



Benemérita Universidad Autónoma de Puebla

FACULTAD DE CIENCIAS DE LA ELECTRÓNICA

**Sistema de monitoreo ambiental basado en UAV,
desde una perspectiva del Internet de las cosas.**

T E S I S

PARA OBTENER EL GRADO DE:

**LICENCIATURA EN INGENIERÍA
MECATRÓNICA**

PRESENTA:

Jesús Alberto Islas Fuentes

ASESORES:

**Dr. José Fermi Guerrero Castellanos
Mtro. Nicolás Quiroz Hernández**

Puebla, Puebla.

Diciembre 2019

Agradecimientos

A mis padres por todo su cariño y apoyo incondicional durante todos estos años; por siempre creer en mí y alentarme a seguir mis sueños cada día.

A mis asesores:

Dr. Fermi Guerrero Castellanos, por el apoyo y la oportunidad de desarrollar este trabajo bajo su tutoría, también por cada uno de los consejos brindados y el conocimiento transmitido que hicieron posible la conclusión de este trabajo de forma satisfactoria.

M.C. Nicolás Quiroz Hernández, por la motivación brindada que me impulso a profundizar en el desarrollo de software, también por sus consejos y asesoría para el desarrollo de este trabajo.

A mis profesores y a la Facultad de Ciencias de la Electrónica la Benemérita Universidad Autónoma de Puebla, por todo el conocimiento y apoyo brindado que me permitió formarme como ingeniero.

Al Ing. Francisco Solano Huitzil Responsable de Calidad del Aire y Cambio Climático de la Secretaría de Desarrollo Rural, Sustentabilidad y Ordenamiento Territorial del Estado de Puebla, por el apoyo y las facilidades brindadas para la realización de este trabajo.

A la M.C Ana María Rodríguez Domínguez, a la M.C. Selene Edith Maya Rueda y a la Dra. Nallely Téllez Méndez por aceptar formar parte del jurado, por la retroalimentación y por los consejos brindados para la mejora de este trabajo de tesis.

Dedicatorias

A mi madre Marina Fuentes Bailón por todo cariño y el apoyo, la educación brindada a lo largo de mi vida que me han llevado a ser la persona que soy hoy en día, por siempre creer en mi sin importar las circunstancias e impulsarme a seguir superándome.

A mi padre Gustavo Islas Morales, por todo el cariño y apoyo recibido a lo largo de mi vida, así como inculcarme desde pequeño el amor al conocimiento y ser un modelo a seguir tanto personal como profesionalmente.

A mi hermana Anabel Islas Fuentes, por estar siempre presente y apoyarme en cualquier circunstancia.

A mis amigos y familiares, por su apoyo y motivación para seguir superándome día con día.

INDICE

LISTADO DE FIGURAS	5
RESUMEN.....	8
CAPÍTULO 1. INTRODUCCIÓN	9
1.1 OBJETIVO GENERAL	9
1.2 OBJETIVOS ESPECÍFICOS	9
1.3 METAS	9
1.4 JUSTIFICACIÓN	10
1.5 DESCRIPCIÓN	10
CAPÍTULO 2. ANTECEDENTES.....	12
2.1 CONTAMINACIÓN DEL AIRE	12
2.2 VARIABLES METEOROLÓGICAS	16
2.3 MONITORIZACIÓN AMBIENTAL.....	18
2.4 MONITORIZACIÓN AMBIENTAL DESDE UNA PERSPECTIVA DEL INTERNET DE LAS COSAS	22
CAPÍTULO 3. DESCRIPCIÓN DEL SOFTWARE	27
3.1 PROTOCOLOS DE INTERNET	27
3.1.1 Protocolo de Internet	27
3.1.2 Protocolo HTTP	29
3.1.3 Protocolo MQTT	30
3.2 APLICACIÓN WEB	32
3.2.1 Lenguajes comunes en la programación de aplicaciones web	34
3.2.2 Elección de lenguaje de programación para el desarrollo de la plataforma web de monitorización	41
3.2.2 DJANGO (Framework para desarrollo de aplicaciones web)	41
3.2.3 MVC (Modelo Vista Controlador)	42
3.2.4 API REST.....	44
3.2.5 Servidor Web.....	46
3.2.6 Base de datos	47
3.2.7 Interfaz Gráfica (GUI)	48
3.3 DESARROLLO DEL SOFTWARE	54
3.3.1 Diagrama general del sistema de monitorización.....	54
3.3.2 Desarrollo de la plataforma de monitorización.....	55
3.3.3 API REST.....	61
3.3.4 API REST para agregar datos de vuelo.....	64
3.4 PANEL DE MONITORIZACIÓN EN TIEMPO REAL.....	67
3.4.1 Módulo Sensores	68
3.4.2 Módulo Monitor	69
CAPÍTULO 4. DESCRIPCIÓN DEL HARDWARE.....	70
4.1 SELECCIÓN DE MÓDULOS Y SENSORES.....	70
4.1.1 Adafruit FONA - Mini Cellular GSM	70
4.1.2 Microcontrolador Atmega328P	71
4.1.3 Gy-GPS6MV2	72
4.1.4 BMP180	73
4.1.5 SI1145	73
4.1.6 MQ7.....	74
4.1.7 Baterías	74
4.2 DIAGRAMA GENERAL DEL HARDWARE.....	75
4.3 FOTOS DEL HARDWARE DE MONITORIZACIÓN MONTADO SOBRE EL UAV	77
4.4 INTEGRACIÓN CON LA PLATAFORMA DE MONITORIZACIÓN	79
CAPÍTULO 5. PRUEBAS	81
5.1 PRUEBAS EN LA ESTACIÓN DE MONITOREO AMBIENTAL “NINFAS”	81
5.1 CRONOGRAMA	82
5.2 FOTOS	83
5.3 RESULTADOS	84
5.2 PRUEBA DE VUELO SOBRE UAV	101
CAPÍTULO 6. CONCLUSIONES	107
6.1 TRABAJO A FUTURO.....	110
APÉNDICES.....	111
APÉNDICE A: FUNCIONAMIENTO DE LA PLATAFORMA DE MONITORIZACIÓN WEB.	111
APÉNDICE B: GLOSARIO DE TÉRMINOS.	117
BIBLIOGRAFÍA.....	119

