UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DISEÑO, IMPLEMENTACIÓN E INTEGRACIÓN DE UN SISTEMA DE MEDICIÓN DE VARIABLES DE ENTORNO EN UN SISTEMA IOT CON SOFTWARE Y HARDWARE LIBRE

AUTOR: JOSÉ DANIEL SERRANO ANDRÉS

TUTORA: YIYAO YE LIN

COTUTOR: FRANCISCO JAVIER GARCÍA CANSADO

Curso Académico: 2016-17

AGRADECIMIENTOS

A mi familia.

A mis tutores tanto en la universidad como en S2 Grupo.

A mis compañeros y amigos.

RESUMEN

El presente proyecto de ingeniería nace impulsado por la necesidad de crear un entorno de prueba y demostraciones de la empresa S2 Grupo, que es una empresa puntera en el ámbito de la ciberseguridad. El proyecto consiste en el diseño y desarrollo de un sistema de monitorización de variables ambientales basado en dispositivos *IoT* en plataformas *open source.* Específicamente se pretende monitorizar las variables de entorno como la contaminación atmosférica y acústica, iluminación, temperatura y humedad. Para ello, primero se ha seleccionado el conjunto de sensores, microcontrolador y servidor que conforman el sistema para la medida de las variables de entorno de interés. Asimismo, se ha desarrollado un programa para el microcontrolador (en el entorno *Arduino IDE*) que es el responsable de leer los datos provenientes de cada uno de los sensores y los envía al servidor vía WiFi, otro programa en servidor (en Python) que recibe los datos enviados por el microcontrolador y los almacenan en una base de datos local (en MongoDB) y una página web en HTML y JavaScript para que el usuario pueda consultar la información en tiempo real desde cualquier terminal de intranet. El sistema desarrollado se encuentra instalado y funcionando actualmente en la empresa.

Palabras Clave: Sistema de instrumentación, dispositivos IoT, programación microcontrolador.

RESUM

El present projecte d'enginyeria naix impulsat per la necessitat de crear un entorn de prova i demostracions de l'empresa S2 Grupo, que és una empresa puntera en l'àmbit de la ciberseguritat. El projecte consisteix en el disseny i desenvolupament d'un sistema de monitorització de variables ambientals basat en dispositius IoT en plataformes "Open Source". Específicament es pretén monitoritzar les variables d'entorn com la contaminació atmosfèrica i acústica, il·luminació, temperatura i humitat. Per a això, primer s'ha seleccionat el conjunt de sensors, microcontrolador i servidor que conformen el sistema per a la mesura de les variables d'entorn d'interés. Així mateix, s'ha desenvolupat un programa en microcontrolador (en l'entorn Arduino IDE) que és el responsable de llegir les dades que provenen de cada un dels sensors i els envia al servidor via WIFI, un altre programa en servidor (en Python) que rep les dades enviades per el microcontrolador i els emmagatzemen en una base de dades locals (en MongoDB) i una pàgina web en HTML i Javascript per a que l'usuari puga consultar la informació en temps real des de qualsevol terminal d'intranet. El sistema desenvolupat es troba instal·lat i funcionant actualment en l'empresa.

Paraules clau: Sistema d'instrumentació, dispositius IoT, programació microcontrolador.

ABSTRACT

The present engineering project is driven by the need to create a test and exposure environment of the company S2 Grupo, that is a leading company in the field of cybersecurity. The project consists of the design and development of a monitoring system for environmental variables based on IoT devices and "Open Source" platforms. Specifically intended to monitor environmental variables such as atmospheric and acoustic pollution, lighting, temperature and humidity. To do this, in first place has been choosen sensors, microcontroller and server that compose the system for the measurement of the environment variables. Also, a microcontroller program (in the Arduino IDE environment) has been developed that is responsible for reading the data from each of the sensors and sends them to the server over WIFI, another server program (in Python) that receives the data sent by the microcontroller and stored in a local database (in MongoDB), and a web page in HTML and Javascript so that the user can query the information in real time from any intranet terminal. The developed system is currently installed and running in the company.

Keywords: Instrumentation system, IoT devices, microcontroller programming

ÍNDICE

DOCUMENTO I

MEMORIA

CAPÍTULO 1: INTRODUCCIÓN Y ANTECEDENTES

1.1. INTRODUCCIÓN.

El proyecto descrito se va a realizar en S2 Grupo, empresa puntera con sede en la Comunidad Valenciana cuyo desarrollo empresarial se despliega en el ámbito de la ciberseguridad. Este TFG forma parte de un proyecto de mayor envergadura del área de seguridad industrial de la empresa.

El proyecto "padre" del que deriva este TFG consiste en la modelización de una casa conectada. Según la empresa Gartner, se entiende como *connected home* aquella que está comunicada con una red para permitir la conexión e interoperabilidad de múltiples dispositivos, servicios y aplicaciones [1]. Dado que este es un concepto muy amplio, las tecnologías empleadas detrás del término pueden agruparse en las siguientes categorías:

- **Redes:** Esta categoría engloba todas las redes y protocolos domésticos que sirven para la comunicación entre dispositivos y servicios. (Wifi, Bluetooth, 3G, LTE, 6LoWPAN, RF, etc).
- **Comunicación y entretenimiento:** Este grupo abarca la mayoría de los sistemas de entretenimiento integrados de la casa. También incluye el acceso e intercambio de contenido digital. Dispositivos como proyectores, Smart TV, sistemas de sonido, consolas de videojuegos, y servicios de steaming y video/música bajo demanda estarían incluidos en este grupo.
- **Seguridad, monitorización y domótica:** Este apartado agrupa las soluciones que se centran en la seguridad y la protección de la casa, la monitorización de la misma, así como el control remoto y automatizado de puertas, ventanas, persianas, cerraduras, calefacción, aire acondicionado, iluminación, electrodomésticos, etc.
- **Gestión de la Energía:** Esta clase está relacionada con las ciudades inteligentes y las iniciativas gubernamentales, pero los servicios y dispositivos como contadores inteligentes se están introduciendo en el mercado de forma masiva permitiendo a los clientes conocer y controlar su consumo de electricidad, gas, agua, etc.
- **Salud, fitness y bienestar:** Este conjunto engloba dispositivos y servicios relacionados con la rápida asistencia sanitaria, no obstante, es un campo aún en desarrollo, y su despliegue está siendo lento, pues engloba a gobiernos, hospitales y planes de salud. Por otra parte, el segmento de fitness y bienestar tiene ecosistemas fuertes, que van desde dispositivos hasta artículos deportivos y aplicaciones que conectadas entre sí mejoran la experiencia proporcionada por ellos individualmente.

El modelo de casa conectada va a servir de entorno de pruebas y demostraciones, donde la empresa S2 grupo podrá desplegar su actividad comercial. Sobre el modelo se estudiarán las

posibles vulnerabilidades y vectores de ataque que presenta la introducción de estos dispositivos en nuestros hogares. Posteriormente se desarrollarán productos comerciales que sirvan para proteger a los hogares de los clientes de estas amenazas. Una vez finalizada la fase de estudio y desarrollo del producto de seguridad, el entorno servirá de demostración a los clientes.

El IoT[1] es un paradigma tecnológico que consiste dotar de conexión a internet a dispositivos que tradicionalmente no disponían de esta capacidad. El riesgo del IoT radica en que los dispositivos, en su mayor parte, no han sido diseñados para ser seguros, si no que han sido diseñados para funcionar. Por ello, un estudio realizado por Hp en 2015 desveló que cerca del 100 por cien de los dispositivos conectados a internet eran vulnerables a ciberataques, y poseían graves vulnerabilidades [2]. Entre estas vulnerabilidades, las más comunes eran:

- **Autorización insuficiente**: Todas las interfaces web basadas en la nube e interfaces móviles no requerían contraseñas de suficiente complejidad y longitud. Además, no incorporaban un sistema de bloqueo de cuentas después de un determinado número de intentos fallidos.
- **Interfaces inseguras**: Debido a la vulnerabilidad anterior, era posible acceder a los datos que almacenaban estas cuentas.
- **Privacidad**: Todos los sistemas recolectaban de alguna forma información personal como nombre, dirección, número de teléfono y hasta tarjetas de crédito. La exposición de toda esta información es preocupante debido a los problemas de recolección de cuentas en todos los sistemas. También muchos de los sistemas de seguridad IoT que hacen uso de la grabación de video para la monitorización de la casa pueden estar exponiendo lo que sucede dentro de ella.
- **Falta de cifrado**: No todos los sistemas incluían sistemas de cifrado para la comunicación con los servidores, o si los incluían no estaban configurados correctamente, permitiendo la interceptación de la información transmitida.

Como la empresa desea desarrollar soluciones lo más amplias y flexibles posibles, dicho modelo deberá contar de una gran diversidad de dispositivos y tecnologías, incluyendo soluciones de diferentes fabricantes catalogadas como propietarias, así como soluciones del ámbito del *open source* y *DIY*[2].

El *open source* o también denominados *hardware* y *software* libre son proyectos en los que su código fuente, planos, construcción, controladores, está público y accesible a cualquier persona. Además, estos proyectos suelen estar bajo una licencia de código abierto que permite que el código fuente, binarios, planos, materiales, etc, sean modificados y redistribuidos libremente [3]. Este tipo de proyectos encaja perfectamente con la filosofía *DIY* que consiste en la fabricación de cosas por uno mismo, de forma que se ahorra dinero y se adquieren conocimientos relacionados con el área de aplicación. La mayoría de estos proyectos están basados en hardware y software libre por sus características de replicación y distribución. Un ejemplo de este tipo de proyectos son las impresoras 3D y su wiki *www.reprap.org.*

[1] IoT: *Internet of Things* o Internet de las Cosas.
[2] DIY: *Do It Yourself o Hágalo usted mismo.*

La mayor parte de dispositivos de una casa conectada también entran dentro de la definición de IoT la cual se profundiza en el siguiente apartado.

1.2. DISPOSITIVOS IOT.

El término Internet de las Cosas fue definido por Kevin Ashton, un precursor en tecnología británico, en referencia a la interconexión digital de objetos cotidianos con internet. Hoy en día el termino IoT expresa una conexión entre dispositivos, sistemas, servicios y personas permitiendo una interacción entre el mundo físico y digital sin precedentes [4]. Las posibilidades que brinda este tipo de tecnología son prácticamente ilimitadas e implantables en cualquier campo. Ejemplos de estas nuevas posibilidades en campos como los descritos a continuación serían:

- **Envases de los productos**: Identificando y conectando cada producto individual, se puede llevar un control de que productos están próximos a caducar y es aplicable tanto a grandes superficies como en los hogares de los clientes. Con esta tecnología también se podría detectar que algún producto se ha acabado, y gestionar su compra de forma automática.
- **Servicios de paquetería**: para reducir al mínimo el extravío de las entregas, además de proporcionar al receptor información sobre la localización en tiempo real del paquete. Mejorando de esta forma la experiencia proporcionada por el servicio.
- **Termostatos, refrigeradores, calderas**: Posibilita controlar y supervisar en todo momento su funcionamiento desde nuevas plataformas conectadas a internet, como podrían ser *smartphones*[3], *smartwatchs*[4] o *wearables*[5]. La inclusión de la tecnología IoT simplifica y hace más cómodo el uso de estos dispositivos.
- **Coches, maquinaria**: Añade la posibilidad de comunicar averías o un mal funcionamiento. También información relevante acerca de los datos recogidos por los sensores del coche o máquina, como puede ser desgaste de ruedas, frenos, necesidad de cambio de aceite. De esta forma, el usuario puede prevenir de una forma más activa el deterioro del coche/maquinaria. Desde el punto de vista de los fabricantes, permite realizar una llamada a revisión de una forma mucho más ágil y efectiva, y proveer de nuevas características o funcionalidades vía actualización de software.

Además, se podría añadir la posibilidad de interactuar con los mismos de forma remota, o añadir capas de inteligencia para que actúen de forma autónoma en base a diversos parámetros. La aplicación de esta inteligencia y control remoto proporciona una gran mejora respecto a los dispositivos tradicionales, facilitando el uso y la vida de sus usuarios.

Por otro lado, aunque no menos interesante, este tipo de tecnología permite recoger una amplia cantidad de datos a todos los niveles. Estos datos pueden estar relacionados con los productos, con los clientes (hábitos, preferencias, etc), con la forma en la que son usados los dispositivos, con los procesos productivos, etc. Los datos pueden servir para detectar los cuellos de botella, mejorar y optimizar los procesos productivos, ahorrar energía, fabricar anticipándose

[3] Smartphone. Teléfono inteligente.
[4] Smartwatch. Reloj inteligente.
[5] Weareable. Dispositivo inteligente que se puede llevar puesto. Por ejemplo, un collar o pulsera.

a la demanda, evitar el desperdicio de productos (perecederos) y materiales, prever averías de maquinaria o transportes, etc.

La automatización y recolección de datos basada en sistemas electrónicos existe desde los años 60, cuando se fabricaron los primeros PLC[6] [5]. Los PLC son sistemas industriales fiables y robustos (Figura 1), y su precio está acorde a sus prestaciones, por tanto, su uso queda limitado al control de procesos productivos, con requerimientos propios del entorno industrial (Velocidad, tiempo real, resistencia física a temperatura y ondas electromagnéticas, tasa mínima de fallos, etc), como cadenas de montaje, fabricación en serie, depuradoras, centrales y subestaciones eléctricas, etc.

Figura 1 PLC Usados En La Automatización De Procesos

La llegada del IoT ha democratizado la creación de sistemas automatizados, su telecontrol y la recolección de datos. La casa conectada nunca hubiese sido económicamente viable si se hubiesen tenido que realizar con sistemas de control industriales.

Pero el ámbito doméstico no es el único lugar donde se está aplicando esta tecnología, en las empresas ha comenzado la transición a lo que se denomina la *Industria 4.0,* que también ha sido denominada como cuarta revolución industrial, industria inteligente o Ciberindustria del futuro. No obstante, aún no es una realidad consolidada, si no una tendencia de futuro que está sentando sus cimientos (véase Figura 2). Estos cimientos se basan en la puesta en marcha de fábricas inteligentes, que dispongan de una alta adaptabilidad a las necesidades y procesos de producción, una mejor y más eficiente gestión de recursos, y una digitalización y telecontrol de todos los procesos (desde la producción hasta la distribución) [6]. Para conseguir estos cimientos se basan en la utilización de las siguientes tecnologías [7]:

- Sistemas ciber-físicos.
- Industria y productos inteligentes.
- Internet of Things and Servicies.
- Hiperconectividad.

[6] PLC: *Programmable Logic Controller.* Sistema informático de control industrial.

- *BigData* and *analitycs*.

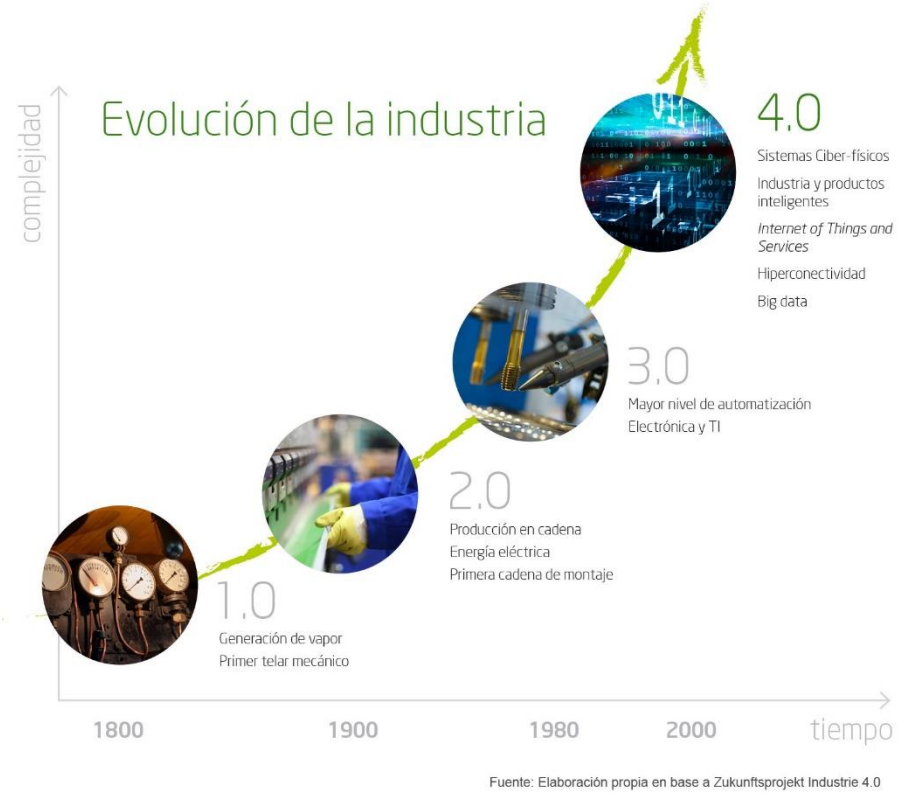


Figura 2 Evolución de la Industria

El concepto IoT empezó a cobrar especial relevancia en los últimos 10 años cuando el número de dispositivos conectados a internet superó la población humana. Analizando la tendencia actual (Tabla 1), se observa un crecimiento exponencial en el número de estos dispositivos, estimándose que para 2020 la cantidad de dispositivos online supere los 20.000 millones [8].

Categoría	**2016**	**2017**	**2018**	**2020**
Domésticos	3.963,0	5.244,3	5.244,3	5.244,3
Empresariales	2.428,7	3.136,4	4.160,3	7.552,4
Total	**6.381,8**	**8.380,6**	**11.196,6**	**20.415,4**

Tabla 1 ¨Dispositivos IoT conectados actualmente y su previsión. Unidades en millones". "IoT Units Installed Base by Category (Millions of Units)¨, [en línea] Disponible en: https://www.gartner.com/newsroom/id/3598917 [Consulta: 6 de agosto de 2017].

Debido al potencial y la rápida implementación de la tecnología se han creado gran cantidad de protocolos y productos en este campo. Grandes multinacionales han invertido en desarrollar los mismos, proporcionando a los consumidores productos completos como pueden

ser los termostatos Nest[7], de Google; o las bombillas Philips Hue[8] (Figura 3). Por otro lado, también existen soluciones de hardware y software libre que están adquiriendo especial importancia debido a que ofrece una gran flexibilidad a un precio realmente competitivo. Existe una gran comunidad en internet detrás de esta plataforma libre, siendo las alternativas predominantes las soluciones basadas en Arduino y Raspberry Pi. No obstante, grandes empresas han visto su potencial, incluyendo en su oferta productos basados en hardware libre como son Siemens con Simatic IoT 2020 o Beaglebone.

Figura 3 Izq: Termostato inteligente Nest. Drch: Bombillas inteligentes Philips Hue

Sin embargo, no todas las soluciones y productos tienen el mismo grado de madurez y aceptación por parte de los consumidores, una buena forma de observar estas tendencias es mediante el ciclo de sobreespectación que Gartner publica en sus informes. En estos ciclos se analiza el estado actual de las diversas tecnologías emergentes y se engloban en una de las siguientes 5 fases.

- **Lanzamiento:** Es el comienzo de una tecnología, cuando es presentada oficialmente al mercado. El grado de madurez es bajo, y aún no ha suscitado interés por ella, aunque con el paso del tiempo y su desarrollo lo irá generando.
- **Pico de expectativas sobredimensionadas:** Los medios de comunicación empiezan a generar noticias sobre la tecnología generando entusiasmo y expectativas poco realistas. También se realizan las primeras implementaciones, pero obteniendo más fracasaos que éxitos.
- **Abismo de desilusión:** Las tecnologías no cumplen con las expectativas creadas. Dejan de estar de moda, y la prensa deja de publicar artículos.
- **Pendiente de consolidación:** A pesar de la falta de cumplimiento de expectativas, y la perdida de la popularidad, algunas empresas continúan desarrollando dichas tecnologías con el objetivo de alcanzar las expectativas creadas.

[7] Nest: Termostato inteligente que permite su control desde plataformas móviles y aprende de los hábitos de los usuarios.

[8] Philips Hue: Bombillas multicolores que permiten control remoto e interconexión con diferentes servicios

- **Meseta de productividad:** Finalmente los beneficios aportados por la tecnología son demostrados y aceptados. Esta se vuelve estable y aparecen segundas y terceras generaciones.

En la Figura 4 se asocian eventos característicos de cada una de las fases del ciclo de sobreexpectación.

