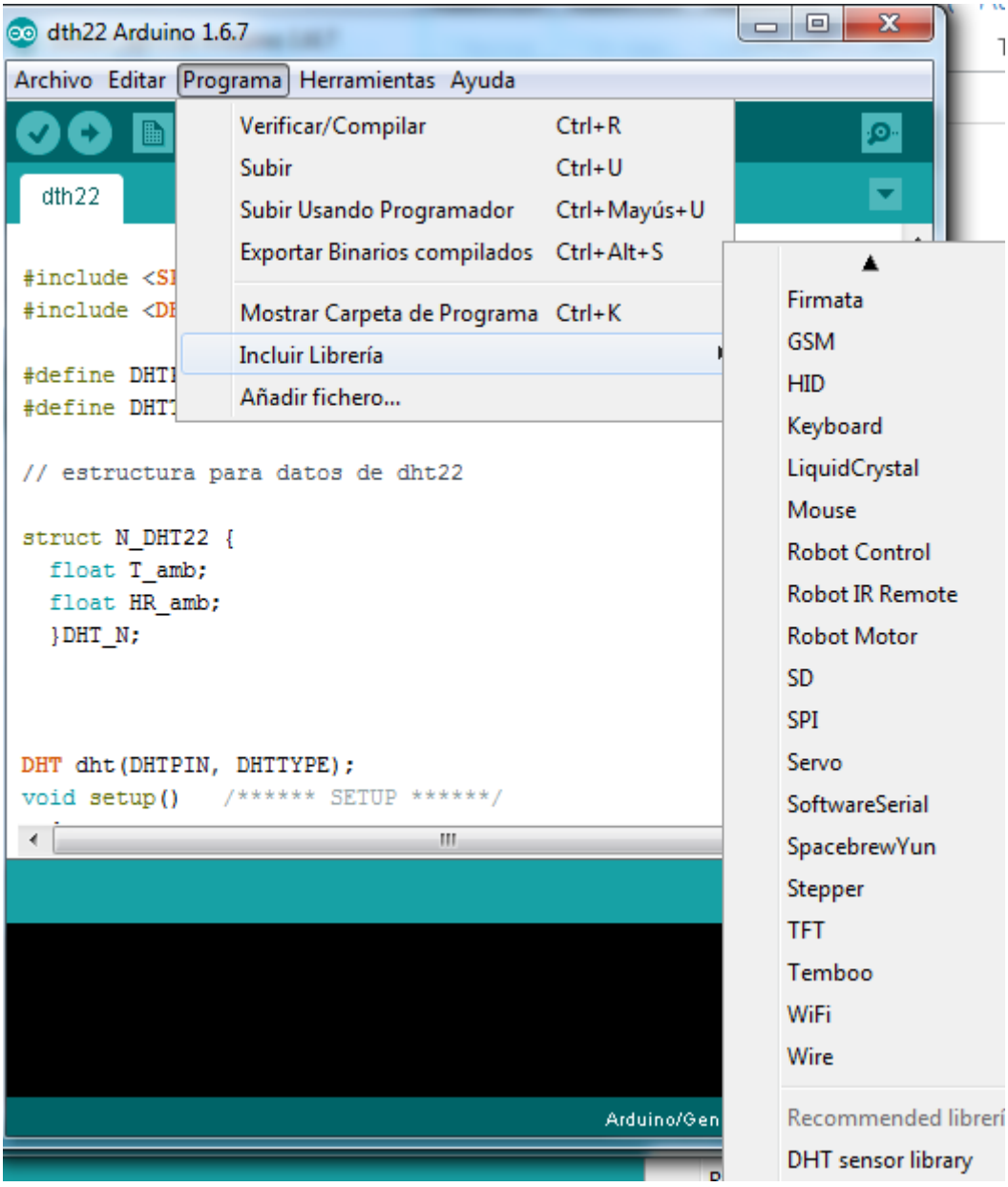

Ilustración 36:IDE Arduino

Una de las herramientas bastante utilizadas de Arduino para conseguir depurar errores es la utilización del monitor serial, enviando los datos hacia el USB de nuestra PC con el comando "Serial.print();" se pueden observar los datos que son enviados al actuador o recibiendo del sensor y observar si funciona correctamente.

5.3 CONEXIÓN REMOTA

Este punto es de bastante importancia para el avance del proyecto ya que nos facilita el acceso remoto al nuestra Raspberry Pi, esto conlleva poder acceder desde nuestra computadora sin tener que estar en el lugar que está colocada físicamente la Raspberry Pi y además no es necesario una segunda pantalla, un teclado y un ratón.

Gracias a SSH (Secure Shell), protocolo de comunicación que permite la conexión remotamente con cualquier máquina que soporte este protocolo en una red local, puede ser una debilidad el que cualquier persona pueda conectarse a nuestro servidor, aunque se puede solventar introduciendo una clave para su acceso.

Para la conexión remota se ha utilizado dos software distintos, Putty y Xming , son programas independientes.

I. PUTTY

Putty es un cliente SSH, es un software de código abierto el cual se puede descargar en la página oficial de Putty [11], en su versión para Windows, es un ejecutable.

Para la conexión mediante Putty se ha de tener activo el servicio SSH en la Raspberry Pi, por defecto viene activado, para iniciar la sesión es necesario introducir la IP asignada a la Raspberry Pi, está es fácil conseguirla pues solamente es necesario ingresar en la interface de configuración del módem y ver la IP asignada por DHCP.

Por comodidad como se pude observar en la imagen siguiente puedes guardar una sesión y cargarla, para no tener que ingresar constantemente la dirección IP, obviamente es necesario configurar una IP estática para la Raspberry Pi.

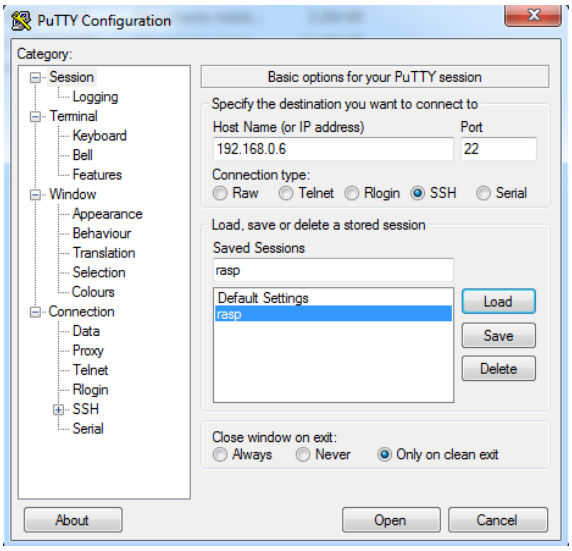

Ilustración 37:Interface configuración Putty

II. XMING

Xming es una aplicación cliente-servidor, para su utilización es necesario la instalación en Windows, una aplicación gratuita. con esta aplicación se puede gestionar el servidor usando un sistema gráfico llamado X11.

Xming transfiere la información gráfica a través del protocolo SSH, por lo que Xming lleva integrado Putty. Solamente se debe ingresar la IP de nuestra Raspberry Pi el usuario y la clave, que por defecto es User= "pi " y Password="raspberry".

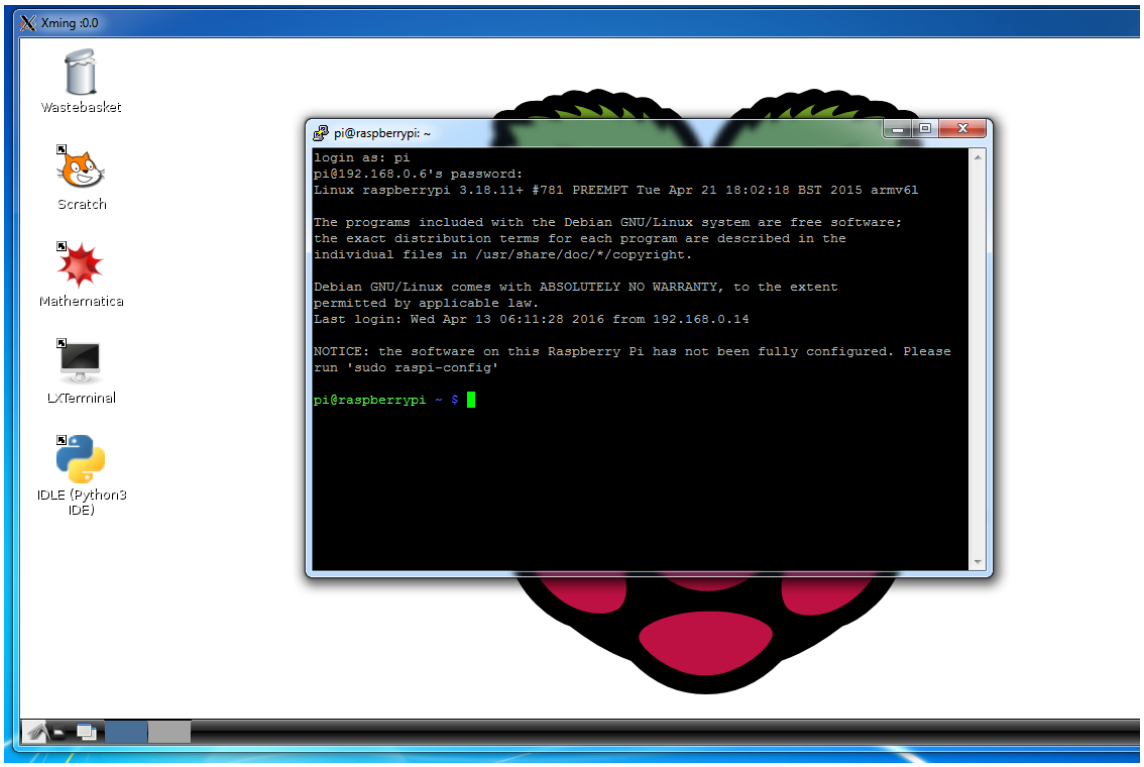

Ilustración 38:Comparación Xming vs Putty

La imagen anterior se puede observar claramente la ventaja que ofrece Xming con respecto a Putty. También cabe destacar que no es necesario el uso de Xming puesto que se puede controlar todo perfectamente desde la terminal ya que es una herramienta muy sencilla y tremendamente poderosa, permite interactuar con el SO sin la necesidad de la interfaz gráfica.

5.4 LAMP

LAMP es el acrónimo usado para definir la infraestructura de red que soportara nuestro servidor web.

Las herramientas que le dan el nombre de LAMP, **L**inux+**A**pache+**M**ySQL+**P**HP, no olvidemos que se está utilizando como sistema operativo Debían distribución de Linux.

No centrare mi atención en explicar la instalación de LAMP, Debido a la gran cantidad de tutoriales en internet y muy bien explicados. Yo personalmente recomiendo geekytheory , web en la cual me he apoyado mucho en los comienzos con este proyecto, gran comunidad en la cual se comparte conocimientos siempre defendiendo el código abierto.

En los siguientes apartados se describiran las herramientas sobre las cuales se levantaran el servidor.

Ilustración 39:LAMP

I. APACHE

Es un servidor de código abierto para plataformas UNIX, el cual implementa el protocolo HTTP, este protocolo de comunicación se sitúa en la capa de aplicación del modelo OSI.

Es uno de los servidores web más populares debido a su gran estabilidad y seguridad, algunas de sus grandes ventajas son:

- Soporta seguridad SSL y TLS
- Realiza la autenticación de datos con SGDB
- Consigue dar soporte a diferentes lenguajes como PHP y PYTHON

Apache será el que nos otorga la capacidad de compartir los archivos de forma segura de nuestra Raspberry Pi a Internet, servirá archivos HTML a través HTTP.

El directorio raíz por defecto una vez instalado Apache es "/var/www/".

Aquí se deberán colocar los archivos HTML ,PHP ,CSS ... Es importante darle permisos para poder mover archivos a esta carpeta puesto que en un principio solo tiene permisos de lectura.

II. PHP

PHP (Hypertext Pre-processor) , se considera uno de los lenguajes más flexibles y potentes que existen hoy en día. Es un lenguaje de programación del lado del servidor, esto quiere decir que el servidor será el encargado de interpretarlo y no el PC el cual se conecte al servidor el solo vera el código HTML generado. ¿El servidor sabe lo que tiene que leer puesto que el código de PHP está delimitado por las etiquetas "<? PHP" y "?>".

Por lo tanto, la persona que se conecte a nuestra Web solo podrá ver el código HTML enviado mediante protocolo HTTP.

Esto hace que la programación con PHP sea segura y confiable, si no fuera el código invisible de cara al cliente podría ver la contraseña de acceso a nuestra base de datos, escrita en el código PHP sin ningún tipo de cifrado.

Algunas de las características más importantes de PHP

- Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- Es un lenguaje muy fácil de aprender.
- El código fuente escrito en PHP es invisible al navegador web y al cliente, ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador.
- Posee una amplia documentación en su sitio web oficial.
- Es un software libre.
- Permite aplicar técnicas de programación orientada a objetos.
- No requiere definición de tipos de variables, aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.

III. MYSQL

MySQL es un gestor de bases de datos, este nos ofrecerá los servicios de almacenamiento y control de tablas en los cuales se guardarán los datos recolectados por los nodos y poder mostrarlos al cliente mediante una interfaz web.

Una de las características más interesantes de MySQL es que permite recurrir a bases de datos multiusuario a través de la web y en diferentes lenguajes de programación que se adaptan a diferentes necesidades y requerimientos, de los cuales en este proyecto se han utilizado PYTHON y PHP.

Manejar Mysql mediante comandos en la terminal de Debían resulta demasiado pesado se toma la opción de instalar phpMyAdmin. Es una herramienta escrita en PHP , su intención es manejar la administraron de MySQL a través de una interfaz gráfica , la cual nos facilita mucho control de la base de datos , poder cambiar valores de la base de datos en línea , acceso a la base de datos desde fuera de la red LAN sin estar en el servidor.

5.5 SERVIDOR FTP

La idea de montar un servidor FTP, es poder tener un total acceso a la Raspberry Pi, en el caso de querer actualizar la web sin tener que entrar al servidor, no tener que ir al lugar el cual está situada, poder hacer cambios desde cualquier lugar con una conexión a internet.

Se procede a la instalación del servidor vsftpd en Raspberry pi con el siguiente comando:

```
pi@raspberrypi ~ $ sudo apt-get install vsftpd
```

Una vez instalado el servidor será necesario la instalación de un cliente FTP , el cliente seleccionado sera Filezilla, software de descarga gratuita desde la página web de Filezilla , dándole los permisos necesarios será posible acceder a cualquier directorio de la Raspberry pi [12].

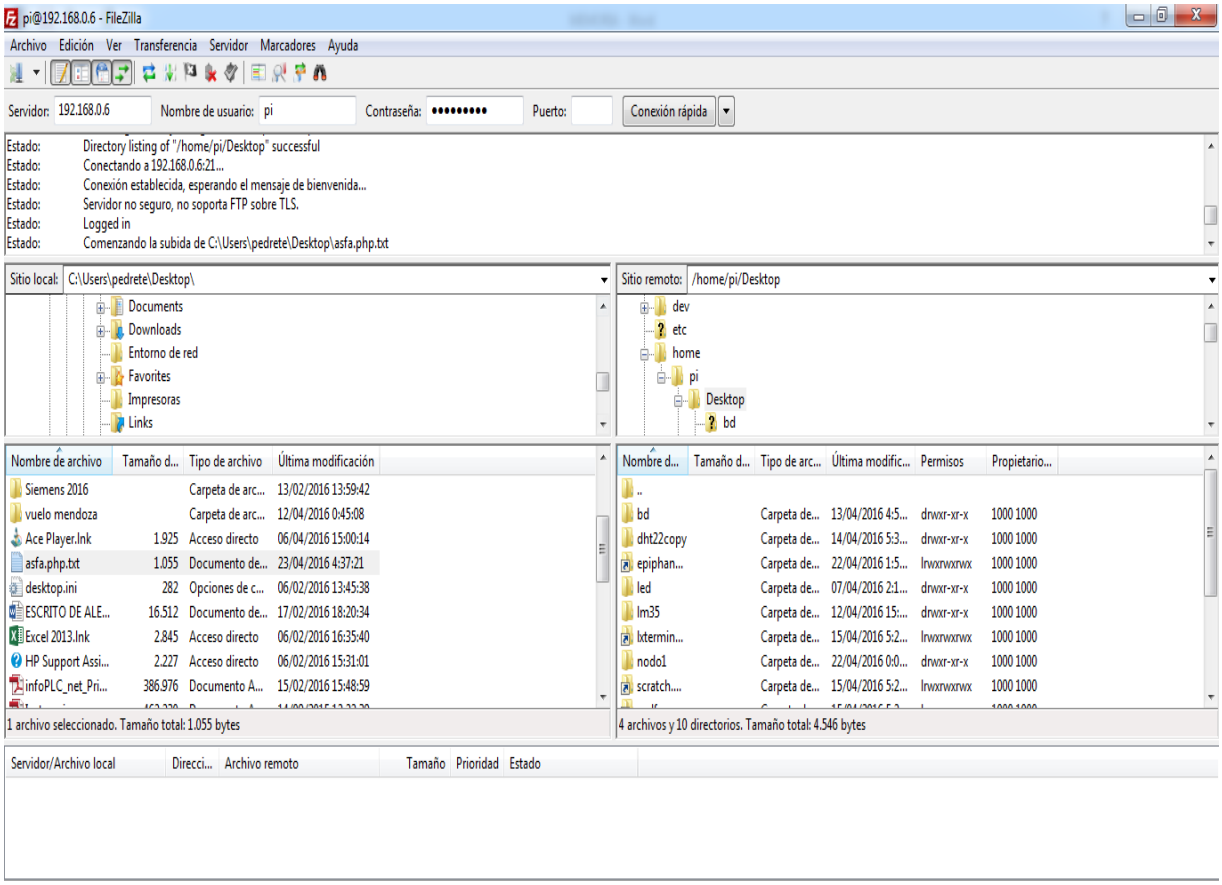

Ilustración 40:Interface de Filezilla

5.6 ACCESO DESDE INTERNET

Para lograr el acceso desde internet al server web, es decir acceder desde fuera de la red local, ha sido necesario la creación de un dominio virtual.

El servicio de DNS de No-ip permite crear un propio dominio gratuitamente, este ha sido el primer paso, seguidamente instalar el cliente en la Raspberry Pi.

Se ha procedido a su instalación y creación de archivos para el arranque automático cada inicio de la Raspberry Pi [13].

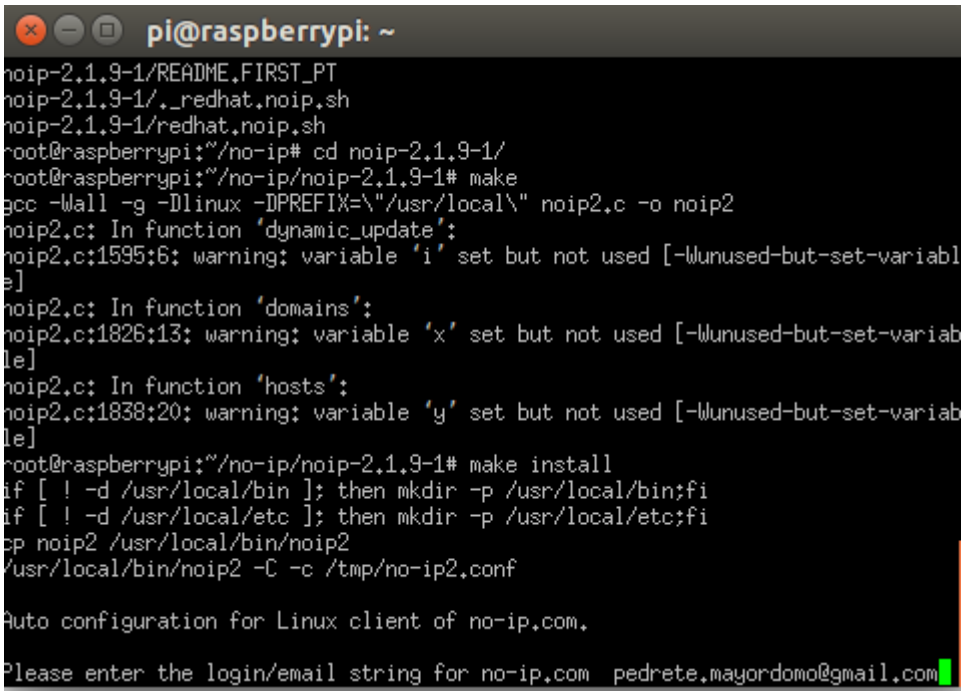

Ilustración 41:Instalación No-Ip

Lo siguiente para poder acceder desde el exterior es abrir los puertos , por lo tanto seria el 80(HTTP), Para aportar una mayor seguridad y no dejarlo por defecto el puerto es cambiado al 8080.