Listado de Figuras

Figura 1 Rango de tamaño de partículas aéreas, se muestran las más relacionadas con la salud como partículas ultra finas, PM2.5, PM10 y el tamaño tipo de otros contaminantes comunes	13
Figura 2 Impacto de fuentes de contaminación estacionaras y móviles en la calidad de aire en una zona urbana [2]	15
Figura 3 Visualización de los diferentes tipos de fuentes de contaminación representados en un inventario de emisiones.....	16
Figura 4 Escala del Índice de Radiación UV	18
Figura 5 La zona metropolitana de la Ciudad de México cuenta con 30 estaciones de monitorización. ...	20
Figura 6 La zona metropolitana de Guadalajara cuenta con 10 estaciones de monitorización.	21
Figura 7 La zona metropolitana de Monterrey cuenta con 8 estaciones de monitorización.....	21
Figura 8 La zona metropolitana de Puebla cuenta con 4 estaciones de monitorización.....	22
Figura 9 El internet de las cosas es la sinergia de diferentes disciplinas tecnológicas	23
Figura 10 Campos dominantes y sus escenarios más relevantes de posibles aplicaciones del IOT	25
Figura 12 Diagrama general del funcionamiento del protocolo de internet	27
Figura 13 Descripción de la suite TCP/IP y sus modelos.	28
Figura 14 Patrón Publicar/Suscribir MQTT.....	31
Figura 15 Arquitectura de una aplicación web LAMP (Linux, Apache, Mysql, PHP)	35
Figura 16 JAVA - Arquitectura Multinivel.....	38
Figura 17 Arquitectura y flujo de procesos en NODE.JS	41
Figura 18 Patrón Modelo Vista Controlador.....	44
Figura 19 Arquitectura Cliente-Servidor	46
Figura 20 Esquema de los componentes de la interfaz de una aplicación web	49
Figura 21 Cumplimiento de los estándares de la W3C por parte de los diferentes navegadores del mercado [30]	50
Figura 22 Estructura básica de un documento HTML	51
Figura 23 CSS Box Model	51
Figura 24 Estructura de una declaración CSS.....	52
Figura 25 Diagrama general del sistema de monitorización.....	54
Figura 26 Diagrama de módulos y componentes de la plataforma de monitorización.....	56
Figura 27 Diagrama entidad relación de la base de datos, parte 1	65
Figura 28 Diagrama entidad relación de la base de datos, parte 2	66
Figura 29 Pantalla - Ver Dispositivos de monitoreo disponibles.....	67
Figura 30 Pantalla - Interfaz Node-Red, diagrama de aplicación	68
Figura 31 Pantalla - Configuración de módulo Sensor	68
Figura 32 Pantalla - Panel de Monitorización en tiempo real.....	69
Figura 43 Módulo Adafruit FONA - Mini Cellular GSM	71
Figura 44 Microcontrolador Atmega328P	71
Figura 45 Especificaciones técnicas Microcontrolador Atmega328P	72
Figura 46 Módulo GPS Gy-GPS6MV2	72
Figura 47 Módulo BMP180	73
Figura 48 Módulo SI1145	73
Figura 49 Módulo MQ7	74
Figura 50 Diagrama electrónico del Hardware de Monitorización.....	75

Listado de Figuras

Figura 51 Diseño del PCB	76
Figura 52 Foto 1 - Hardware de Monitorización sobre UAV	77
Figura 53 Foto 2 - Hardware de Monitorización sobre UAV	77
Figura 54 Foto 3 - Hardware de Monitorización sobre UAV	78
Figura 55 Foto 4 - Hardware de Monitorización sobre UAV	78
Figura 56 Diagrama de integración de la plataforma de monitorización.....	79
Figura 57 Foto 5 - Estación de Monitoreo Atmosférico "Ninfas"	81
Figura 58 Foto 6 - Estación de Monitoreo Atmosférico "Ninfas"	82
Figura 59 Cronograma de monitoreo "Estación Ninfas"	82
Figura 60 Foto 7 - Monitoreo en Estación "Ninfas"	83
Figura 61 Foto 8 - Monitoreo en Estación "Ninfas"	83
Figura 62 Grafica - Medición de temperatura en el 24 de septiembre de 2018.....	84
Figura 63 Grafica - Medición de Índice de Radiación UV el 24 de septiembre de 2018.....	84
Figura 64 Grafica Medición de Presión atmosférica el 24 de septiembre de 2018	85
Figura 65 Grafica Medición de Monóxido de carbono el 24 de septiembre de 2018.....	85
Figura 66 Tabla Comparación de datos con Estación "Ninfas"	86
Figura 67 Tabla Mediciones del día 24/09/18	87
Figura 68 Grafica Temperatura (°C): 24/09/18.....	87
Figura 69 Grafica Presión Atmosférica (mb): 24/09/18	88
Figura 70 Grafica Índice de radiación UV: 24/09/18	88
Figura 61 Tabla Mediciones del día 25/09/18	89
Figura 62 Grafica Temperatura (°C): 25/09/18.....	89
Figura 73 Grafica Presión Atmosférica (mb): 25/09/18	90
Figura 74 Grafica Índice de radiación UV: 25/09/18	90
Figura 75 Tabla Mediciones del día 26/09/18	91
Figura 76 Grafica Temperatura (°C): 26/09/18.....	91
Figura 77 Grafica Presión Atmosférica (mb): 26/09/18	92
Figura 78 Grafica Índice de radiación UV: 26/09/18	92
Figura 79 Tabla Mediciones del día 27/09/18	93
Figura 80 Grafica Temperatura (°C): 27/09/18.....	93
Figura 81 Grafica Presión Atmosférica (mb): 27/09/18	94
Figura 82 Grafica Índice de radiación UV: 27/09/18	94
Figura 83 Tabla Mediciones del día 28/09/18	95
Figura 84 Grafica Temperatura (°C): 28/09/18.....	95
Figura 85 Grafica Presión Atmosférica (mb): 28/09/18	96
Figura 86 Grafica Índice de radiación UV: 28/09/18	96
Figura 87 Tabla Mediciones del día 01/10/18	97
Figura 88 Grafica Temperatura (°C): 01/10/18.....	97
Figura 89 Grafica Presión Atmosférica (mb): 01/10/18	98
Figura 90 Grafica Índice de radiación UV: 01/10/18	98
Figura 91 Tabla Mediciones del día 02/10/18	99
Figura 92 Grafica Temperatura (°C): 02/10/18.....	99
Figura 93 Grafica Presión Atmosférica (mb): 02/10/18	100
Figura 94 Grafica Índice de radiación UV: 02/10/18	100
Figura 95 Vista del UAV con el sistema de monitorización acoplado	101
Figura 96 Despegue durante la prueba de vuelo	102
Figura 97 UAV y sistema de monitorización durante la prueba de vuelo	103
Figura 98 Vista de Ciudad Universitaria BUAP, durante la prueba de vuelo	103