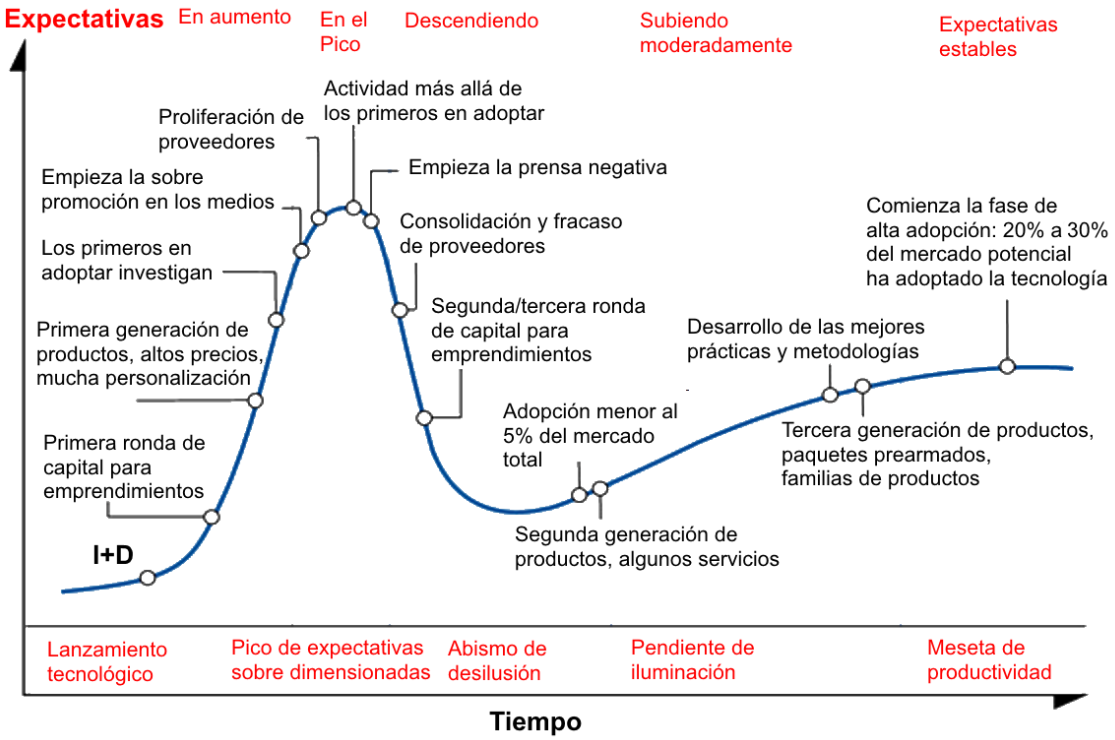


Figura 4 Fases del ciclo de sobreexpectación

Como se ha mencionado anteriormente, el grafico de sobreexpectación analiza el desarrollo de las tecnologías emergentes. Es por ello que en el año 2016 Gartner elaboró un estudio sobre el IoT obteniendo la Figura 5.

En cuanto a la tecnología IoT en el año 2016 Gather realizo este gráfico (Figura 5) incluyendo las tecnologías presentadas hasta ese momento en el mercado, indicando su estado en el ciclo y su previsión acerca del tiempo que tardarían en consolidarse.

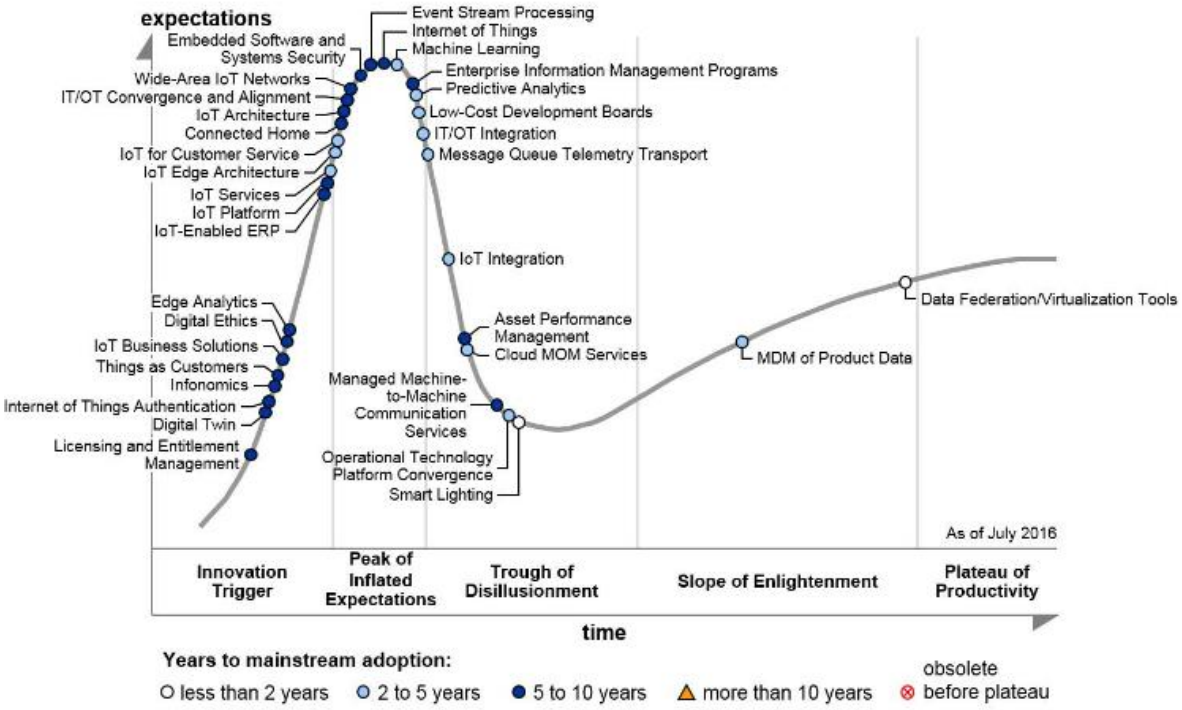


Figura 5 Gráfico de sobreexpectación tecnología IoT

En ella se observa el estado de muchas de las tecnologías usadas en este proyecto que se sitúan en el pico de expectativas sobredimensionadas. Así pues, *Low-Cost Development Boards*[9]*, Internet of Things y Connected Home* son conceptos tecnológicos, en los que muchas empresas están investigando y desarrollando soluciones tratando de alcanzar el éxito, pero que aún se encuentran en una fase temprana de su ciclo vital.

1.3. SENSORES.

Un sensor es un dispositivo capaz de detectar y cuantificar variaciones en una magnitud física. La salida más extendida de los sensores es la eléctrica pues permiten la utilización de gran cantidad de componentes electrónicos Estos componentes se usan para tratar y transmitir la señal proporcionada por los sensores.

En la mayoría de los casos la amplitud de la medida que proporciona un sensor es muy pequeña, para poder trabajar con la señal es necesario el uso de amplificadores. Por otro lado, también existe ruido en la señal de salida de los sensores causado por diversos motivos, como la frecuencia de la red eléctrico. Para evitar, estas señales indeseadas se hacen uso de los filtros. Dependiendo la señal que se quiera eliminar se puede confeccionar de paso alto y paso bajo. Por último, puede ser necesario convertir la señal analógica captada por el sensor en digital. Hoy en día la mayoría de sistemas de control son digitales, para ello se hace uso de un CAD[10]. Una vez digitalizados, el ordenador puede trabajar con esos datos representando la información,

[9] *Low-Cost Development Boards.* Placas de prototipado y desarrollo de bajo coste.
[10] Convertidor analógico-digital.

almacenadola, o ejecutando algoritmos de control automático. En la Figura 6 se puede observar cada una de las fases. En este caso el sensor tendría ya implementado el amplificador

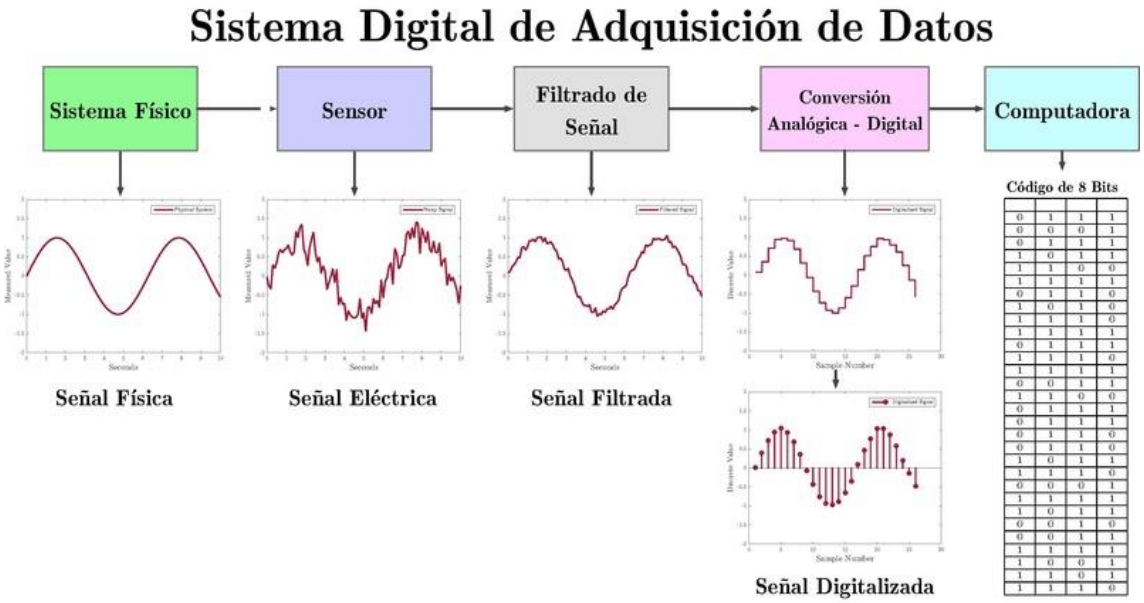


Figura 6 Sistema Digital de Adquisición de Datos

Hay gran variedad de tipos de sensores, que podemos clasificar a grandes rasgos en grupos dependiendo de [9]:

- **Aporte de energía**: Pasivos o activos dependiendo de si tienen o no alimentación externa.
- **Modo de funcionamiento**: Deflexión o comparación. Depende de si el sensor produce alguna deformación o almacenamiento de energía en el sensor (dinamómetro) o si se intenta evitar esta deflexión mediante contrarrestando el efecto de la magnitud (báscula)
- **Relación de entrada salida**: Dependiendo del número de almacenadores de energía independiente que cuente el sensor pueden ser de orden cero, de primer orden, de segundo orden, u orden superior.
- **Señal de salida**: Analógicos y digitales.
- **Magnitud medida**: Atendiendo a la magnitud física a la que sean sensibles.
- **Parámetro variable**: Pueden ser:
    - Sensores resistivos
    - Sensores capacitivos
    - Sensores inductivos y electromagnéticos
    - Sensores generadores de tensión y corriente
    - Sensores digitales

Las innovaciones tecnológicas han traído una nueva clase en la clasificación llamada sensor inteligente, que se define como aquél que combina funciones de medición con funciones de amplificación, filtrado, linealización, compensación de temperatura, digitalización, identificación, auto-calibración, auto-diagnóstico, etc . Estas funciones suelen ser propias de un

microprocesador, por lo que también se denomina sensor inteligente al conjunto de ambos (véase Figura 7). Debido al nivel de integración de los circuitos de silicio, disponer de todo en un mismo sensor encapsulado supone claras ventajas en términos potencia consumida, fiabilidad, facilidad de uso y precio. Estos sensores han sido clave en el desarrollo de la tecnología IoT.

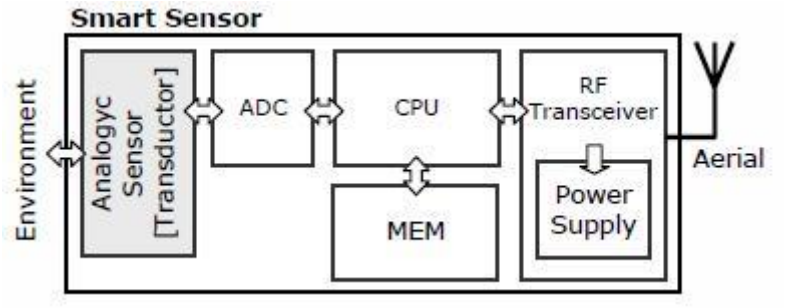


Figura 7 Esquema sensor Inteligente

1.4. DESCRIPCIÓN DEL ENTORNO

El entorno de pruebas y demostraciones va a representar una casa real con sus diferentes estancias. En lo concerniente a este TFG, el dispositivo IoT está situado en la representación del garaje de la casa. Por lo que se han definido las variables a monitorizar en función de esta ubicación.

- Fugas: En un garaje suelen coexistir diferentes instalaciones en que podrían producirse fugas, como son las acometidas de agua, calderas, depósitos, etc.
- Calidad del aire: Un garaje puede tener multitud de usos (almacenamiento, taller, etc) aunque su principal función es aparcar los vehículos. Controlar la calidad del aire puede ser importante para detectar concentraciones toxicas de contaminantes producidas por los vehículos a motor, o derrames de productos como disolventes.
- Sonido y Luz: Estas variables son útiles a hora de detectar actividad dentro del garaje. Encendido del motor del coche, apertura/apertura cierre del portón, encendido de luces, etc.
- Temperatura y humedad: Variables útiles para determinar el estado del ambiente del garaje.

CAPITULO 2: JUSTIFICACIÓN Y OBJETIVOS DEL TRABAJO

2.1. JUSTIFICACIÓN

Para una empresa como S2 Grupo disponer de un entorno de pruebas donde poder investigar y desarrollar soluciones de seguridad es fundamental. El despliegue y adopción de los dispositivos conectados es exponencial y es por ello por lo que los cibercriminales se aprovechan de esta tecnología para llevar a cabo sus ataques. Un ejemplo de ello es Mirai, un malware cuyo fin es crear una botnet (red de dispositivos "zombies" que siguen las ordenes de su dueño) de dispositivos IoT. Con esta red se realizaron los dos ataques de denegación de servicio DDoS (Distributed Denial of Service) más grandes de la historia [10]. En muchas ocasiones estos dispositivos son cámaras, micrófonos, dispositivos domóticos, etc. Pudiendo formar el ecosistema IoT parte de nuestra casa, por lo que la privacidad de los usuarios y la integridad de la infraestructura también pueden estar en peligro.

Para poder desarrollar soluciones de seguridad ante este tipo de ataques se requiere conocer el funcionamiento de cada uno de los dispositivos conectados y descubrir sus vulnerabilidades para, a continuación, poder desarrollar sistemas de protección y estrategias de defensa. Por ello, es necesario disponer de un entorno de pruebas que cuente con una amplia variedad de dispositivos, marcas, tecnologías y protocolos con el fin abarcar la mayor cuota de mercado disponible. Este es el marco que justifica el proyecto en el que se incluye este TFG.

El proyecto cuenta con 3 puntos principales.

El primer punto consiste en el diseño construcción y configuración del entorno de pruebas (casa conectada). Se van a reunir la mayor cantidad de tecnologías y dispositivos diferentes posibles. Todos los dispositivos seleccionados para la creación de este entorno son productos de consumo basados en las tecnologías propietarias. Por ello este TFG va a proporcionar el único dispositivo del entorno basado en tecnologías de "hardware" y "software" libre. Dentro de esta primera fase se incluye además de la instalación y configuración de todos los dispositivos, la integración con otros servicios y/o dispositivos compatibles, por ejemplo, plataformas como Alexa, Google Assistant o IFTTT. Y el diseño y montaje del *layout*[11] con las diferentes estancias (salón, garaje, cocina, habitaciones, etc) y los subcircuitos necesarios para el correcto funcionamiento del entorno (eléctrico, calefacción, red, etc).

El segundo punto consiste en realizar un estudio de seguridad del entorno. El estudio consiste en analizar los dispositivos y sus posibles vulnerabilidades utilizando técnicas de ataque.

[11] Layout: Diseño y disposición de elementos.

El tercer punto abarca el desarrollo de soluciones para segurizar el entorno como, por ejemplo, el desarrollo de sistemas de monitorización. Una vez obtenidos resultados del trabajo con el entorno experimental se aplicarán a la oferta de servicios de seguridad que ofrece S2 Grupo.

El alcance de este TFG se limita, dentro del primer punto de este proyecto, a la creación, configuración y puesta en marcha de ese único dispositivo basado en Hardware y Software libre del entorno.

Por otra parte, con el *open source* es el usuario el que se encarga de la gestión de los datos recogidos por el dispositivo, permitiendo elegir entre una plataforma en la nube como podría ser io.adafruit.com o mantener los datos en la red local utilizando una base de datos. Esto es una ventaja que los productos comerciales no permiten, pues suelen existir una línea de negocio encaminado en el tratamiento y/o venta de estos datos.

Esta tecnología también permite una completa adaptabilidad a las necesidades, es posible añadir o quitar sensores, funcionalidades, integración con otros dispositivos, etc. Cualidad de la que no disponen los productos disponibles para él público, que cuentan con normalmente cuentan con unas funcionalidades prefijadas y poco margen de cambio.

Por el contrario, para poder hacer uso de esta adaptabilidad son necesarios conocimientos específicos, tanto en electrónica como en protocolos y programación. En cambio, los dispositivos ofrecidos por las empresas vienen preparados para enchufar y funcionar, ampliando su target de usuarios finales respecto al *open source.* También los dispositivos comerciales poseen un mejor diseño, tanto en el producto como en el software que lo acompaña, pues es uno de sus reclamos diferenciadores.

En definitiva, el *open source* ofrece muchas más posibilidades y flexibilidad, sacrificando diseño y facilidad de uso.

2.2. OBJETIVOS

El objetivo es diseñar e implementar un sistema de adquisición de variables de entorno, con tecnología IoT, basado en hardware y software libre, y que se integrará en el entorno de pruebas en S2 Grupo. Se ha definido que en el modelo de casa conectada la ubicación será el garaje y por tanto deberá monitorizar las variables indicadas en el apartado 1.4. El dispositivo deberá ser capaz de almacenar esa información en una base de datos para su posterior consulta y análisis. También se pretende que los datos puedan ser visualizados. Por tanto, los objetivos a alcanzar son:

- Utilizar hardware y software libre.
- Diseñar e implementar la monitorización de las variables indicadas en el apartado 1.4.
    - Fugas
    - Calidad de aire
    - Ruido y luminosidad
    - Temperatura y humedad.

- Adquisición de los datos en tiempo real.
- Diseñar e implementar la conectividad.
- Diseñar e implementar almacenamiento de los datos
- Dotar al dispositivo visualización a los datos.

CAPÍTULO 3. DESCRIPCIÓN DE LA SOLUCIÓN

Para resolver el problema planeado en los objetivos se ha decidido implementar un sistema con la siguiente estructura (véase Figura 8).

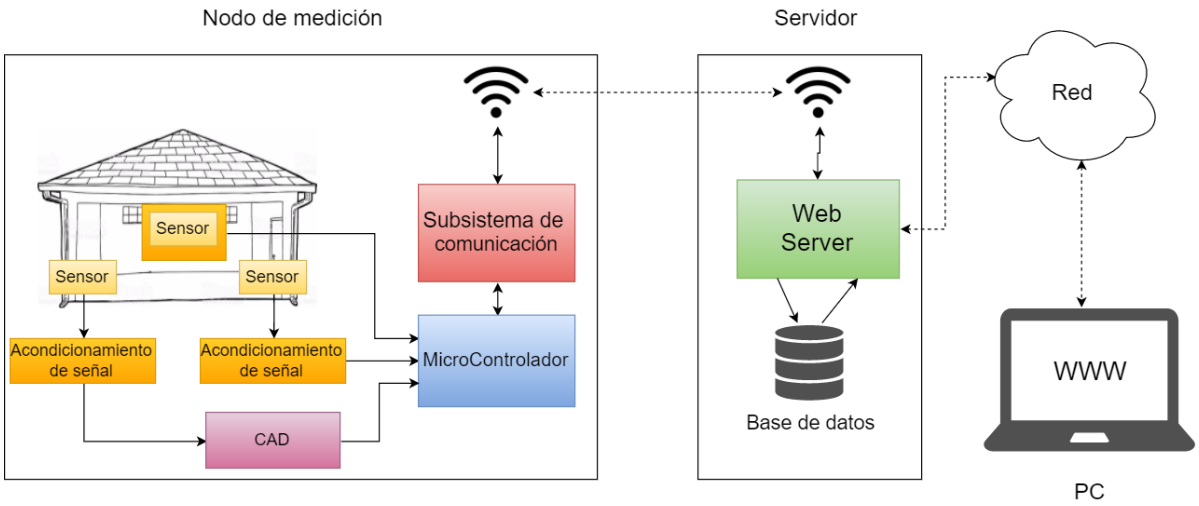


Figura 8: Esquema de la solución.