Para conseguir abrir los puertos, la primera es conectarse al router vía cable Ethernet, ya que no tenía la opción de conexión remota habilitada.

Local			External							
IP Address	Start Port	End Port	IP Address	Start Port	End Port	Prot	Description	Enabled		Remove All
192.168.0.6	8080	8080	0.0.0.0	8080	8080	BOTH	web server2	Yes	Edit	Remove

Ilustración 42:Redirección de puertos

IP adress es la dirección designada a la Raspberry Pi , el puerto al cual el router le enviara las peticiones de datos que será 8080. IP Address externa es "0.0.0.0 " se refiere a la IP asignada por el proveedor de servicios de Internet (*ISP*, por la sigla en inglés de *Internet service provider), además de poner el puerto 8080 en el cual el cliente realizara la petición.*

Con esto conseguimos que los mensajes que se reciben por ese puerto en el router automáticamente sabrá a que dirección debe enviar el paquete dentro de nuestro host, para eso la dirección de la Raspberry Pi es estática.

Aquí reside el funcionamiento de No-ip, el cual nos brinda una oportunidad de obtener una administración remota, fuera de nuestro Host. Nos permitirá tener un dominio virtual y asignarlo a una dirección IP la cual será la asignada por nuestro ISP, también se encargará de actualizar los servicios DNS, en el caso de que nuestro router sea reiniciado y asignado una nueva IP externa, actualizará nuestra ubicación.

El nombre asignada a nuestra dirección IP gracias al servicio DNS de No-Ip es " balcarce.hopto.org" , puesto que se está trabajando con el puerto 8080 , será "balcarce.hopto.org:8080".

También se pueden utilizar los puertos 22 para conexión SSH .

5.7 ALMACENAMIENTO EN LA NUBE

Se ve necesario en este proyecto incluir un espacio en la nube donde se recogen todos los datos trasferidos por los dispositivos.

De esta manera es posible tener siempre disponibles en la nube los archivos generados de la información recogida por los sensores, los archivos que son almacenados serán, archivos de texto, donde se encontraran los muestreos de datos de cada nodo y la hora en la que fueron tomados.

Para ello ha sido de gran utilidadel kit de desarrollo de software (SDK) para Python ofrecido por Dropbox,de esta manera el servidor ofrece una mayor capacidad a la plataforma IOT.

La intención de esta conexión es realizar Backup cada hora de los archivos generados del muestreo de datos y además poder acceder a ellos sin estar conectado al host en el cual se encuentra la Raspberry Pi o en caso de caída del servidor.

Ilustración 43:Dropbox

Para comenzar será necesario la creación de una App de Dropbox para poder tener acceso a una interfaz de programación de aplicaciones o API (del inglés Application Programing Interface).

App folder, se selecciona acceso a la carpeta de la aplicación que se desea realizar, no a Dropbox completo. El nombre de la App elegido es TFG_UPCT, a la cual se le ha dado permisos de escritura y lectura.

La conexión a Dropbox será mediante el protocolo Oauth, el cual es un protocolo abierto que permite la autorización segura de una API, en este caso la de Dropbox, sin el intercambio de contraseñas.

Un ejemplo habitual se puede encontrar en las redes sociales las cuales han adoptado este medio de autorización seguro, además de promover la integración de otras aplicaciones.

Por ejemplo, delegar en Twitter una autorización para que pueda publicar en tu muro de Facebook o registrarse en cualquier aplicación con tu cuenta de Google o facebook. Estos gestos cotidianos no serían viables si no existiese este protocolo OAuth, puesto que deberíamos darle las claves de nuestra cuenta y esto otorgaría demasiado poder a las aplicaciones además de quedar expuesta la información personal.

Con Outh el propietario de los recursos autoriza al cliente durante un tiempo limitado, por la expiración del token de acceso, a poder acceder a los recursos o realizar las operaciones que el propietario permita acceder en el servidor al cliente.

En este caso nuestra App será quien se asocia con la API ofrecida por Dropbox, otorgando yo permisos de acceso para poder leer y escribir dentro de la carpeta creada.

Si bien se ofrece toda una serie de documentación para poder realizarlo con SDK(del inglés software development kit) , se decide utilizar una librería la cual se encontraba en los repositorios de GITHUB, de Andrea Fabrizi [14], la cual es una aplicación que permite automatizar los procesos de subida de archivos de cualquier tipo utilizando comandos BASH , simplemente en primer inicio pide las claves para la asociación con nuestra APP creada , una vez que le damos permiso a la aplicación se guarda el token de acceso, el resto de sincronizaciones no es necesario volver a darle .

Esta librería facilita la tarea de automatizar la subida de archivos a Dropbox, los comandos son muy simples. Como se puede observar en la siguiente ilustración.

```
Commands:
        upload   <LOCAL_FILE/DIR ...>  <REMOTE_FILE/DIR>
        download <REMOTE_FILE/DIR> [LOCAL_FILE/DIR]
        delete   <REMOTE_FILE/DIR>
        move     <REMOTE_FILE/DIR> <REMOTE_FILE/DIR>
        copy     <REMOTE_FILE/DIR> <REMOTE_FILE/DIR>
        mkdir    <REMOTE_DIR>
        list     [REMOTE_DIR]
        share    <REMOTE_FILE>
        saveurl  <URL> <REMOTE_DIR>
        info
        unlink
```

Ilustración 44:Comandos de librería andreafabrizi.

En la ilustración anterior se pueden observar las funciones que nos ofrece la librería, subir archivos, descargar, borrar, crear carpetas...Todo ello de una forma muy sencilla.

Para ejecutar comandos de la Shell desde Python, se importa la librería “OS” y la función system. Es un comando simple y directo para ejecutar comandos en la Shell de Debian.

El comando que se utiliza para subir estos archivos:

```
os.system(" bash /home/pi/Desktop/..../Dropbox_uploader.sh  upload
'/home/pi/.../archivo.txt'    '/home/nodo/../archivo.txt'    ")
```

- “/home/pi/Desktop/..../Dropbox_uploader.sh “es la ruta del ejecutable de la librería.
- “upload “la acción que se desea realizar, ver ilustración 35.
- “'/home/pi/.../archivo.txt' “archivo el cual se quiere subir a Dropbox.
- “‘/home/nodo/../archivo.txt’ “ carpeta donde quieres almacenar el archivo en la carpeta de Dropbox , además del nombre que se le va a dar.

El comando anterior ejecutará el comando BASH y cuando termine devolverá el flujo a nuestro código Python.

CAPÍTULO 6

6 SOLUCION DESARROLLADA

En este capítulo se explicarán todo el proceso tomado para llegar a la aplicación Web final. Se explicarán las partes más relevantes de los códigos realizados.

6.1 INTRODUCCION

Antes de entrar a detallar los códigos de los diferentes lenguajes de programación usados, se define la topología implementada y como se comunican entre los diferentes elementos que engloban el proyecto.

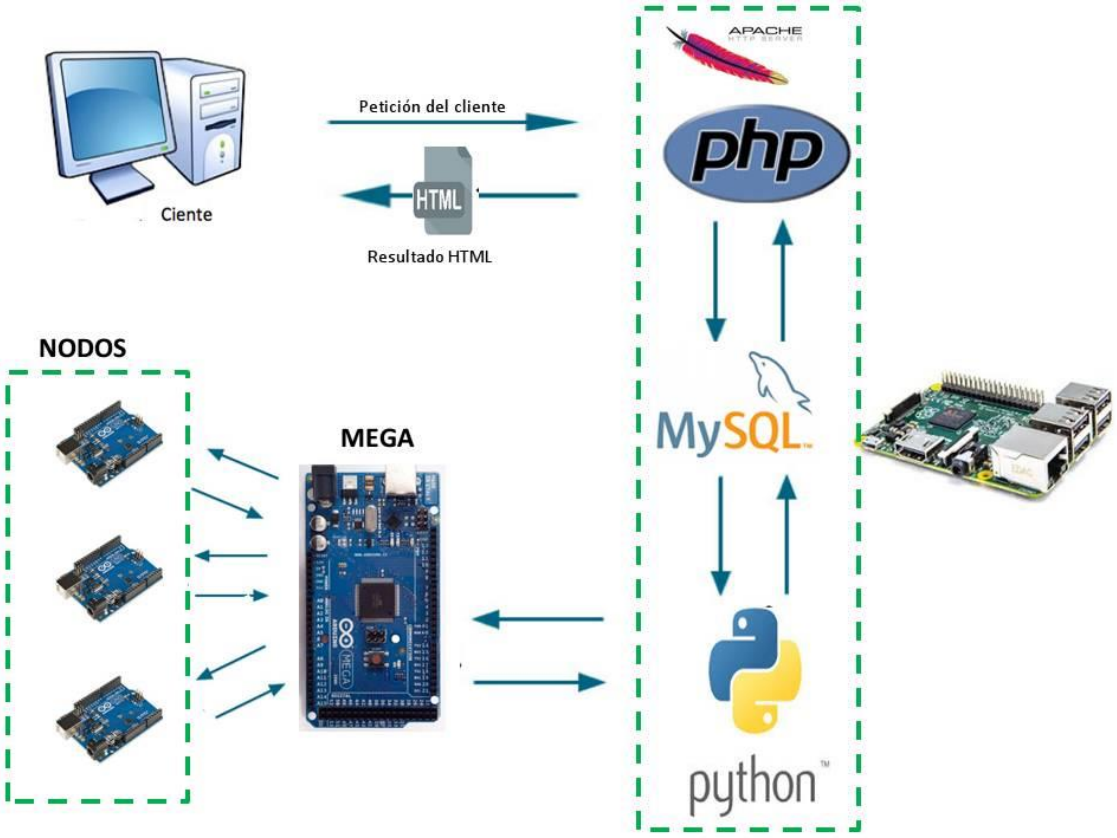

Ilustración 45:Configuración de Topología del proyecto

En la anterior ilustración queda representada las partes integrantes del proyecto, desde el usuario final hasta la recolecta de información, además de cómo llega la información o los datos recogidos por los nodos y es tratada para presentarla de una forma legible al usuario.

Para tratar de explicar bien la ilustración anterior y poder entender con mayor claridad las soluciones tomadas, se pasará a dividir el gráfico en 3 secciones en las cuales tendremos:

- **Monitorización y control**:

La tarea de monitorizar las variables del entorno recae en los nodos, el código de estos esta implementado sobre Arduino UNO.

Arduino MEGA es el encargado de controlar y gestionar él envió de los datos además de realizar un control para la apertura de las válvulas, con las cuales se puede controlar directamente la humedad del sustrato.

La tarea del intercambio de datos entre los nodos y el maestro funcionara con la arquitectura de un modelo Maestro/Esclavo.

Se tienen varios bucles en paralelo, es decir los procesos de adquisición de datos son contaste por parte de los esclavos y el bucle del maestro el cual recoge los datos de los esclavos de forma ordenada.

- **Código Python**

Es encargado de recibir la información de Arduino MEGA, pero en este caso se ha utilizado un modelo de comunicación Productor / Consumidor, para la comunicación realizada entre Raspberry Pi y Arduino Mega por puerto serial.

Rasperry Pi actuara de consumidor realizando peticiones a Arduino MEGA, y este le entregara los datos.

Además, el script de Python será encargado de:

- Subir datos a BD, valores instantáneos.
- Realizar una media de los valores recibidos, la media será por hora y los datos serán subidos a la BD para ser graficados.
- Escribir archivos de texto y guardarlos de forma organizada.
- Proceso de subida de archivos de texto a Dropbox cada hora.
- Lectura de niveles de consigna de humedad en la BD y envío a Arduino para que realice control.

- **Servidor**

El servidor web atenderá las peticiones http del cliente.

Para aprovechar al máximo las características del servidor web instalado en Raspberry Pi, servidor web Apache, se utiliza PHP y Mysql, donde PHP será el encargado de acceder al sistema de gestión de base de datos Mysql, ejecutando scripts del lado del servidor.

De esta forma es posible hacer llegar al cliente, en formato HTML , los datos de la base de datos y poder cambiar los niveles de consigna de humedad del sustrato manualmente desde la interfaz web.

6.2 ESTRUCTURA DE NODOS

I. NODOS

Los nodos son la parte más importante en la topología puesto que son los que captan la información de nuestro entorno, ellos son los esclavos no realizan ningún tipo de control en los datos simplemente muestrean y envían sin interrupción, el maestro leerá los datos que ellos envían cuando los necesite.

Cabe destacar la aplicación de los modelos Maestro/Esclavo y Productor/ Consumidor en este proyecto además de explicar el uso de su elección mediante la explicación del código.

En el caso de a ver usado una comunicación de Arduino MEGA y Raspberry Pi Maestro/esclavo el buffer seria colapsado muy rápido y sin ningún control, esto podría ocasionar fallos y en consecuencia la parada del programa o entrada en bucle.

En cambio, aplicando el modelo Productor/ Consumidor, el nodo maestro o sumidero solo leerá de los nodos cuando el Consumidor (Rasperry Pi), le pida datos, con este planteamiento se evita colapsar el Buffer y no producir errores de lectura.

La comunicación de los nodos "esclavos" con el maestro funciona con el Modelo Maestro / Esclavo como se ha mencionado anteriormente, los nodos están constantemente enviando las tramas de datos de las variables muestreadas y el nodo maestro leerá del nodo que necesite, aquí no se tiene problema de buffer puesto que el medio de comunicación es el aire.

En los sketch's de los nodos se pueden encontrar tres funciones leer_datos (), envio_datos () y comprobación () las cuales se repiten cíclicamente.

Primero lee los datos, posteriormente los envía por el canal de comunicación establecido con el nodo maestro, se comprobará si el mensaje que el nodo ha enviado ha sido leído por el maestro. Y vuelve a repetirse el Loop sucesivamente.

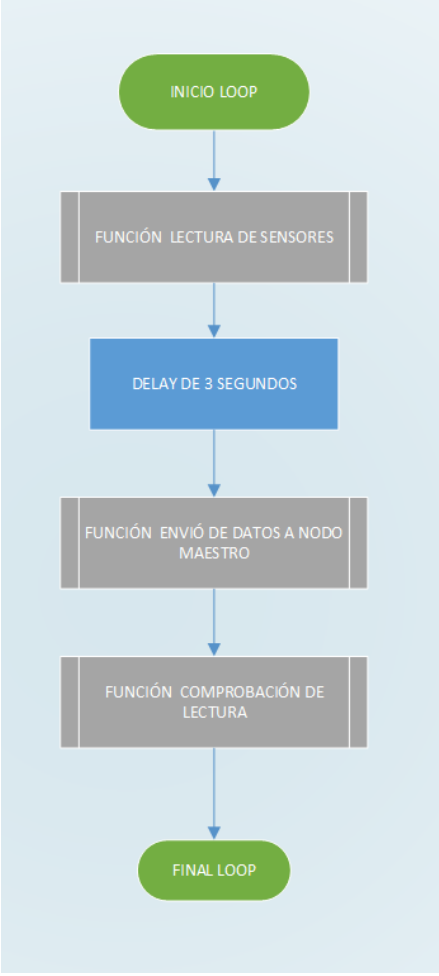

Ilustración 46:Diagrama de flujo Nodos.

II. NODO SUMIEDERO

Sera el encargado de recibir los datos captados por los nodos donde se encuentran todos los datos recogidos por los sensores, realizar comprobación que los datos recogidos son del nodo al cual se los está pidiendo, realiza el tratamiento de los datos correspondientes del nodo y crea la trama para enviárselos por puerto serial a Raspberri Pi y además realiza el control de las válvulas con las consignas introducidas por el usuario desde la interfaz web.

El Loop del nodo maestro se puede dividir en tres funciones generales.

1º Control ():

La primera función del Loop, comprobara el estado de las válvulas comparando con los niveles de HR introducidos por el usuario desde la interface Web.

Para el control de las válvulas se ha implementado un control On /Off con histéresis, cuando la variable controlada, en este caso la humedad relativa del sustrato baja del nivel mínimo la válvula abrirá hasta llegar al nivel máximo. De este modo se asegura un rango de humedad a controlada por el usuario.

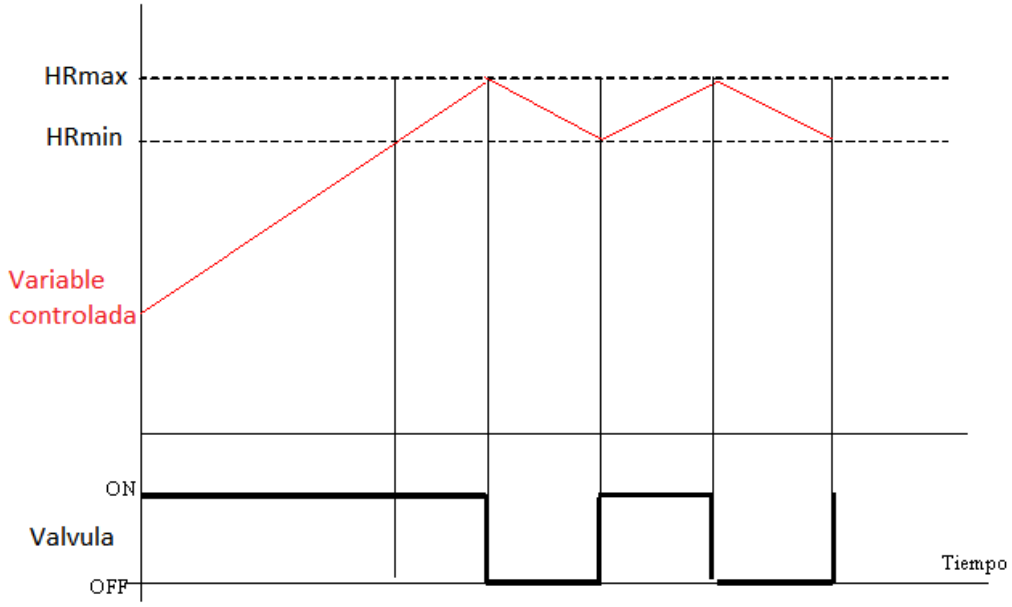

Ilustración 47:Control Válvula

2º Escuchar_rasp ()

Se comprueba si se está recibiendo alguna petición de datos o, hay un mensaje en el buffer si es así se procederá a leer.

La trama que Python envía para recibir los datos sigue la siguiente estructura:

```
N:nodo:HRmin:HRmax

Ej:  N:1:20:80
```

Se introduce por seguridad La letra N, para asegurar que no recibimos valores extraños, además de separar la trama por ':' para poder separar los valores fácilmente.

Una vez se ha escuchado la trama recibida, se guardan los valores de HRmin y HRmax en el nodo correspondiente y se realiza lectura del nodo pedido.

Se responde a la petición de los nodos con la siguiente trama:

N:dia:mes:año:h:min:seg;nodo:Tamb:Hramb:Tsut:Hrsut;estado_valvula:IRRADIACION:
Ej:N:1:9:16:14:30:30;1:25.00:60.00:16.50:40.10;0

La hora proveniente del sensor RTC, para poder comprobar la hora exacta en la cual fue realizada la lectura posteriormente en el txt, es separada con un ';' con respecto a los datos muestreados al igual que separados estos con el estado de la válvula en el momento del envío

En el ejemplo anterior el estado de la válvula seria 0, eso quiere decir que está dentro del rango de humedad introducido por el usuario.

3º Actualización de valores:

Se trata de refrescar los datos de los nodos para poder realizar un control de las válvulas, con las variables lo más actualizadas posibles.

Es posible controlar el tiempo en el cual fue la última actualización de los nodos con la función millis (), la cual devuelve el tiempo desde que la placa Arduino fue alimentada. El valor de milli() desborda a los 50 días es decir a los 50 días vuelve a 0.