Figura 99 Grafica de la Altura durante el vuelo de prueba.....	104
Figura 100 Grafica de la presión atmosférica durante la prueba de vuelo.....	104
Figura 101 Grafica de la Temperatura durante la prueba de vuelo.....	105
Figura 102 Detalle de variable de vuelo (Presión atmosférica), se puede observar el mapa de muestras y grafica vs tiempo.....	106
Figura 103 Detalle de variable de vuelo (Temperatura), se puede observar el mapa de muestras y grafica vs tiempo.....	106
Figura 104 Grafica unificada Temperatura °C	108
Figura 105 Grafica unificada Presión atmosférica mb.....	108
Figura 106 Grafica unificada Índice de radiación UV	109

Resumen

En este documento se describe el desarrollo de un hardware de monitorización de variables ambientales que tiene dos características fundamentales, ser lo suficientemente compacto para ser montado sobre un vehículo aéreo no tripulado (UAV) y contar con la capacidad de transmitir los datos ambientales en tiempo real hacia internet de forma inalámbrica a través de la red celular GPRS.

También se describe el diseño y desarrollo del software del sistema de monitorización que consiste en tres partes fundamentales, la comunicación en tiempo real entre el hardware de monitorización de variables ambientales a través e internet, el panel de monitorización en tiempo real de las variables ambientales y la plataforma de monitorización web.

Del lado del software la plataforma de monitorización web es la columna vertebral de este trabajo, consiste en una aplicación web a la cual se puede acceder desde cualquier parte del mundo con conexión a internet, tiene la tarea de facilitar la administración a través de una interfaz intuitiva de los sistemas de monitorización (hardware), gestión de zonas de monitoreo y recorridos, almacenamiento y organización de los datos recolectados por zona, recorrido, tipo de variable, geolocalización, fecha y hora, finalmente esta plataforma cuenta con la característica de ser abierta y flexible ya que permite exportar los datos de monitoreo en formato CSV y además cuenta con un API (interfaz de programación de aplicaciones) que facilita el acceso a los datos de monitoreo desde aplicaciones externas, características fundamentales para su uso en aplicaciones de investigación.

Capítulo 1. Introducción

1.1 Objetivo General

Diseñar e implementar un sistema de monitorización de variables ambientales, que sea ligero, flexible y expandible, para poder ser colocado en un vehículo aéreo no tripulado, y que permita tener conexión directa a internet a través de la red GPRS (del inglés, General Packet Radio Service) que permita enviar los datos en tiempo real a una plataforma web desarrollada para este fin.

1.2 Objetivos Específicos

- Investigación de sensores y tarjetas para adquisición de datos a utilizar para la monitorización de las variables ambientales seleccionadas.
- Implementación la tarjeta de adquisición de datos y sensores seleccionados.
- Diseño de un sistema utilizando un microcontrolador y GPRS para permitir comunicación con un servidor.
- Investigación de tecnologías, estándares y modelos para el desarrollo de una aplicación web.
- Desarrollo de la plataforma web de monitorización ambiental.
- Realización de pruebas del sistema, plataforma web y hardware para la adquisición de datos.

1.3 Metas

- Desarrollo de hardware para la adquisición de datos y comunicación con Internet.
- Desarrollo del firmware con implementación del estándar de transmisión de datos ambientales a la plataforma web.
- Desarrollo de plataforma web de monitorización con las siguientes características:
 - Interfaz gráfica intuitiva y de fácil uso.
 - Gestión de zonas de monitorización y programación de recorridos.
 - Organización de datos por día, semana, mes y año.
 - Panel de instrumentos virtuales para monitorización de datos en tiempo real.
 - Geolocalización de la información, usando el API de Google Maps
 - Facilidad para agregar nuevos dispositivos de monitorización.
 - Diseño de un API REST para el fácil acceso a los datos desde aplicaciones externas.
 - Capacidad de exportar datos en formato CSV (Excel)
- Despliegue de la plataforma web ya sea un servidor privado o bien un servidor de la universidad, para la realización de pruebas.
- Realización de pruebas del sistema en conjunto Hardware-Plataforma y solución de posibles errores.

1.4 Justificación

Este proyecto pretende convertirse en una alternativa eficiente para el monitorización de variables ambientales, con ventajas frente a los sistemas convencionales de monitorización ambiental, entre ellas ser un sistema móvil de muestreo con la capacidad de programar su trayectoria a través del UAV, esto nos permite monitorear diversas áreas geográficas con un mismo equipo, lo que nos da la posibilidad de abarcar áreas geográficas que no son consideradas por los sistemas estáticos, además de poder monitorear áreas específicas que se encuentren en eminente riesgo ambiental.

Otra ventaja sobre los sistemas tradicionales es la posibilidad de tomar muestras a diferentes altitudes algo que no es posible realizar con un sistema convencional, que son datos muy valiosos para los investigadores. Además, al tener conexión directa vía internet con la plataforma de monitorización es posible tomar decisiones durante el vuelo del UAV con el objetivo de cambiar la ruta o bien incrementar las muestras en cierta área geográfica de interés, lo que se traduce en un ahorro de tiempo y recursos. Contar con un estándar para la transmisión de datos ambientales a través de un firmware para el hardware de monitorización, compatible con la plataforma web a desarrollar, permitirá la adición de nuevos sistemas de monitorización ambiental de forma rápida, eliminando los obstáculos que existen actualmente en cuanto a compatibilidad de protocolos, esto será reflejado en un ahorro de tiempo y recursos puesto que se evitará el desarrollo de estándares cada vez que se requiera agregar un nuevo sistema de monitorización o sensor.

Actualmente en el mercado existe una gran variedad de software especializado en la recolección y visualización de datos, sin embargo la mayoría de ellos no se especializa en monitorización ambiental ni en la transmisión de datos en tiempo real, además su rendimiento es bajo al tener un propósito general, además cabe destacar que el costo de las licencias de este tipo de software puede llegar a ser muy alto para aplicaciones comerciales y de investigación.