Se utilizarán distintos sensores para monitorizar las distintas variables de entorno mencionadas anteriormente. Dado que la amplitud de señal proporcionada por el sensor suele ser pequeña, de orden de pocos mV o menor, se precisa un circuito de acondicionamiento de señal responsable de amplificar y ajustar la señal en el ancho de banda correspondiente. Lógicamente, el circuito de acondicionamiento requerido para los distintos sensores podría ser distinto. En este proyecto, tal como se describirá más adelante, todos los sensores seleccionados integran su circuito de acondicionamiento. Tras el procesamiento analógico, la señal salida de cada sensor es digitalizado mediante un CAD para su posteriormente visualización y análisis. Esta función podría realizarse por el propio microcontrolador, mediante un CAD externo o incluso podría estar integrado en el propio sensor en el caso de sensor inteligente. El microcontrolador es el responsable de la lectura de datos de los distintos sensores y a su vez está conectado a un subsistema de comunicación. Este subsistema se encarga de transmitir los datos recogidos por sensores y procesados por el microcontrolador y enviarlos al servidor (*server*). El servidor recibe la información y la almacena en una base de datos. En él también se aloja una página web donde se pueden consultar los datos en tiempo real y un histórico de los mismos.

Una característica importante de este tipo de solución es que el servidor puede recibir y administrar la información de distintos nodos compuestos por sensores, acondicionadores de señal y un microcontrolador, adoptando así un modelo de solución escalable y distribuido. Esto

supone una ventaja si en un futuro se decidiese ampliar los espacios a monitorizar, pues solo sería necesario instalar un nuevo nodo y reconfigurar correctamente el servidor.

Además, disponer de una base de datos centralizada en el servidor permitiría la extracción de los datos de todos los sensores de una forma más rápida y eficaz que si estuviesen distribuidos. Sobre estos datos se podrían realizar análisis en profundidad facilitando el descubrimiento de patrones o relaciones interesantes que a priori pasarían desapercibidas. No obstante, esta última característica no será explotada en este TFG ya que queda fuera del alcance de este TFG.

CAPÍTULO 4. DESARROLLO DEL HARDWARE

En este capítulo se expondrán y justificarán las elecciones de Hardware que compondrá el sistema.

4.1. SENSORES.

Tal como se describe en el apartado 1.4, se pretende monitorizar las siguientes variables: fugas de agua, calidad de aire, luminosidad, ruido, temperatura y humedad.

4.1.1. FUGAS DE AGUA.

La detección de fugas de agua se lleva a cabo mediante el registro de cambios en la conductividad del medio debido a la presencia de agua. Las especificaciones requeridas son:

- Rango de medida: [0-5cm] Profundidad de agua.
- Rango temperatura funcionamiento: [5°C - 45°C].
- Precisión: Fiabilidad en la presencia/ausencia de agua.
- Resolución: No se precisa resolución alta.
- Linealidad

Analizando el mercado encontramos dos tipos de sensores que podrían servir (véase Figura 9), aunque se rigen por el mismo principio de funcionamiento y por tanto responden de la misma manera a las especificaciones. Lo que los diferencia es la forma en la que están construidos para adaptarse mejor a cada una de las situaciones.

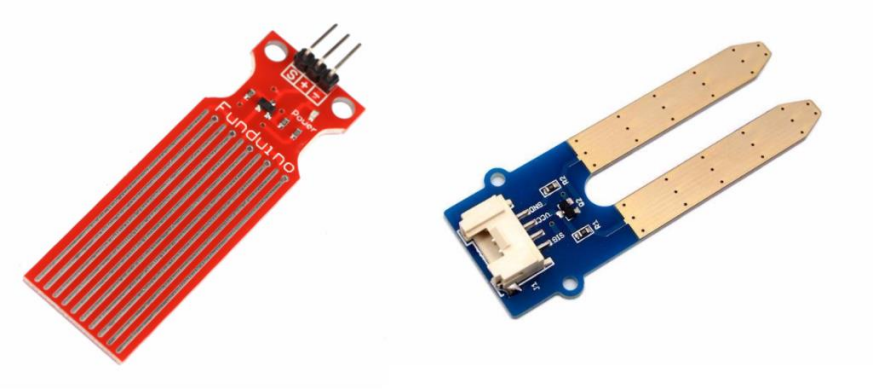


Figura 9 Izquierda: sensor nivel de agua de agua. Derecha: Sensor humedad en suelo.

- **Sensor de nivel de agua**: Forma rectangular, ideado para colocarse verticalmente y determinar la altura a la que se sitúa el nivel de agua.

- **Sensor de humedad en suelo**: Forma de "U", construido para clavarse en la tierra y detectar la presencia o ausencia de agua.

Finalmente, se ha decidido optar por el sensor de humedad en suelo debido a su diseño en forma de "U" facilita la detección de pequeñas fugas al entrar los extremos en contacto con el agua de la fuga con más mayor facilidad.

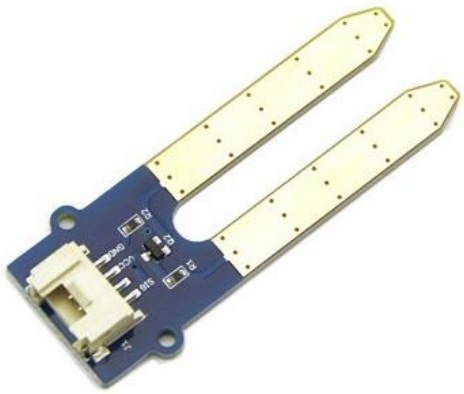


Figura 10 Sensor humedad en suelo

Las características del sensor son las siguientes:

- Alimentación: 5V
- Voltaje de Salida: 0 a 4.2V DC
- Consumo: 35mA
- Dimensiones: 5.5x2.3x0,7cm

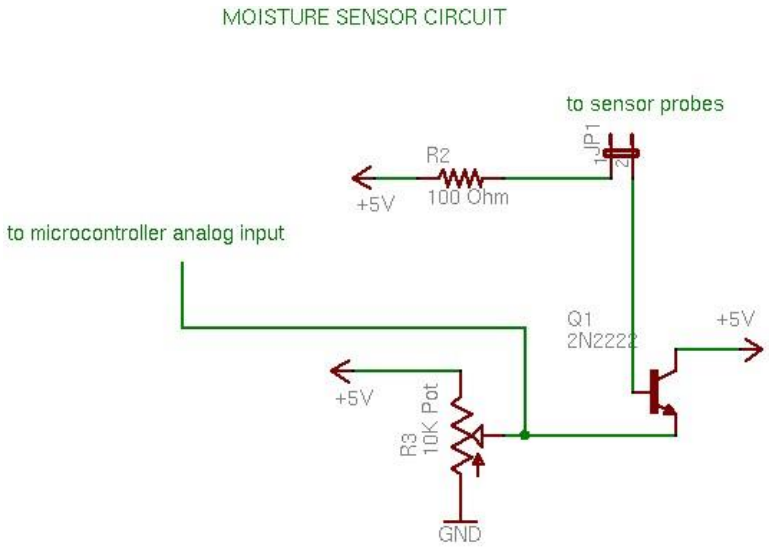


Figura 11 Esquema circuito de acondicionamiento del sensor de humedad en suelo

La salida del sensor es analógica, es decir, proporciona un voltaje en función de la cantidad de agua detectada. Debido a la falta de precisión en la documentación del sensor elegido se han realizado unas pruebas de calibración del sensor, introduciendo el mismo en un vaso de agua y anotando la longitud del sensor introducida y salida del mismo. Se han usado las

marcas de los terminales para tener una mayor precisión a la hora de introducir el sensor en el agua. El sensor se ha conectado a la entrada analógica A1 de una placa de Arduino y se ha usado el valor digitalizado de dicha entrada (0-1023) como salida del sensor. La medición se ha realizado tres veces y se ha calculado la media de cada uno de los valores para la obtención de la siguiente gráfica y su línea de tendencia.

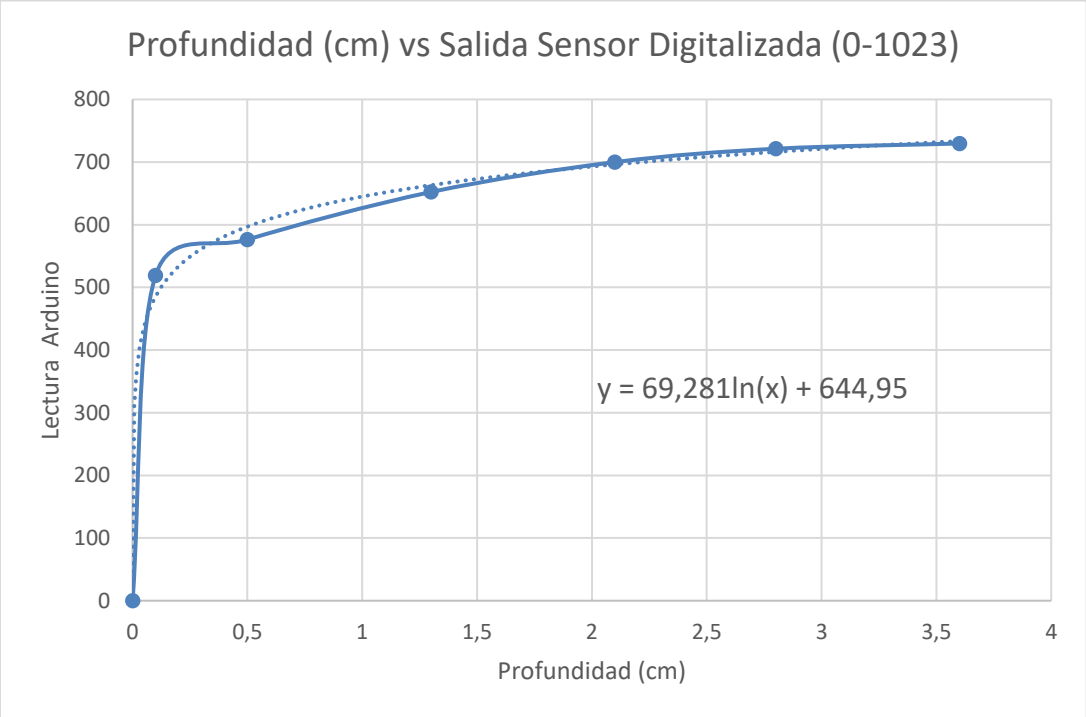


Gráfico 1 Calibración Sensor Fugas

A partir de la línea de tendencia se ha despejado la incógnita x que representa la profundidad, obteniendo la ecuación 1, que es la fórmula usada para convertir salida del sensor en profundidad.

$$x(cm) = e^{\frac{y-644,95}{69,281}} \quad (1)$$

4.1.2. CALIDAD DEL AIRE.

Para evaluar la calidad del aire se ha propuesto usar un sensor de calidad de aire especifico. Analizando el mercado se han encontrado tres tipos de sensores sus características se definen a continuación en esta tabla comparativa.

	MQ135	**TP401**	**MP503**
Material	Nano-SnO2 (dióxido de estaño) dopado con catalizador	Nano-SnO2 (dióxido de estaño) dopado con catalizador	-

Gas de detección	NH3, NOx, alcohol, benceno, humo, CO, CO2, etc.	Humo, CO, alcohol, acetona, insecticida, benceno, formaldehído, etc.	CO, alcohol, acetona, disolvente, formaldehído, etc.
Condición de uso estándar	Temp: 20 ± 2 Humedad: 65% ± 5%	-	Temp: 20 ± 2 Humedad: 65% ± 5%
Gráfico de la sensibilidad a los gases	Sí	-	Sí

Tabla 2 Tabla comparativa sensores de calidad del aire extraida de LeAirf: Air Quality Detection Necklace – Stream's ITP Archive, Wp.nyu.edu, [en línea] Disponible en: https://wp.nyu.edu/streamgao/?p=1750 [Consulta: 9 de agosto de 2017].

El sensor elegido es el MP503 Air-Quality Gas que viene integrado en el módulo Air Qualitiy Sensor en su versión 1.3 diseñado por Seeedstudio.

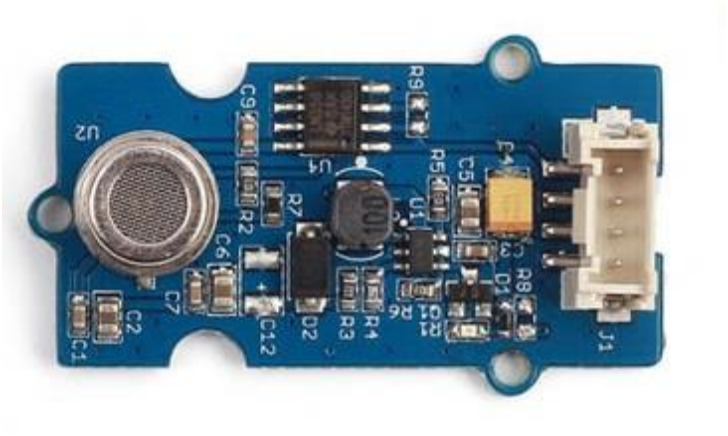


Figura 12 Sensor calidad del aire. Air Quality Sensor v1.3

Según indica su fabricante, debido al mecanismo de medición no puede ofrecer datos cuantitativos sobre la concentración de gases, pero si datos cualitativos sobre la contaminación del aire, estableciéndose diferentes niveles: “Fresh air”, “Low pollution”, “High pollution”, y “High pollution” con “Force signal active” [11].

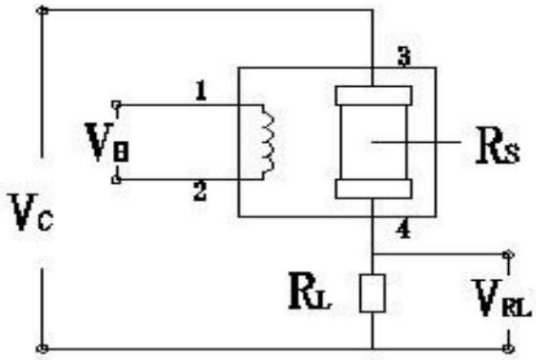


Figura 13 Esquema circuito de acondicionamiento del sensor MP503

El funcionamiento del sensor MP503 se muestra en la Figura 13. La tensión Vc es la alimentación del circuito del sensor, mientras que la tensión VB a la que está conectar una resistencia sirve para mantener el sensor en su temperatura de trabajo. La resistencia Rs varía en función de calidad de aire, y consecuentemente la tensión de salida Vrl. La sensibilidad el sensor ante los diferente agentes gaseosos queda definido en la Figura 14

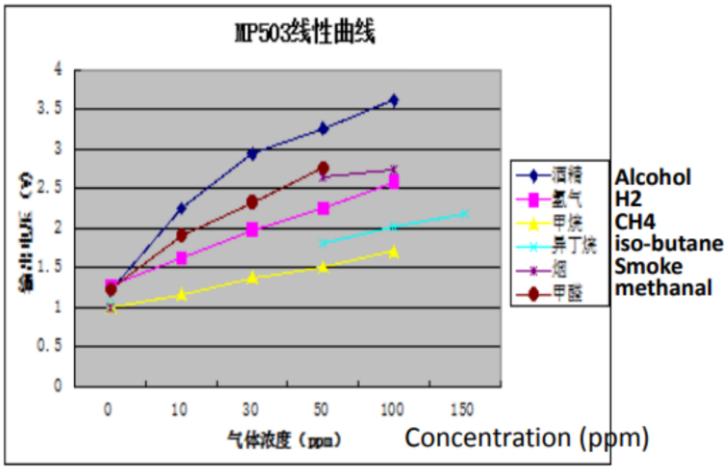


Figura 14 Salida (Voltios) del sensor MP503 en función de la concentración de diferentes gases.

Este sensor presenta diversas ventajas como son su amplio número de gases detectables, su bajo precio y su durabilidad. Pero también presenta inconvenientes como la necesidad de aire limpio para iniciar las mediciones, y que la exposición prolongada en atmosferas contaminadas disminuye su sensibilidad considerablemente [12].

4.1.3. RUIDO.

Para el sensor de ruido se ha reutilizado el sensor KY-037 de un proyecto anterior, el cual se compone de un micrófono de alta sensibilidad y un comparador LM393. El sensor tiene dos salidas. Una analógica que amplifica la señal recogida por el micrófono y un digital que se activa cuando el sonido supera una determinada amplitud.

Figura 15 Sensor de sonido KY-037

Sus especificaciones son las siguientes:

- Voltaje de fuente: 5V
- Salida: Digital o Analógica
- Respuesta de frecuencia: 20Hz - 20 KHz
- Umbral regulable

4.1.4. LUMINOSIDAD.

Existen tres tipos fundamentales de sensores de luz que son fotorresistencias, fotodiodos y fototransistores en función en que tecnología estén basados.

Figura 16 Sensor de luminosidad MM109.

El sensor elegido ha sido un TEMT 6000 que viene integrado en la placa MM109 de la marca Velleman. Se trata de un sensor basado en la tecnología de fototransistores con el siguiente acondicionamiento eléctrico.

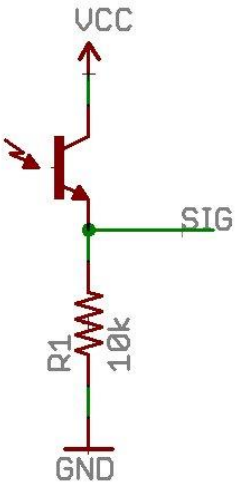


Figura 17 Esquema circuito de acondicionamiento del sensor TEMT6000

El circuito se ha implementado como un divisor de tensión, en el que el sensor actúa como una de las resistencias del divisor cambiando su valor según la incidencia de la luz en él.

Sus especificaciones técnicas son:

- Rango de Temperatura: -40 a 85 °C
- Rango de Medición (Lux): 1 a 1,000 Lux
- Rango espectral:
- Voltaje: 5V
- Salida analógica.

La gráfica que relaciona la iluminancia y la corriente que atraviesa el fototransistor es la siguiente:

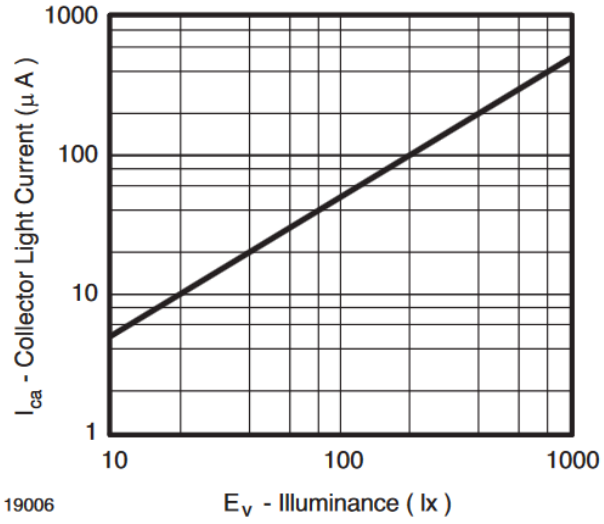


Figura 18 Relación entre iluminancia y corriente que atraviesa el fototransistor. Collector Light Current vs. Illuminance Extraido de TEMT600 SataSheet [en línea] Disponible en:

https://www.sparkfun.com/datasheets/Sensors/Imaging/TEMT6000.pdf [Consulta: 10 de agosto de 2017].

Para realizar convertir la señal obtenida por el sensor en lux usamos la Figura 18 para extraer la relación entre iluminancia y la corriente que atraviesa el fototransistor obteniendo:

$$\frac{100 \cdot 10^{-6}}{200} = \frac{I_{ca}}{E_v} \quad (2)$$

De esta relación despejamos I_{ca} y sustituimos su resultado en la ecuación resultante de calcular el voltaje de salida (sig) de la Figura 17.

$$V_{sig} = I_{ca} \cdot R1 \quad (3)$$

$$V_{sig} = \frac{100 \cdot 10^{-6}}{200} \cdot E_v \cdot R1 \quad (4)$$

Sustituyendo el valor de la resistencia R1 y operando obtenemos:

$$2000 \cdot V_{sig} = E_v \quad (5)$$

Teniendo en cuenta que el conversor AD de la placa de Arduino es de 10 bit (1024 pasos), y la tensión máxima de salida del sensor es 5V la fórmula de conversión resultará:

$$E_v = 2000 \cdot \frac{5}{1023} \cdot AD = 9{,}77 \cdot AD \quad (6)$$

Siendo AD la lectura proporcionada por el convertido Analógico-Digital.

4.1.5. TEMPERATURA Y HUMEDAD.

Existen gran multitud de sensores que proporcionan las medidas de temperatura y humedad deseadas, no obstante, nos vamos a centrar en el mercado *low-cost* y en la gama DHTXX.