Cada vez que se llama a la función Petición_datos (), se guarda en la variable T [3], el cual tiene 3 posiciones una para cada nodo, además se guarda el tiempo en el que se realizó la última actualización de los nodos.

Por lo tanto, con una simple resta entre millis () y T[nodo], es posible saber cuánto tiempo se lleva sin actualizar la variable o lo que es lo mismo cuánto tiempo lleva Python sin hacer ninguna petición.

Con esta función se fija un valor como máximo unos 5 segundos, y en el caso de no haber sido actualizado el nodo será actualizado para cuando vuelva a inicializar el lazo, pueda realizar el control de válvulas con variables refrescadas.

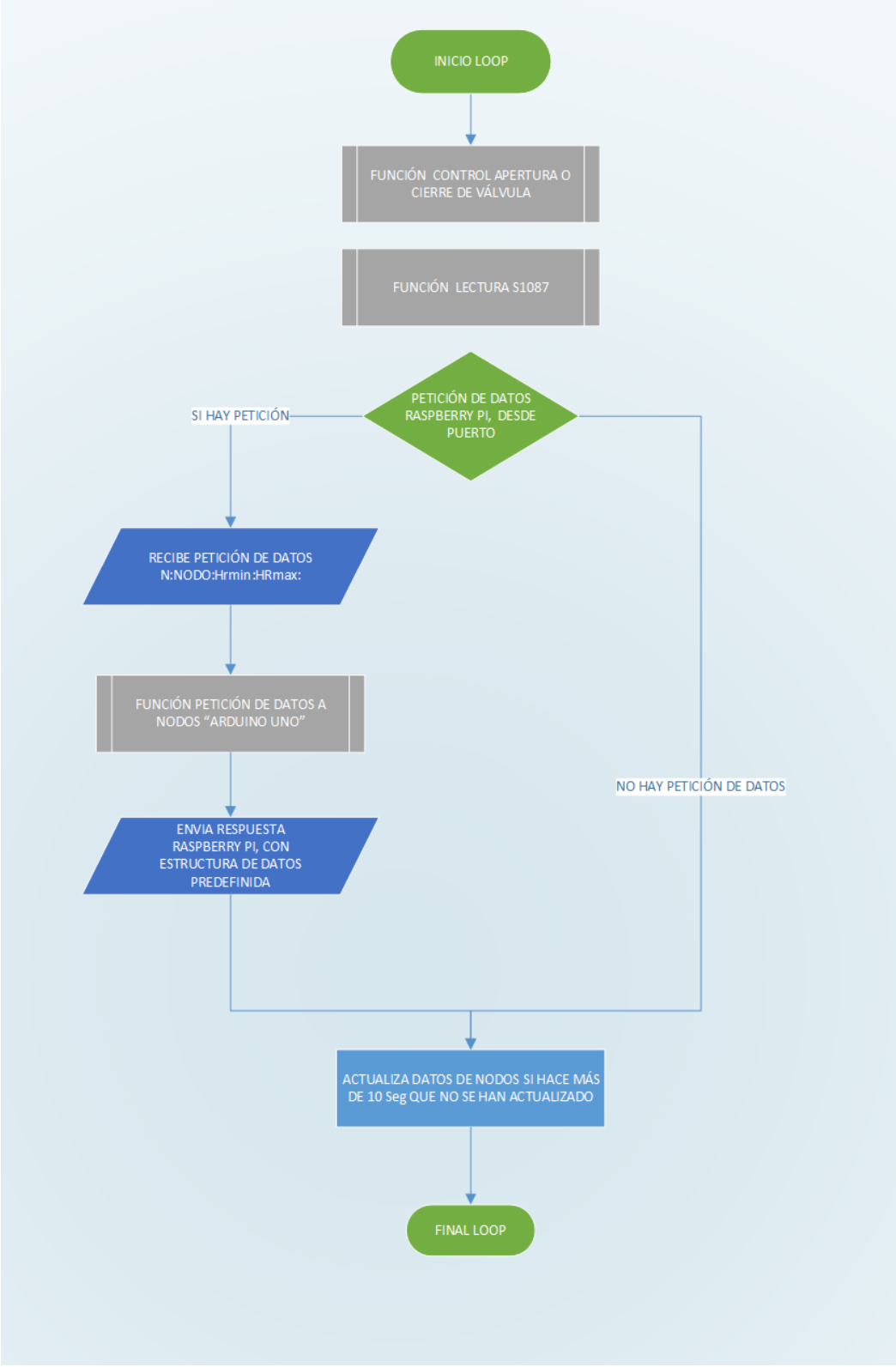

Ilustración 48:Diagrama de flujo Nodo Sumidero.

6.3 CÓDIGOPYTHON

I. DESARROLLO CÓDIGO PYTHON

Como bien se ha comentado anteriormente se utiliza el modelo Productor/ Consumidor, Arduino MEGA será el productor, el cual pide los datos uno por uno de cada nodo y se encargará de enviarle los datos por comunicación Serial al consumidor cuando él se los demande.

Las tramas enviadas y de respuesta han sido mencionadas anteriormente en el apartado 6.2.2.

Este Script de Python será el encargado de realizar la Comunicación entre Arduino MEGA y Raspberry Pi, además de realizar el tratamiento de los datos y subirlo a la Base de Datos.

El Script tiene las siguientes funciones:

-Subir valores de Gauge a BD

- Realiza media de valores cada hora

-Sube valores medios por hora a BD

-Crea archivos de texto de los valores recibidos, separando y ordenando los TXT en carpetas para cada nodo y separando por días

-Subida de archivos TXT a Dropbox, actualización cada hora.

-Envió de correo Electrónico, cuando se recibe un mensaje de error.

En este apartado se explicarán algunos de los puntos más interesantes utilizados en el código, puesto que al ser un lenguaje Ordenado y limpio es bastante sencillo de leer, el script esta comentado para lograr una mejor compresión de este.

- **str.partition("cararter_separador")**:

Una de las funciones que facilita el tratamiento de las tramas recibidas por puerto serial.

Simplemente se utiliza ":" para separar los valores, esta función retorna una tupla, la cual contiene 3 cadenas, la parte de la derecha y izquierda de lo dividido y el carácter separador.

-**Mutabilidad e Inmutabilidad:**

Ha diferencia de lenguajes como C, en el cual existen los denominados punteros, los cuales son referencia a una variable en Python no existen punteros, todas las variables son referencias a una porción de memoria o "objeto", cuando igualamos una variable a otra le asignamos esa porción de memoria.

Existen dos tipos de variables Mutables e Inmutables, las variables inmutables, como puede ser una cadena o un entero, no es posible asignar un nuevo contenido, pero si modificar la variable.

Si modificamos la variable lo que ocurrirá será que se creará un nuevo objeto, por lo tanto, la variable tendrá otra referencia a ese objeto. Explicándolo con concepto de punteros se podría decir que las variables se tratan de punteros a objetos, no contienen el valor sino la referencia a ese valor.

En cambio, las variables mutables, pueden ser modificadas, como es el caso de las listas.

Conociendo bien estos conceptos se puede trabajar de forma sencilla y rápida, evitando los pasos por referencia. Es un lenguaje muy poderoso, se consiguen buenos resultados con pocas líneas de código, fácil y muy flexible, además de una gran comunidad de tras

```
#M_N1=[T_ini,Tamb,Hamb,Tsut,Hsut,C_Tamb,C_Hamb,C_Tsut,C_Hsut]
M_N1=[]
M_N2=[]
M_N3=[]
for i in range(9):
        M_N1.append(0)
        M_N2.append(0)
        M_N3.append(0)
-----------------------------------------------------------------------------------------
if nodo == '1':
        N= 'nodo1'
        MEDIA= M_N1
elif nodo == '2':
        N= 'nodo2'
        MEDIA= M_N2
else  :
        N= 'nodo3'
        MEDIA= M_N3
```

Como se puede ver en este extracto de código, la primera parte se declaran las variables en este caso array de 9 para almacenar los valores de media por hora de cada nodo.

En la parte de abajo de la ilustración se copia la referencia a ese valor dependiendo del nodo, de esta manera desde la variable MEDIA se apunta al objeto de media de cada nodo y de esta forma es posible simplificar las funciones.

-**Acceso a base de datos:**

Para el acceso a base de datos con Python se trabaja con el módulo MySQLdb.

Para ello se crea una función la cual en función de las referencias a valores que se le envíen realizara una acción u otra.

El procedimiento es el mismo que se realiza para PHP:

-Abrir conexión y crear puntero

-Ejecutar la consulta

-Hacer efectiva la escritura o traer los datos de la selección realizada

- Cerrar puntero y la conexión

La función creada para realizar todas las consultas es la siguiente:

```
# ********************FUNCION ESCRIBIR BASE DE DATOS ******************


def run_query(query=' ' , DB_NAME=' ', temp=' '):


        datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]

        db = MySQLdb.connect(datos[0], datos[1], datos[2],datos[3]) # Conecta a la base de datos

        cursor = db.cursor()       # preparamos  cursor

        if temp=='r':   #  Para leer

                try:

                        cursor.execute(query)

                        data = cursor.fetchall() # Este método obtiene todos los registros de un
conjunto de resultados de una consulta a la BD

                except:

                        print "Error , no se obtienen datos"


        elif temp=='w':   # Para escribir

                try:

                        cursor.execute(query)

                        db.commit()          # Hacer efectiva la escritura de datos


                except:

                        print "Error que le den a error  al modificar"

                        db.rollback()     #revierte el error

        cursor.close()           # Cerrar el cursor

        db.close()              # Cerrar la conexión

    if temp=='r':


                return data[0]
```

Para poner en marcha el script de Python se ha optado por iniciarlo en segundo plano llamándolo desde la Shell de Debian, ingresando el nombre “tfg_upct” y para ejecutarlo en segundo plano añade “&”.

Para eso se crea un bash, el cual es un intérprete de ordenes basado en la Shell de UNIX, este bash se guarda en la carpeta con ruta “*/usr/local/bin* “y se le da al archivo permisos de ejecución.

```
#!/bin/bash

echo Abriendo script python

cd  /home/pi/Desktop/fft/

python General.py
```

También se podría arrancar una vez inicializada la Raspberry Pi como se hace con apache o no-ip.

-Ejemplo de registro de datos en archivos txt :

Los datos recibidos por puerto serie son escritos en un txt , rellenado con los datos muestreados. Cada día se crea un TXT nuevo además de que cada NODO tiene su propio TXT.

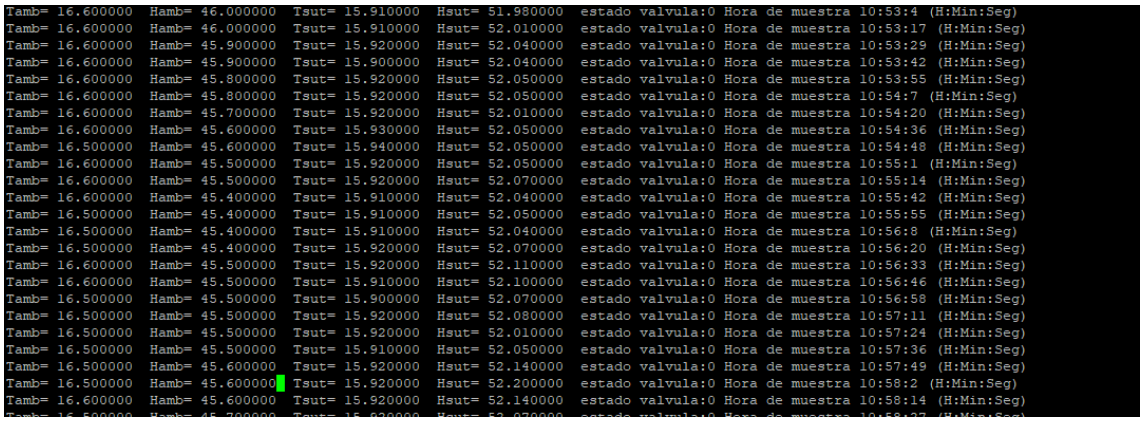

Ilustración 49:Registro de datos txt

-Envió de correos electrónicos:

En el caso de no a ver comunicación con los nodos el nodo sumidero ante una petición de datos, enviara la siguiente trama:

```
Error:dia:mes:año:h:min:seg;nodo:Tamb:Hramb:Tsut:Hrsut;estado_valvula:IRRADIACION:

E :1:9:16:14:30:30;1:25.00:60.00:16.50:40.10;0
```

Siendo la cabecera del mensaje “E”, el código de Python lo interpreta y enviara un correo avisando del error del nodo del cual proviene el error.

```
pi@raspberrypi:/var/www/Python $ python General4.py
N:1:23:90:
 Error Comunicación en nodo:'1' Revisar Estación
N:2:42:24:
 Error Comunicación en nodo:'2' Revisar Estación
N:1:23:90:
 Error Comunicación en nodo:'1' Revisar Estación
```

Ilustración 50:Petición por parte de Raspberry Pi , recepción de error.

Tras recibir la trama de error, se envía el correo con el siguiente mensaje.

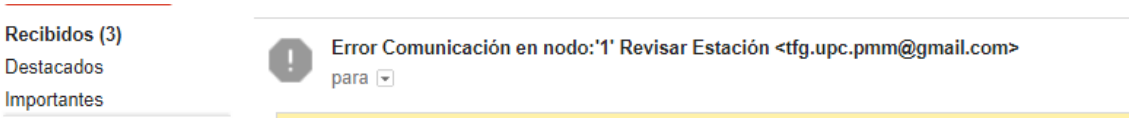

Ilustración 51:Mensaje de correo, ante error de Nodo.

La función creada para enviar correo es la siguiente:

```
def Emails (fromAdress,ToAdress,PW,msg):


        # Enviando el correo
        server = smtplib.SMTP('smtp.gmail.com:587')
        server.ehlo()
        server.starttls()


        server.login(fromAdress,PW)
        server.sendmail(fromAdress,ToAdress, msg)
        server.quit()
```

Función creada para usarla en el código se importa archivo y se llama a la función.

II. DIAGRAMA DE FLUJO CÓDIGO PYTHON

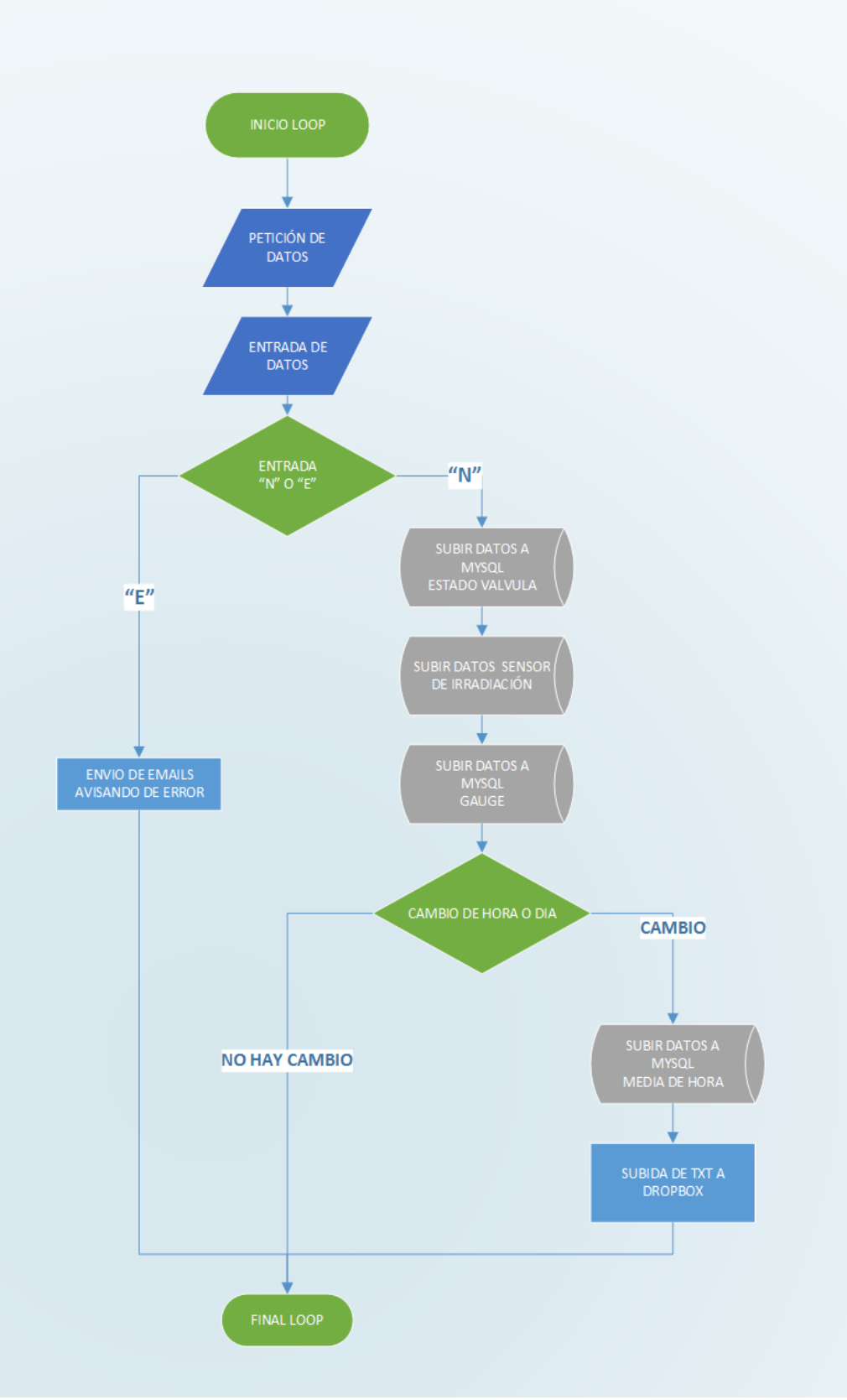

Ilustración 52:Diagrama de flujo Python.

6.4 SERVIDOR

I. CONFIGURACION RASPBERRY

Primero la instalación del sistema operativo Raspbian en la Raspberry Pi, desde la página oficial de Raspberry pi [8] en la sección de descargas.

Para la instalación se utiliza el programa Win32Disklmanger, simplemente seleccionando el archivo imagen descargado y escribiéndolo en la tarjeta microSD de 8Gb utilizada para este proyecto.

Como ya se comentó la primera parte de la configuración debería ser habilitar la configuración SSH, en nuestro caso se encontraba habilitada por defecto, por lo tanto, directamente se puede conectar con Putty conociendo la IP asignada por el router dinámicamente mediante *DHCP (*Protocolo de configuración de host dinámico).

La siguiente cuestión será fijar una IP estática, para no tener que estar buscando la dirección asignada en cada arranque, accediendo a la siguiente dirección:

Y dejando el archivo de configuración, con la dirección estática con la cual te deseas conectar al router evitando el servicio DHCP:

```
auto eth0
allow-hotplug eth0
iface eth0 inet static
address 192.168.0.6
netmask 255.255.255.0
gateway 192.168.0.1

auto wlan0
```

Ilustración 53:Configuración interfaces

Una buena práctica también es acceder a la página de configuración del router y filtrar por MAC o en otro caso cambiar el rango de asignación de direcciones del servicio DHCP y comenzar a repartir direcciones DCHP a partir de la asignada a Raspberry Pi hacia arriba.

Sino no se hace lo anteriormente dicho puede darse el caso de que la dirección que se use para redireccionar los puertos puede estar ocupada por otra persona.

Lo siguiente será instalar el servidor LAMP (Linux, Apache, MySQL, PHP), explicados anteriormente.

II. BASE DE DATOS

La base de datos guardara los datos de las variables recolectadas por los nodos, gracias al gestor de bases de datos MySQL instalado se tiene la compatibilidad para poder acceder a la base de datos con Python y PHP.

Con la interfaz de phpmyadmin se crearán las siguientes Bases de datos denominadas Actuadores y una base de datos para cada nodo las cuales corresponde a nodo1, nodo2 y nodo3.

En la base de datos actuadores se pueden observar tres tablas, una para los valores de consigna de Hrmin y Hrmax para cada nodo, los estados de las válvulas y otra con el valor instantáneo de la irradiación solar que se está recibiendo.

Para mostrar las aplicaciones que nos ofrece phpMyAdmin se han analizado la base de datos de los nodos a modo de ejemplo. En cada nodo es posible encontrar las tablas de valores de gauge, S_amb y S_sut como aparece en la siguiente ilustración.