Es por ello que el desarrollo de la plataforma web contribuirá a tener una herramienta desarrollada a la medida basada en tecnologías libres, para cubrir las necesidades del proyecto la plataforma web tendrá las siguientes características: compatibilidad con el estándar de comunicación implementado en el hardware de monitorización, posibilidad de agregar nuevos dispositivos de monitorización de forma sencilla, gestión de zonas de monitorización y programación de recorridos, concentrar, almacenar y organizar los datos obtenidos de los dispositivos de monitorización, mapas con geolocalización y finalmente será una herramienta diseñada para compartir la información de forma sencilla a través de un API REST o bien de archivos CSV.

1.5 Descripción

El sistema de monitorización se colocará sobre un UAV (vehículo aéreo no tripulado) con lo cual se obtendrá una unidad de monitorización ambiental móvil con la capacidad de transmitir los datos en tiempo real para su almacenamiento, análisis y posible toma de decisiones.

Se propone definir un estándar para la transmisión de las variables medidas por el sistema de adquisición de datos compatible con la plataforma web donde se almacenará y organizarán los datos, esto con la finalidad de tener un sistema con posibilidades de escalar fácilmente y de esta forma poder crear redes de monitorización ambiental dinámicas y reducir costos.

Para lograr estos objetivos nos enfocaremos en las siguientes vertientes:

- Transmisión de datos desde el sistema de adquisición hacia la nube.
- Plataforma web desarrollada a la medida para almacenar, organizar, consultar y analizar los datos recibidos del sistema de monitorización.
- API (Application Programming Interface) para hacer disponible los datos para su consulta y análisis a través de software externo.

Este proyecto cuenta con diversos componentes esenciales que trabajando en conjunto conforman el sistema de monitorización ambiental, a continuación, se describen cada uno de los elementos que conforman el sistema.

- Hardware para la adquisición de datos y comunicación con Internet, los datos que reciba este sistema los procesara y adaptara para poder ser transmitidos a través de internet y ser compatibles con la plataforma web, para esto se desarrollará un firmware basado en el protocolo MQTT que implemente este estándar con el fin de que pueda ser reutilizado de forma sencilla en la implementación de nuevos sistemas de monitorización a bajo costo y de forma rápida.
- La plataforma en la nube será desarrollada bajo la arquitectura de software MVC (Modelo-Vista-Controlador), por lo cual consistirá en un modelado de las diferentes variables que serán monitoreadas con respecto al tiempo, ubicación de la muestra, etc., también se implementara una base de datos para almacenar la información generada por el sistema de adquisición y sus relaciones descritas en los modelos.

Se implementará la programación de las vistas que tendrán la función de gestionar zonas de monitorización y programar recorridos, desplegar la información recolectada que puede ser de forma cronológica, ubicación, poder realizar consultas en base hora, día, semana, mes y año, así como georreferenciación en mapas.

La plataforma estará diseñada para que sea sencillo agregar nuevos sistemas de monitorización y puedan comenzar a recolectar datos de forma inmediata, se podrá configurar que variable medirán, así como sus unidades.

Finalmente se implementará una capa de acceso a los datos (API REST) en formato JSON para que puedan ser consultados por aplicaciones externas.

El sistema se construirá basado en la siguientes tecnologías: Node.js para obtener los datos provenientes del sistema de adquisición de forma asíncrona y diseñar una interfaz de monitorización Real Time, Python-Django para realizar el modelado de la información y mapeo de base de datos, para realizar consultas y desplegar la información, la interfaz se construirá en base a tecnologías estándares HTML 5, CSS 3 y JavaScript.

Se correrá en un servidor NGIX, una base de datos SQL y de ser necesario una base de datos no relacional que sirva como puente entre Node.js y Python-Django esto para mejorar el rendimiento del sistema, estas tecnologías son compatibles para funcionar en un servidor basado en UNIX.

Capítulo 2. Antecedentes

2.1 Contaminación del aire

Existen 2 tipos de contaminantes atmosféricos [1], los contaminantes primarios que son aquellos que son emitidos hacia la atmósfera por agentes naturales como pueden ser las emisiones volcánicas o bien agentes humanos como puede ser la actividad industrial o emisiones producidas por automotores, entre otras fuentes de contaminación atmosférica, algunos de los contaminantes primarios son: dióxido de sulfuro, óxidos de nitrógeno, monóxido y dióxido de carbono, compuestos orgánicos volátiles y partículas suspendidas primarias PM.

El segundo tipo de contaminantes atmosféricos son los secundarios, aquellos que se forman en la atmósfera resultado de reacciones químicas entre los contaminantes primarios y algunos compuestos naturales como el oxígeno, hidrogeno y agua, entre los principales contaminantes secundarios podemos encontrar el Ozono, óxidos de nitrógeno y partículas secundarias PM.

Ambos tipos de contaminantes determinarán la calidad del aire en una área geográfica, actualmente existen diferentes técnicas para medir e incluso predecir la calidad del aire, pero estas técnicas requieren de datos como lo son los inventarios de emisiones que consisten en determinar y cuantificar las fuentes de cada uno de los contaminantes primarios por área geográfica, estos datos en combinación con modelos de dispersión hacen posible poder predecir la calidad del aire y ser usados para modelar condiciones locales, regionales e incluso globales además de observar tendencias de emisiones espacial y temporalmente.

Otros métodos alternativos como los modelos de receptor utilizan datos de calidad del aire en conjunto con información meteorológica para determinar y cuantificar como impactan ciertos tipos de fuentes contaminantes a la concentración de contaminantes en el aire.

La naturaleza de los contaminantes secundarios hace muy difícil predecir sus concentraciones con estos métodos sin embargo es posible estimar un índice de formación de contaminantes secundarios por unidad volumétrica por tiempo.

Para comprender mejor la naturaleza de los contaminantes y sus efectos se pueden clasificar también en 2 grupos según su estado físico, contaminantes gaseosos y partículas contaminantes:

Contaminantes gaseosos

Son aquellos contaminantes en estado gaseoso o vapores, físicamente son aquellos contaminantes que se encuentran como pequeñas moléculas individuales, son capaces de pasar a través de un filtro por lo cual no pueden ser capturados o no reaccionan con el medio del filtro así que este tipo de contaminantes pueden penetrar fácilmente espacios interiores e ingresar al sistema respiratorio humano.