	DHT11	**DHT22**
Rango temperatura (°C)	0 a 50 °C ± 2°C	-40 a 125 °C ± 0,5 °C
Rango humedad (%)	20 a 80 % (5% precisión)	0 a 100% (2% precisión)
Frecuencia máxima de muestreo (Hz)	1 Hz	0.5 Hz
Tensión alimentación	5V	5V
Salida	Digital	Digital

Tabla 3 Comparativa sensores DHTXX

Atendiendo a las características presentadas por los sensores y sabiendo que el sensor va a estar situado en el interior no es necesario que soporte temperaturas por debajo de 0 °C ni

por encima de 50 °C, por lo que se selecciona el sensor DHT11 (Figura 19). Además, la frecuencia máxima de muestreo es mayor por lo que nos resulta más interesante (Tabla 3).

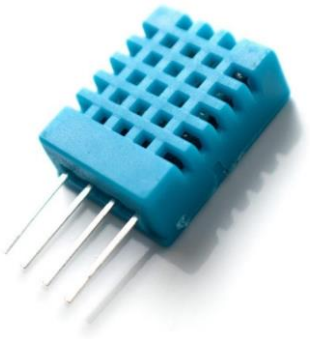


Figura 19 Sensor de humedad y temperatura

La salida del sensor es digital pero no utiliza un protocolo estándar I2C, SPI o 1Wire. El protocolo viene definido en la documentación y su funcionamiento queda descrito en la Figura 20.

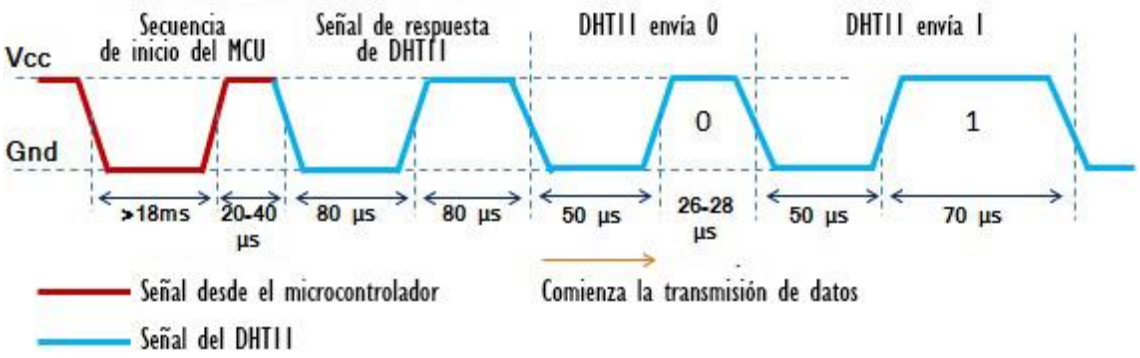


Figura 20 Funcionamiento del protocolo de comunicación del sensor DHT11

4.2. SUBSISTEMA DE COMUNICACIÓN.

Para que el entorno IoT funcione, los diferentes dispositivos deben poder comunicarse. Para ello debe existir un medio por el que hacerlo, en este apartado analizamos las diferentes posibilidades y las ventajas e inconvenientes de cada una de las soluciones.

Existen muchas formas de otrogar conectividad a un dispositivo, no obstante para la realización de este proyecto se han estudiado las 4 tecnologías más extendidas. Ellas son Wifi, 4G, Bluetooth, y Ethernet.

- **Bluetooth:** Según describe en su página web [13], es una Tecnología de conectividad inalámbrica de baja potencia utilizada para transmitir audio, transferir datos y difundir información entre dispositivos. Existen distintas versiones, las apellidadas LE (Low Energy) son de bajo consumo y son las más indicadas para ser usadas en dispositivos IoT. Por otro lado, la próxima versión en ser comercializada (Bluetooth 5.0) está específicamente pensada para estos dispositivos.

- **Ethernet:** Es una conexión física mediante cable con el *router* o *switch.* Proporciona un ancho de banda mucho mayor que las conexiones inalámbricas y se rige por el estándar IEEE 802.3. Su conector habitual es el RJ45.
- **4G:** Es la tecnología usada por los móviles para la realización de llamadas y el envío y recepción de datos. En su cuarta generación se caracteriza por una alta velocidad de transferencia (hasta 1 Gbit/s en conexiones estáticas) y está basada en la arquitectura LTE (Long term Evolution).
- **Wifi:** Es el nombre comercial del estándar de comunicaciones inalámbrico IEEE 802.11. El uso de esta tecnología está ampliamente extendido en hogares y empresas, y prácticamente cualquier dispositivo inteligente lo incorpora.

La elección de la tecnología se va a basar en las siguientes especificaciones:

- **Ancho de Banda:** No se necesita un gran ancho de banda, no se van a transmitir grandes cantidades de datos.
- **Alcance de la red (área extensa, área local, área personal):** Se requiere como mínimo una conexión de área local para la comunicación servidor-microcontrolador.
- **Tipo de conexión:** Inalámbrica o cableada.
- **Necesidad de infraestructura:** A ser posible, se desea aprovechar la infraestructura ya preexistente en la empresa.
- **Escalabilidad:** Que permita una fácil introducción de nuevos nodos al conjunto.
- **Flexibilidad y movilidad:** En principio no es necesaria la movilidad, pues una vez instalado no se va a desplazar ni transportar. No obstante, si dispone de ella facilita la instalación, permitiendo instalar de forma independiente microcontrolador y servidor.
- **Madurez de la tecnología y compatibilidad con otros dispositivos:** Es interesante que sea compatible con la mayor cantidad de dispositivos para la futura integración en la solución de control centralizado y multiplataforma. Una mayor madurez de la tecnología también proporciona una menor tasa de fallos.

	WiFi	**4G**	**Bluetooth 4.2**	**Ethernet**
Ancho de Banda	Alto	Alto	Medio	Muy Alto
Alcance de la red	Área local	Área extendida	Área personal	Área local
Tipo de conexión	Inalámbrica	Inalámbrica	Inalámbrica	Cableada
Infraestructura	Necesita Punto de acceso WiFi	Necesita Redes 4G	Sí, Red Bluetooth	Sí

Escalabilidad	Alta	Media	Alta	Baja
Flexibilidad y movilidad	Media	Alta	Media	Baja
Madurez / compatibilidad	Alta / Muy alta	Alta / Media	Alta / Media	Alta / Media

Tabla 4 Comparativa conectividad

En Tabla 4 se muestra una comparativa entre las distintas tecnologías.

La conexión mediante Ethernet queda descartada por ser cableada, lo que necesitaría que el servidor y el microcontrolador estuviesen muy cerca. También es la tecnología menos escalable, pues todos los nodos necesitarían estar conectados por cable al servidor.

Actualmente no existe una red bluetooth en la empresa por lo que se descarta esta tecnología por falta de escalabilidad del sistema. Además, las versiones LE de bajo consumo disponen de un alcance reducido 10-15 metros lo que limita la posición relativa de instalación entre microcontrolador y servidor.

Por otro lado, la tecnología 4G sería la más flexible, pues ofrecería una conexión de datos prácticamente en cualquier lugar del mundo. No obstante, necesita la contratación de una línea mensual de datos a una operadora de telefonía. Esto supone un inconveniente en términos de escalabilidad, pues se necesita contratar una nueva línea con cada microcontrolador que se quiera ampliar. Este coste extra mensual puede hacer que el producto pierda interés, además se plantea colocarlo en un garaje por lo que la movilidad que permite sería desaprovechada.

Por último, el subsistema de comunicación elegido es WiFi. El amplio uso y disponibilidad de redes Wifi domésticas como empresariales permite que el dispositivo IoT pueda instalarse en prácticamente cualquier lugar. También es importante las altas posibilidades de escalabilidad y compatibilidad con otros dispositivos, pues la mayoría de dispositivos IoT disponen de esta tecnología de comunicación.

4.2. MICROCONTROLADOR

En los apartados anteriores se ha realizado la selección de los sensores y el subsistema de comunicaciones. En este apartado va a seleccionar el microcontrolador que se encarga de leer y procesar los datos obtenidos de los sensores. Por ello necesitamos un microcontrolador que disponga como mínimo de las siguientes especificaciones:

- 4 entradas analógicas
- 2 entradas digitales
- Conexión wifi o posibilidad de expansión mediante sheilds o similares.
- Disponibilidad de alimentación de 5V para alimentar directamente los sensores.

- Software de programación bajo licencia *open source* o libre.

Existen multitud de placas compatibles con estas características como son las Arduino, prácticamente en toda su gama, las Beaglebone, las Teensy o las de Nanode. Otro tipo de placas como la Raspberry Pi se ha descartado pues no disponen de entradas analógicas, lo que obligaría a usar ADC (conversores analógico-digital) externos.

Se ha seleccionado de cada marca la placa que mejor se adapta a las especificaciones expuestas anteriormente.

	Arduino uno	**Beaglebone Green**	**Teensy 2.0**	**Nanode classic**	**Arduino yun**
Procesador	ATmega328P	AM3358 ARM Cortex-A8	ATMEGA32U4	ATmega328P	ATMEGA32U4 + Atheros AR9331
Memoria	32KB Flash + 2KB SRAM	512Mb RAM + 4GB eMMC + MicroSD	32KB Flash + 2,5KB SRAM	32KB Flash + 2 KB SRAM	32KB Flash +2,5KB SRAM / 64 MB RAM + 16 MB Flash + MicroSD
Entradas A/D	6 analógicas / 14 digitales (6 PWM)	7 analógicas / 65 digitales (8 PWM)	7 analógicas / 25 digitales (7 PWM)	6 analógicas / 14 digitales (6 PWM)	7 analógicos / 20 digitales (7 PWM)
Interfaces	UART, I2C, SPI	UART, GPMC, SPI, I2C	UART, I2C, SPI	UART, I2C, SPI	UART, I2C, SPI
Conectividad	No. (Wifi mediante sheild)	Ethernet	No, (Wifi mediante sheild)	Ethernet	Wifi + Ethernet

Tabla 5 Comparativa microcontroladores

En su mayoría las diferentes propuestas de microcontroladores son bastante parecidas, pues se basan en el mismo hardware, no obstante, existen dos alternativas que destacan sobre el resto. Se trata de las placas Beaglebone Green y Arduino Yun. La BeagleBone Green lleva un microprocesador en vez de un microcontrolador y trabaja sobre un sistema operativo Linux al igual que realiza una Raspberry Pi, aunque no cuenta con conexión Wifi. La placa Arduino Uno y la Teeennsy si son compatibles con WiFi pero necesitan un módulo adicional. Por su parte la placa Arduino Yun (véase Figura 21) cuenta con dos procesadores, un microcontrolador programable y un microprocesador que también trabaja sobre Linux (*Linino*) y controla las interfaces de red (véase Figura 22). Esta última solución es la placa elegida debido a que resulta realmente interesante que incluya el módulo wifi integrado, además del segundo microprocesador permitiendo también la ejecución de scripts.

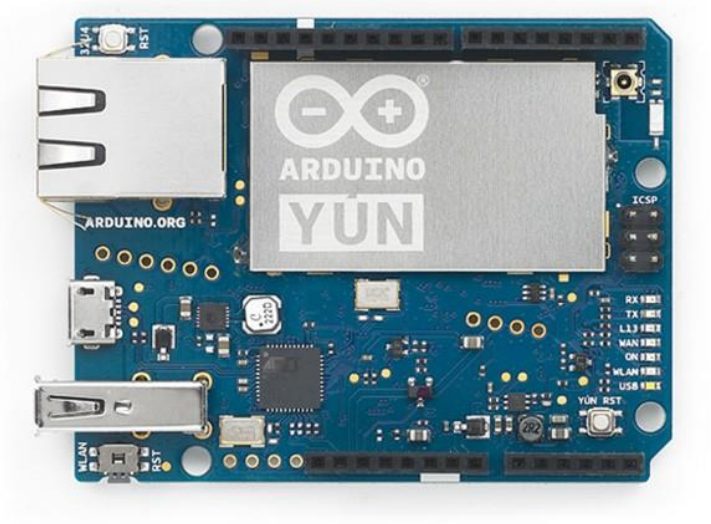


Figura 21 Placa Arduino Yun

En esta dualidad de procesadores, el microcontrolador ATmega32u4 se encarga de gestionar las entradas y salidas analógicas y digitales, así como los diferentes buses de comunicación (Serie, I2C, SPI, etc). También se encarga de gestionar la lógica a bajo nivel. Por otro lado, el microprocesador AR9331 corre una versión ligera de OpenWRT basada en Linux que se denomina Linino. Gestiona las interfaces de red y posee la mayor parte de las funciones de este sistema operativo, entre ellas ejecutar scripts en Python u otros lenguajes.

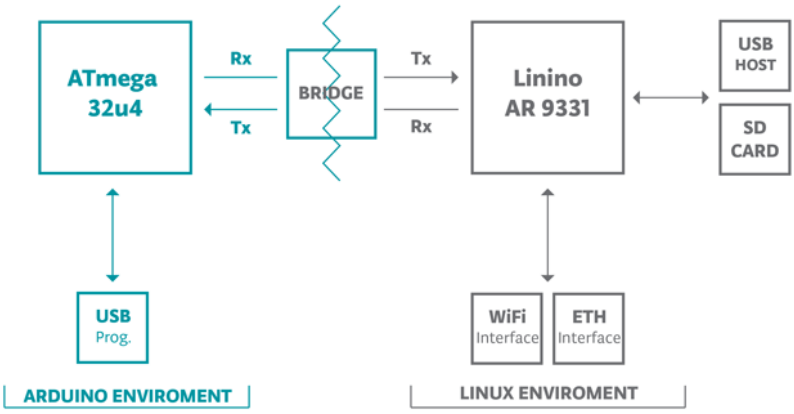


Figura 22 Esquema funcionamiento Arduino Yun

La comunicación entre ambos procesadores es en serie, y la librería que gestiona esta conexión se llama “bridge” (véase Figura 22). Esto es importante pues nos permite acceder realizar gran cantidad de funciones como lanzar programas de Linux mediante el microcontrolador o consultar el estado de las entradas y salidas a través de internet.

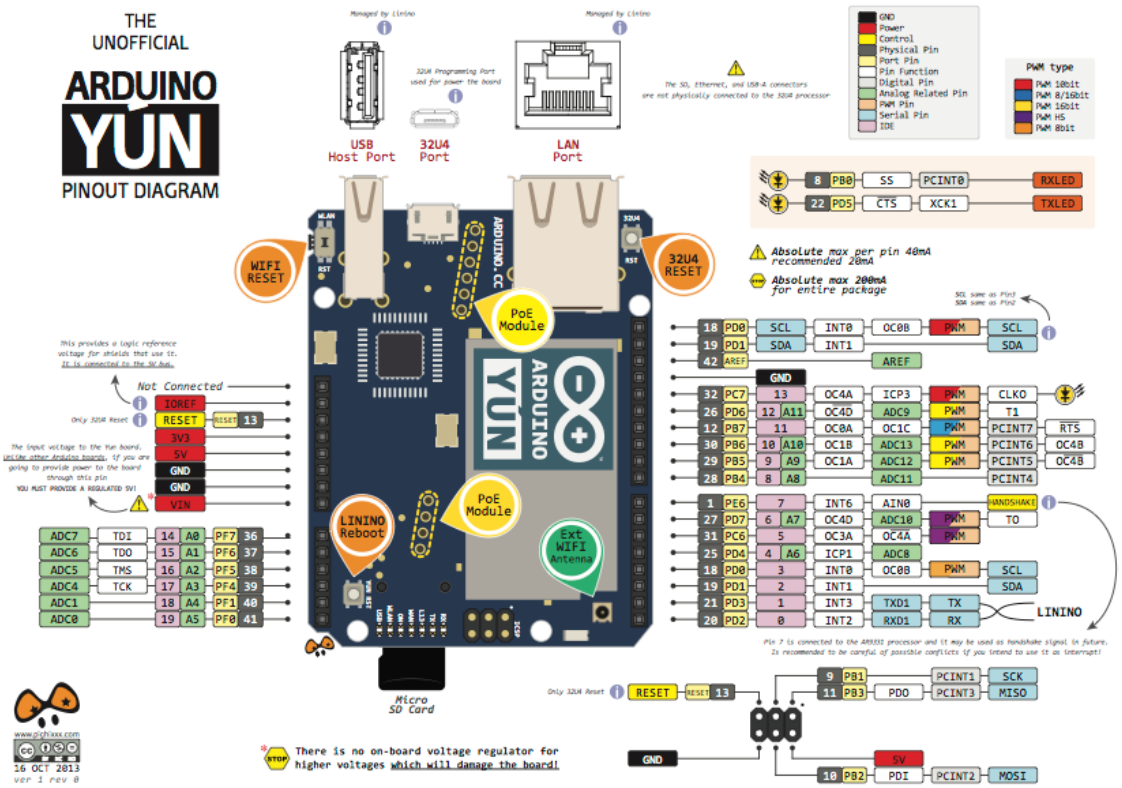


Figura 23 Arduino Yun pinout

Por último, en la Figura 23 se observan la distribución de los diferentes pines de entrada/salida y las funciones y protocolos que llevan asociados.

4.3. SERVIDOR

Como se ha descrito en el capítulo 3, el servidor tendría dos funciones principales. El almacenamiento de los datos y proporcionar una interfaz de interacción con el usuario. Existe un gran abanico de posibilidades. Se puede usar desde un ordenador personal, servicios como AWS (Amazon Web Services), un VPS (Virtual Private Server), u ordenadores de bajo consumo y coste basados en arquitectura ARM. En este caso se ha elegido esta última opción, usando una Raspberry Pi 3 por diversos motivos:

- Facilidad de uso, con una gran cantidad de foros y tutoriales.
- Software adaptado a esta plataforma.
- Bajo consumo y precio.
- Planos y diseño públicos (“hardware libre”).
- Rápida instalación y configuración.
- Wifi integrado en la placa.
- Capacidad de proceso suficiente.

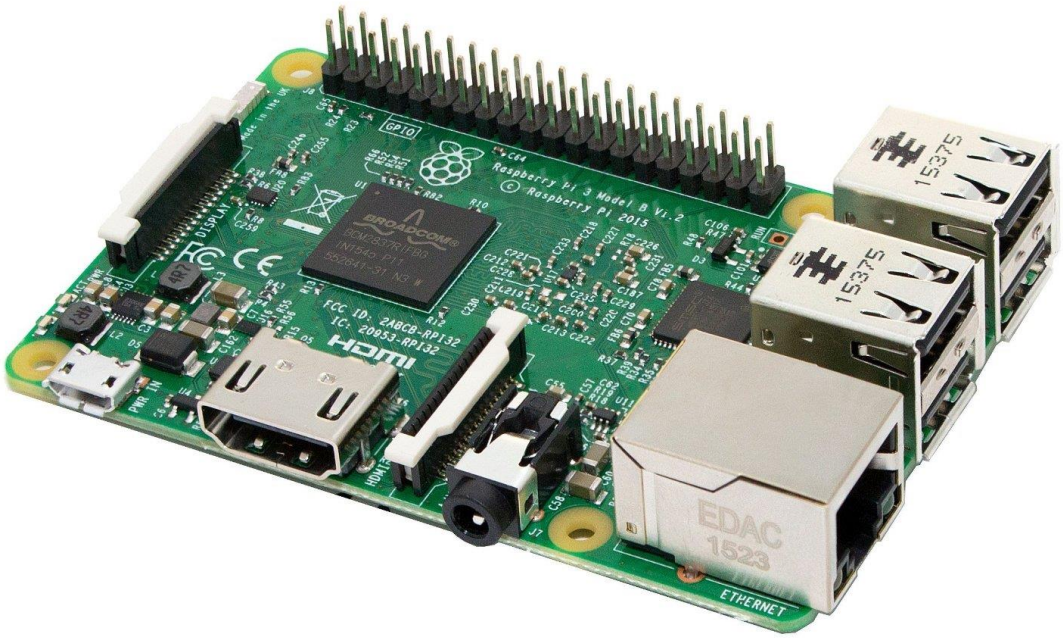


Figura 24 Raspberry Pi 3 Modelo B

Las especificaciones completas de la Raspberry Pi 3 son:

- Procesador Quad Core a 1.2GHz modelo Broadcom BCM2837 64bit.
- 1GB RAM.
- BCM43438 Wifi y Bluetooth Low Energy (BLE) incluido en la placa.
- 40 pins GPIO.
- 4 puertos USB 2.0.
- Salida estero de audio Jack 2.5mm.
- Puerto HDMI tamaño completo.
- Micro SD para cargar sistema operativo y almacenamiento de datos.
- Fuente de alimentación USB hasta 2.5

La Raspberry Pi actuará de servidor recibiendo y almacenando los datos que le envíe el microcontrolador y también alojará la página web. Como se planea que esté funcionando de forma constante también se adquirirá una memoria USB extraíble de 32 GB marca SandDisk para que la base de datos se guarde en ella. Esto supone dos ventajas: la primera es que en caso de fallo los datos estarán a salvo en la memoria externa, y la segunda es que en caso de que se llene esta memoria es posible sustituirla por una nueva sin que haya que parar el dispositivo (“en caliente”).