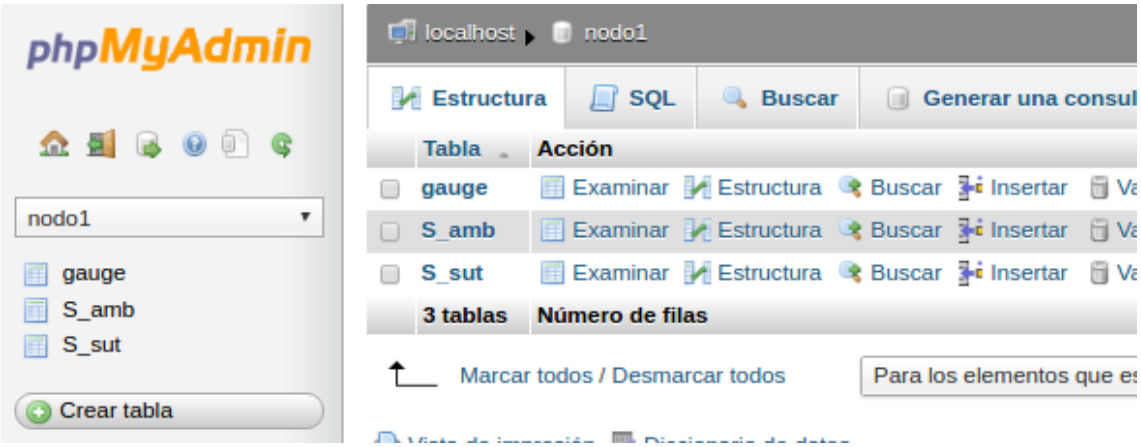

Ilustración 54:phpMyAdmin ejemplo de base de datos de nodo

La tabla de “S_amb” y “S_sut” contienen los valores medios por hora subidos por Python,los cuales son representados en gráficas leyendo los valores a través de consultas a la base de datos con PHP como se muestra posteriormente.

Ambas tablas contienen 4 campos id, humedad, temperatura y hora, en la siguiente ilustración se muestran las características de cada variable.

Ilustración 55:phpMyAdmin campos de S_amb y S_sut

También se puede examinar los datos insertados por Python desde la interfaz phpMyAdmin, incluso editar en línea para la realización de pruebas.

Ilustración 56:phpMyAdmin interfaz de nodo

6.5 DISEÑO INTERFAZ

El cuerpo de la interfaz está desarrollado en HTML y los estilos con CSS, pero el lenguaje HTML es estático por tanto se complementa con la programación en PHP para hacer una interfaz dinámica. Este lenguaje nos permite realizar las consultas a la base de datos, actualizar los valores de la base de datos, y mostrar estos valores en la interface web.

El código para realizar la conexión a la base de datos es el siguiente:

```
<?php
        $con=mysqli_connect("localhost","root","raspberry","Base_datos");
        if(mysqli_connect_error()){
        echo "No accede a la base :".mysqli_connect_error();}

?>
```

Trabajando con Mysqli el cual tiene un mayor rendimiento que Mysql, en la forma de optimizar el acceso a consultas.

Un ejemplo de consulta se puede observar en la pantalla de inicio, mostrada en la siguiente ilustración.

Ilustración 57:Pantalla de inicio

En la anterior ilustración es posible ver las consultas realizas a base de datos mediante PHP:

- Para saber el estado de la válvula, abierta o cerrada
- El valor actual de HR_min y HR_max para cada nodo

- Se realiza una consulta de escritura al realizar un cambio en los valores de consigna, mediante un POST.

El usuario introduce los valores de consigna en el formulario y se envía este valor en una variable a un archivo PHP el cual es ejecutado y sube el valor a la base de datos y devuelve al usuario a la página de inicio.

El siguiente código muestra cómo se realizan las consultas del estado de válvula, valor actual y envió de datos en formulario mediante método POST.

```
        <div>
                <?php
                        $result = mysqli_query($con,"SELECT * FROM valve where id='1'  ");//selecciona tabla y busca fila con
id=1
                        $row = mysqli_fetch_row($result);
                        if ($row[1]=="0") {  // Si esta cerrada entramos con esta condición
                        $estado1 = "images/cerrada.gif";}
                        else  {
                        $estado1 = "images/abierta.gif";}
                        //**************************Valores de HRmin y HR max ********************
                        $result = mysqli_query($con,"SELECT *   FROM `offset` where id='1'  ");
                        $row = mysqli_fetch_row($result);
                        $hrmax1 = $row[2];
                        $hrmin1 = $row[1];
                ?>
        <center><p><IMG SRC="/<?php echo "$estado1"; ?>" WIDTH=140 HEIGHT=140 ALT="family"></p></center>
                <!--********** HRMAX*******1******** -->
                <form action="/PHPs/HRmax1.php" method="POST">
                        <label for="valor">valor HR maximo</label>
                        <input type="text"   value=" actual:<?php echo "$hrmax1"; ?>" name="valor" class="form-input"
required/>
                        <!-- name="valor" , es la variable que enviamos por POST -->
                        <center> <input class="form-btn" name="submit" type="submit" value="enviar" />
                        </center>

                </form>
        </div>
```

Para la representación de las gráficas y los gauges se optó por Google Charts por varias razones:

-facilidad de uso

-Configuración variada, colores, formatos.

-No satura el servidor en la generación de imágenes.

También cabe mencionar que se puede llegar a tener el problema de que Google cierre este servicio en cualquier momento y además de ser necesario tener una conexión para su uso.

La API de Google Chart nos hace sencilla la tarea puesto que el código es sencillo solamente es necesario la consulta a base de datos con un “mysqli_fetch_array ($selector)”, devolverá la fila y cuando no queden filas devuelve un NULL.

```
<script type="text/javascript" src="https://www.google.com/jsapi"></script>
<script>
        google.load("visualization", "1", {packages:["corechart"]});
        google.setOnLoadCallback(dibujarGrafico);
        function dibujarGrafico() {
        var data = google.visualization.arrayToDataTable([   ['fecha', 'Hs'],
        <?php
                while ($row = mysqli_fetch_array($result)){
        ?>  ['<?php echo" $row[0]=>$row[3]";?>',<?php echo $row[1] ;?>],   <?php
        }//cierre while
        ?>
        ]);
        <?php
                mysqli_free_result($result); // Liberamos los registros
                mysqli_close($con); // Cerramos la conexión con la base de datos
        ?>
        var options = {
        title: 'Valor Medio por hora _ Humedad Sustrato'
        }
        new google.visualization.ColumnChart(
        document.getElementById('GraficoGoogleChart-ejemplo-1')
        ).draw(data, options);
        }
</script>
```

Las peticiones realizadas mediante HTTP a la URL y en función de los parámetros que se han seleccionado en el código creara una imagen PNG que se mostrara al usuario en la aplicación

Web, el código con el que se realizan las peticiones de imágenes PNG es el anteriormente mostrado.

El código anteriormente mostrado, el cual accede a las bases de datos y toma los valores de las mismas y además realiza la petición para la obtención de las gráficas genera las siguientes imágenes PNG:

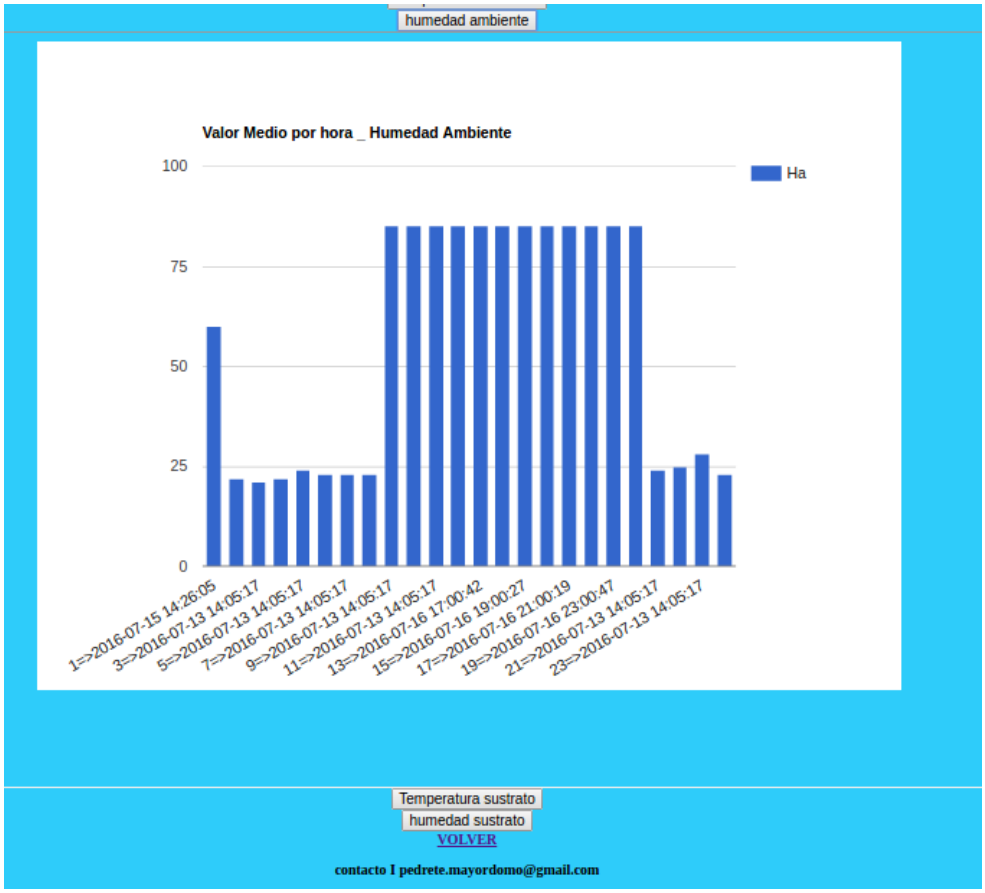

Ilustración 58:Imagen de graficas Interface Web

Con el fin de obtener una mejor organización de las gráficas y una interfaz simple para el usuario, se crea un formulario en HTML en la cual cada botón abrirá una nueva ventana utilizando iframe, lo cual nos ayuda a insertar un documento HTML dentro de otro.

Cuando se hace clic se ejecuta el archivo PHP correspondiente al botón seleccionado, en el siguiente código se puede ver un extracto del código correspondiente a la ventana de grafico generada de "Temperatura ambiente "donde se llama al archivo "Ta.php".

Se puede observar las dimensiones del marco creado donde se mostrará el gráfico generado por Google Chart anteriormente visto:

```
<!-- pestaña numero 1  -->
        <center>
                <button
                onclick="var ifreim1 = document.getElementById('ifreim1');
                ifreim1.style.display = ifreim1.style.display == 'none' ? 'block'
                : 'none'; "/>  Temperatura ambiente </button>
                <iframe src="Ta.php"
                id="ifreim1" style="display:none" width=100% height=80%
                display=inline></iframe>
        </center>
```

El aspecto de la interface Web de graficas sin accionar ningún botón sería el siguiente:

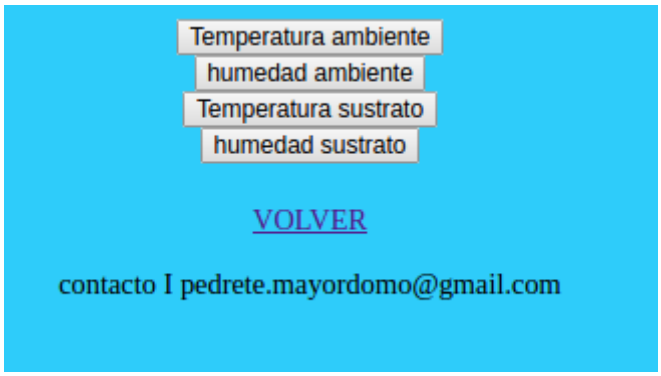

Ilustración 59:Aspecto de formulario para graficas de interface web

La siguiente ilustración muestra la interfaz que se observa en la ventana de cada nodo, en la cual se puede ver el estado actual de los sensores y el acceso a las gráficas comentadas anteriormente.

Se puede ver la radiación solar actual, para ayudar al usuario y no tener que visualizar las gráficas para saber la longitud de onda actual. Se ha optado por simplificar el resultado de los datos obtenidos por el sensor ya que la finalidad es saber si se está en el rango de luz visible o no.

La interfaz que se muestra en la pestaña de nodos es la siguiente:

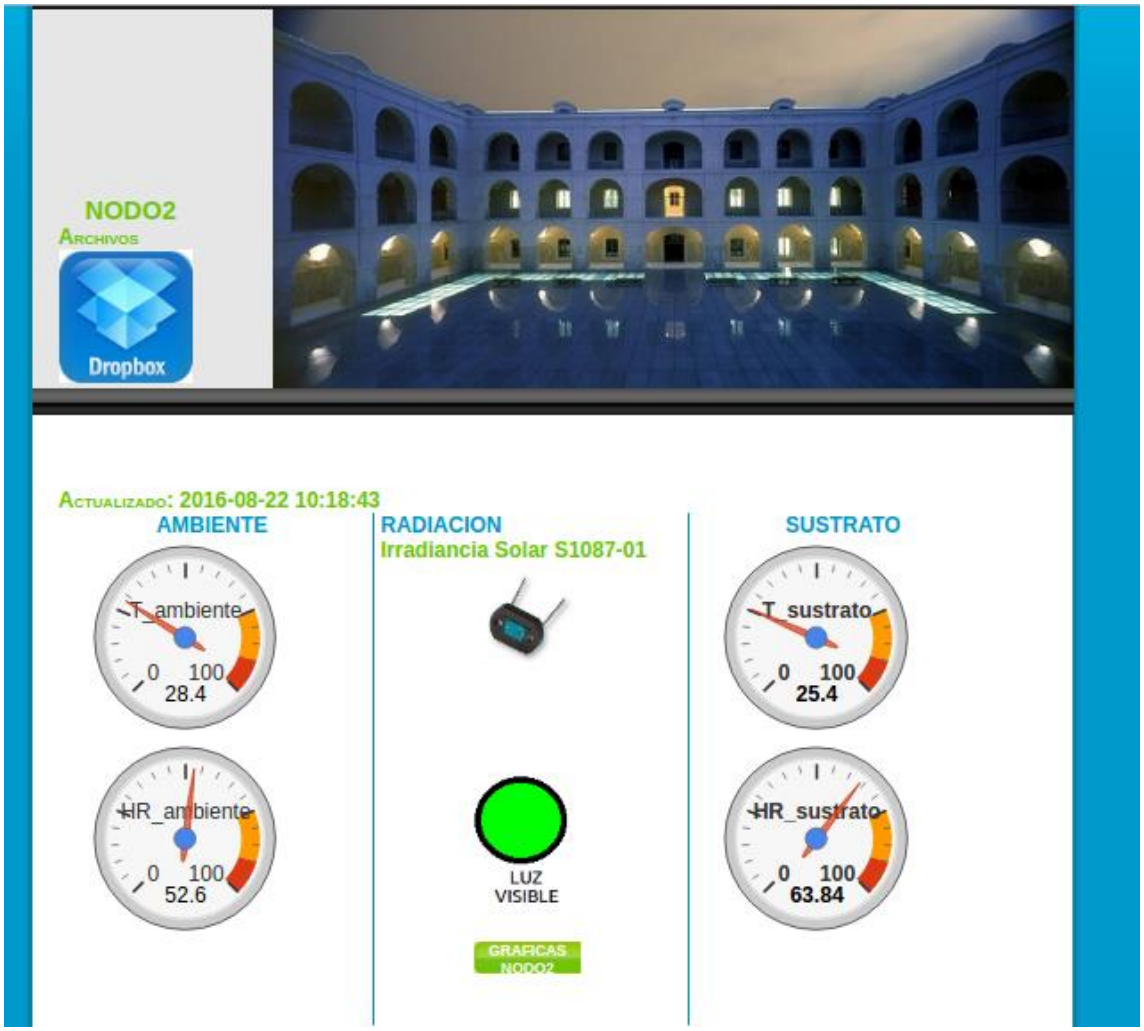

Ilustración 60:Interfaz de nodos

6.6 CREACIÓN DE SESIONES CON PHP Y MYSQL

Con el fin de limitar el acceso al servidor se optado por crear un sistema de Login de usuarios.

Para ello se a utilizado PHP y base de datos MySQL. Ha sido necesario la creación de una base de datos donde queden registrados los usuarios y sus contraseñas, de esta manera se podrán crear sesiones de usuario.

Han sido necesarios los siguientes códigos de PHP:

- Validación de Login , para poder obtener el acceso .

```
<?php  // Código  PHP lectura de  usuario y contraseña en basde de datos


 $con=mysqli_connect("localhost","root","raspberry","Usuarios");
        if(mysqli_connect_error()){ echo "No accede a la base :".mysqli_connect_error();}
        // Acceso a Base de datos tabla “usuarios”
        if(isset($_POST['login'])){
                $usuario = $_POST['user'];
                $pw = $_POST['pw'];
                $log = mysqli_query($con,"SELECT * FROM admin WHERE user='$usuario' AND pw='$pw'");
                // Se compara “user” y “pw” en el caso de encontrar alguno idéntico el usuario existe , por lo que permite abrir sesión y redirecciona a  pagina inicial .


                if (mysqli_num_rows($log)>0) {
                        $row = mysqli_fetch_array($log);
                        $_SESSION["user"] = $row['user'];
                        echo 'Iniciando sesión para '.$_SESSION['user'].' <p>';
                        echo '<script> document.location="Principal.php"; </script>';
                }
                else{
                        echo '<script> alert("Usuario o contraseña incorrectos.");</script>';
                        echo '<script> document.location="index.php"; </script>';
                }
```

En el caso de no abrir sesión e intentar saltar Login , al inicio de cada página existe una comparación. Si la comparación resulta negativa vuelve a la página de Login, la cual se muestra a continuación:

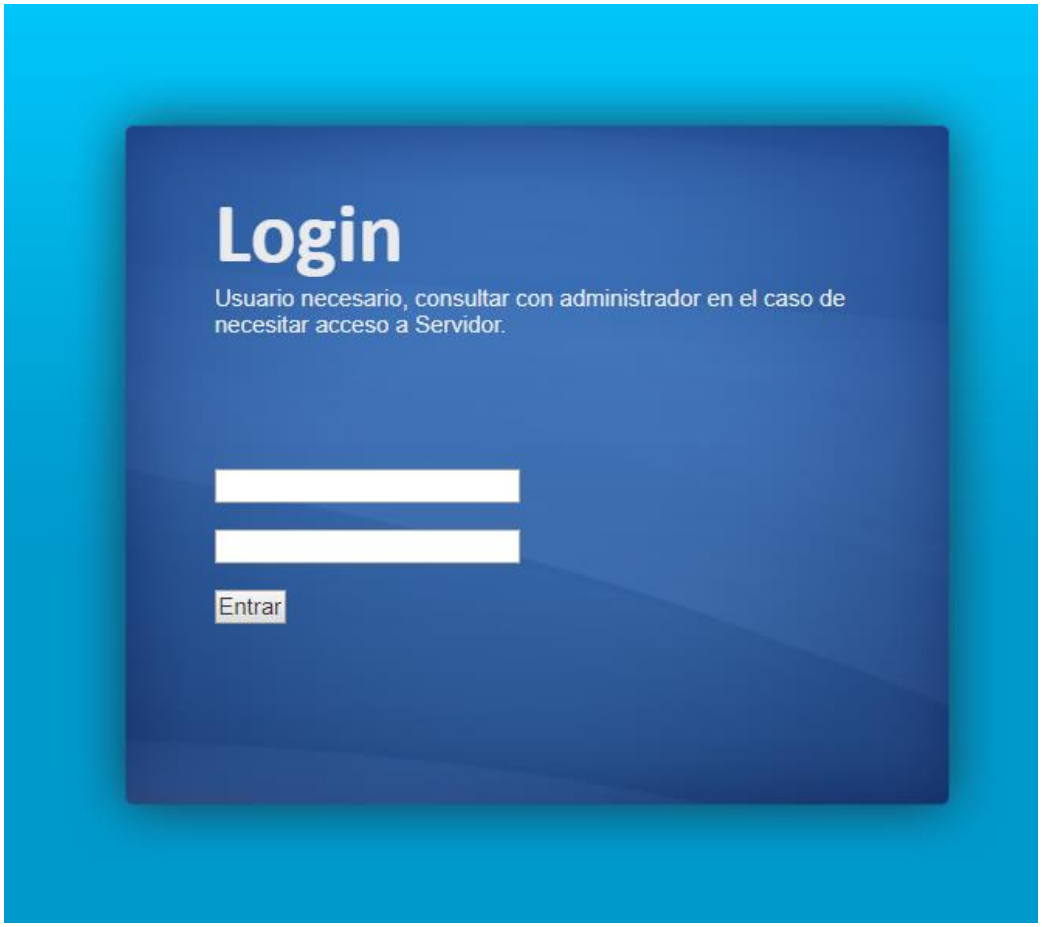

Ilustración 61:Login Interface Web

CAPÍTULO 7

7 PROBLEMAS Y SOLUCIONES

En este capítulo se desarrollan los diversos pasos que se han ido siguiendo a lo largo del proyecto en el cual se han encontrado distintos problemas conforme se integraba al proyecto las diferentes tecnologías o dispositivos, además las soluciones dadas a estos errores.