Partículas contaminantes

Este tipo de contaminantes pueden ser partículas sólidas o líquidas suspendidas en la atmósfera, pueden ser de origen primario o secundario, las partículas secundarias son las más pequeñas entre 1 y 2 nm de diámetro, mientras que las partículas más grandes como polvo pueden llegar a 0.1mm (100 μ m) de

Capítulo 2. Antecedentes

diámetro, entre más grandes sean estas partículas debido al efecto de la gravedad estas se asientan rápidamente y su existencia en la atmósfera es mucho más corta y además su dispersión es menor por lo cual a menudo solo se encuentran cerca de la fuente que las emite, en contra parte las partículas más pequeñas menores a $20\mu\text{m}$ su tiempo de existencia y propagación en la atmósfera es mayor por lo cual este tipo de partículas representan un mayor riesgo para salud ya que además pueden ingresar al cuerpo humano y penetrar tejidos, por su riesgo a la salud y también con fines de monitorización este tipo de partículas se clasifican en PM_{10} y $\text{PM}_{2.5}$, las partículas PM_{10} son todas aquellas de un diámetro igual o menor a $10\mu\text{m}$ y las partículas $\text{PM}_{2.5}$ son todas aquellas de un diámetro igual o menor a $2.5\mu\text{m}$ es decir las partículas $\text{PM}_{2.5}$.

Hasta hace algún tiempo las partículas PM_{10} era el estándar para clasificar las partículas contaminantes suspendidas en el aire, sin embargo reciente la OMS ha recomendado utilizar las $\text{PM}_{2.5}$ como nuevo indicador de las partículas contaminantes en el aire, esto hace mucho sentido si tomamos en cuenta que las partículas contaminantes provenientes la quema de combustible fósiles se encuentran en su mayoría bajo esta clasificación $\text{PM}_{2.5}$, quedando un espectro mucho menor de partículas contaminantes mayor a $2.5\mu\text{m}$.

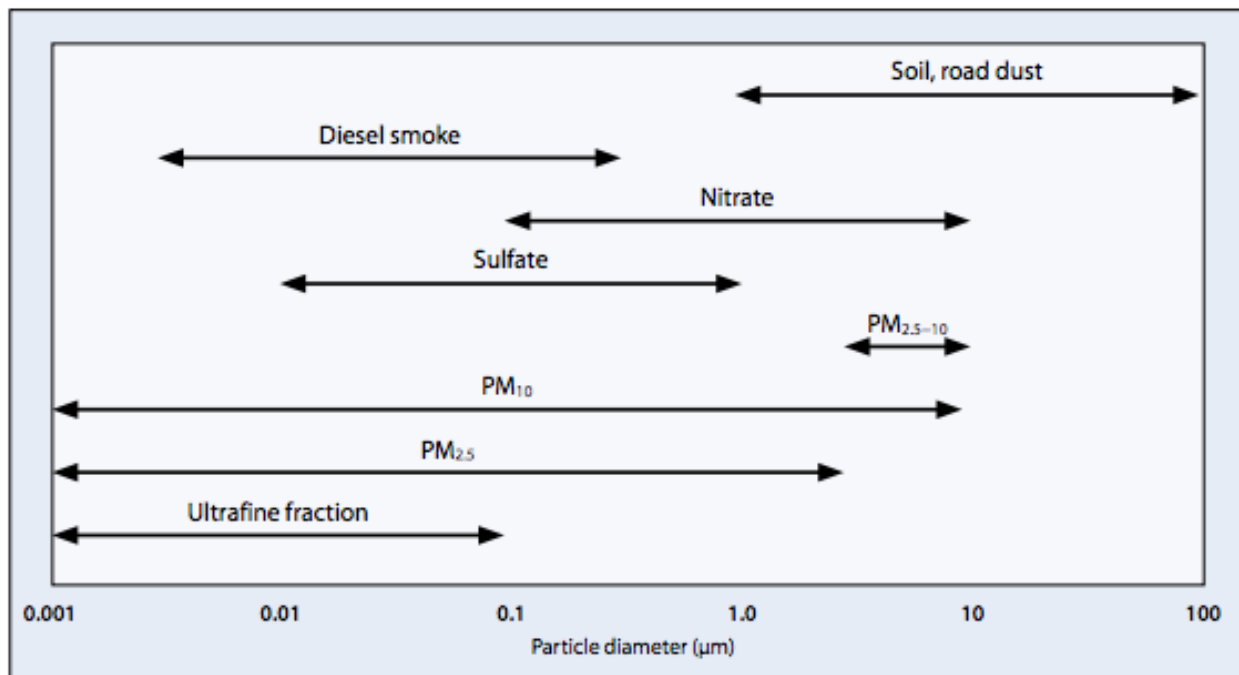


Figura 1 Rango de tamaño de partículas aéreas, se muestran las más relacionadas con la salud como partículas ultra finas, $\text{PM}_{2.5}$, PM_{10} y el tamaño tipo de otros contaminantes comunes

La contaminación del aire se produce acorde a los diferentes tiempos de vida que tienen los contaminantes a partir de la emisión desde su fuente puesto que este tiempo de vida repercute directamente en el rango de propagación que pueden tener en el espacio, acorde a la escala de propagación se puede clasificar la contaminación del aire en escala local, escala urbana, escala regional y finalmente escala global.

Escala Local

Los contaminantes tienen diferentes tiempos de vida debido a efectos físicos o químicos, por ejemplo, las partículas contaminantes más grandes (mayor a 20µm) duran poco tiempo en la atmósfera ya que por acción de la gravedad son decantados al suelo, otros contaminantes como el 1,3-Butadieno tiene un tiempo de vida muy corto por lo regular de 1 hora debido a que se degrada rápidamente al contacto con la atmósfera. Por lo cual el tiempo de vida de los contaminantes es un factor primordial que determina la propagación geográfica que pueda llegar a tener, entre más corto el tiempo de vida del contaminante menor será su propagación y viceversa, por lo cual los contaminantes con un periodo de vida corto solo se podrán encontrar en elevadas concentraciones cerca de su fuente de emisión por lo regular industrias, vías de alta circulación, entre otras fuentes y es solo cerca de su fuente donde tienen un efecto en la contaminación del aire local, es por ello que este tipo de contaminación se le denomina escala local.

Escala Urbana

La contaminación del aire a escala urbana es producida por contaminantes con un tiempo de vida mayor a los contaminantes de escala local, sin embargo, no es muy grande por lo regular de solo unas horas, tiempo suficiente para propagarse dentro de la ciudad donde fueron emitidos y en ocasiones dependiendo de las condiciones meteorológicas y geográficas pueden llegar a impactar a zonas rurales aledañas.

Ejemplos de contaminantes de escala urbana son los óxidos de nitrógeno y monóxido de carbono, ambos producidos por la quema de combustibles fósiles que mueven el parque vehicular de las ciudades, estos contaminantes se encuentran en grandes concentraciones dentro de la ciudad donde fueron emitidos generando un impacto negativo en la calidad del aire urbano por lo local se le denomina escala urbana.