4.4. CONEXIÓN FINAL DE LOS COMPONENTES

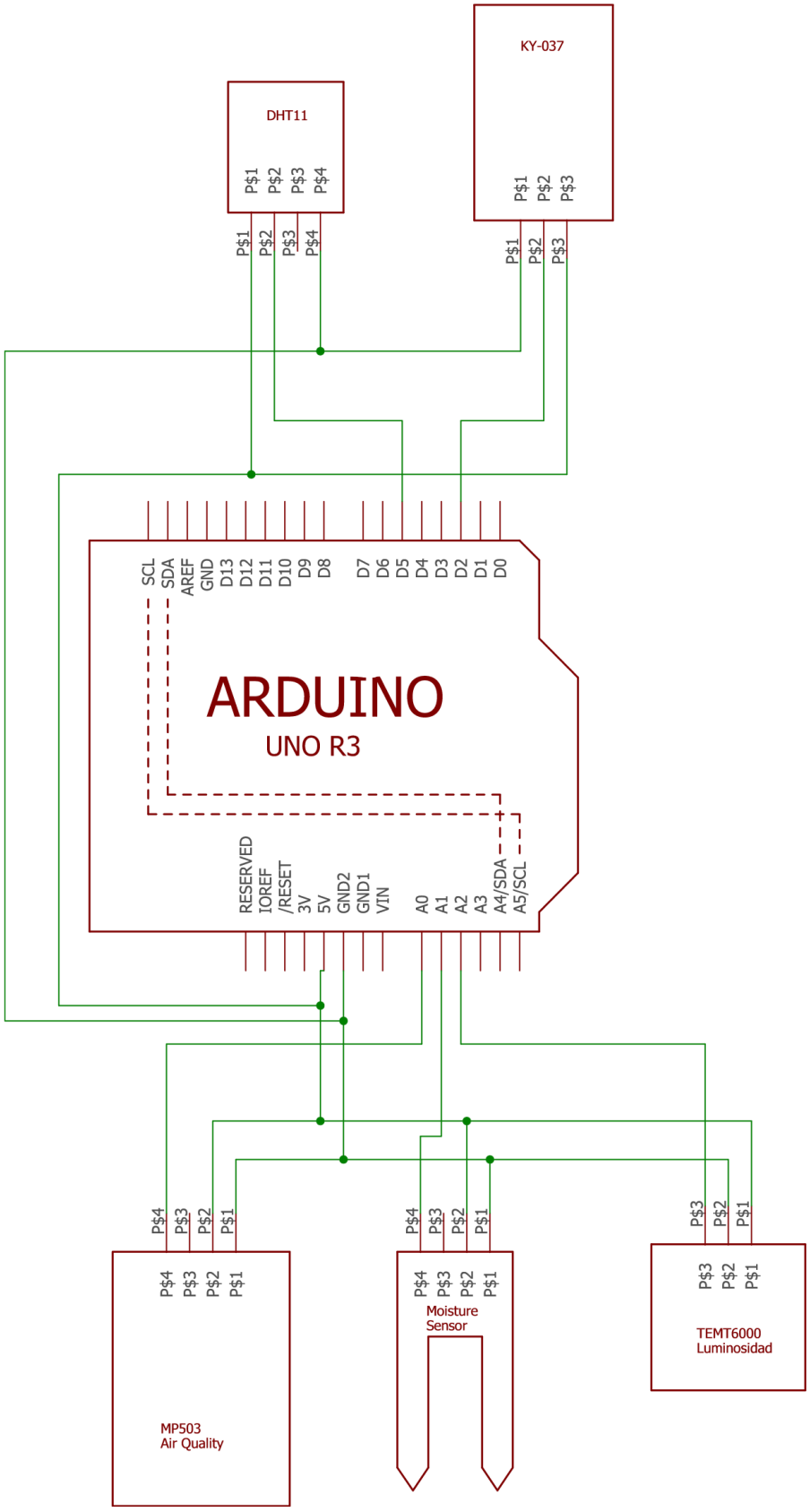


Figura 25 Esquema de conexión de los sensores con la placa de Arduino Yun.

En la figura 25 se pueden observar las diferentes conexiones entre los sensores y la placa de Arduino. Todos los sensores están conectados a *Vcc*(+5 V) y a tierra (*ground*).

Los sensores de fugas, calidad del aire y luminosidad se conectan a las entradas analógicas A1, A0 y A2 respectivamente. Por otro lado, el sensor de humedad y temperatura, al usar su propio protocolo se conecta al pin digital 5, mientras que el sensor de sonido se conecta al pin interrupción *int1* (pin digital 2).

CAPÍTULO 5: DESARROLLO DEL SOFTWARE

En este capítulo se describirán las principales funciones que realiza el programa de software desarrollado para el microcontrolador y el servidor incluyendo los diagramas de flujo correspondientes. También se explicará que entornos de programación se han usado en las diferentes partes del dispositivo.

En primer lugar, para programar el microcontrolador se ha usado Arduino IDE, que es el Entorno de Desarrollo Integrado oficial ofrecido gratuitamente por Arduino. Su funcionamiento es sencillo, una vez instalado en el PC de programación se puede escribir el código, compilarlo y cargar los programas en la placa de Arduino. Permite la utilización de librerías y extensiones que le proporcionan compatibilidad con un gran número de sensores y placas de diferentes marcas. También permite manejar la comunicación serie con dichas placas. El desarrollo del código de los programas está basado en el lenguaje C++, que es una evolución de C en la que se incluye la posibilidad de trabajar con objetos y clases. Los programas de Arduino, también conocidos como *sketch*, incluyen dos funciones principales por defecto, que son la función *setup()* y la función *loop()* (véase Figura 26). La función *setup()* solo se ejecuta cuando la placa es encendida, y en ella deben configurarse todos los parámetros necesarios (por ejemplo, si los pines son de entrada o salida), e inicializarse las librerías que se vayan a utilizar. La función *loop()*, en cambio, es un bucle que se ejecuta constantemente. En ella se debe escribir el código que el microcontrolador ejecutará, por ejemplo, la lectura del valor de un sensor.

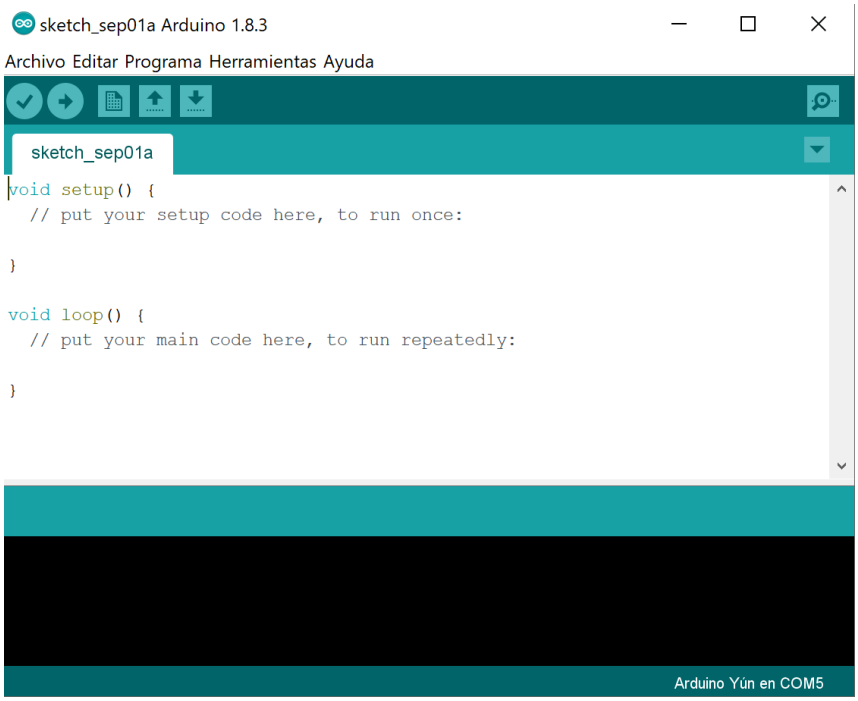


Figura 26 Interfaz gráfica de Arduino IDE

En segundo lugar, se ha utilizado *Visual Studio Code* (VSC) como editor de texto para el código que componen los *scripts* y programas, en Python, del servidor y del microprocesador de la placa de Arduino. Este editor fue desarrollado por Microsoft en 2015, dispone de una gran cantidad de lenguajes predefinidos, autocompletado y control de versiones. Además, mediante las adecuadas extensiones puede compilar y depurar el código. El lenguaje Python usado para desarrollar los programas y scripts es un lenguaje de alto nivel y propósito general. Está orientado a la legibilidad, y su sintaxis facilita programar en muchas menos líneas que otros lenguajes como C o Java. También es importante destacar que es un lenguaje interpretado y no compilado, esto quiere decir que las instrucciones se ejecutan de forma secuencial.

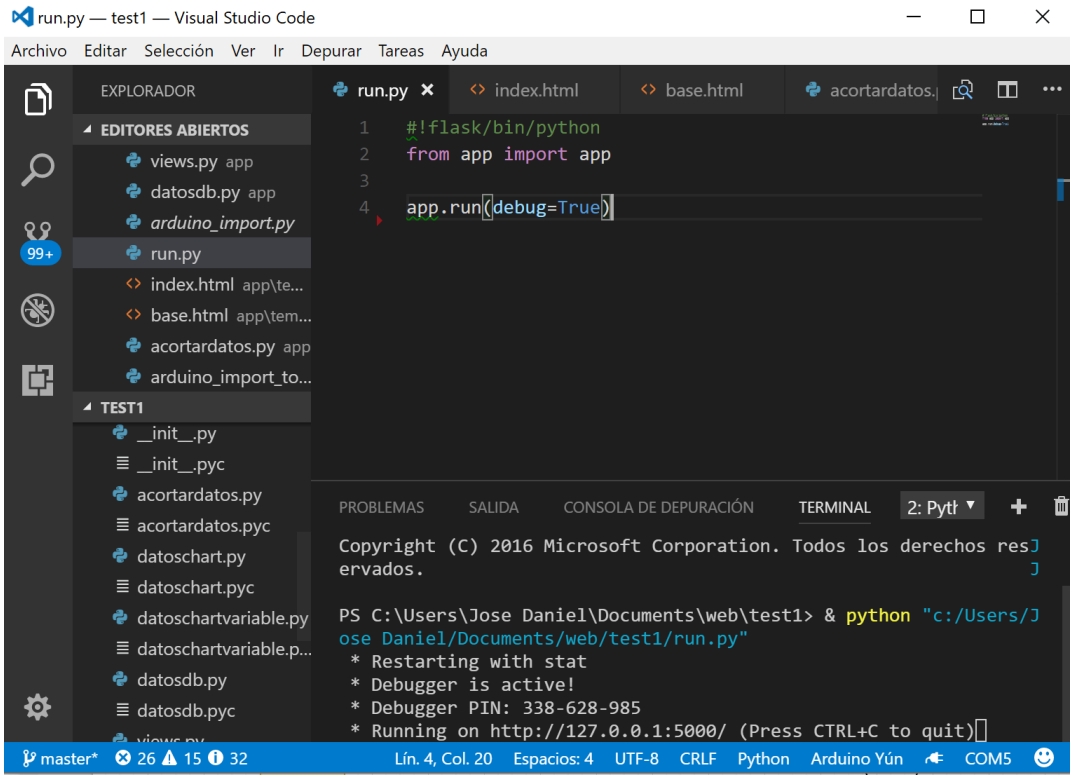


Figura 27 Interfaz gráfica Visual Studio Code.

En los siguientes apartados se entrará en detalle sobre el funcionamiento y software de cada una de las partes.

5.1. MICROCONTROLADOR.

El microcontrolador cuenta con dos procesadores, los cuales han sido definidos en el apartado 4.2. En este proyecto se han usado ambos, a continuación, se define que software se ha programado para cada uno de ellos.

Como se ha indicado en el apartado 4.4, tres de los cinco sensores usados presentan una salida analógica, por tanto, hacen el uso del convertidor A/D integrado en la placa Arduino Yun. El sensor de sonido se conecta al pin 2 de la placa de Arduino Yun y hace uso de las interrupciones para entregar la información. Por otro lado, el sensor DHT11 se conecta mediante uno de los pines digitales y su propio protocolo para entregar la información.

Se han incluido tres librerías en el programa para dotar de comunicación (*Bridge.h*) y una correcta interpretación de los valores de los sensores de calidad del aire (*AirQuality.h*) y de humedad y temperatura (*DTH.h*). Se explicará su funcionamiento a lo largo de este apartado.

La función *loop()* de Arduino se ha diseñado de manera modular, creando una función específica para cada sensor. Cada una de estas funciones engloba todas las acciones necesarias para leer los datos, procesar la información y enviarla a través de la librería *bridge*. En la Figura 28 se puede observar este diseño modular, y el orden en el que se ejecutan las lecturas.

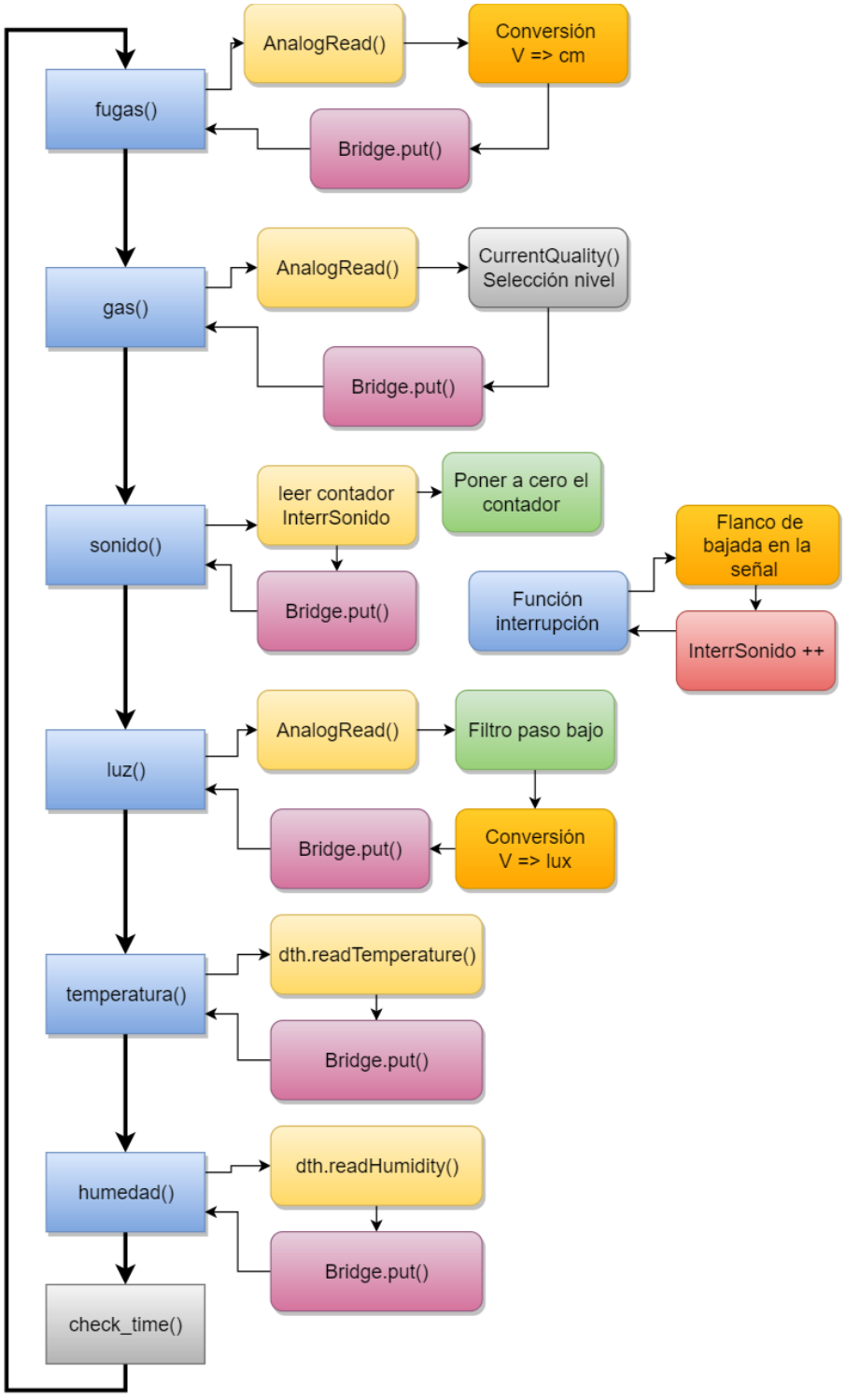


Figura 28 Esquema de funcionamiento del programa del microcontrolador

A continuación, se describen las funciones asociadas a cada sensor en profundidad.

- **Sensor de Fugas.**

A este sensor se le asocia la función *fugas()* que lee el valor del voltaje proporcionado por el sensor en la entrada analógica A1. Este valor es convertido mediante un CAD integrado en la placa y se obtiene un dato entre 0 y 1023. Posteriormente se aplica una conversión, especificada en la ecuación 1 para obtener el valor de profundidad que está sumergido el sensor en el agua de la fuga. Por último, mediante la función *bridge.put(etiqueta*, valor*)* se envían los datos al microprocesador. En este caso la etiqueta será *fugas*. Esta última acción es común en todas las funciones y se describirá más adelante con mayor detalle.

- **Sensor de calidad del aire**.

Para la implementación de este sensor se ha usado la función *gas().* En este caso se ha usado la librería *AirQuality.h* para inicializar el sensor y tratar la información obtenida de él. En primer lugar, al principio del código se incluye la librería y se declara un objeto de clase *AirQuality,* posteriormente, en la función *setup()* se inicializa ese objeto. Estas acciones ponen en marcha el sensor y lo calibran, tienen una duración estimada de dos minutos e informan si ocurre algún error. Una vez finalizado el inicio correctamente, en la función *gas*() dentro de la función *loop*() se lee el valor proporcionado por el sensor y la librería *AirQuality* lo interpreta, devolviendo un nivel de la calidad del aire que se corresponde con lo siguiente:

  - Nivel 3: “Fresh air”
  - Nivel 2: “Low pollution”
  - Nivel 1: “High pollution”
  - Nivel 0: “High pollution, Force signal active”

Finalmente, el resultado se envía al microprocesador mediante la función *bridge.put*().

- **Ruido:**

El sensor de ruido se compone de un comparador, que cuando supera el umbral especificado emite una señal. Cuando no detecta ningún ruido por encima del nivel establecido la salida del sensor es *HIGH*, y cuando se detecta cambia a *LOW*. Por tanto, se ha decidido tratar la información proporcionada por el sensor almacenando el número de veces que el ruido sobrepasa el umbral durante el tiempo de muestreo.

Implementar la función *sonido()* de la misma manera que se han implementado las demás (cuando se llama a la función se realiza la medición) planteaba un problema de pérdida de información, pues si se producía algún sonido mientras se estaba realizando mediciones que no fuese *sonido()* no quedaba registrado. Para solucionar este problema, se han utilizado interrupciones. Una interrupción es un mecanismo que permite interrumpir el hilo principal del procesador para ejecutar una rutina de acciones especificadas. Una vez finalizada la interrupción el procesador sigue el hilo principal en el lugar en el que se interrumpió como se observa en la Figura 29. El evento causante de la interrupción puede ser externa (de hardware) o interna (de software).

Un ejemplo claro del uso de interrupciones por un procesador son las señales enviadas por el teclado. El ordenador se encuentra ejecutando un conjunto de instrucciones, por

ejemplo, procesando datos, pero reacciona ante la pulsación de las teclas. Estas pulsaciones están programadas como interrupciones y evitan la perdida de información (pulsaciones).