7.1 PROBLEMAS

La evaluación del proyecto y poder ir comprobando los avances y corregir los errores cometidos, ha contribuido a evitar que los errores pequeños, los cuales se pueden corregir de forma sencilla den pie a un error mayor que nos lleve a tener que invertir una mayor cantidad de tiempo en corregirlo.

Han sido necesario realizar multitud de pruebas debido a la integración de tantos dispositivos.

Aquí se comentan algunos de los problemas que han ido apareciendo a lo largo del proyecto

I. Problema de comunicación Arduino:

   Problema en la comunicación entre módulos de radio frecuencia, resultaba que al encontrarse todos los canales de comunicación leyendo, se producían lecturas erróneas, y en ocasiones entraba en bucle al no cumplirse las condiciones, puesto que no se recibía contestación por parte del nodo maestro.

   La solución, fue comenzar a plantear un nuevo código desde lo más básico dejando de lado la estructura del código inicial. Ya que se quedaba estancado al enviar datos y no recibir ninguna contestación.

Ahora la contestación si no es recibida en 200ms se sabe que no se envió correctamente el mensaje, puesto que no he encontrado en la librería usada ninguna manera de leer la ID del canal que recibe para declinar el recibimiento de datos de ese nodo.

II. ESPACIO MEMORIA, LIBERAR

Con el fin de no saturar la memoria con archivos antiguos, copias de librerías, programas o archivos de logger antiguos, lo cual puede ocasionar errores en la base de datos por falta de memoria. Se obta por investigar los archivos que no son útiles para poder liberar memoria de la Raspberry Pi.

El gestor de paquetes guarda una copia de todos los paquetes descargados en /var/cache/apt/archives, esto puede ir bien para acelerar instalaciones repetitivas, pero también ocupa un espacio en disco que muchas veces es innecesario,es posible liberar la memoria fácilmente con:

$ sudo aptitude clean

```
root@raspberrypi:/var/cache/apt/archives# ls
dos2unix_6.0-1_armhf.deb
filezilla_3.5.3-2_armhf.deb
filezilla-common_3.5.3-2_all.deb
fonts-liberation_1.07.2-6_all.deb
gnuplot_4.6.0-8_all.deb
gnuplot-nox_4.6.0-8_armhf.deb
groff_1.21-9_armhf.deb
libdjvulibre21_3.5.25.3-1_armhf.deb
libdjvulibre-text_3.5.25.3-1_all.deb
libexiv2-12_0.23-1_armhf.deb
liblensfun0_0.2.5-2_armhf.deb
liblensfun-data_0.2.5-2_all.deb
liblinear1_1.8+dfsg-1_armhf.deb
liblinear-tools_1.8+dfsg-1_armhf.deb
liblqr-1-0_0.4.1-2_armhf.deb
liblua5.1-0_5.1.5-4+deb7u1_armhf.deb
libnetpbm10_2%3a10.0-15_armhf.deb
libpcap0.8_1.3.0-1_armhf.deb
libsvm-tools_3.12-1_armhf.deb
libtinyxml2.6.2_2.6.2-1_armhf.deb
libwmf0.2-7_0.2.8.4-10.3+deb7u1_armhf.deb
libwxbase2.8-0_2.8.12.1-12_armhf.deb
libwxgtk2.8-0_2.8.12.1-12_armhf.deb
lock
netpbm_2%3a10.0-15_armhf.deb
nmap_6.00-0.3+deb7u1_armhf.deb
partial
psutils_1.17.dfsg-1_armhf.deb
python2.6_2.6.8-1.1_armhf.deb
python2.6-minimal_2.6.8-1.1_armhf.deb
python-pip_1.1-3_all.deb
python-pkg-resources_0.6.24-1_all.deb
python-setuptools_0.6.24-1_all.deb
ufraw-batch_0.18-2_armhf.deb
vsftpd_2.3.5-3_armhf.deb
root@raspberrypi:/var/cache/apt/archives#
```

Ilustración 62:Copia de paquetes descargados

Tambien se opta por liberar espacio borrando los archivos antiguos del directorio /var/log en el cual se guardan los logger de los programas como apache.

III. ARRANQUE CÓDIGO PYTHON, ERROR PUERTO SERÍE ARDUINO.

Si ha pasado mucho tiempo desde la última vez que se inició Raspberry Pi o por casualidad se ha conectado algún dispositivo más a Raspberry Pi, el Código de Python puede dar un error de puerto dirección de puerto serie donde se encuentra conectado Arduino MEGA.

Para ello se puede observar los dispositivos conectados antes de conectar Arduino MEGA, con el comando:

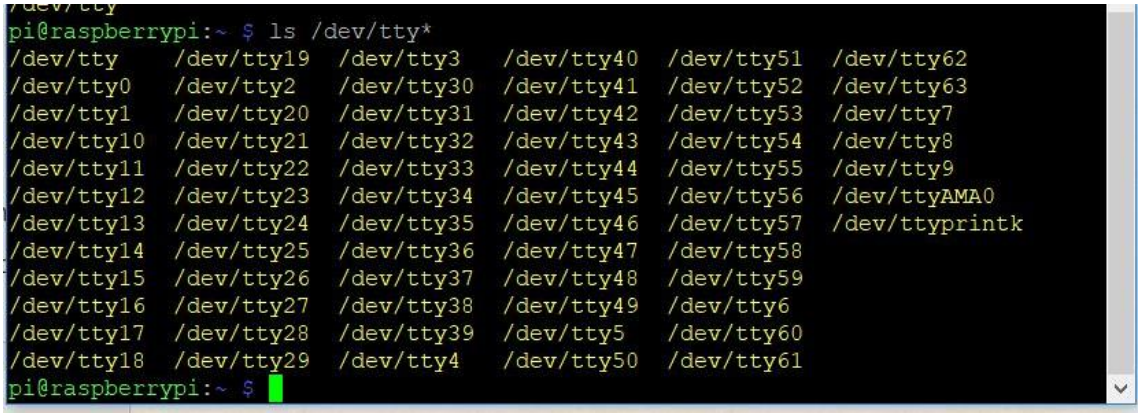

Ilustración 63:Dispositivos Conectados a Raspberry Pi sin conectar Arduino.

Ilustración 64:Dispositivos Conectados a Raspberry Pi al conectar Arduino.

Como se puede apreciar en la anterior ilustración, la señalada con el recuadro es la de Arduino MEGA , la cual será la que utilizaremos para recibir y trasmitir las tramas de comunicaciones .

CAPÍTULO 8

8 CONCLUSIONES Y TRABAJOS FUTUROS

8.1 CONLUSIONES

La realización de este trabajo ha sido de gran ayuda para poder aprender diferentes lenguajes de programación y consolidar el uso de diferentes tecnologías.

Gracias a la realización de este trabajo, he adquirido una serie de competencias muy importantes. Una capacidad para analizar los errores, comprenderlos y solventarlos.

Los objetivos de este proyecto el cual era la implementación de un sistema de monitorización de huertos urbanos de bajo costo, ha sido desarrollado de la forma explicada en los anteriores puntos.

Todos los procesos han sido desarrollados cumpliendo los objetivos marcados, además de a ver podido añadir funcionalidades al proyecto, las cuales se detallan a continuación:

- Gráficas para mejorar la interface Web y facilitar al cliente el reconocimiento de los datos
- Subida de datos a la nube automáticamente
- Datos guardados en formato de texto en la memoria interna de Raspberry
- Obtener un trabajo con una gran capacidad de escalabilidad para en un futuro poder realizar una ampliación con facilidad
- Control de válvulas desde interface Web

Se puede concluir que las exigencias expuestas en un principio se han cumplido además de poder añadir al trabajo algunos extras para mejorar la funcionalidad del proyecto.

8.2 PRESUPUESTO

En este capitulo se puede observar la repercusión económica de cada uno de los elementos utilizados en este proyecto.

	CANTIDAD	PRECIO UD (€)	TOTAL (€)
Raspberry			
Raspberry Pi Model B+	1	33.00 €	33.00 €
Fuente de alimentacion 5 V 3A	1	5.00 €	5.00 €
Kingston MicroSDHC 8GB	1	6.80 €	6.80 €
Total Raspberry			**44.80 €**

Tabla 11:Presupuesto Raspberry pi

	CANTIDAD	PRECIO UD (€)	TOTAL (€)
Arduino Uno			
Arduino Uno	3	22.00 €	66.00 €
Fuente de alimentacion Arduino ,9V 1A	3	3.95 €	11.85 €
Total Uno			**77.85 €**

Tabla 12:Presupuesto Arduino Uno

	CANTIDAD	PRECIO UD (€)	TOTAL (€)
Arduino MEGA			
Arduino MEGA 2560 REV3	1	38.50 €	38.50 €
Cable Usb 2.0 tipo A a Tipo B macho	1	3.00 €	3.00 €
Total Otros			**41.50 €**

Tabla 13:Presupuesto Arduino MEGA

	CANTIDAD	PRECIO UD (€)	TOTAL (€)
Nodos			
Sensor Temperatura y Humedad Ambiente. DHT22	3	2.38 €	7.14 €
Sensor Temperatura y Humedad Sustrato. SHT10	3	25.00 €	75.00 €
Caja Estanca 150 x 150 x 80 mm	3	5.00 €	15.00 €
Total Nodos			**97.14 €**

Tabla 14:Presupuesto Nodos

	CANTIDAD	PRECIO UD (€)	TOTAL (€)
Sensores Otros			
Modulo Reloj DS1307 RTC	1	2.20 €	2.20 €
Trascentor NRF24L01	4	2.50 €	10.00 €
Rele 4 canales salida 230V	1	5.00 €	5.00 €
Sensor Irradiación solar S1087-01	1	4.00 €	4.00 €
Total Otros			**21.20 €**

Tabla 15:Presupuesto Otros

Categoría	Importe
Raspberry	44.80 €
Arduino Uno	77.850 €
Arduino Mega	41.50 €
Nodos	97.140 €
Otros	21.20 €
Importe Total	**282.490 €**

Tabla 16:Presupuesto Importe final

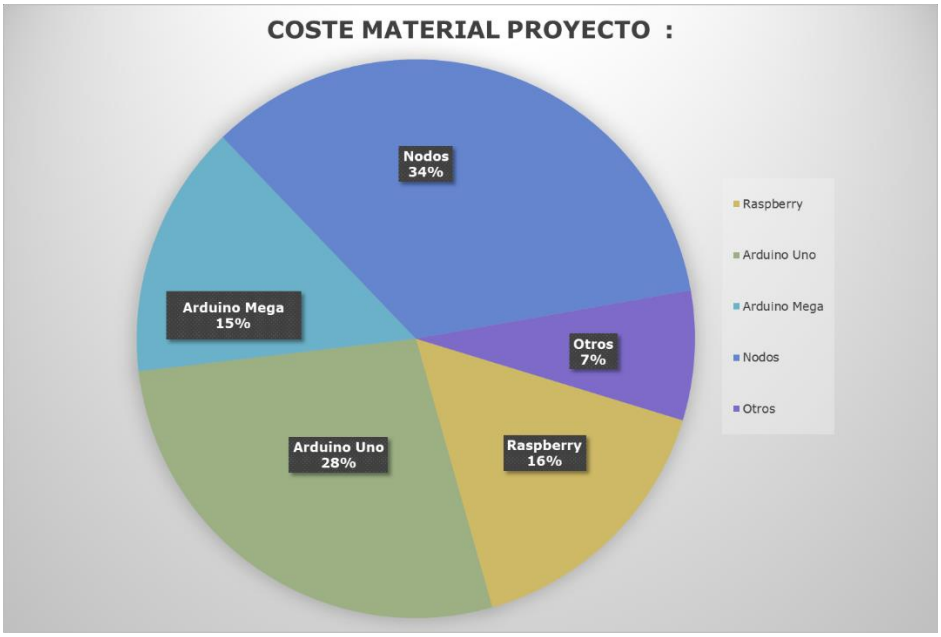

Ilustración 65:Diagrama Sectores Presupuesto

Como se puede analizar de la ilustración anterior la mayor parte del presupuesto recae en los sensores utilizados para cada nodo.

El análisis revela el impacto económico del 34% del coste final, invertido en la instrumentación de cada nodo, el precio aumenta considerablemente al haber elegido sensor de temperatura y humedad Sustrato el módulo" SHT10". Seleccionado por su alta calidad y precisión, además de ser un módulo compacto.

En la tabla 16, se muestra el Importe final del proyecto finalizado, incluyendo 3 nodos. La siguiente tabla muestra el importe final del proyecto, pero con un nodo.

Categoría	Importe
Raspberry	44.80 €
Arduino Uno	25.950 €
Arduino Mega	41.50 €
Nodos	32.380 €
Otros	16.20 €
Importe Total	**160.830 €**

Tabla 17:Presupuesto Importe final, 1 nodo

Por cada nodo se sumará al importe Total mostrado en la "Tabla 17" un coste de **60.830 €.**

8.3 TRABAJOS FUTUROS

Gracias a trabajar con Raspberry Pi y Arduino, se puede observar una gran escalabilidad y flexibilidad, para poder seguir ampliando las funcionalidades de este proyecto.Algunas de las mejoras que se podrían realizar modificando pequeñas partes del código o cambiando parte de hardware son las siguientes.

- El sumidero “nodo maestro” podrá ser conectado a internet ya sea vía Wi-Fi o Ethernet y envía los datos al servidor creado con Rapberry Pi, para poder hacer independiente ambas partes, de esta forma se eliminaría la conexión serie.
- Realizar comunicación con la API de Twitter y enviar avisos de alarma cada vez que se abran las válvulas, estas notificaciones serán públicas o privadas, pero llegaran instantáneamente al teléfono del cliente.
- Ampliación de sensores en nodos.
- Realizar PCB con ATmega328 y eliminar Arduinos UNO.
- Creación de librerías,simplificacndo el código principal de

ANEJO I

PUESTA EN MARCHA

Para la puesta en marcha de la instalación se debería seguir unos pasos para corroborar que todo va según lo previsto, por esta razón se han preparado unos sketch's para poder probar la correcta conexión de los dispositivos en los nodos.

Para comenzar las pruebas con Arduino se debe recordar que la velocidad debe estar a 9600 baudios y elegir la tarjeta adecuada a la cual se le está cargando el programa.

Lo primero será probar la comunicación entre los módulos de radiofrecuencia, con este Sketch se puede comprobar el funcionamiento correcto de los módulos además de corroborar si se ha realizado las conexiones correctas.

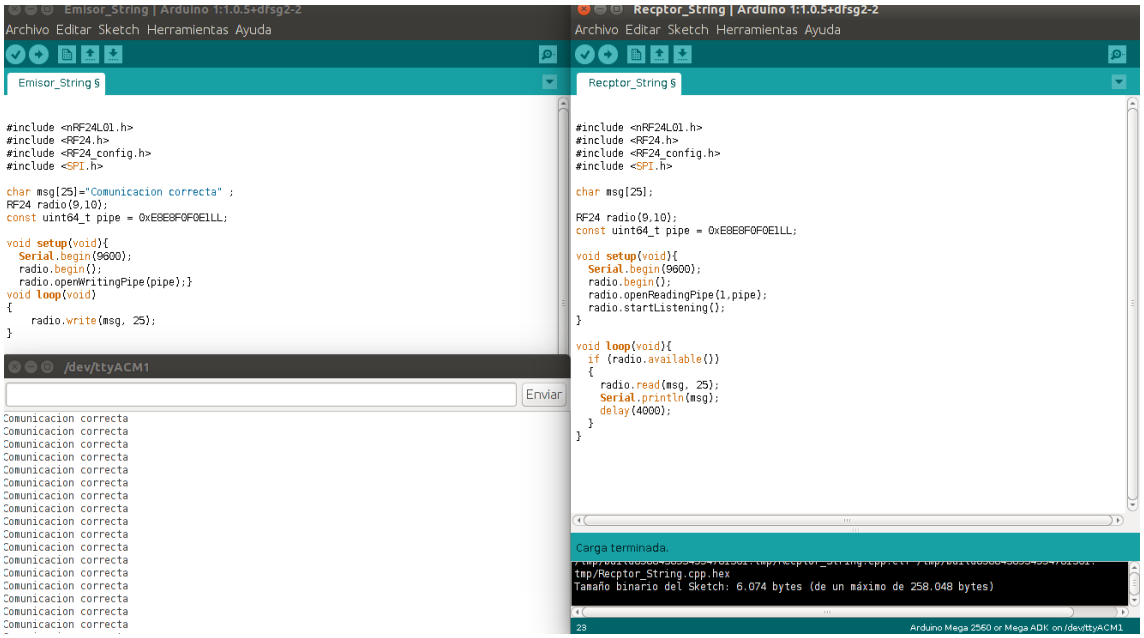

Ilustración 66:Prueba Comunicación

Una vez se tiene probada la comunicación de todos los nodos, se comprueba la correcta instalación de los sensores de Humedad DHT22, en los nodos.

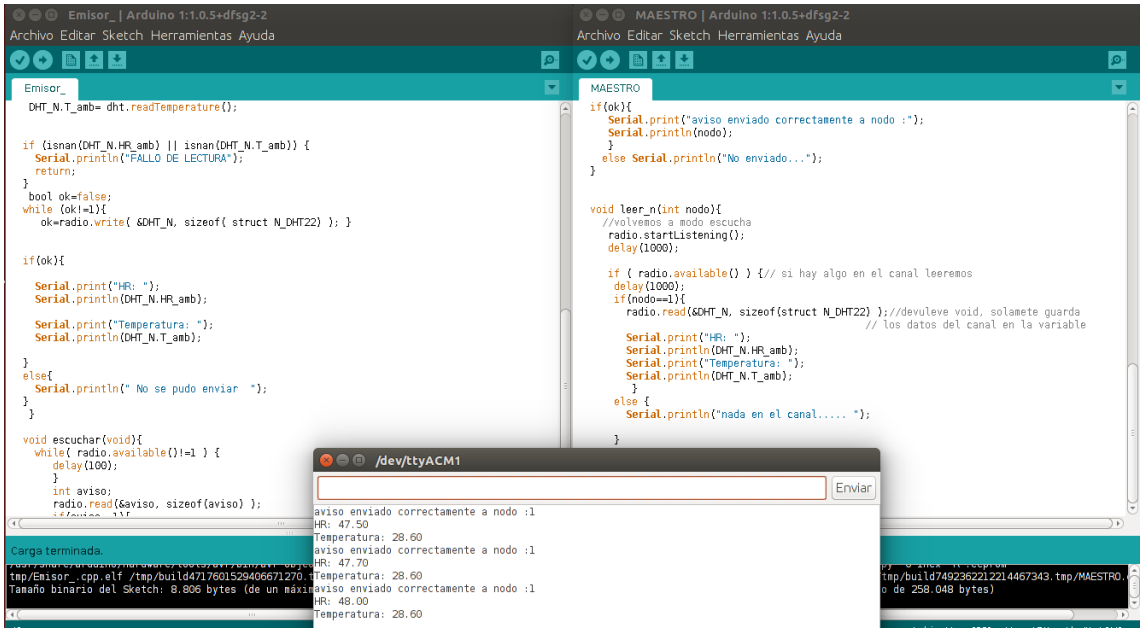

Ilustración 67:Prueba Sensor DHT22

Para comprobar el correcto funcionamiento del sensor SHT10, se ofrece el Sketch ofrecido por el fabricante, para comprobar su buena conexión y funcionamiento.