Escala Regional

Contaminantes con algunas partículas PM_{2.5} y algunos contaminantes gaseosos como el ozono, tienen periodos de vida largos de días e incluso semanas lo cual les permite propagarse fácilmente desde su fuente de emisión en una escala regional, pasando ciudades, estados e incluso países de acuerdo a las condiciones meteorológicas y geográficas.

Escala Global

Existen contaminantes con periodos de vida muy largos en el rango de años, lo que les permite propagarse a grandes distancias llegando a impactar todo un hemisferio e incluso todo el mundo, algunos de estos contaminantes están relacionados con el calentamiento global como lo son el dióxido de carbono, óxido de nitrógeno y el metano, es por ello que se le denomina contaminación a escala global.

Tipos de fuentes de contaminación

Para poder comprender mejor la metodología de cómo se crean los inventarios de emisiones es importante conocer más acerca de las fuentes de contaminación, como hemos mencionado una fuente de contaminación del aire es todo aquel agente que emite uno o más contaminantes hacia la atmósfera, una forma sencilla de categorizar los tipos de fuentes es en móviles y estáticas [2], dentro de las fuentes móviles podemos encontrar a los automotores, aviones, barcos, entre otros medios de transporte que se mueven gracias a combustibles fósiles. Dentro las fuentes de contaminación estáticas podemos encontrar por ejemplo a las casas e industrias.

Sin embargo, para fines de estudio es poco práctica esta clasificación, ya que seguir el rastro de cada una de las fuentes móviles y estáticas sería muy difícil, por lo cual los científicos muy pocas veces se interesan en estudiar las fuentes bajo esta clasificación, pero si están interesados en los efectos que producen por ejemplo la cantidad de contaminantes emitidos en una autopista durante un periodo de tiempo determinado.

Es por ello que existen otras clasificaciones mucho más prácticas para efectos de estudio, una de estas clasificaciones son, Puntos fuente, Líneas fuente y Áreas fuente.

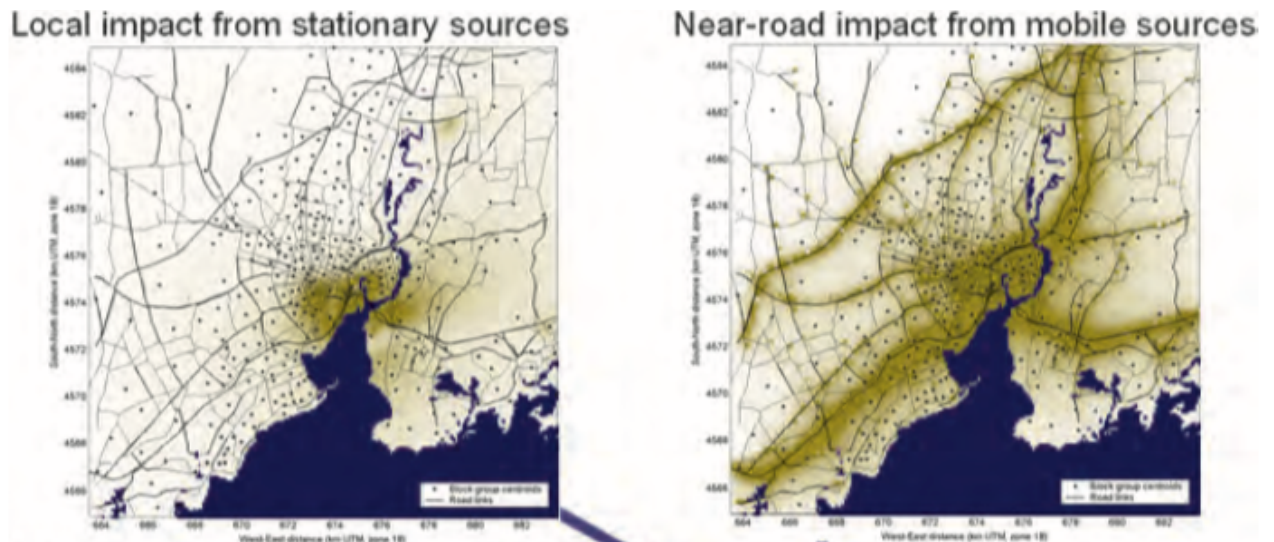


Figura 2 Impacto de fuentes de contaminación estacionaras y móviles en la calidad de aire en una zona urbana [2]

Esta clasificación se basa en el contexto metodológico de un inventario de emisiones, por lo cual se considera a la espacio geográfico dentro de la malla, de esta forma un punto fuente es aquel que se puede representar como puntos individual de la malla, siempre y cuando el área de emisiones sea igual o menor a 1 x 1 km, de esta forma se puede considerar como un punto fuente a una planta termoeléctrica aun cuando los contaminantes provienen de diferentes chimeneas dentro del complejo, sin embargo se le considera como un solo punto fuente ya que el principal interés de los inventarios de contaminantes radica en conocer la masa de contaminantes emitidos y no la forma en que fueron emitidos.

Como hemos mencionado las fuentes de contaminación móviles son raramente estudiadas individualmente, sin embargo, cuando estas fuentes móviles emiten contaminantes lo hacen formando líneas bien definidas, entre este tipo de fuentes encontramos a las autopistas, vías principales, vías de tren, etc. es por ello que a este tipo de fuente se les llama “Líneas fuente.”