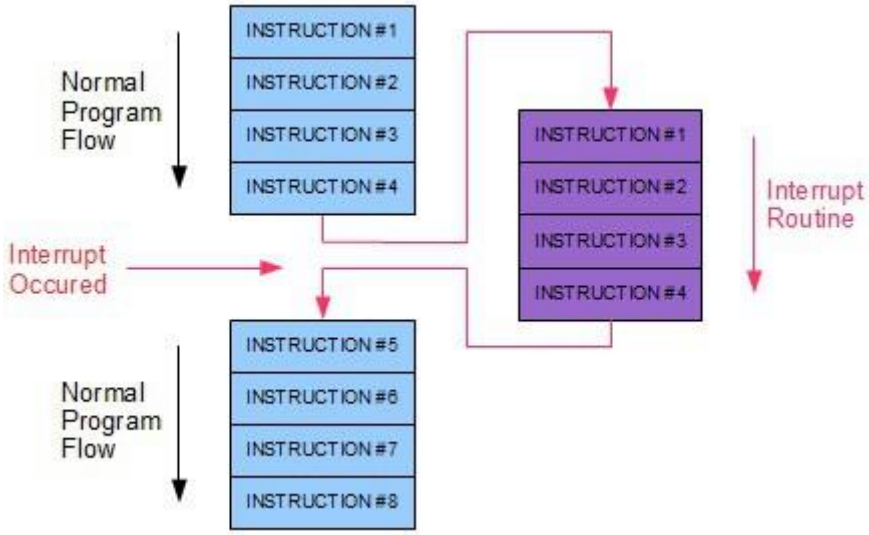


Figura 29 Funcionamiento de las interrupciones

En el caso del sensor de sonido se va a usar la interrupción externa que va a estar asociada a un flanco de bajada detectado en el pin 2. La rutina programada consiste en incrementar el contador de veces que el ruido ha superado el umbral. También se ha programado una función anti-rebote [12]de forma que un mismo sonido no provoque más de una interrupción [14]. Se considera un mismo sonido aquel que ocurre en un intervalo de tiempo inferior a 10 milisegundos.

La función *gas()* es simple, lee el contador de interrupciones producidas, lo reinicializa a cero, y envía los datos al microprocesador. Por último, indicar que el potenciómetro de este sensor se ha ajustado para que emita señal cuando el ruido detectado por micrófono supere los 70dB. Este valor es un valor intermedio entre una conversación y el tráfico de una ciudad.

- **Luz:**

En las pruebas realizadas a este sensor se observaban ciertas oscilaciones debidas a los armónicos producidos por la red eléctrica en el alumbrado. Por ello se decidió implementar un filtro de paso bajo mediante software de forma que eliminase la componente senoidal de la salida del mismo. Así mismo, se toman 4 conjuntos de medidas y se realiza la media. Por tanto, la secuencia de la función *luz()* empieza con la lectura de la entrada analógica, aplica el filtro de paso bajo y el promedio le las mediciones, y usando las ecuaciones descritas en el apartado 4.1.4 se convierte el valor de la entrada en lux. Finalmente se envía el resultado al microprocesador.

- **Temperatura y Humedad:**

Los sensores de la familia DHT utilizan un protocolo de comunicación propio, descrito en el apartado 4.1.4 en la Figura 20. Este protocolo viene implementado en la librería "DHT.h". Al igual que ocurre con el sensor de calidad del aire, en primer lugar, se incluye la

[12] Función que elimina interrupciones no deseadas debidas al ruido producido en la señal cuando se usan interrupciones externas.

librería y se declara un objeto de la clase *DHT*, en este objeto se indica el pin al que está conectado el sensor (pin 5) y el tipo de sensor (DTH11). El desarrollo de funciones *temperatura()* y *humedad()* sigue la misma estructura. Mediante las funciones *dth.readTemperature() y dth.readHumidity()* se obtienen los valores necesarios. Este sensor calcula directamente la temperatura y la humedad y solo es necesario implementar en Arduino la comunicación (placa-sensor) para obtener los datos. Por último, se envían los datos al microprocesador.

El periodo de muestreo se ha establecido en 7500 milisegundos (7,5 segundos). Se ha definido este periodo debido a que el microcontrolador tarda 1930 microsegundos en obtener los datos de los sensores, y el microprocesador unos 4 segundos en enviar los datos y recibir la confirmación de que han sido correctamente insertados en la base de datos. El tiempo del microprocesador puede variar en función de la red a la que esté conectado, de forma que se ha tomado un margen de aproximadamente 1,5 segundos respecto al tiempo total del proceso para asegurar el correcto funcionamiento del dispositivo y evitar la pérdida de datos.

Como la ejecución de las instrucciones del microcontrolador dura 1930 milisegundos añadimos una función extra que se encargue de supervisar que el tiempo entre muestra y muestra sea el establecido. Esta función va a hacer uso de *millis()* y un bucle *while.* La función *millis()* devuelve un entero con la cantidad de milisegundos transcurrida desde el encendido de la placa. El control del tiempo se realizad de la siguiente forma. Se almacena el valor de *millis()* al terminar la función setup(). Se realizan las mediciones, cuando estas terminan se entra en el bucle *while* a la espera de que hayan transcurrido 7500 milisegundos desde que comenzaron las mediciones, y una vez transcurridos se vuelve a guardar el valor de *millis()* cerrando el bucle principal y empezando todo el proceso de nuevo.

La librería *Bridge* habilita la transmisión de datos entre los dos procesadores de la placa simplificando el uso de la comunicación. Proporciona un espacio de almacenamiento para compartir, por ejemplo, las lecturas de sensores en internet, y recibir comandos de Internet y enviarlos a Arduino. En este dispositivo se han usado las funciones *put()* y *get()* que permiten envíar y extraer información del espacio compartido que crea la librería. En el skecth de Arduino se ha usado bridge.put() para enviar los datos. Esta función toma dos parámetros que son la palabra clave, y la información a transmitir (véase Figura 30). La palabra clave sirve para recuperar los datos enviados a través la función *bridge.get().*

Figura 30 Ejemplo de uso de funciones de la librería Bridge.

Una vez los datos están en el espacio compartido, desde el microprocesador ejecutamos un script de Python llamado *send_data_to.db*. Este script recoge los datos compartidos por el

microcontrolador mediante la función bridge.get(), les añade la hora en formato UNIX[13] y le aplica un cambio formato para convertirlos en tipo *json*, debido a que este es el formato requerido para la correcta recepción y procesado de información de la base de datos. Finalmente se envían los datos a través de la comunicación WiFi a la base de datos (véase Figura 31).

Para enviar los datos se ha usado la librería *pymongo* de Python. Esta librería gestiona la conexión con la base de datos, permitiendo la autenticación y el tratamiento de datos (añadir, eliminar, buscar, actualizar, etc). Para introducir nuevos datos en la base de datos la librería utiliza una petición *http* tipo *post* a la dirección *IP* del servidor y al puerto por defecto (27017). En el cuerpo de la propia petición se incluye el fichero *json* con los datos a añadir.

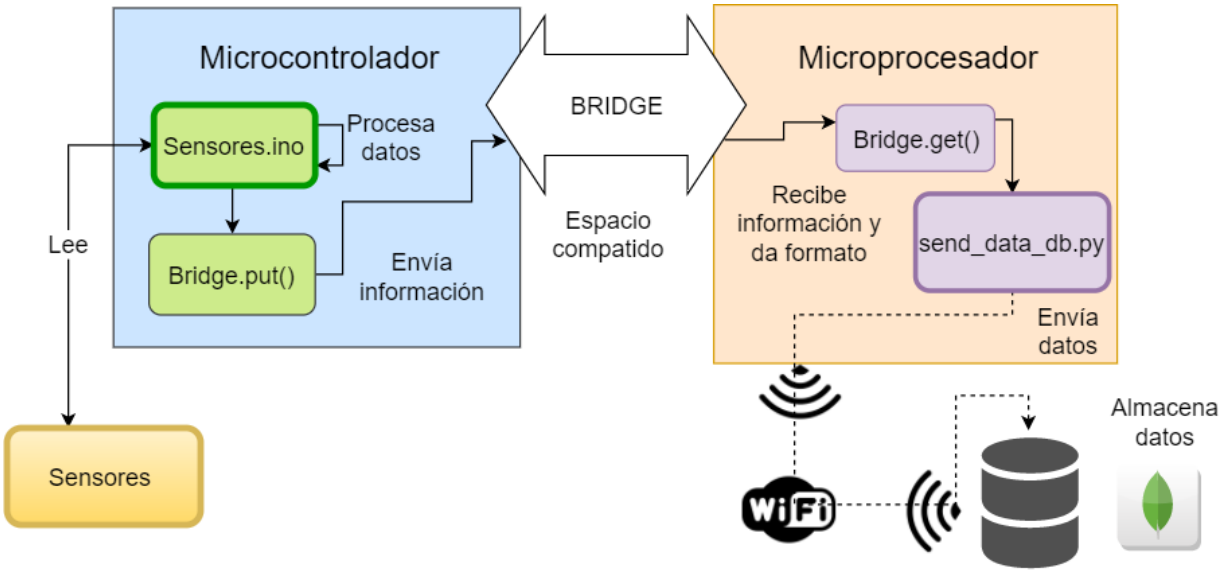


Figura 31 Diagrama funcionamiento completo de la placa Arduino Yun.

5.2. SERVIDOR.

El servidor es una parte muy importante en el sistema pues permite almacenar los datos y también la interacción del usuario con el dispositivo vía página web. La Raspberry Pi 3 es proveedora de estos 2 servicios. La Raspberry Pi corre, sobre una arquitectura ARM, una distribución de Linux llamada *Raspbian* que está basada en la distribución *Debian*. Esta distribución otorga la mayoría de funciones del sistema operativo *Linux*, siendo ideal para el desempeño de tareas que no requieran una alta capacidad de proceso. A continuación, se describe cómo y con qué herramientas se han implementado estos servicios y su relación con el resto del dispositivo IoT.

Para la creación del *back-end,* es decir, la parte que no interacciona con el usuario directamente, y se encarga de procesar las peticiones web, buscar los datos y enviar las respuestas se ha usado el *framework* de Python Flask [15]. Flask permite el desarrollo de aplicaciones web de una forma sencilla. Además es muy popular y ofrece gran cantidad de documentación.

[13] Formato de tiempo ampliamente usado sistemas Linux/unix y computación. Se basa en contar los segundos transcurridos desde el Epoch, que corresonde con medianoche del 1 de enero de 1970.

Para almacenar los datos se ha elegido una base de datos no relacional, por su flexibilidad, facilidad de uso y escalabilidad, además de ser mucho más eficientes a la hora de manejar grandes cantidades de datos y funcionan mejor en dispositivos poco potentes. La base de datos usada es MongoDB, que es de código abierto [16].

Por último, para el *front-end* se ha creado una página web usando HTML y CSS. Y JavaScript para mostrar los datos de manera dinámica. El diseño se ha basado en una plantilla gratuita de Bootstrap [17], y para los gráficos se ha usado JavaScript y las librerías Chart.js [18].

En la Figura 32 se describe de forma global la solución creada, indicando las relaciones entre las diferentes partes, y los principales programas usados para que el dispositivo funcione. En color verde se ha representado los programas creados mediante Arduino IDE, en color naranja los programas escritos en Python. En azul y amarillo, equivalen a HTML y JavaScript respectivamente. También se han indicado los sistemas operativos que incorporan los diferentes microprocesadores.

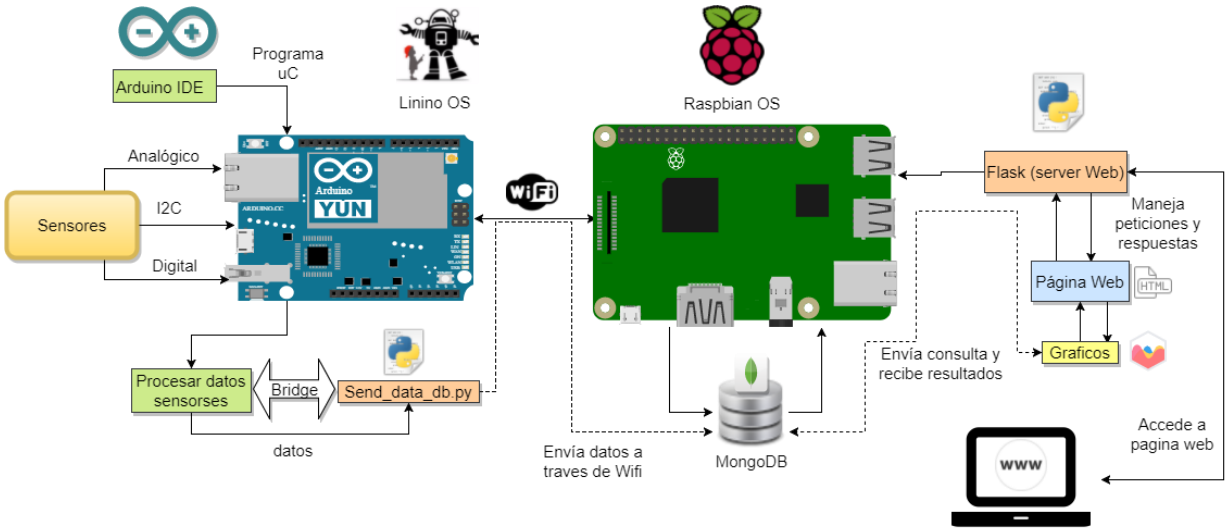


Figura 32 Esquema general de funcionamiento del dispositivo IoT

5.2.1. BASE DE DATOS (MONGODB).

Como hemos descrito anteriormente el software elegido para manejar y almacenar los datos es MongoDB. Se ha configurado la base de datos para que almacene los datos en una unidad USB externa de 32GB de forma que, en caso de fallo, los datos sigan disponibles sin necesidad de recurrir a técnicas avanzadas de recuperación de datos. Con MongoDB se puede disponer de varias bases de datos, que a su vez se subdividen en colecciones. Para cargar o consultar datos se debe especificar en que base de datos y que colección se encuentran los mismos.

Por otro lado, el funcionamiento de la base de datos es muy sencillo. La placa Arduino Yun se conecta al puerto abierto por la base de datos (27000), y carga los datos mediante una petición *post*. Estos datos deben llevar estar en formato *json*. La base de datos asigna un "object_id" único que permite identificar el objeto de forma inequívoca. En la Figura 33 se muestra el formato en el que se almacenan los datos dentro de la base de datos, con el *id* ya añadido.

```
{
        "_id" :ObjectId("59a71fbb5f1d6515b5aeaf4d"),
        "sonido" : "0.00",
        "temp" : "25.00",
        "hum" : "67.00",
        "luz" : "110.99",
        "gas" : "3",
        "fugas" : "0.07",
        "time" : 1504124843.417332
}
```

Figura 33 Ejemplo formato del almacenamiento de datos en la Base de Datos

La consulta de los datos, se pueden realizar de dos maneras. Mediante una petición tipo GET a la *IP* y puerto de la base de datos, o mediante un cliente. Un cliente es una librería que incorpora la comunicación de forma predefinida, facilitando el uso de la base de datos. En nuestro caso hemos usado *Pymongo* que es un cliente para Python. En esencia, las dos formas permiten realizar exactamente las mismas consultas (*queries*). Las consultas realizadas tienen el objetivo de proveer de datos las gráficas y tiene la forma indicada en la Figura 34

```
datos = datadb.find({"time": {"$gte": d}}).sort("time",pymongo.ASCENDING)
```

Figura 34 *Query* usada para buscar información en MongoDB

En la consulta a la base de datos se realiza una petición de búsqueda sobre todos los elementos de la base de datos mediante la instrucción "find()", pero solo devolverá aquellos que cumplan con el parámetro *({"time": {"$gte": d }}).* Este parámetro filtra los resultados por el valor del campo *time* y devuelve aquellos que sean igual o mayor (*$gte)* que el parámetro *d.* Es decir, devuelve aquellos en los que la fecha es mayor que la indicada en el parámetro *d.* La función "sort()" ordena los resultados devueltos por la base de datos. En este caso ordena también en función del parámetro *time* y de forma ascendente. Una vez obtenidos los datos, el servidor web los envía para ser representados en el navegador web del cliente. En la Figura 35 se representa el proceso de interacción con la base de datos.

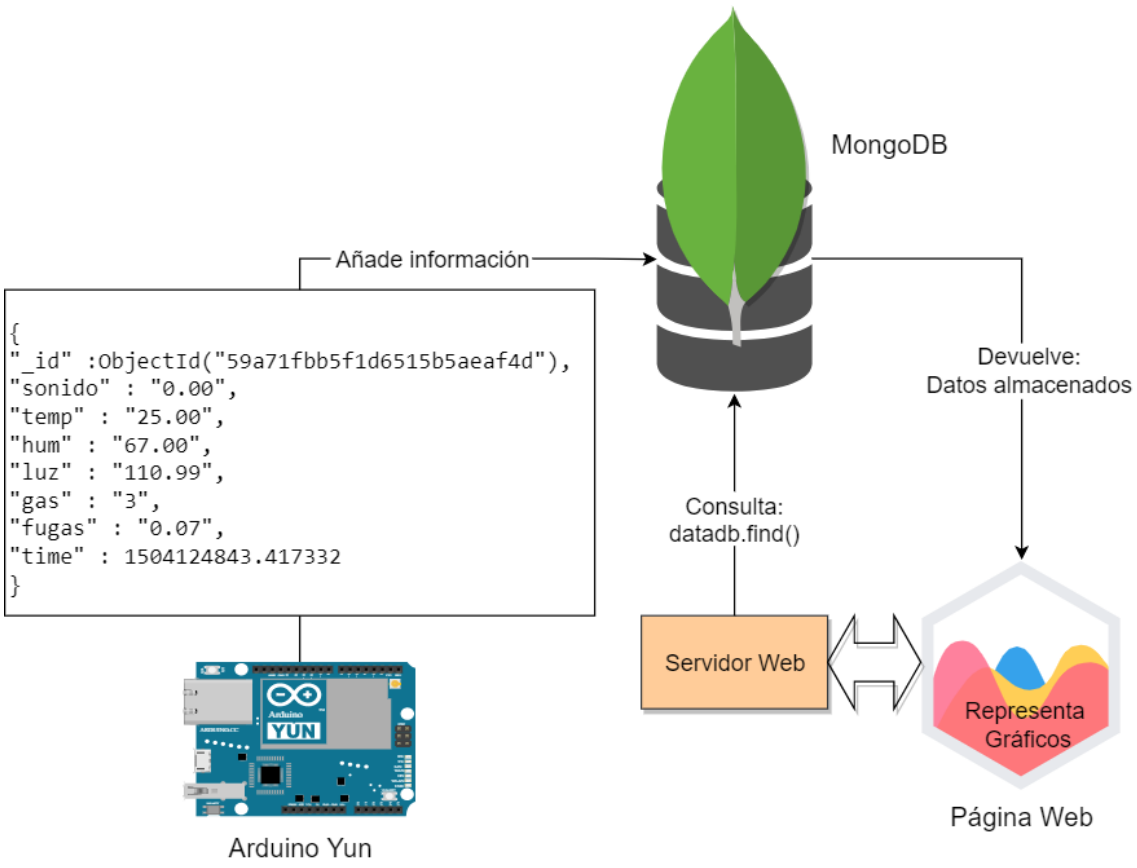


Figura 35 Funcionamiento de la base de datos.

5.2.2. SERVIDOR WEB.

El servidor web tiene como objetivo ofrecer un cuadro de mando personalizado para la visualización de los datos obtenida de los sensores. Se ha basado en el *framework* Flask. En él se han desarrollado diferentes funciones para implementar la funcionalidad deseada. Para crear una aplicación web con Flask se necesitan dos programas. El primero es *run.py* que se encargar de inicializar el servidor y todos sus componentes. El segundo es *views.py*, que gestiona los directorios y permite la navegación a través del servidor web. En este programa de Python se definen todo el árbol de directorios de la aplicación y las respuestas del servidor antes las peticiones de los clientes. En este dispositivo se ha definido como respuesta para el directorio "/" el renderizado de la página *index.html*. De forma que cualquier petición al directorio "/" devolverá la página web *index.html*. También se ha asociado a este directorio, la consulta de la base de datos mediante la función *datosdb.py*. Esta función devuelve un *array*[14] con cada uno de los datos a representar (temperatura, humedad, luminosidad, etc). Estos datos son usados para generar los gráficos embebidos en la página web *index.html*. En la Figura 36 se muestra un flujograma del funcionamiento del servidor web.