En este caso la prueba se realiza individualmente en cada nodo.

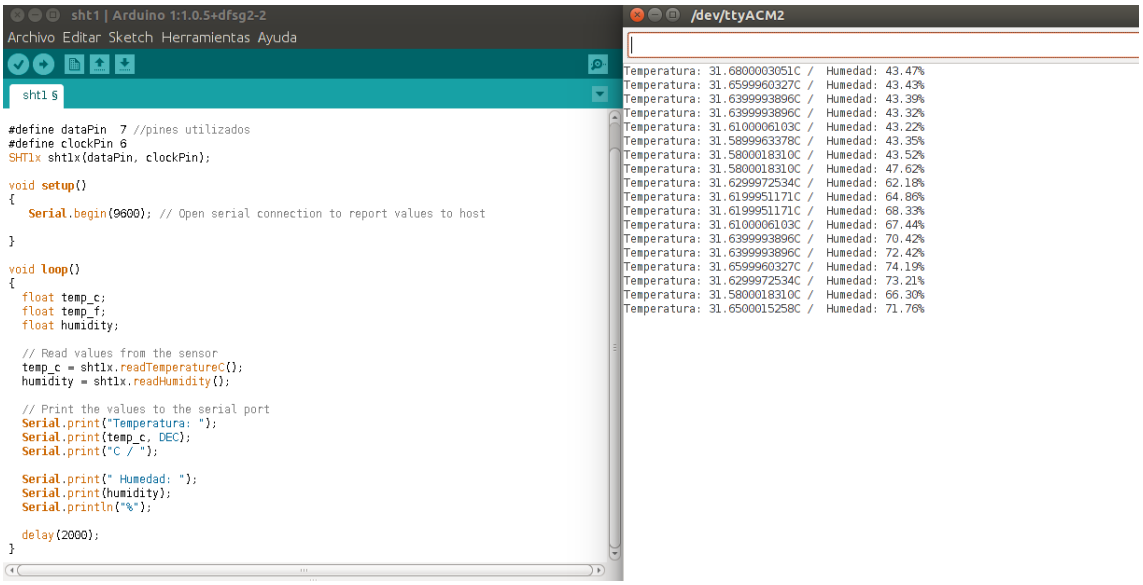

Ilustración 68:Prueba de sensor SHT11

Para la configuración del reloj en tiempo real, lo primero es cargar en la memoria la hora y fecha. Para eso se dispone de un sketch que permite cargar la hora exacta y otro sketch para comprobar la que la carga ha sido correcta.

Primero cargamos la hora, con el sketch con nombre “RTC”.

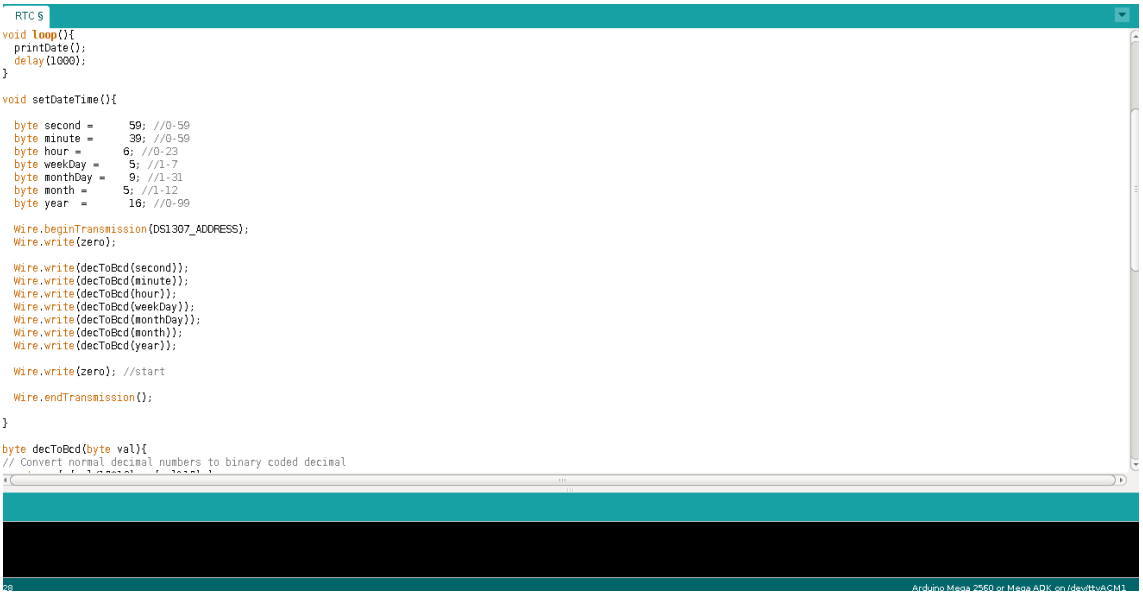

Ilustración 69:Carga fecha en RTC

Seguidamente comprobamos que se carga correctamente la hora en el sensor RTC, con el Sketch utilizado para leer los datos de la memoria.

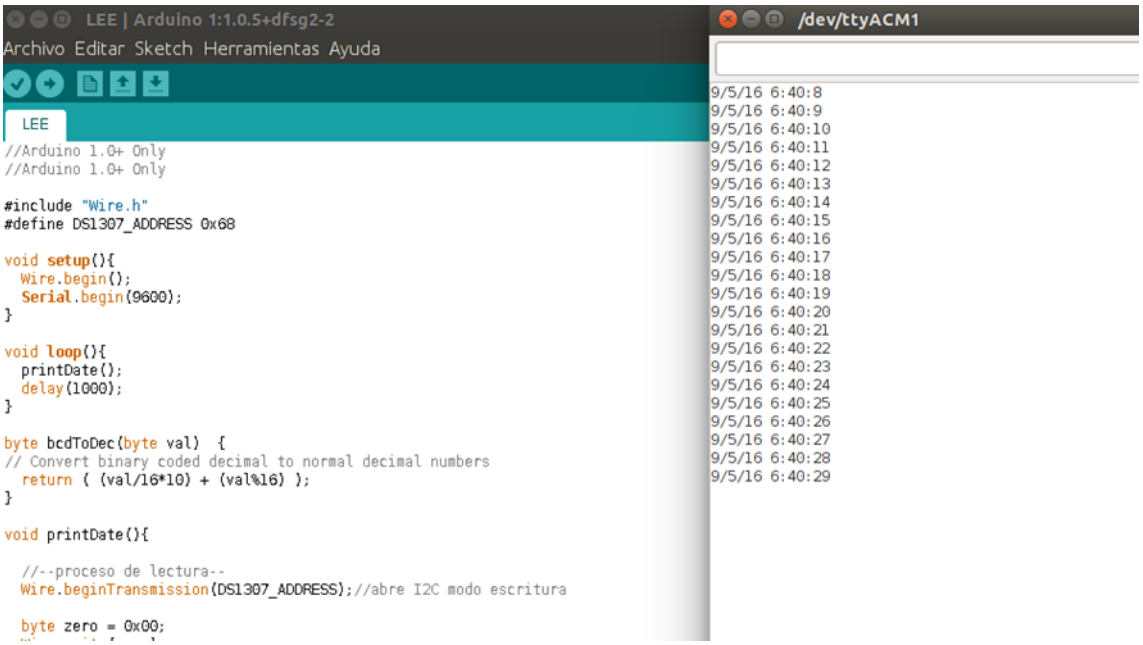

Ilustración 70:Lectura RTC

Una vez comprobadas las conexiones y el correcto funcionamiento de los sensores, además de tener configurado el servidor y nuestra Rasperry Pi con los pasos seguidos durante la Memoria de este proyecto , el proyecto estará preparado para la puesta en marcha.

A falta de copiar los archivos en el servidor apache, en la carpeta /var/www/ , se debe de cargar los scketch finales en los Arduinos y arrancar el script de Python para comenzar la comunicación.

ANEJO II

CODIGOS DE SISTEMA

Seguidamente se exponen los códigos desarrollados en este proyecto, los lenguajes usados son los siguientes:

- PYTHON
- PHP
- HTML
- C/C++ (ARDUINO)

I. CÓDIGOPYTHO

```
#-*- coding: utf-8 -*-
# -*- coding: iso-8859-1 -*-
#-*- coding:latin-1 -*-

import serial  #libreria para arduino , poder leer puerto serial
import MySQLdb
import os,time
import Func_Emails
#T tiempo=[dia,mes,año,hora,minuto,seg]
T_tiempo=[]
for i in range(6):
    T_tiempo.append(0)


#T dia=[nodo1,nodo2,nodo3]      utilizadas para saber el nombre del txt antiguo y comparar
si cambia de nombre
T_dia=[]
T_mes=[]
T_ano=[]
for i in range(3):
    T dia.append(0)
    T_mes.append(0)
    T_ano.append(0)
#M_N1=[T_ini,Tamb,Hamb,Tsut,Hsut,C_Tamb,C_Hamb,C_Tsut,C_Hsut]      Para subir la media de
los nodos , contadores ,Tini=tiempo de primer valor
M_N1=[]             #MEDIA NODO1
M_N2=[]
M_N3=[]             # NODO 3
for i in range(9):
    M_N1.append(0)
    M_N2.append(0)
    M_N3.append(0)
HR1=[]
HR2=[]
HR3=[]
#HR=[HRmin,HRmax]
for i in range(2):
        HR1.append(0)
        HR2.append(0)
        HR3.append(0)

# VARIABLES GLOBALES DE BASE DE DATOS
data=[]
for i in range(3):
        data.append(0)  #array que devuelve sel SELECT para HR  data[]=(id,HRmin,HRmax)

DB_HOST = 'localhost'
DB_USER = 'root'
DB_PASS = 'RASPBERRY'
# ********************FUNCION ESCRIBIR BASE DE DATOS *****************

def run_query(query=' ' , DB_NAME=' ', temp=' '):

    datos = [DB_HOST, DB_USER, DB_PASS, DB_NAME]
    db = MySQLdb.connect(datos[0], datos[1], datos[2],datos[3]) # Conecta a la base de datos
    cursor = db.cursor()          # preparamos  cursor


    if temp=='r':
        try:
            cursor.execute(query)
            data = cursor.fetchall() # Este método obtiene todos los registros de un
            conjunto de resultados de una consulta a la BD


        except:
            print "Error , no se obtienen datos"
```

```
    elif temp=='w':
            try:

            cursor.execute(query)
            db.commit()              # Hacer efectiva la escritura de datos
            print "subida bien"
        except:
            print "Error Escritura"
            db.rollback()    #revierte el error

    cursor.close()               # Cerrar el cursor
    db.close()                   # Cerrar la conexión
        if temp=='r':
        return data[0]

#***********TRATAMIENTO DATOS DE NODO ***********

def nodos (temp,tiempo,estado):
    T_tiempo[0],void,tiempo=tiempo.partition(":")
    T_tiempo[1],void,tiempo=tiempo.partition(":")
    T_tiempo[2],void,tiempo=tiempo.partition(":")
    T_tiempo[3],void,tiempo=tiempo.partition(":")
    T_tiempo[4],void,T_tiempo[5]=tiempo.partition(":")
    map(int,T_tiempo)                        #Aseguramos que que no los trate como string ,
    mapea a int

    nodo,void,temp=temp.partition(":")
    Tamb,void,temp=temp.partition(":")
    Hamb,void,temp=temp.partition(":")
    Tsut,void,Hsut=temp.partition(":")
    if nodo == '1':
        N= 'nodo1'
        MEDIA= M_N1

    elif nodo == '2':
        N= 'nodo2'
        MEDIA= M_N2
    else  :
        N= 'nodo3'
        MEDIA= M_N3
                        #En MEDIA se encuentra Copiado el array de Valores  del nodo
                        correspondiente
    gauge(N,float(Tamb),float(Hamb),float(Tsut),float(Hsut))
    graficas(int(nodo),N,MEDIA,float(Tamb),float(Hamb),float(Tsut),float(Hsut))
    txt(int(nodo),N,float(Tamb),float(Hamb),float(Tsut),float(Hsut),int(estado))


#***********FUNCIONE PARA GRAFICAS ***********
def graficas(nodo,N,MEDIA,Tamb,Hamb,Tsut,Hsut):
    DB_NAME=N
                    #aclaraciones:
                        #si valor (dato) es  cero no  se suma, por lo tanto no incrementa
                        :corregimos fallos  de esta manera
                        #En T_ini guardamos hora  de inicio  y comparamos con hora actual
                        #Contador nos ayudara a saber entre cuanto hay que dividir para
                        hacer la media
                        #Todo puesto a cero menos T ini  el cual nos dice desde cuando se
                        comienza ha muestrear
    #primera Inicializacion
    if (MEDIA[0]==0 and T_mes[nodo-1]==0 ):

        ##if (T_mes[nodo-1]==0 ): # si se dan ambas condiciones seria la primera vez que
        este nodo envia copiamos hora

        MEDIA[0]=T_tiempo[3]
        T_dia[int(nodo)-1]=T_tiempo[0]
        T_mes[int(nodo)-1]=T_tiempo[1]
```

```
            T_ano[int(nodo)-1]=T_tiempo[2]


        #Si es cero, algún problema hay en la lectura o envio de datos
        #Por lo tanto si es cero no incrementariamos la media
        if Tamb!=0:
            MEDIA[1]=MEDIA[1]+Tamb
            MEDIA[5]+=1                #incrementos contadores , para la media
        if Hamb!=0:
            MEDIA[2]=MEDIA[2]+Hamb
            MEDIA[6]+=1
        if Tsut!=0:
            MEDIA[3]=MEDIA[3]+Tsut
            MEDIA[7]+=1
        if Hsut!=0:
            MEDIA[4]=MEDIA[4]+Hsut
            MEDIA[8]+=1

        if T_tiempo[3]!=MEDIA[0]:#si la hora es distinta significa que ha pasado una hora

            Tamb=MEDIA[1]/MEDIA[5]     #VALORES FINALES DE MEDIA, QUE SE SUBIRAN A BD
            Hamb=MEDIA[2]/MEDIA[6]
            Tsut=MEDIA[3]/MEDIA[7]
            Hsut=MEDIA[4]/MEDIA[8]

            for i in range(8):
                MEDIA[i+1]=0   #ponemos a 0 todos, menos el valor de la hora , puesto que lo
                usaremos para entrar a la tabla , como id


            query = "UPDATE S_amb SET humedad='%f',temperatura='%f', `hora`=NOW()   WHERE id=
            '%i' " % (Hamb, Tamb,int(MEDIA[0]))
            run_query(query,DB_NAME,'w')  #se pasa la estructura para la BD ademas de el nodo
            que selecionara


            query = "UPDATE S_sut SET humedad='%f',temperatura='%f', `hora`=NOW()  WHERE id=
            '%i' " % (Hsut, Tsut,int(MEDIA[0]))
            run_query(query,DB_NAME,'w')  #se pasa la estructura para la BD ademas de el nodo
            que selecionara


                    print "dropbox"
                    os.system(" bash
                    /home/pi/Desktop/Dropbox/Dropbox-Uploader/dropbox_uploader.sh upload
                    /home/pi/Desktop/NODO/%s/%i:%i:%i.txt  /home/NODO/%s/%i:%i:%i.txt"%(N,int(
                    T_dia[nodo-1]),int(T_mes[nodo-1]),int(T_ano[nodo-1]),N,int(T_dia[nodo-1]),int
                    (T_mes[nodo-1]),int(T_ano[nodo-1])))


            MEDIA[0]=T_tiempo[3]    #Colocamos el nuevo tiempo

        #calculamos las medias , y MEDIA[0] guarda el valor de la hora anterior despues de subirlo
        # a BD guardamos de nuevo la hora siguiente


    def txt(nodo,N,Tamb,Hamb,Tsut,Hsut,estado):



        fo = open("/home/pi/Desktop/NODO/%s/%i:%i:%i.txt"%(str(N),int(T_tiempo[0]),int(T_tiempo[1
        ]),int(T_tiempo[2])), "a")
        n="Tamb= %f  Hamb= %f  Tsut= %f  Hsut= %f  estado valvula:%i Hora de muestra %i:%i:%i
        (H:Min:Seg) \n"%(Tamb,Hamb,Tsut,Hsut,estado,int(T_tiempo[3]),int(T_tiempo[4]),int(
        T_tiempo[5]))
        fo.write(n)
        fo.close()           #CERRAMOS OBJETO
```

```
                        # con 'nodo' controlamos si cambia el archivo

    if T_tiempo[0]!=T_dia[nodo-1]:   #si cambia el dia lo subimos a dropbox , puesto que ha
    cambiado de archivo
        print "dropbox"
        os.system(" bash /home/pi/Desktop/Dropbox/Dropbox-Uploader/dropbox_uploader.sh
        upload  /home/pi/Desktop/NODO/%s/%i:%i:%i.txt  /home/NODO/%s/%i:%i:%i.txt"%(N,int(
        T_dia[nodo-1]),int(T_mes[nodo-1]),int(T_ano[nodo-1]),N,int(T_dia[nodo-1]),int(T_mes[
        nodo-1]),int(T_ano[nodo-1])))
        T_dia[nodo-1]=T_tiempo[0]
        if T_mes[nodo-1]!=T_tiempo[1]:
            T_mes[nodo-1]=T_tiempo[1]
        if T_ano[nodo-1]!=T_tiempo[1]:
            T_ano[nodo-1]=T_tiempo[1]

    ##creamos txt ponemos fecha  e ese dia ,  y vamos abriendo y escribiemdo por nodos
    #if se debe subir a dropbox porque se  a cambiado de dia pues llamamos a funcion para
    que lo suba


#*********ACTUALIZA_GAUGE
def gauge(N,Tamb,Hamb,Tsut,Hsut):
    DB_NAME=N                    #Para seleccionar BD
    a = [Tamb,Hamb,Tsut,Hsut]    # tomamos len(a)  que seran 4
    for i in range(len(a)):
        query = "UPDATE gauge SET valor= %f, `tiempo`=NOW() WHERE nombre = %i" % (a[i],int(i+
        1))
        run_query(query,DB_NAME,'w')

#*********Lectura HRmin HRmax

def HR_BD (i):
    DB_NAME='actuadores'
    query = "SELECT   `id`, `HRmin`, `HRmax`    FROM `offset` where id=%i" % (i+1)
    data=run_query(query,DB_NAME,'r')

        if  i == 0:
            HR1[0] = data[1]
            HR1[1] = data[2]
        elif i == 1:
                HR2[0] = data[1]
        HR2[1] = data[2]

        else :
                HR3[0] = data[1]
                HR3[1] = data[2]



#*********ACTUALIZA_VALVE
def valve(estado):
    DB_NAME= 'actuadores'
    query = "UPDATE `valve`  SET salida='%i' WHERE id = %i" % (int(estado), i+1)
    run_query(query,DB_NAME,'w')
#*********ACTUALIZA_SENSOR RADIACION SOLAR
def S_sensor(estado):
    DB_NAME= 'actuadores'
    query = "UPDATE `Irradiacion`  SET valor='%f' WHERE id =1" % (float(estado))
    run_query(query,DB_NAME,'w')


#****-----------------COMIENZA EL
CODIGO************************************************************

arduino = serial.Serial('/dev/ttyACM0',9600) # Serial de arduino
time.sleep(80) #espera que arranque mysql
arduino.flush()
```

```
while True :
    time.sleep(5) ##no saturar pantalla .............pruebas
    for i in range(3) : #LLama a función para leer los  valores introducidos por usuario de
    HR
        HR_BD(i)
    #time.sleep(2)
        for i in range(2):  #Peticion de datos
        if(i==0):
            a=int(HR1[0])
            b=int(HR1[1])
        if(i==1):
            a=int(HR2[0])
            b=int(HR2[1])
        c=("N:%i:%i:%i:"%(i+1,a,b))
        arduino.write(c)
        time.sleep(1)
        print c
        #Asegura que a contestado al mensaje
        while  True:
            #trama-->
            N:dia:mes:año:h:min:seg;nodo:Tamb:Hramb:Tsut:Hrsut;estado_valvula:IRRADIACION:
            temp,void,void=arduino.readline().partition("\n") #Lee puerto serie y realiza
            particion desde caracter "\n"
            cabezera,void,temp=temp.partition(":")
            if cabezera == 'N':
                print temp
                tiempo,void,temp=temp.partition(";")            #Obtenemos valores de tiempo
                temp,void,temp2=temp.partition(";")             #Separamos Valores sensados
                y valor actual de valvula (temp2)
                estado_valv,void,temp2=temp2.partition(":")
                S_radiacion,void,void=temp2.partition(":")
                #llamadas a función
                valve(estado_valv)
                S sensor(S radiacion)
                nodos(temp,tiempo,estado_valv)
                break
            elif cabezera == 'E':
                msg_Emails = " Error Comunicación en nodo:'%i' Revisar Estación" % (int(i+1))
                print msg_Emails
                username = 'tfg.upc.pmm@gmail.com'
                password = '[redacted]'
                toaddrs = 'pedrete.mayordomo@gmail.com'
                Func_Emails.Emails(username,toaddrs,password,msg_Emails)
                break
arduino.close()
```