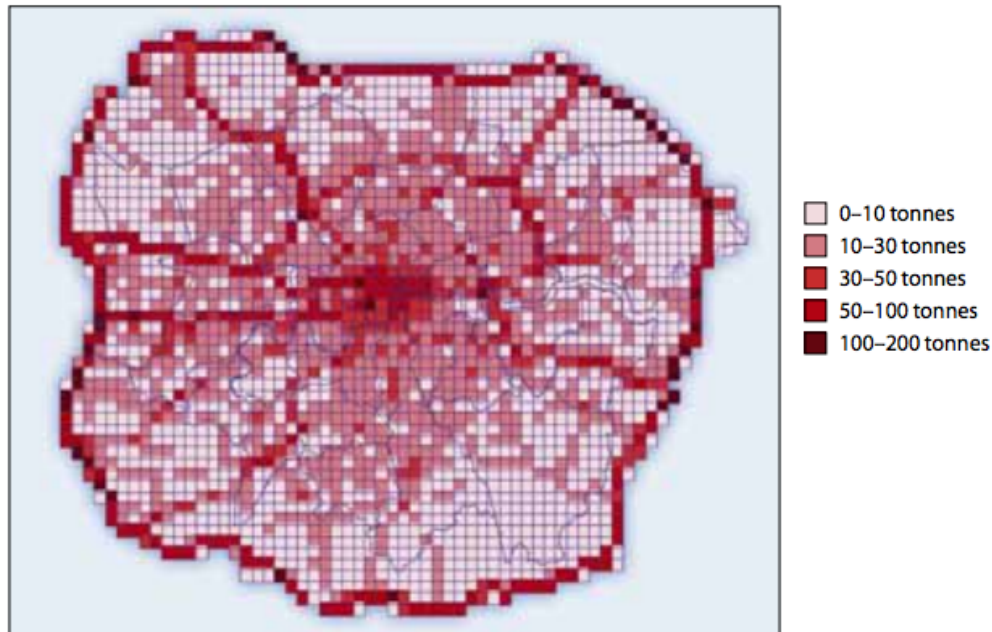


Figura 3 Visualización de los diferentes tipos de fuentes de contaminación representados en un inventario de emisiones.

Existen otro tipo de fuentes que su emisión de contaminantes no encaja en ninguna de las anteriores (puntos y líneas), ya que estas fuentes se encuentran dispersas en una determinada área geográfica y debido a que su contribución individual es pequeña no se le puede considerar como punto fuente, sin embargo su impacto tiene relevancia si se considera su efecto en conjunto, es por ello que se les conoce como áreas fuente, un ejemplo de este tipo de fuente son los boilers usados en las casas, su impacto individual es muy pequeño pero es considerablemente grande si se toman en cuenta todos los boilers de una ciudad como una sola fuente de emisión de contaminantes.

2.2 Variables Meteorológicas

Además de la calidad del aire, medir variables meteorológicas para conocer el clima actual y futuro es igual de importante ya que la calidad del aire no solo depende la concentración de partículas contaminantes sino también de las condiciones climáticas que se presenten en combinación.

Además, el clima es un factor muy importante en nuestras vidas ya que de ello dependen muchas de las actividades económicas y cotidianas de la sociedad moderna, por lo que conocer el clima actual y futuro brinda certidumbre.

La monitorización meteorológica hace posible llevar un seguimiento de diferentes comportamientos climáticos como la temperatura, humedad, presión atmosférica, radiación UV, velocidad y dirección del

viento, esta información es relevante para hacer pronósticos del clima a futuro, además en combinación con los datos de contaminantes atmosféricos permite definir cómo se comportará la concentración de los mismos y por lo tanto como se reflejará en la calidad del aire en el presente y futuro.

Para este trabajo se tomarán mediciones de las siguientes variables:

Temperatura

La temperatura es una medida comparativa de calor o frío, usualmente se mide con un termómetro, existen diversas escalas de medición las más comunes son, Celsius °C (también conocida como Centígrado), también se mide en Fahrenheit °F y Kelvin °K [3].

Sus equivalencias se pueden encontrar en las siguientes formulas:

$$1^{\circ}\text{K} = 273 + ^{\circ}\text{C}$$

$$1^{\circ}\text{F} = 32 + 9/5 ^{\circ}\text{C}$$

Para el objeto de este trabajo la medición de temperatura se realizará en °C.

Presión Atmosférica

La presión atmosférica en resumen es el peso del aire en la atmosfera de la tierra, es decir la fuerza que ejerce la atmosfera sobre un cuerpo en la superficie terrestre, por lo tanto, la presión atmosférica varía según la altitud donde se mida, a mayor altitud menor cantidad de aire en la atmósfera y por lo tanto menor presión atmosférica y a menor altitud se tendrá una mayor presión atmosférica.

La unidad de medida para la presión atmosférica pueden ser mili bares, pascales o atmosferas, a continuación se define su equivalencia [3]:

$$1 \text{ mili bar} = 1 \text{ hecto Pascal (100 Pascales)}$$

La presión a nivel del mar depende de la región geográfica, una medida normalizada de presión atmosférica media a nivel del mar se tomó como 101,324 Pascales y se definió como 1 atmosfera.

$$1 \text{ atmosfera} = 101,324 \text{ Pascales o } 101.324 \text{ mili bares.}$$

Para el objeto de este trabajo la medición de presión atmosférica se realizará en mili bares (mb).

Índice de radiación UV

La radiación UV proviene de diversas fuentes artificiales, pero la principal fuente de exposición para los humanos es el Sol, la radiación UV es un tipo de radiación electromagnética en un rango de longitud de onda entre 100 y 400 nm, se divide en tres bandas [4]:

UVA 315 - 400 nm

UVB 280 - 315 nm

UVC 100 - 280 nm

Cuando la luz solar pasa por la atmosfera toda la radiación UVC y aproximadamente el 80% de la radiación UVB es absorbida por la capa de ozono, vapor de agua, oxígeno y dióxido de carbono presente en la atmosfera, la radiación UVA es poco afectada en su paso por la atmosfera, por lo cual la radiación UV que llega a la superficie terrestre en su mayoría está compuesta por radiación UVA y un pequeño componente de radiación UVB.

En pequeñas cantidades la radiación UV es benéfica para la salud humana ya que es un componente esencial para la producción de vitamina D, sin embargo, en grandes cantidades es perjudicial para la salud siendo factor de riesgo para diversas enfermedades cutáneas incluyendo cáncer de piel.

Diversos factores geográficos y físicos influyen en los niveles y sus efectos de la radiación UV, por lo cual se desarrolló el Índice Global de Radiación UV como un estándar para conocer los niveles de radiación UV y la peligrosidad ante su exposición, el índice va del 0 al valor más alto de la escala 11+, entre más grande el valor el potencial de daño a la piel y ojos es mayor y al mismo tiempo significa menos tiempo para que el daño ocurra.

Categoría de exposición	Intervalo IUV
Baja	<2
Moderada	3 a 5
Alta	6 a 7
Muy alta	8 a 10
Extremadamente alta	11+

Figura 4 Escala del Índice de Radiación UV

2.3 Monitorización Ambiental

La monitorización ambiental cada día tiene más relevancia especialmente cuando se trata de la medición de la contaminación del aire, puesto que es uno de los principales problemas que afectan la calidad de vida en las grandes ciudades debido al grave impacto que tiene en la salud de las personas, este problema va en aumento en la mayoría de las ciudades de países emergentes debido a que aún no se tiene una clara regulación para disminuir la emisión de gases contaminantes.