[14] Vector de datos

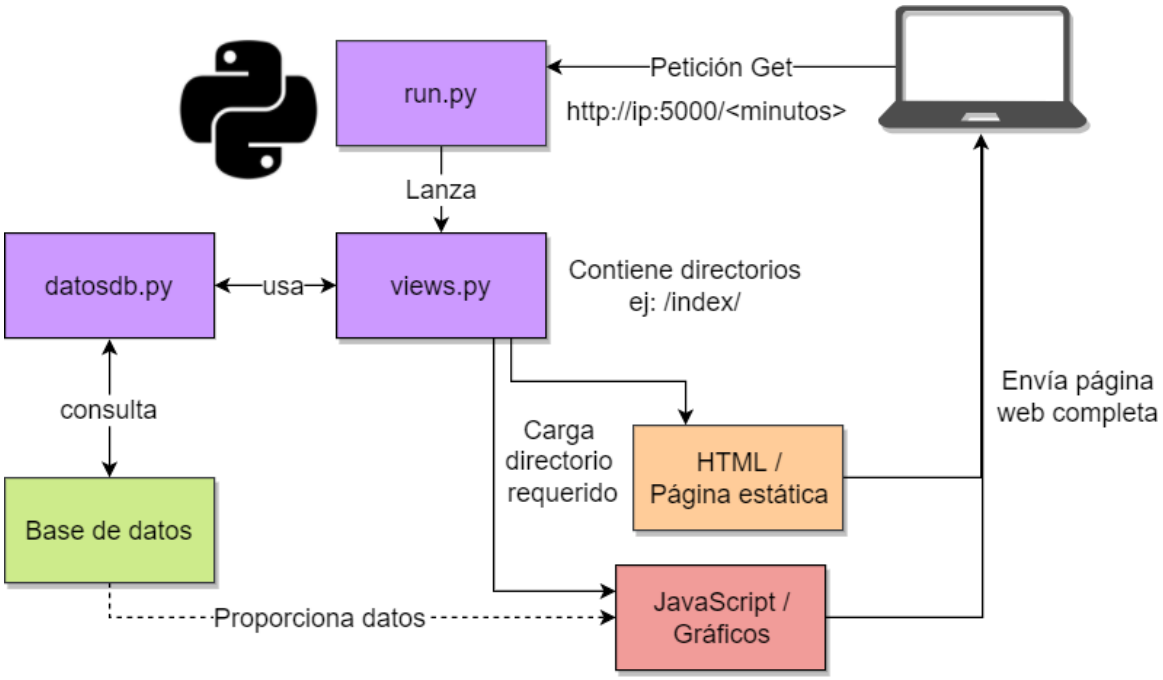


Figura 36 Funcionamiento servidor web.

También se ha añadido la funcionalidad de seleccionar la ventana de tiempo que se desea ver en los gráficos. Para ello, se ha utilizado un parámetro en la URL, de forma que accediendo a la dirección *http://ip-raspberry:5000/<minutos-visualización>* y dando un valor a *minutos-visualización* se puede variar el tiempo representado en las gráficas. Este valor es por defecto 30 minutos y da valor al parámetro "*d*" utilizado en las *queries* a la base de datos (véase apartado 5.2.1).

La página web o *front-end* se ha diseñado para que el usuario pueda ver el estado actual de los sensores y un histórico de los mismos. Esta es una parte importante del proyecto, pues el simple almacenamiento de datos sin las correctas herramientas de visualización es inútil. El *front-end* personalizado aporta un valor extra sobre otras soluciones de medición de variables como podría ser un termómetro. Como se acaba de comentar, es posible variar la escala de tiempo añadiendo los minutos deseados a la URL. También se ha definido el periodo de auto-actualización de los gráficos a 15 segundos.

La página web se ha programado en HTML y JavaScript y dispone de un gráfico específico para cada una de las variables y una pequeña descripción. El aspecto visual de la misma es el mostrado en la Figura 37.

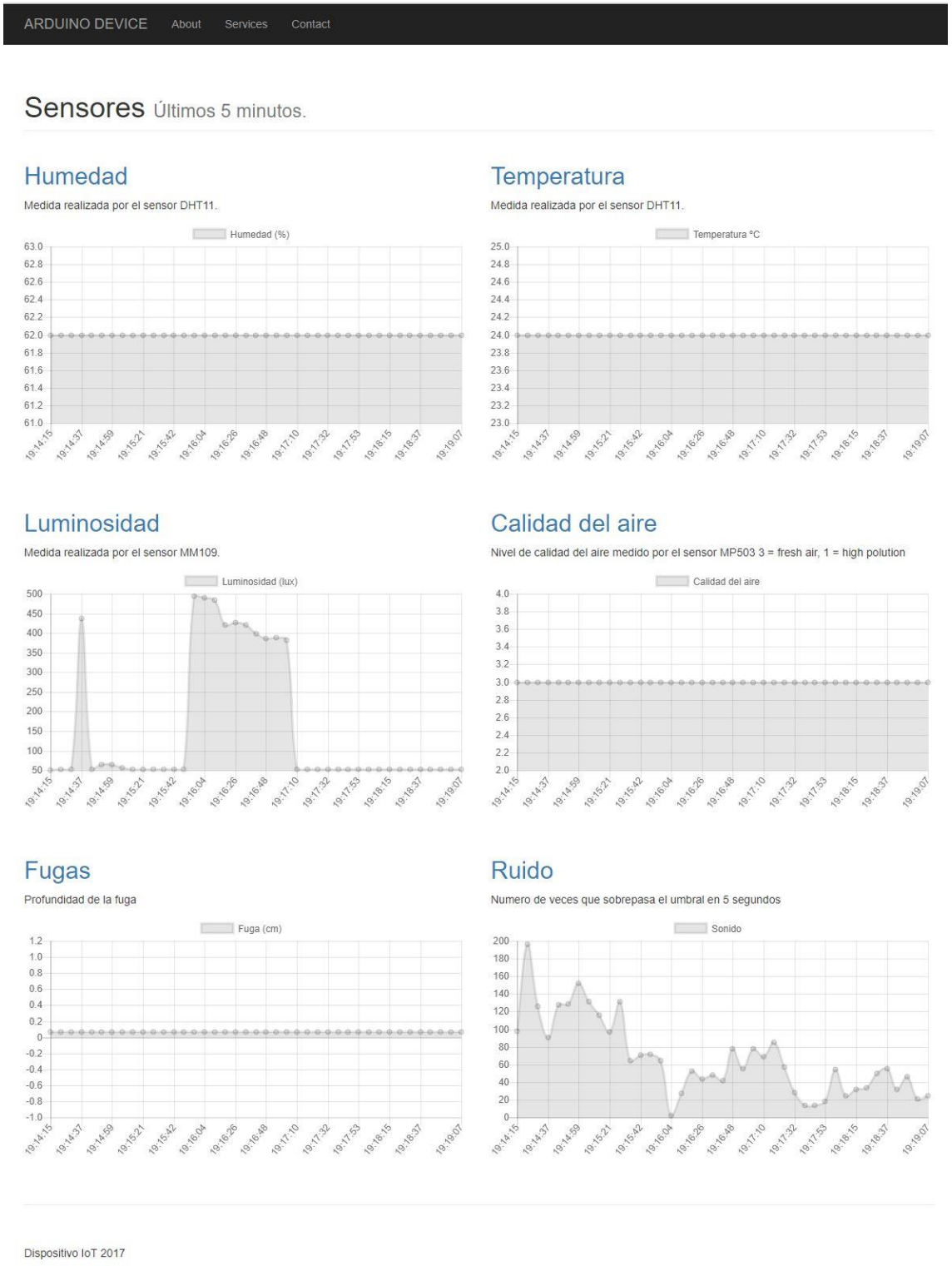


Figura 37 Aspecto de la página web.

Se ha utilizado una plantilla gratuita de Bootstrap para la realización del diseño web, por otro lado, el motor de representación de los gráficos ha sido Chart.js. Flask soporta *templates* interactivos, de forma que soporta el envío de variables desde el *Back-end* al *Front-end*. Este envío de información proporciona a los scripts de Chart.js y la página web los datos que deben representar.

Por último, existe la posibilidad de dar visibilidad al servidor y la página web a todo internet, para poder ser consultado desde cualquier parte. No obstante, el dispositivo está conectado a la red interna de la empresa por lo que cualquier fallo de seguridad podría afectar a toda la red acarreando graves consecuencias y comprometiendo los activos de la empresa. Por ello, se ha restringido a la red local, pudiendo acceder al cuadro de mando desde cualquier terminal conectado a esta red. Cumpliendo así las políticas de seguridad de S2 Grupo.

5.2.3. INTEGRACIÓN CON OTROS SERVICIOS.

Uno de los pilares del proyecto en el que se engloba e integra este dispositivo la automatización de tareas. En una primera aproximación la automatización se está llevando a cabo mediante plataformas como IFTTT (*If This Then That).* IFTTT es una plataforma en la nube que permite la integración de multitud de dipositivos y plataformas. Su funcionamiento se basa en la creación de lógicas condicionales simples, es decir, si se produce un evento (por ejemplo, se abre una puerta) se ejecuta la acción asociada (se enciende la luz). Su potencial radica en la interconectividad de dispositivos de una forma sencilla de cara a los usuarios.

Para integrar el dispositivo objeto de este TFG con el resto del entorno e IFTTT se ha decidido usar la plataforma *adafruit.io*. Se trata de una plataforma que permite la monitorización de sensores en la nube en tiempo real. Para monitorizar un sensor se debe añadir un *feed* de entrada a la plataforma. Para enviar la información de los sensores existen multitud posibilidades, pero se ha utilizado la librería de Python. De esta forma añadiendo otra nueva función, que haga uso de esta librería, en el *script* ejecutado en la placa Arduino Yun se consigue la integración del servicio *adafruit.io* en el dispositivo IoT. En la Figura 38 se puede ver los *dashboards* creados que permiten visualizar que la información llega al dispositivo en tiempo real.

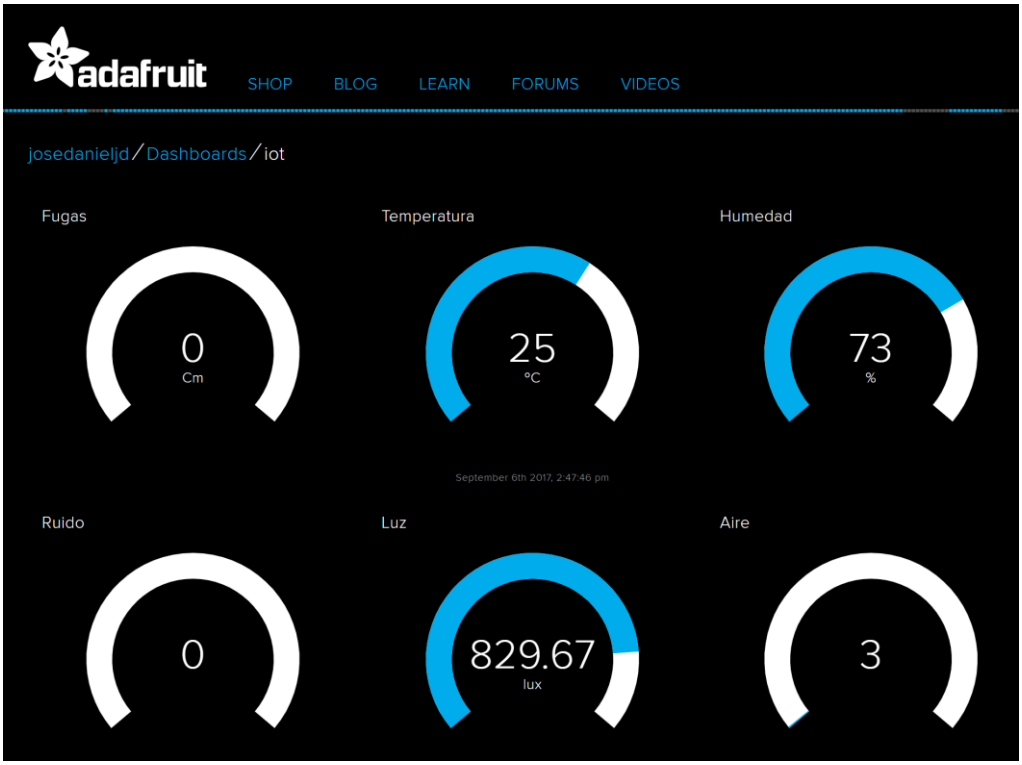


Figura 38 Dashboards creados en io.adafruit.com

La plataforma IFTTT permite la integración con *adafruit.io* permitiendo usar los valores de los sensores del dispositivo como eventos disparadores. De esta forma podemos, por ejemplo, crear un *log* en una hoja de cálculo de Google (Figura 39) cuando los valores excedan los límites establecidos, encender o pagar luces en función de la luminosidad detectada, o un sinfín de posibilidades. Como el resto del entorno de la casa conectada no está finalizado, se han realizado algunas pruebas de concepto del potencial de esta tecnología.

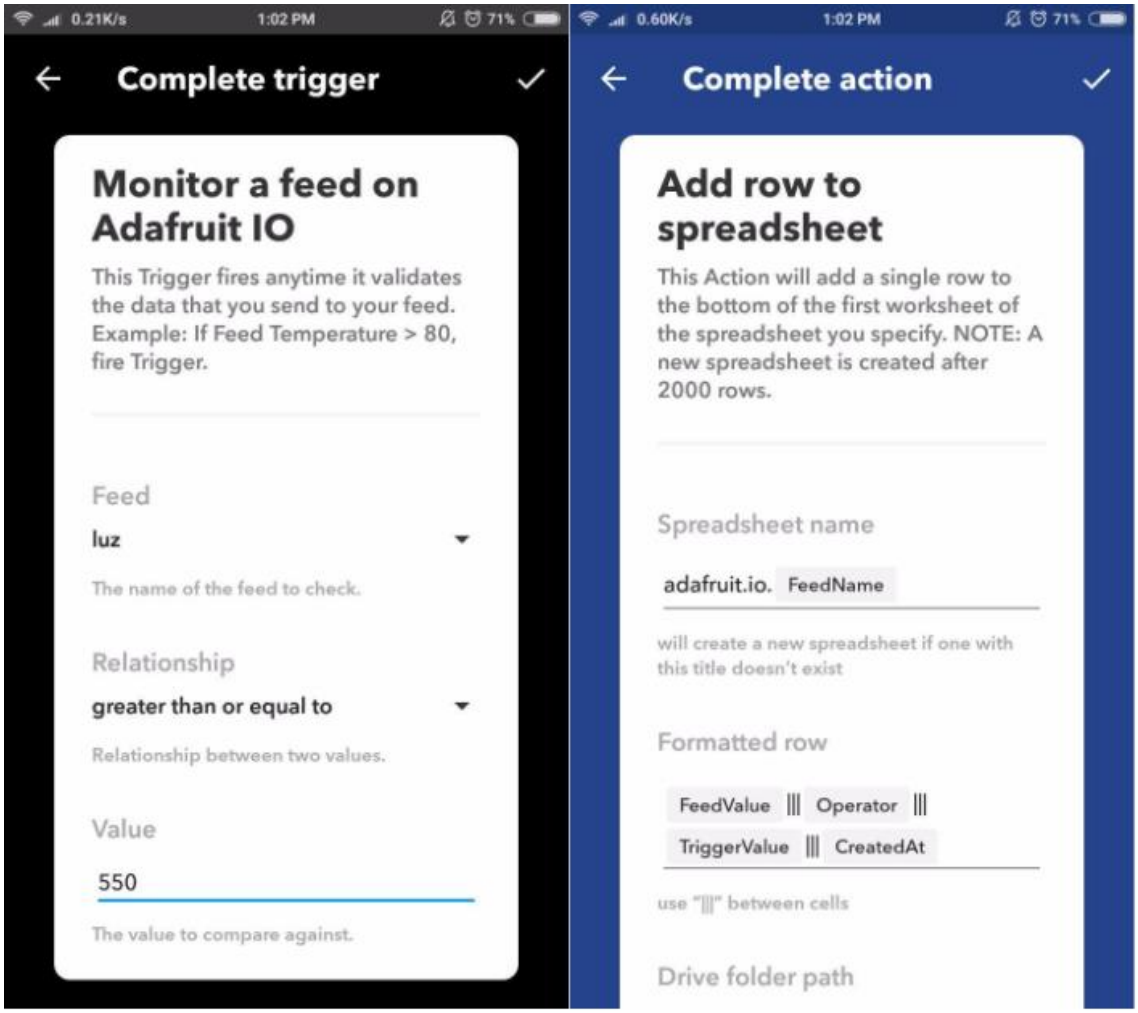


Figura 39 Configuración de disparadores y actuadores en IFTTT

CAPÍTULO 6: RESULTADOS Y MEJORAS FUTURAS

6.1. RESULTADOS

Tras el diseño, la implementación y la integración de los distintos componentes, se ha obtenido un dispositivo funcional. Se han realizado diferentes pruebas de funcionamiento: individualmente a cada uno de los sensores y de conjunto. También se han realizado diversas pruebas de estabilidad, en las que se ha puesto a funcionar el dispositivo durante 3, 5 y 12 horas ininterrumpidas. El resultado de estas pruebas ha sido positivo, no presentando problemas. En ninguna de ellas.

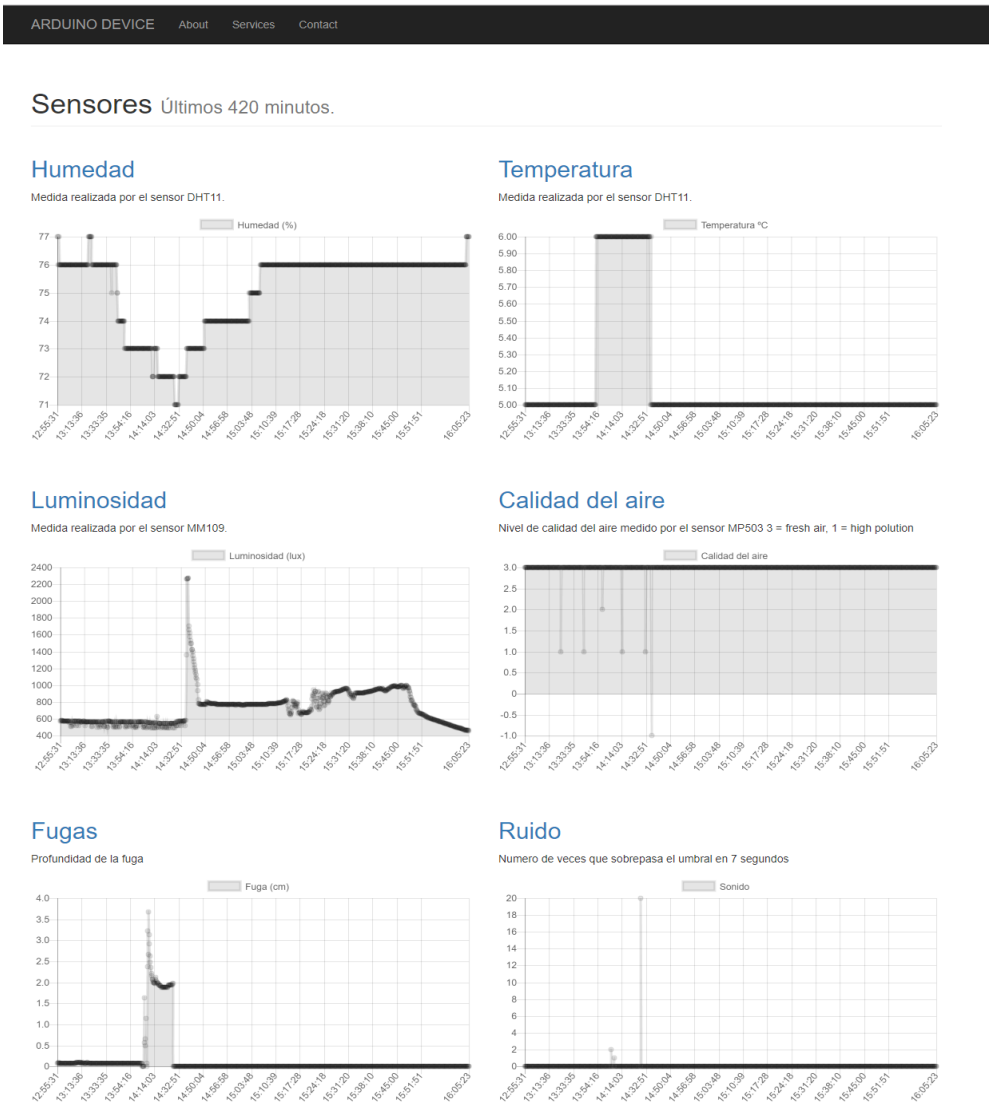


Figura 40 Resultados tras 5 horas funcionando.