II. CODIGOS PHP

i. Index

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <?php
        $con=mysqli_connect("localhost","root","raspberry","actuadores");
        if(mysqli_connect_error()){
        echo "No accede a la base :".mysqli_connect_error();}

    ?>

    <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <link rel="stylesheet" media="screen" type="text/css" title="style" href="style.css" />
    <title>CASA BALCACER</title>
    </head>
    <body>
    <div
    id="holder">









    <div id="footer_terms">
    <h1><a href="http://upct.es/">
    <strong>UNIVERSIDAD POLITECNICA DE CARTAGENA </strong></a></h1>
    </div>
    <!--END OF TERMS OF USE-->
    <!-- HEADER -->
    <div id="header">
    <a href="index.html">
    <img src="images/logo_gif.gif" alt="logo" width="150" height="150"/>
    </a> </div>
    <!-- END HEADER -->
    <div id="shadow">
    <!-- MENU -->
    <ul id="menu">
    <li> <a href="index.php">INICIO</a> </li>
    <li> | </li>
    <li> <a href="nodo.php">NODO1</a> </li>
    <li> | </li>
    <li> <a href="nodo2.php">NODO2</a> </li>
    <li> | </li>
    <li> <a href="nodo3.php">NODO3</a> </li>
    <li> | </li>
    <li> <a href="#">Memoria</a> </li> <!-- memoria en html </div>-->
    <li> | </li>
    </ul>
    <!-- END MENU -->
    <div id="edito">
    <h1> <em>Archivos </em>  </h1>

    <a
    href="https://www.dropbox.com/sh/7ttkwjmccsl4t3o/AACbW406ZvwWA16b9fX-Cq87a?dl=0"><img
    src="images/Dropbox.gif"  alt="Dropbox" width="70" height="70" /></a>
    <P>
    </div>
```

```
<div id="toal"> </div> <!-- Separa ambas partes-->
<!-- CONTENT -->

<div id="content">
<!-- ****************************************NODO
1********************************** -->
<div>
    <h1>NODO 1</h1>

    <?php
        $result = mysqli_query($con,"SELECT * FROM valve where id='1'  ");//selecciona
        tabla y busca fila con id=1
        $row = mysqli_fetch_row($result);

        if ($row[1]=="0") {  // para cuando esta cerrada
        $estado1 = "images/cerrada.gif";}

        else  {
        $estado1 = "images/abierta.gif";}
        //***************************Valores de HRmin y HR max ********************
        //NODO1
        $result = mysqli_query($con,"SELECT *   FROM `offset` where id='1'  ");
        $row = mysqli_fetch_row($result);
        $hrmax1 = $row[2];
        $hrmin1 = $row[1];
        //NODO2
        $result = mysqli_query($con,"SELECT *   FROM `offset` where id='2'  ");
        $row = mysqli_fetch_row($result);
        $hrmax2 = $row[2];
        $hrmin2 = $row[1];
        //NODO3
        $result = mysqli_query($con,"SELECT *   FROM `offset` where id='3'  ");
        $row = mysqli_fetch_row($result);
        $hrmax3 = $row[2];
        $hrmin3 = $row[1];


    ?>


    <center><p><IMG SRC="/<?php echo "$estado1"; ?>" WIDTH=140 HEIGHT=140
    ALT="family"></p></center>
    <!--********** HRMAX*******1******** -->
    <form action="/PHPs/HRmax1.php" method="POST">

        <label for="valor">valor HR maximo</label>
        <input type="text"   value=" actual:<?php echo "$hrmax1"; ?>" name="valor"
        class="form-input" required/>
        <!-- name="valor" , es la variable que enviamos por POST -->

        <center> <input class="form-btn" name="submit" type="submit" value="enviar" />
        </center>

    </form>
    <!--************* HRMIN***************1******* -->
    <form action="/PHPs/HRmin1.php" method="POST">

        <label for="valor">Valor HR  minimo</label>
        <input type="text"  value=" actual:<?php echo "$hrmin1"; ?>" name="valor"
        class="form-input" required/>
        <!-- name="valor" , es la variable que enviamos por POST -->

        <center> <input class="form-btn" name="submit" type="submit" value="enviar" />
        </center>

    </form>


</div>
```

```
<!-- ******************************************NODO
2********************************** -->
<div id="vertical_barr">
    <h1>NODO 2 </h1>

    <?php
        $result = mysqli_query($con,"SELECT * FROM valve where id='2'  ");//selecciona
        tabla y busca fila con id=1
        $row = mysqli_fetch_row($result);

        if ($row[1]=="0") {  // para cuando esta cerrada
        $estado2 = "images/cerrada.gif";}

        else  {
        $estado2 = "images/abierta.gif";}
    ?>


    <center><p><IMG SRC="/<?php echo "$estado2"; ?>" WIDTH=140 HEIGHT=140
    ALT="family"></p></center>
    <!--********** HRMAX*******2******** -->
    <form action="/PHPs/HRmax2.php" method="POST">

        <label for="valor">valor HR maximo</label>
        <input type="text"   value=" actual:<?php echo "$hrmax2"; ?>" name="valor"
        class="form-input" required/>
        <!-- name="valor" , es la variable que enviamos por POST -->

        <center> <input class="form-btn" name="submit" type="submit" value="enviar" />
        </center>

    </form>
    <!--************* HRMIN***************1******* -->
    <form action="/PHPs/HRmin2.php" method="POST">

        <label for="valor">Valor HR  minimo</label>
        <input type="text"  value=" actual:<?php echo "$hrmin2"; ?>" name="valor"
        class="form-input" required/>
        <!-- name="valor" , es la variable que enviamos por POST -->

        <center> <input class="form-btn" name="submit" type="submit" value="enviar" />
        </center>

    </form>
</div>

<!-- ******************************************NODO
3********************************** -->
<div>
    <h1>NODO 3</h1>

    <?php
        $result = mysqli_query($con,"SELECT * FROM valve where id='3'  ");//selecciona
        tabla y busca fila con id=1
        $row = mysqli_fetch_row($result);

        if ($row[1]=="0") {  // para cuando esta cerrada
        $estado3 = "images/cerrada.gif";}

        else  {
        $estado3 = "images/abierta.gif";}
    ?>
    <center><p><IMG SRC="/<?php echo "$estado3"; ?>" WIDTH=140 HEIGHT=140
    ALT="family"></p></center>
    <!--********** HRMAX*******3******** -->
    <form action="/PHPs/HRmax3.php" method="POST">

        <label for="valor">valor HR maximo</label>
```

```
            <input type="text"   value=" actual:<?php echo "$hrmax3"; ?>" name="valor"
            class="form-input" required/>
            <!-- name="valor" , es la variable que enviamos por POST -->

            <center> <input class="form-btn" name="submit" type="submit" value="enviar" />
            </center>

        </form>
        <!--************ HRMIN***************1******* -->
        <form action="/PHPs/HRmin3.php" method="POST">

            <label for="valor">Valor HR  minimo</label>
            <input type="text"  value=" actual:<?php echo "$hrmin3"; ?>" name="valor"
            class="form-input" required/>
            <!-- name="valor" , es la variable que enviamos por POST -->

            <center> <input class="form-btn" name="submit" type="submit" value="enviar" />
            </center>

            </form>
    </div>
    </div>

    <!-- END CONTENT -->
    <div class="clear"></div>
    <!-- END SHADOW -->
    </div>
    <!-- FOOTER -->
    <div id="footer">
    <p>contacto  I <a href="pedrete.mayordomo@gmail.com"></a>pedrete.mayordomo@gmail.com </p>
    </div>
    <!-- END FOOTER -->
    </div>
    <!-- END HOLDER -->
    </body>
</html>
```

ii. Nodos

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

        <?php   //*************************Ciclo For para recoger valores de gauge***********

            $con=mysqli_connect("localhost","root","raspberry","nodo1"); //Base de datos
            if(mysqli_connect_error()){
            echo "No accede a la base :".mysqli_connect_error();}

        ?>
        <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <link rel="stylesheet" media="screen" type="text/css" title="style" href="style.css"
        />
        <title>NODO1</title>

        <script type="text/javascript"
        src="https://www.gstatic.com/charts/loader.js"></script>
<!-- **********FUNCIONES DE GAUGE ********  -->
        <script type="text/javascript">
        google.charts.load('current', {'packages':['gauge']});
        google.charts.setOnLoadCallback(drawChart);
        function drawChart() {

            var data = google.visualization.arrayToDataTable([
            ['Label', 'Value'],

            <?php
                $result = mysqli_query($con,"SELECT * FROM gauge where nombre='1'
                ");//selecion de tabla
                $row = mysqli_fetch_array($result)

            ?>

            ['T_ambiente',<?php echo $row[1] ;?>]


            ]);

            var data2 = google.visualization.arrayToDataTable([
            ['Label', 'Value'],

            <?php
               $result = mysqli_query($con,"SELECT * FROM gauge where nombre='3'
               ");//selecion de tabla
             $row = mysqli_fetch_array($result)
            ?>

            ['T_sustrato',<?php echo $row[1] ;?>]

            ]);

            var data3 = google.visualization.arrayToDataTable([
            ['Label', 'Value'],
            <?php
                $result = mysqli_query($con,"SELECT * FROM gauge where nombre='2'
                ");//selecion de tabla
                $row = mysqli_fetch_array($result)
            ?>

            ['HR_ambiente',<?php echo $row[1] ;?>]

            ]);

            var data4 = google.visualization.arrayToDataTable([
            ['Label', 'Value'],
            <?php
                $result = mysqli_query($con,"SELECT * FROM gauge where nombre='4'
```

```
            ");//selecion de tabla
            $row = mysqli_fetch_array($result)
        ?>

        ['HR_sustrato',<?php echo $row[1] ;?>]
        ]);

        <!--opciones las mismas -->
        var options = {
            width: 150, height: 150,
            redFrom: 90, redTo: 100,
            yellowFrom:75, yellowTo: 90,
            minorTicks: 5
            };

        var chart = new google.visualization.Gauge(document.getElementById('chart_div'));
        chart.draw(data, options);


        var juan = new google.visualization.Gauge(document.getElementById('chart_div2'));
        juan.draw(data2, options);

        var humedad = new google.visualization.Gauge(document.getElementById('humedad1'));
        humedad.draw(data3, options);

        var humedad2 = new
        google.visualization.Gauge(document.getElementById('humedad2'));
        humedad2.draw(data4, options);


         }
</script>
</head>
<body>
<div
id="holder">
```

```
<div id="footer_terms">
<h1><a href="http://upct.es/">
<strong>UNIVERSIDAD POLITECNICA DE CARTAGENA </strong></a></h1>
</div>

<div id="header">
<a href="index.html">
<img src="images/logo_gif.gif" alt="logo" width="150" height="150"/>
</a> </div>

<div id="shadow">

<ul id="menu">
<li> <a href="index.php">INICIO</a> </li>
<li> | </li>
<li> <a href="nodo.php">Nodo1</a> </li>
<li> | </li>
<li> <a href="nodo2.php">Nodo2</a> </li>
```

```
    <li> | </li>
    <li> <a href="nodo3.php">Nodo3</a> </li>
    <li> | </li>
    <li> <a href="">Memoria</a> </li>
    </ul>
    <div class="clear"></div>
    <div id="edito">
<!-- **********NODO1 ********  -->
    <h2> NODO1</h2>
     <P>
     <h1> <em>Archivos </em>  </h1>
    <a
    href="https://www.dropbox.com/sh/7ttkwjmccsl4t3o/AACbW406ZvwWA16b9fX-Cq87a?dl=0"><img
    src="images/Dropbox.gif"  alt="Dropbox" width="100" height="100" /></a>
        <P>
    </div>
    <div id="toal"> </div>
    <div id="content">
    <h1><em>Actualizado: <?php echo $row[2] ;?></em></h1>
    <div >
        <center><h1> AMBIENTE </h1>
        <div id="chart_div"     ></div>

        <div id="humedad1"     ></div> </center>
    </div>
    <div id="vertical_barr">
        <h1>RADIACION</h1>
        <em>Irradiancia Solar S1087-01</em><strong>
        <center> <a><img
        src="http://uk.farnell.com/productimages/standard/en_GB/42250055.jpg"
        alt="graficas" width="80" height="80" /></a></p></center>
        <?php
            $con=mysqli_connect("localhost","root","raspberry","actuadores");
            if(mysqli_connect_error()){
            echo "No accede a la base :".mysqli_connect_error();}

            $result = mysqli_query($con,"SELECT * FROM Irradiacion  where id='1'
            ");//selecciona tabla y busca fila con id=1
            $row = mysqli_fetch_row($result);

            if ($row[1]>0.2 and $row[1]<0.4 ) {  // para cuando esta cerrada
            $estado1 = "images/visible.gif";}

            elseif($row[1]>0.4 ) {
            $estado1 = "images/inflarrojo.gif";}
            else {
            $estado1 = "images/rayosx.gif";}
        ?>

    <center><IMG SRC="/<?php echo "$estado1"; ?>" WIDTH=140 HEIGHT=140 ALT="family"></center>

        <center> <a href="/nodo1/index.php" class="content_button">  GRAFICAS   NODO1
        </a></center>
    </div>

    <div>
        <center><h1>SUSTRATO</h1>
        <div id="chart_div2"></div>
        <div id="humedad2" ></div></center>

    </div>

    </div>

    <div class="clear"></div>

    </div>

    <div id="footer">
```

```
    <p>contacto  I <a href="pedrete.mayordomo@gmail.com"></a>pedrete.mayordomo@gmail.com </p>
    </div>

    </div>

    </body>
</html>
```

iii. Hrmax

```php
<?php

    $con=mysqli_connect("localhost","root","raspberry","actuadores");
    if(mysqli_connect_error()){
        echo "No accede a la base :".mysqli_connect_error();}
    else{

        $result = mysqli_query($con,"SELECT *   FROM `offset` where id='1'  ");
        $row = mysqli_fetch_row($result);

        $hr = $_POST["valor"];
        if ( $hr>$row[1] &&  $hr<100 && $hr>0){

            $sql = "UPDATE `offset` SET `HRmax`=$hr  WHERE id=1";
            $resultado =$con -> query ($sql) || die("no se puede conectar");

            if($resultado){
                echo " Grabando en BD $valor <p>";
                sleep(2);
                echo "correcto ";
            }
            else{
                 echo " error";}

    }}

    mysqli_close($con);
    header('Location: /index.php');
?>
```

iv. Hrmin

```php
<?php

    $con=mysqli_connect("localhost","root","raspberry","actuadores");
    if(mysqli_connect_error()){
        echo "No accede a la base :".mysqli_connect_error();}
    else{

        $result = mysqli_query($con,"SELECT *   FROM `offset` where id='1'  ");
        $row = mysqli_fetch_row($result);

        $hr = $_POST["valor"];
        if ( $hr<$row[2] &&  $hr<100 && $hr>0){

            $sql = "UPDATE `offset` SET `HRmin`=$hr  WHERE id=1";
            $resultado =$con -> query ($sql) || die("no se puede conectar");

            if($resultado){
                echo " Grabando en BD $valor <p>";
                sleep(2);
                echo "correcto ";
            }
            else{
                 echo " error";}

    }}

    mysqli_close($con);
    header('Location: /index.php');
?>
```

v. GRAFICAS DE NODOS

```
<html>
    <head>

     <center><title>GRAFICA</title> </center>
    <META HTTP-EQUIV="REFRESH" CONTENT="200;URL=/nodo1/index.php">
    <!--<BODY BACKGROUND="fondo.gif" BGPROPERTIES="fixed">-->
    <body style="background:#2ECCFA">

    </head>
    <body>



    <H5><B>
        <!-- pestaña numero 1  -->
        <center>

            <button
            onclick="var ifreim1 = document.getElementById('ifreim1');
            ifreim1.style.display = ifreim1.style.display == 'none' ? 'block'
            : 'none'; "/>  Temperatura ambiente </button>
            <iframe src="Ta.php"
            id="ifreim1" style="display:none" width=100% height=80%
            display=inline></iframe>
        </center>
        <!-- pestaña numero 2  -->
        <center>

            <button
            onclick=" var  ifreim2 = document.getElementById('ifreim2');
            ifreim2.style.display = ifreim2.style.display == 'none' ? 'block'
            : 'none'; "/>humedad ambiente </button>
            <iframe src="ha.php"
            id="ifreim2" style="display:none" width=100% height=80%
            display=none></iframe>
        </center>
        <!-- pestaña numero 3  -->
        <center>

            <button  onclick=" var ifreim3 = document.getElementById('ifreim3');
            ifreim3.style.display = ifreim3.style.display == 'none' ? 'block'
            : 'none'; "/> Temperatura sustrato </button>
            <iframe src="Ts.php"
            id="ifreim3" style="display:none" width=100% height=80%
            display=none></iframe>
        </center>
        <!-- pestaña numero 4  -->
        <center>

            <button
            onclick="var ifreim4 = document.getElementById('ifreim4');
            ifreim4.style.display = ifreim4.style.display == 'none' ? 'block'
            : 'none'; "/>humedad sustrato </button>
            <iframe src="hs.php"
            id="ifreim4" style="display:none" width=100% height=80%
            display=inline></iframe>
        </center>
    </B>
```

```
    <!-- Volver -->
    <center><a href="/index.php" class="content_button">VOLVER</a></center>
    <div id="footer">
    <center><p>contacto  I <a
    href="pedrete.mayordomo@gmail.com"></a>pedrete.mayordomo@gmail.com </p></center>

    <!-- END FOOTER -->
    </div>

    </body>
</html>
```

III. Arduino

i. Maestro

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"
#define CE_PIN   9
#define CSN_PIN 10
#include "Wire.h"
#define DS1307_ADDRESS 0x68
#define S1087 0
float S1087_READ;
float  CON_S1087 ;

RF24 radio(CE_PIN, CSN_PIN); // Pines de conexion de radio
/***************************VAMOS A SIMULARLOS ***********************************/

// ESTRUCTURA PARA NODOS
struct NODOS {
  int cabezera;
  float T_amb;
  float HR_amb;
  float T_sut;
  float HR_sut;
};

NODOS NODO1;  //CREAMOS VARIABLES TIPO NODO
NODOS NODO2;
NODOS NODO3;

// VALORES DE HRsut , para control marcados por   filas -->nodos columanas-->[HRmin,HRmax]
int HR[][2]= { {20 ,90} ,{20 ,90},{20 ,90}};

// ***estado de valvulas ****
    //V[]={ nodo 1,  nodo2,nodo3}
 int V[3]={0,0,0,};

//*  almacenaran el tiempo en el cual fue ala ultima actualizacion del nodo
   //    T[]={ nodo 1,  nodo2,nodo3}
unsigned long T[3]={0,0,0};


int i,nodo,flag,seg ,minutos ,h ,dia_semana ,dia_mes,mes, ano ;
const uint64_t  Emisor[2] ={ 0x88E8F0F0E1LL ,0x88E8F0F0E2LL,0x88E8F0F0E3LL }; //donde escribe
const uint64_t  Receptor[2] ={ 0x38E8F0F0E1LL ,0x38E8F0F0E2LL ,0x38E8F0F0E3LL }; //donde
escuchamos


int pinRead0 ;
float pVolt0 ;



void setup(void)
{
    Serial.begin(9600);
    Wire.begin();
    radio.begin();
    radio.setRetries(15,15);
    radio.openReadingPipe(1,Receptor[0]);
    radio.openReadingPipe(2,Receptor[1]);
    radio.openReadingPipe(3,Receptor[2]);
    radio.startListening();