El ensamblaje de los diferentes componentes tiene el siguiente aspecto aspecto:

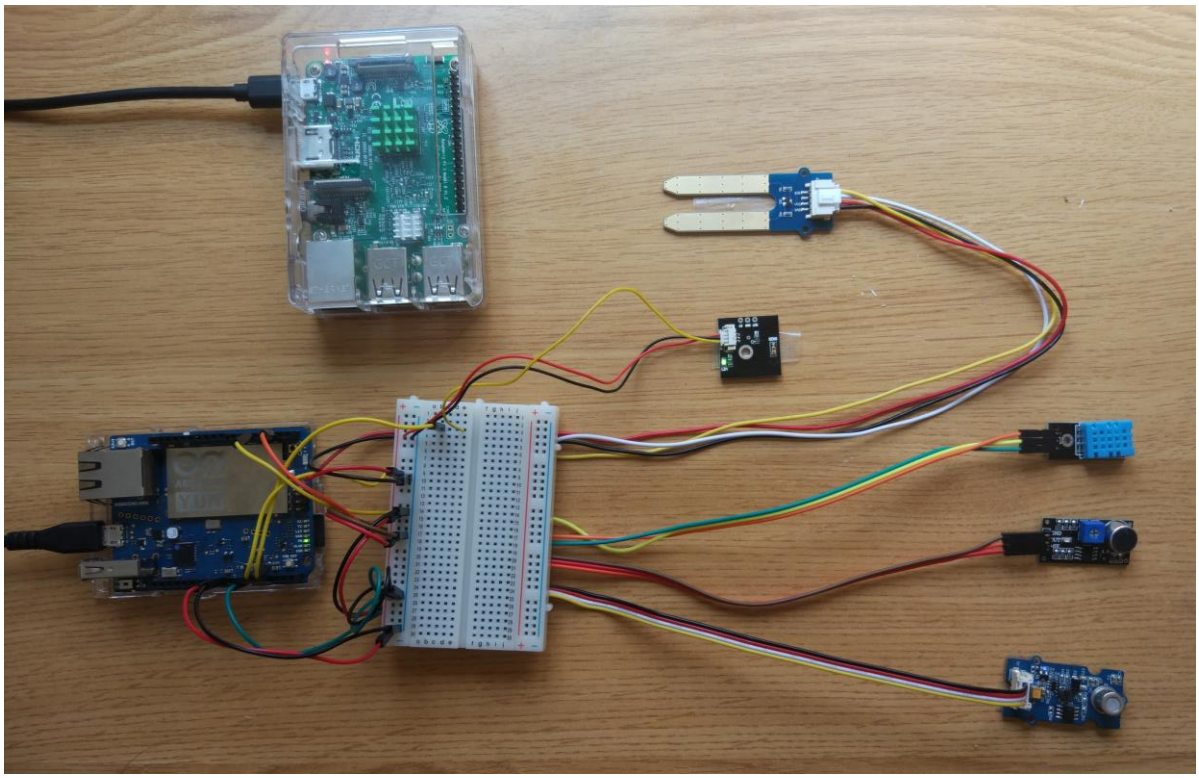


Figura 41 Montaje de los componentes. En orden descendente los sensores son de fugas, de luminosidad, de temperatura y humedad, de ruido y de calidad del aire.

Las distintas pruebas de concepto de integración con Adafruit e IFTTT también han resultado positivas, los datos son correctamente recibidos por parte de Adafruit *(veasé* Figura 42*)* y *logs* deseados (véase Figura 43).

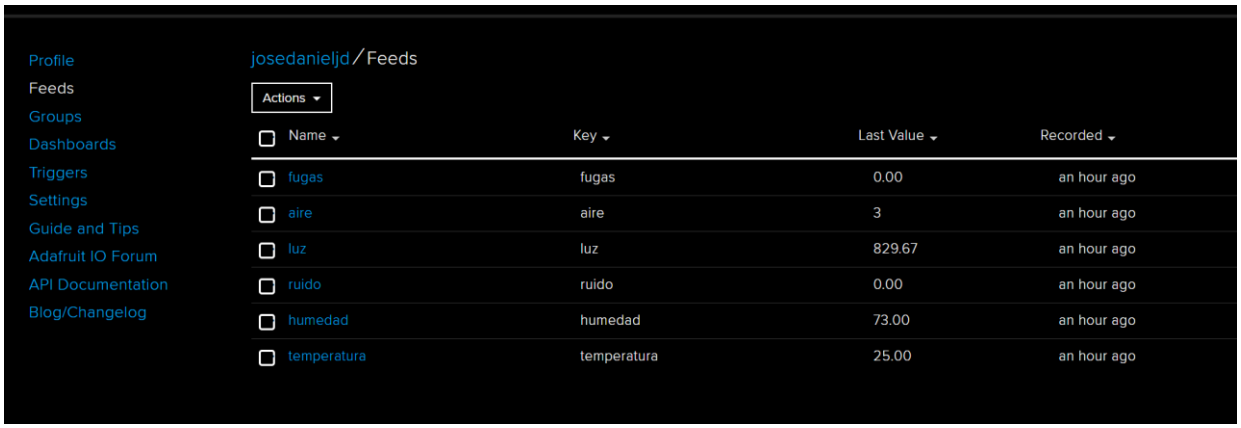
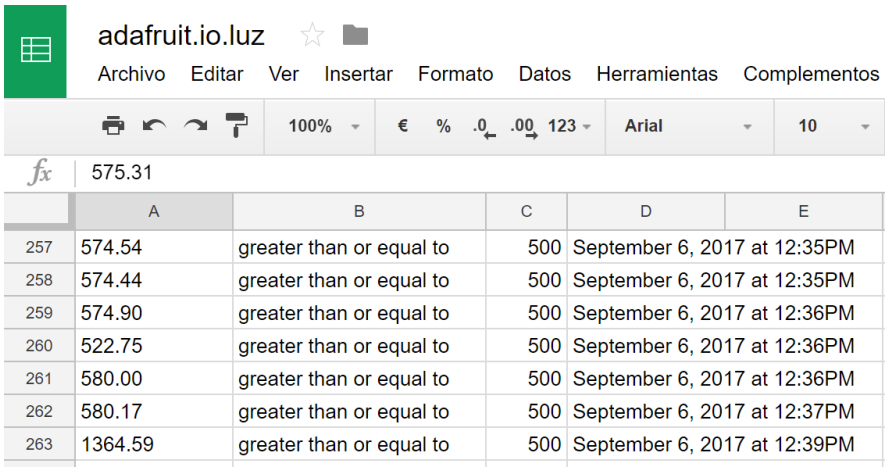


Figura 42 "Feeds" de entrada de Adafruit.io



adafruit.io.luz

Archivo Editar Ver Insertar Formato Datos Herramientas Complementos

100% € % .0 .00 123 Arial 10

fx 575.31

	A	B	C	D	E
257	574.54	greater than or equal to	500	September 6, 2017 at 12:35PM	
258	574.44	greater than or equal to	500	September 6, 2017 at 12:35PM	
259	574.90	greater than or equal to	500	September 6, 2017 at 12:36PM	
260	522.75	greater than or equal to	500	September 6, 2017 at 12:36PM	
261	580.00	greater than or equal to	500	September 6, 2017 at 12:36PM	
262	580.17	greater than or equal to	500	September 6, 2017 at 12:37PM	
263	1364.59	greater than or equal to	500	September 6, 2017 at 12:39PM	

Figura 43 Log creado a partir de IFTTT

6.2. MEJORAS FUTURAS

Como mejoras futuras se plantean:

- Migrar la web de http:// a https:// con certificados válidos para aumentar el nivel de seguridad de la página.
- Incluir algún método de autentificación de usuarios, como por ejemplo un formulario de usuario y contraseña.
- Soldar los componentes a una placa de prototipado, de forma que sea posible una reducción de tamaño y un aumento de la resistencia
- Diseño e impresión de una carcasa en 3D, para mejorar el aspecto estético del resultado final.
- Integración con más servicios relacionados con la casa conectada.
    - Desarrollar compatibilidad con algún asistente de voz.
    - Estudio de plataformas abiertas de integración.
    - Creación de nuevas rutinas de IFTTT que relacionen el dispositivo con el resto del entorno.
- Aumentar el número de sensores/nodos de medición. De forma que se puedan monitorizar diferentes estancias o nuevas variables como la cantidad de polvo.
- Añadir al conjunto actuadores, de forma que sea posible la interacción en ambas direcciones. (Dispositivo=>Cliente y Cliente=>Dispositivo)

CAPÍTULO 7: CONCLUSIONES

El proyecto ha sido finalizado alcanzando los objetivos propuestos inicialmente. Se ha desarrollado un sistema de monitorización de variables ambientales en plataformas open source, entre estas variables se incluyen la contaminación atmosférica y acústica, iluminación, temperatura y humedad. Para ello, se ha utilizado un Arduino Yun que permite la comunicación con los distintos sensores y los envía a través de WiFi, el servidor los recoge y los almacena en una base de datos local (MongoDB). Como interfaz al usuario, se ha diseñado una plataforma de visualización mediante una página web en HTML que se puede acceder desde cualquier terminal de la intranet de la empresa para consultar la información en tiempo real. El sistema diseñado es totalmente escalable añadiendo otros nodos donde considere apropiado. El dispositivo desarrollado forma parte del entorno de prueba y demostraciones de la empresa S2 Grupo, y más adelante las soluciones obtenidas del entorno podrían incluirse en su catálogo de productos.

Además, se ha trabajado en la integración de del mismo con el resto del entorno de pruebas mediante plataformas como IFTTT, aumentando de esta forma el valor útil del dispositivo en la empresa.

En vista al desarrollo del proyecto, se puede concluir que es posible la creación de dispositivos IoT open source, no obstante, el dispositivo final es factible de ser comercializado al gran público en gran escala, por diversos motivos. El primero es la necesidad de tener conocimientos básicos de electrónica y programación. El segundo es el aspecto estético y resistencia del resultado final. Si se desease la comercialización del mismo deberían plantearse diferentes mejoras como la creación de una PCB en la que se incluyan todos los componentes necesarios en lugar de usar plataformas de desarrollo como es Arduino, o los sensores utilizados. Ello unido a un buen diseño exterior mejoraría sin dudas su aceptación y adopción entre los clientes potenciales. Sin embargo, el objetivo final no era la comercialización ni fabricación a gran escala del dispositivo, si no desarrollar un dispositivo específico y completamente adaptado a las necesidades un cliente. Es en esta última situación donde esta plataforma ofrece todo su potencial, siendo además de económica, completamente adaptable a las necesidades específicas del cliente, y eso es algo que no puede ofrecer ninguna solución comercial de fabricación a gran escala.

En vistas de futuro, la casa conectada dispondrá tanto de soluciones comerciales fabricadas en masa, como soluciones específicas para necesidades particulares. Y es en estas necesidades donde reside el futuro de este tipo de dispositivos open source (además de el desarrollo de prototipos).

CAPÍTULO 8: BIBLIOGRAFÍA

8.1 REFERENCIAS.

[1] Connected Home - Gartner IT Glossary, Gartner IT Glossary, [en línea] 2017, Disponible en: https://www.gartner.com/it-glossary/connected-home [Consulta: 10 de julio de 2017].

[2]. HP News - HP Study Finds Alarming Vulnerabilities with Internet of Things (IoT) Home Security Systems, Www8.hp.com, [en línea] 2017, Disponible en: http://www8.hp.com/us/en/hp-news/press-release.html?id=1909050 [Consulta: 19 de julio de 2017]..

[3] What is open source?, Opensource.com, [en línea] 2017, Disponible en: https://opensource.com/resources/what-open-source [Consulta: 17 de julio de 2017].

[4] AWS IoT – Amazon Web Services, Amazon Web Services, Inc., [en línea] 2017, Disponible en: https://aws.amazon.com/es/iot/ [Consulta: 19 de julio de 2017].

[5] Advanced Micro Controls Inc :: What is a PLC?, Amci.com, [en línea] 2017, Disponible en: https://www.amci.com/industrial-automation-resources/plc-automation-tutorials/what-plc/[Consulta: 3 de agosto de 2017].

[6]. Industria 4.0, Es.wikipedia.org, [en línea] Disponible en: https://es.wikipedia.org/wiki/Industria_4.0 [Consulta: 4 de septiembre de 2017].

[7] Informe preliminar Transformación digital de la Industria Española, [en línea] [ebook], Ed. Ministerio de Industria, Energía y Turismo, Disponible en: http://www6.mityc.es/IndustriaConectada40/informe-industria-conectada40.pdf [Consulta: 6 de agosto de 2017].

[8] Gartner Says 8.4 Billion Connected, Gartner.com, [en línea] 2017, Disponible en: https://www.gartner.com/newsroom/id/3598917 [Consulta: 6 de agosto de 2017].

[9] FRANCO MARTÍNEZ, E.: Instrumentación. 06 Tipos de Sensores, [en línea] [ebook], ed. , Disponible en: http://www.eafranco.com/docencia/instrumentacion/files/06/Diapositivas06.pdf [Consulta: 13 de agosto de 2017].

[10] Mirai (malware), Es.wikipedia.org, [en línea] Disponible en: https://es.wikipedia.org/wiki/Mirai_(malware) [Consulta: 7 de agosto de 2017].

[11] TEAM, S.: Grove - Air Quality Sensor v1.3 - Seeed Wiki, Wiki.seeed.cc, [en línea], Disponible en: 1http://wiki.seeed.cc/Grove-Air_Quality_Sensor_v1.3/ [Consulta: 4 de septiembre de 2017].

[12] SENSOR, G.: Grove - Air Quality Sensor :: Solarbotics, Solarbotics.com, [en línea], Disponible en: https://solarbotics.com/product/29054/ [Consulta: 4 de septiembre de 2017].

[13] Bluetooth Technology, Bluetooth.com, [en línea] Disponible en: https://www.bluetooth.com/ [Consulta: 11 de agosto de 2017].

[14] Leer un pulsador con Arduino con interrupciones y debounce, Luis Llamas, [en línea] Disponible en: https://www.luisllamas.es/debounce-interrupciones-arduino/ [Consulta: 25 de agosto de 2017].

[15] Welcome | Flask (A Python Microframework), Flask.pocoo.org, [en línea] 2017, Disponible en: http://flask.pocoo.org/ [Consulta: 22 de julio de 2017].

[16] The mongo Shell — MongoDB Manual 3.4, Docs.mongodb.com, [en línea] 2017, Disponible en: https://docs.mongodb.com/manual/mongo/ [Consulta: 24 de agosto de 2017].

[17] Two Column Portfolio - Free Bootstrap Template, Start Bootstrap, [en línea] 2017, Disponible en: https://startbootstrap.com/template-overviews/2-col-portfolio/ [Consulta: 12 de julio de 2017].

[18] Chart.js | Open source HTML5 Charts for your website, Chartjs.org, [en línea] 2017, Disponible en: http://www.chartjs.org/ [Consulta: 5 de septiembre de 2017].

8.2 TABLA DE FIGURAS.

DOCUMENTO II

PRESUPUESTO

PRESUPUESTO GENERAL

1.1. INTRODUCCIÓN

En este capítulo se procede al cálculo del presupuesto aproximado necesario para realización del proyecto. El presupuesto se ha subdivido en tres cuadros de obra: mano de obra, materiales y unidades de obra.

Se ha calculado la amortización del ordenador portátil utilizado un periodo de 5 años. Considerando que un año tiene 220 días hábiles de 8 horas.

1.2. DESGLOSE PRESUPUESTO.

Mano de obra

Ref	*Descripción*	*l*	*Precio*
MO.ING	Graduado GITI		20 €/h

Materiales

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio en Euros*	*Importe (€)*
MT.PORT	Surface Pro 4	u	1	1.400,00	1.400,00

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio en Euros*	*Importe (€)*
MT.ARYU	Arduino Yun	u	1	77,14	77,14
MT.TEMP	DHT11	u	1	4,10	4,10
MT.MIC	Sensor Arduino de sonido KY-036	u	1	5,89	5,89
MT.LUZ	Sensor de luz Velleman MM109	u	1	5,04	5,04
MT.AIR	Grove-air Quality sensor V1.3	u	1	8,39	8,39
MT.FU	Sensor de humedad del suelo	u	1	7,97	7,97
MT.COM	Kit básico de componentes Arduino	u	1	21,97	21,97
MT.JUM	Jumpers conexión protoboard machos a hembras	u	1	5,90	5,90
MT.RASP	Raspberry Pi 3 Modelo B	u	1	36,72	36,72
MT.CARG	Cargador Micro USB 5V + Caja + Disipadores Raspberry	u	1	12,99	12,99
MT.SD	Tarjeta Micro SD Clase 10 de 32 GB	u	1	18,79	18,79
				Total	*186,11 €*

Unidades de obra

CAPITULO 1: MICROCONTROLADOR

Código 1.01 **Diseño programa controlador**

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio (€/h)*	*Importe (€)*
MO.ING	Graduado GITI	h	20	20,00	400,00
MT.PORT	Surface Pro 4	h	20	0,16	3,18
	Costes directos complementario	%	2%		8,06
				Subtotal	*411,25 €*

Código 1.02 **Diseño comunicación**

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio (€/h)*	*Importe (€)*
MO. ING	Graduado GITI	h	20	20,00	400,00
MT.PORT	Surface Pro 4	h	20	0,16	3,18
	Costes directos complementarios	%	2%		8,06
				Subtotal	*411,25 €*

Total Microcontrolador	*822,50 €*

CAPITULO 2: SERVIDOR

Código 2.01 **Programación servidor web**

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio(€/h)*	*Importe*
MO.ING	Graduado GITI	h	45	20,00	900,00
MT.PORT	Surface Pro 4	h	45	0,16	7,16
	Costes directos complementarios	%	2%		18,14
				Subtotal	*925,30€*

Código 2.02 **Programación Db**

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio(€/h)*	*Importe*
MO.ING	Graduado GITI	h	20	20,00	400,00
MT.PORT	Surface Pro 4	h	20	0,00	0,00
	Costes directos complementarios	%	2%		8,00
				Subtotal	*408,00 €*

Código 2.03 **Diseño y programación página web**

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio(€/h)*	*Importe*
MO. ING	Graduado GITI	h	45	20,00	900,00
MT.PORT	Surface Pro 4	h	45	0,16	7,16
	Costes directos complementarios	%	2%		18,14
				Subtotal	*925,30 €*

Código 2.04 **Implementación con otras APIs /Servicios**

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio(€/h)*	*Importe*
MO. ING	Graduado GITI	h	20	20,00	400,00
MT.PORT	Surface Pro 4	h	20	0,16	3,18
	Costes directos complementarios	%	2%		8,06
				Subtotal	*411,25 €*

Total Servidor	*2.669,85 €*

CAPÍTULO 3: MONTAJE

Código 3.01 **Diseño y elección de componentes**

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio*	*Importe*
MO. ING	Graduado GITI	h	20	20,00	400,00 €
MT.PORT	Surface Pro 4	h	20	0,16 €	3,18 €
	Costes directos complementarios	%	2%		8,16 €
				Subtotal	*411,25 €*

Código 3.02 **Montaje componentes y puesta a punto**

Ref	*Descripción*	*Ud.*	*Cantidad*	*Precio*	*Importe*
MO. ING	Graduado GITI	h	25	20,00 €	500,00 €
MT.PORT	Surface Pro 4	h	15	0,16 €	2,39
MT.ARYU	Arduino Yun	u	1	77,14 €	77,14 €
MT.TEMP	DHT11	u	1	4,10 €	4,10 €
MT.MIC	Sensor Arduino de sonido KY-036	u	1	5,89 €	5,89 €
MT.LUZ	Sensor de luz Velleman MM109	u	1	5,04 €	5,04 €
MT.AIR	Grove-air Quality sensor V1.3	u	1	8,39 €	8,39 €
MT.FU	Sensor de humedad del suelo	u	1	7,97 €	7,97 €
MT.COM	Kit básico de componentes Arduino	u	1	21,97 €	21,97 €
MT.JUM	Jumpers conexión protoboard machos a hembras	u	1	5,90 €	5,90 €
MT.RASP	Raspberry Pi 3 Modelo B	u	1	36,72 €	36,72 €
MT.CARG	Cargador Micro USB 5V + Caja + Disipadores Raspberry	u	1	12,99 €	12,99 €
MT.SD	Tarjeta Micro SD Clase 10 de 32 GB	u	1	18,79 €	18,79 €
	Costes directos complementarios	%	2%		14,15 €
				Subtotal	*721,43 €*

Total montaje	*1.132,68 €*

Presupuesto total.

Microcontrolador	822,50 €
Servidor	2.669,85 €
Montaje	1.132,68 €
PEM	**4.628,27 €**
Gastos generales (13%)	601,67 €
Beneficio industrial (6%)	277,70 €
	5.507,64 €
IVA (21%)	1.156,60 €
Total presupuesto	**6.664,24 €**

El presupuesto total del dispositivo IoT asciende a 6.664,24 €.