}


void loop(void){


    control ();
    lectura_S1087 ();
    escuchar_rasp();
    for (i=1;i<4;i++){
```

```
            if ((millis()-T[i-1])>10000){ // 10s esta puesro es decir 10000ms  , cuanto
            menor se se asegura de que la variable este actualizada para el control
                Peticion_datos(i);
                T[i-1]=millis();  //actualizo
            }
    } delay(3000);
}



//*******************************FUNCIONES*******************************
void  Peticion_datos(int i){
     do{
      if ( radio.available() ) {
          leer_n(i);
          Respuesta_n(i);
          comprobacion(i);
        }
     }while(flag==0);
  flag=0;

}


void escuchar_rasp(){

if (Serial.available()){
char char_serial;
 char_serial=Serial.read();

 int b ;
 int mensaje=0;
 if (char_serial== 'N'){

     b=Serial.parseInt();

     HR[b-1][0]=   Serial.parseInt();

     HR[b-1][1]=   Serial.parseInt();

     Peticion_datos((int)b);
     envio(b);
     T[b-1]=millis();
     Serial.flush();//Limpiar buffer

    }
  }//if
}


void   control (){

   int aux=0;
    for (int i=0; i < 3; i++) {
        if (i==0) aux=NODO1.HR_sut;
        if (i==1) aux=NODO2.HR_sut;
        if (i==2) aux=NODO3.HR_sut;

        if (aux< HR[i][0] ) V[i]=1;
        else   V[i]=0;


      }

}



void leer_n(int i){
```

```
  bool done = false;

  int contador=0;
         //do para que entre la primera vez, y pueda comparar
     do{
       switch (i) {
            case 1:

               done=radio.read(&NODO1, sizeof(struct NODOS) );
               nodo=NODO1.cabezera;

            break
            case 2:


               done=radio.read(&NODO2, sizeof(struct NODOS) );
               nodo=NODO2.cabezera;

            break;
            default:


               done=radio.read(&NODO3, sizeof(struct NODOS) );
               nodo=NODO3.cabezera;

            break;

       }

        delay(20);  // Para dar tiempo al emisor
        contador +=1;
        if(contador==5){
        nodo=i;//hacemos que salga del bucle

        }


     }while (!done && nodo!=i);


}


void Respuesta_n(int i){

  radio.stopListening();
  radio.openWritingPipe(Emisor[i-1]);// Dejamos d escuchar para poder hablar

  radio.write( &i, sizeof(int) );
 // Serial.println("Enviando Respuesta");
  radio.startListening();    // Volvemos a la escucha para recibir mas paquetes


}


void comprobacion(int i){
  if(nodo==i) flag=1;
}

void lectura_S1087 (){


 S1087_READ= analogRead(S1087);
 CON_S1087 = (S1087_READ/ 1024) * 5;

 delay(100);
}
void envio(int i){
```

```
  bool done = false;

  int contador=0;
         //do para que entre la primera vez, y pueda comparar
     do{
       switch (i) {
            case 1:

              done=radio.read(&NODO1, sizeof(struct NODOS) );
              nodo=NODO1.cabezera;

           break
           case 2:


              done=radio.read(&NODO2, sizeof(struct NODOS) );
              nodo=NODO2.cabezera;

           break;
           default:


              done=radio.read(&NODO3, sizeof(struct NODOS) );
              nodo=NODO3.cabezera;

           break;

       }

        delay(20);  // Para dar tiempo al emisor
        contador +=1;
        if(contador==5){
        nodo=i;//hacemos que salga del bucle

        }


     }while (!done && nodo!=i);


}


void Respuesta_n(int i){

  radio.stopListening();
  radio.openWritingPipe(Emisor[i-1]);// Dejamos d escuchar para poder hablar

  radio.write( &i, sizeof(int) );
 // Serial.println("Enviando Respuesta");
  radio.startListening();    // Volvemos a la escucha para recibir mas paquetes


}


void comprobacion(int i){
  if(nodo==i) flag=1;
}

void lectura_S1087 (){


 S1087_READ= analogRead(S1087);
 CON_S1087 = (S1087_READ/ 1024) * 5;

 delay(100);
}
void envio(int i){
```

```
switch (i) {
   case 1:


        Serial.print("N");
        Serial.print(":");
        Actualizar_hora();
        Serial.print(";");//para separar bloque hora en python
        Serial.print(NODO1.cabezera);
        Serial.print(":");
        Serial.print(NODO1.T_amb);
        Serial.print(":");
        Serial.print(NODO1.HR_amb);
        Serial.print(":");
        Serial.print(NODO1.T_sut);
        Serial.print(":");
        Serial.print(NODO1.HR_sut);
        Serial.print(";");
        Serial.print(V[0]);
        Serial.print(":");
        Serial.print(CON_S1087);
        Serial.println(":");

     break;

    case 2:

        Serial.print("N");
        Serial.print(":");
        Actualizar_hora();
        Serial.print(";");
        Serial.print(NODO2.cabezera);
        Serial.print(":");
        Serial.print(NODO2.T_amb);
        Serial.print(":");
        Serial.print(NODO2.HR_amb);
        Serial.print(":");
        Serial.print(NODO2.T_sut);
        Serial.print(":");
        Serial.print(NODO2.HR_sut);
        Serial.print(";");
        Serial.print(V[1]);
        Serial.print(":");
        Serial.print(CON_S1087);
        Serial.println(":");

     break;
     default:

        Serial.print("N");
        Serial.print(":");
        Actualizar_hora();
        Serial.print(";");
        Serial.print(NODO3.cabezera);
        Serial.print(":");
        Serial.print(NODO3.T_amb);
        Serial.print(":");
        Serial.print(NODO3.HR_amb);
        Serial.print(":");
        Serial.print(NODO3.T_sut);
        Serial.print(":");
        Serial.print(NODO3.HR_sut);
        Serial.print(";");
        Serial.print(V[2]);
        Serial.print(":");
        Serial.print(CON_S1087);
        Serial.println(":");

     break;
}
```

```
}


byte bcdToDec(byte val)  {
// Convert binary coded decimal to normal decimal numbers
  return ( (val/16*10) + (val%16) );
}

void Actualizar_hora(){

  //--proceso de lectura--
  Wire.beginTransmission(DS1307_ADDRESS);//abre I2C modo escritura

  byte zero = 0x00;
  Wire.write(zero);
  Wire.endTransmission();

  Wire.requestFrom(DS1307_ADDRESS, 7);//abre I2C modo lectura

  int seg = bcdToDec(Wire.read());
  int minutos = bcdToDec(Wire.read());
  int h = bcdToDec(Wire.read() & 0b111111); //24 hour time
  int dia_semana = bcdToDec(Wire.read()); //0-6 -> domingo  a  sabado
  int dia_mes = bcdToDec(Wire.read());
  int mes= bcdToDec(Wire.read());
  int ano = bcdToDec(Wire.read());

  //print   3:1:11:23:59:59
        Serial.print(dia_mes);
        Serial.print(":");
        Serial.print(mes);
        Serial.print(":");
        Serial.print(ano);
        Serial.print(":");
        Serial.print(h);
        Serial.print(":");
        Serial.print(minutos);
        Serial.print(":");
        Serial.print(seg );

}
```

ii. Nodos

```
#include <SPI.h>
#include "nRF24L01.h"
#include "RF24.h"
#include "DHT.h"
#include <SHT1x.h>
#define DHTPIN 2        //PIN DE DATOS SENSOR
#define DHTTYPE DHT22  // MODELO A UTILIZAR
#define CE_PIN   9
#define CSN_PIN 10
#define dataPin  7
#define clockPin 6
#define NODO   2
SHT1x sht1x(dataPin, clockPin);

struct NODOS {
  int cabezera=2;
  float T_amb ;
  float HR_amb;
  float T_sut;
  float HR_sut;
}NODO2;

RF24 radio(CE_PIN, CSN_PIN); // Create a Radio
DHT dht(DHTPIN, DHTTYPE);
const uint64_t  Emisor[3] ={ 0x38E8F0F0E1LL ,0x38E8F0F0E2LL,0x38E8F0F0E3LL}; //donde
escuchamos
const uint64_t Receptor[3] ={ 0x88E8F0F0E1LL ,0x88E8F0F0E2LL,0x88E8F0F0E3LL }; //donde
escribe

void setup(void)
{
    Serial.begin(9600);
    radio.begin();
    delay(5000);
    radio.setRetries(15,15);    // Maximos reintentos
    radio.openWritingPipe(Emisor[NODO-1]);
    radio.openReadingPipe(NODO,Receptor[NODO-1]);
}

void loop(void)
{
  leer_datos();
  delay(3000);
  Envio_datos(2);
  Comprobacion();


}
void leer_datos(void){
  NODO2.HR_amb= dht.readHumidity();
  NODO2.T_amb= dht.readTemperature();
  if (isnan(NODO2.HR_amb) || isnan(NODO2.T_amb)) {
      Serial.println("FALLO DE LECTURA");
      leer_datos();
      }
  NODO2.T_sut = sht1x.readTemperatureC();
  NODO2.HR_sut = sht1x.readHumidity();




 }
  // *****ENVIO MANUAL **********PARA PRUEBAS


 void leer_datos2(void){
    if (Serial.available() > 0){
```

```
    NODO2.HR_sut = Serial.parseInt();
    NODO2.T_amb = Serial.parseInt();
    NODO2.T_sut = Serial.parseInt();
        Serial.print(NODO2.cabezera);
        Serial.print(":");
        Serial.print(NODO2.T_amb);
        Serial.print(":");
        Serial.print(NODO2.HR_amb);
        Serial.print(":");
        Serial.print(NODO2.T_sut);
        Serial.print(":");
        Serial.println(NODO2.HR_sut);
      }
 }


void Envio_datos(int i){
  radio.stopListening();     // Paramos la escucha,para poder hablar

  Serial.print("Enviando...  ");
   bool ok=false;
  while(ok==false){
  ok=radio.write( &NODO2, sizeof( struct NODOS) );}
        Serial.print(NODO2.cabezera);
        Serial.print(":");
        Serial.print(NODO2.T_amb);
        Serial.print(":");
        Serial.print(NODO2.HR_amb);
        Serial.print(":");
        Serial.print(NODO2.T_sut);
        Serial.print(":");
        Serial.println(NODO2.HR_sut);


  if (ok)
      Serial.println("ok...");
  else
      Serial.println("failed");

}


void Comprobacion(void){
    radio.startListening();   //Volvemos a la escucha
    //Funcion millis() te recoge el valor en milisegundos desde que comenzo el programa
    unsigned long tiempo = millis();
    bool timeout = false;
    // 'radio.available'= TRUE si  nodo maestro comunica en nuestro canal
    // estar esperando mientra !"no" disponible y !"no" tiempo

    while ( ! radio.available() && ! timeout ) { // Esperasmos repsuesta hasta 200ms
        if (millis() - tiempo > 300 )
            timeout = true;
    }//while


    if ( timeout )
        Serial.println("Failed, response timed out");
    else
        { // Leemos el mensaje recibido
        int respuesta;
        radio.read( &respuesta, sizeof(int) );

        Serial.print(" enviado corectamente Respuesta = ");
        Serial.println(respuesta);
    }
    delay(600);
```

ANEJO III

BACKUP Y RESTAURACIÓN BASE DE DATOS

Con el fin de obtener un respaldo de seguridad de la base de datos creada tras el trabajo de insertar las tablas con el formato correspondiente para cada variable, es necesario el backup de las bases de datos, en el caso de fallo de servidor o restaurar el proyecto en otra base de datos reduces el tiempo de trabajo.

Para ello desde “PHPMyAdmin” el gestor de base de datos, permite exportar la base de datos de esta manera se puede guardar un backup de la base de datos con todas sus tablas creadas además de los datos guardados en ese instante de tiempo.

Seleccionando la base de datos que se desea exportar se puede ver en la parte superior sobre las opciones, el botón de “exportar”:

Ilustración 71:Exportación base de datos

El formato guardado de la exportación de la base de datos es ".sql"

Para restaurar las tablas recuperadas se debe de tener creada la base de datos donde se desean importar las tablas guardadas anteriormente.

Se muestra de ejemplo la restauración de nodo3:

1) Primero crear base de datos, donde se desean recuperar las tablas

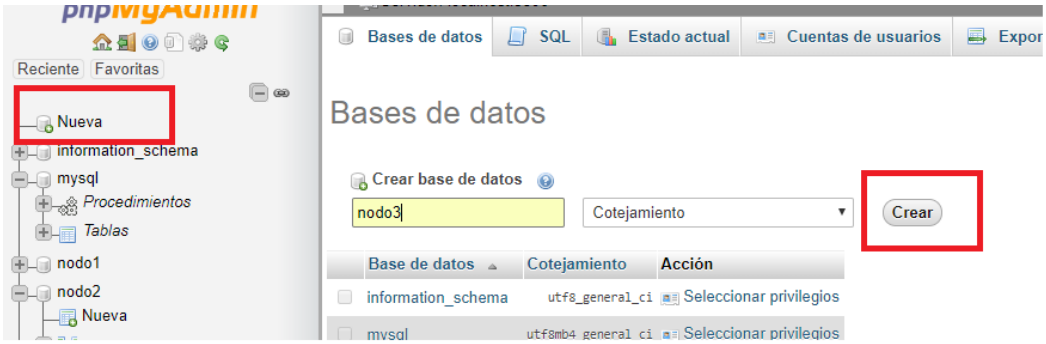

Ilustración 72:Creación de Base de Datos

2) Se selecciona base de datos y se pincha en "importar ", el archivo a seleccionar debe de ser del formato ".sql", archivo exportado de base de datos completa.

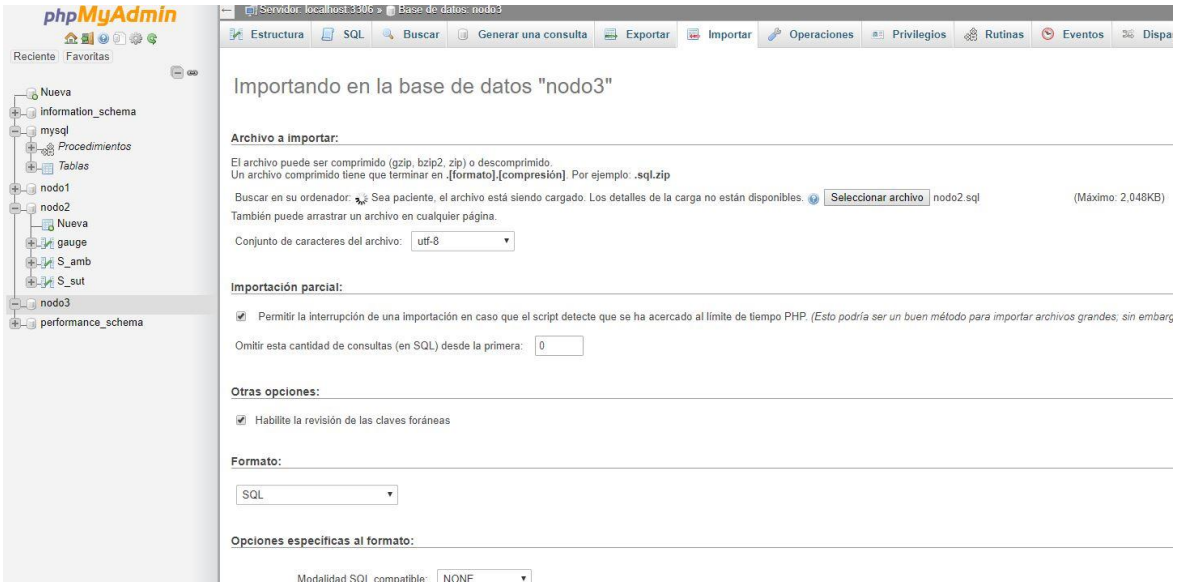

Ilustración 73:Importar Base de Datos

3) Si todo ha sido correcto, las tablas se deben de añadir a la base de datos creada:

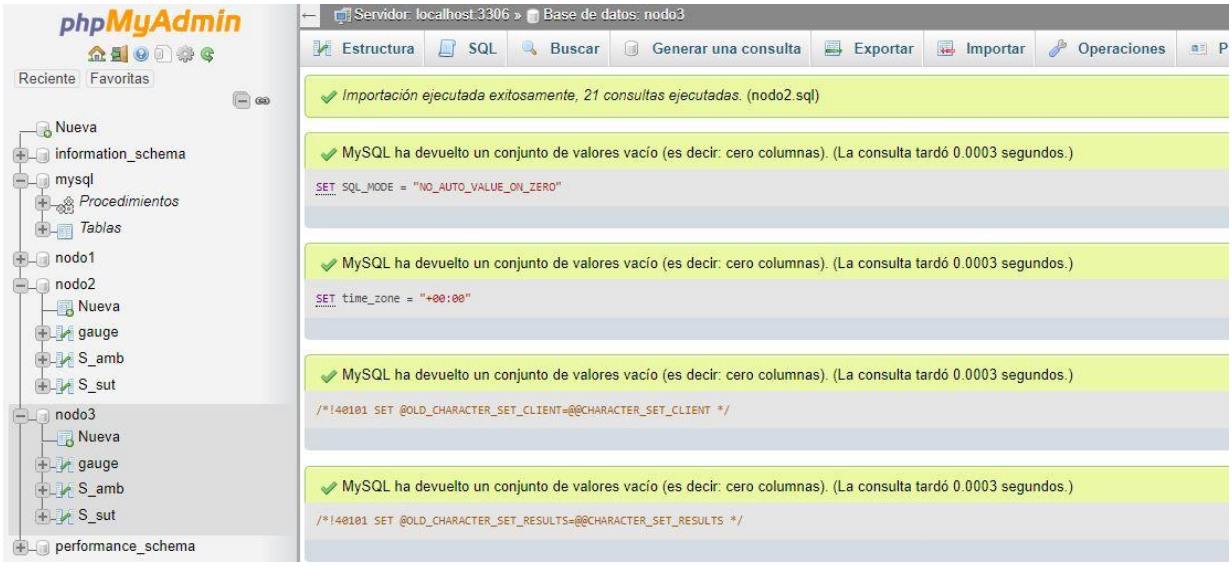

Ilustración 74:Importar Base de Datos finalizado

BIBLIOGRAFÍA

[1] Luis Pérez Bermejo. (2008, marzo). Radiocomunicaciones: el estado del arte. Recuperado de http://www.coitt.es/res/revistas/05b_Radiotrans.pdf

[2] Ana Belén Corral Ignoto. (2005, Julio). Diseño e implementación de un entorno de simulación para redes de sensores inalámbricos (Capitulo 2, página 3-23). Recuperado de http://repositorio.upct.es

[3] CSIRT-CV Centro de Seguridad TIC de la Comunitat Valenciana. Seguridad en Internet de las Cosas – CSIRT-CV. Recuperado de http://www.csirtcv.gva.es/

[4] Dave Evans. (2011, abril). Internet de las cosas. Como la próxima evolución de Internet lo cambia todo. Recuperado de http://www.cisco.com

[5] Cooking-hacks. Open Garden Shield . Recuperado de https://www.cooking-hacks.com/blog/category/waspmote/

[6] Parrot. Parrot flower power. Recuperado de http://global.parrot.com/

[7] Waspmote. Libelium. Recuperado de http://www.libelium.com

[8] Arduino . Recuperado de https://www.Arduino .cc/

[9] Raspberry Pi. Recuperado de https://www.raspberrypi.org/

[10] Maniacbug. (2013, Octubre). Librería RF24. Recuperado dE https://github.com/maniacbug/RF24

[11] *Simon Tatham, Owen Dunn, Ben Harris, Jacob Nevins.Putty.* Recuperado de http://www.putty.org/

[12] FileZilla. FileZilla Client.
Recuperado de https://filezilla-project.org/

[13] No-Ip. Servicio de DNS. Recuperado de https://my.noip.com/
[14] Andrea Fabrizi. Dropbox Uploader. Recuperado de
https://github.com/andreafabrizi/Dropbox-Uploader/