Según estudios epidemiológicos la contaminación del aire puede causar graves daños a la salud de la población expuesta por largos periodos de tiempo, siendo las personas mayores de 65 años la población más vulnerable, algunos de los problemas de salud derivados de la contaminación del aire son cáncer, daños en el sistema respiratorio, nervioso y cardiovascular [5].

En la mayor parte de las ciudades de países desarrollados se tienen redes de monitorización ambiental sin embargo su distribución puede ser deficiente ya que las estaciones de monitorización se encuentran

en lugares fijos lo que da como resultado solo un muestreo de la calidad del aire en ciertas zonas, sin embargo la contaminación ambiental está muy relacionada con el área geográfica puesto que depende principalmente de factores localizables como lo son las áreas industriales, la concentración de población y trasportes motorizados, o bien la altitud de la zona geográfica [6] lo que da como resultado que no se pueda brindar información sobre la calidad del aire a los habitantes de una determinada zona de la ciudad que les permitan tomar medidas para cuidar su salud.

Este problema es mucho más grave en las ciudades de países emergentes donde aún muchas de ellas no cuentan con una red de monitorización de calidad del aire y las que cuentan con una red de monitorización no tienen suficientes estaciones distribuidas, considerando la elevada densidad de población en ciertas zonas, esto es un serio problema puesto que una medición de la calidad de aire de una zona a otra puede ser diferente y por lo tanto las recomendaciones a la población pueden no ser las adecuadas afectando la salud de una gran cantidad de habitantes.

Datos de la organización mundial de la salud indican que solo en 2012 se dieron 3.7 millones de muertes relacionadas con la contaminación del aire en zonas urbanas de todo el mundo [7].

En países emergentes como el nuestro la calidad del aire es un serio problema que va en aumento, según datos de un estudio realizado por el Clear Air Institute, nuestro país es el segundo de América Latina en registrar más muertes relacionadas con la contaminación del aire con 15,000 muertes por año, solo superado por Brasil con 23,000 muertes al año, aunado a este problema el Clear Air Institute señala que en América Latina incluido México no existen técnicas estandarizadas de monitorización ni de recolección de datos [8].

Otro estudio realizado por el Instituto Mexicano de la Competitividad en donde se analizaron datos de las 34 ciudades con más de 500,000 habitantes del país que de acuerdo a la norma NOM-156-SEMARNAT-2012 son las ciudades obligadas a operar un sistema de monitoreo de la calidad del aire sin embargo en la práctica la mayoría de ellas no cuenta con uno, solo 12 ciudades fueron las que reportaron datos de calidad del aire, el estudio relaciona la contaminación del aire en México con el impacto económico que genera, entre sus resultados se tiene que el promedio de concentración de partículas PM10 es de 57.71 $\mu\text{g}/\text{m}^3$ cuando la recomendación de la OMS es de 20 $\mu\text{g}/\text{m}^3$ en base a estos datos el impacto económico anual se estima en 4,123,716,987 MXN entre pérdidas de productividad y gastos en salud [9].

Como podemos ver los datos son alarmantes en nuestro país y de no implementar medidas eficaces para mejorar la calidad del aire aunado con el envejecimiento de la población, en las próximas décadas se convertirá en un grave problema de salud pública.

En México actualmente existen programas de monitorización compuestos por redes automáticas o manuales y en algunos casos mixtas, cabe destacar que este tipo de estaciones son fijas, evalúan la concentración de los siguientes contaminantes: Dióxido de Azufre (SO_2), Monóxido de Carbono (CO), Partículas Suspendidas (PST, PM10 y PM2.5), Óxidos de Nitrógeno (NO_2 – NO – NO_x), Ozono (O_3), Plomo (Pb), ácido sulfhídrico (H_2S), metales pesados, sulfatos, nitratos, y otros parámetros como: radiación solar y depósitos atmosféricos (seco y húmedo) además se cuenta con dispositivos para determinar los parámetros meteorológicos como son dirección y velocidad del viento, temperatura y humedad relativa que en conjunto con los datos de calidad del aire permiten realizar predicciones de concentraciones futuras al nivel de la superficie [10].

Los principales sistemas de monitorización en México se indican a continuación:

- Sistema de Monitoreo Atmosférico de la Ciudad de México
- Red de Monitoreo de la Calidad del Aire de la Zona Metropolitana de Guadalajara
- Sistema Integral de Monitoreo Ambiental (SIMA) de Monterrey
- Red de Monitoreo de la Calidad del Aire de la Zona Metropolitana de Toluca
- Red de Monitoreo de la Calidad del Aire de Ciudad Juárez
- Redes de Monitoreo de la Calidad del Aire de Tijuana y Mexicali
- Red Estatal de Monitoreo Automático de Puebla (REMA)
- Sistema de Monitoreo de la Calidad del Aire en el Corredor Industrial del Bajío

A continuación se muestra la distribución geográfica de las estaciones de monitorización de calidad del aire de las principales zonas metropolitanas del país [11].



Figura 5 La zona metropolitana de la Ciudad de México cuenta con 30 estaciones de monitorización.



Figura 6 La zona metropolitana de Guadalajara cuenta con 10 estaciones de monitorización.



Figura 7 La zona metropolitana de Monterrey cuenta con 8 estaciones de monitorización.



Figura 8 La zona metropolitana de Puebla cuenta con 4 estaciones de monitorización.

Como se puede ver existen esfuerzos en cuestión de monitorización ambiental sin embargo no son suficientes puesto que como ya se ha mencionado existen zonas que requieren mayor atención como son zonas de alta densidad de población, alta industrialización, zonas de posible riesgo ambiental, entre otras y la información que se genera es insuficiente para poder tomar acciones en cuestión de políticas públicas que ayuden a mejorar la calidad del aire, para generar estrategias de contingencias ambientales e incluso para poder dar recomendaciones más específicas y confiables a los habitantes de una determinada zona geográfica dentro de las ciudades, en base a esto podemos ver que es indispensable una mayor cantidad de información con una mayor representatividad de las áreas geográficas, sin embargo crecer la red de monitorización en cada una de las ciudades a través de estaciones fijas es costoso en su instalación y mantenimiento, por lo cual es necesario explorar otras alternativas como la presentada en esta tesis.

2.4 Monitorización ambiental desde una perspectiva del internet de las cosas.

El internet de las cosas es un concepto nuevo que ha estado ganando relevancia rápidamente, el internet como lo conocemos tradicionalmente es una red que interconecta a usuarios finales (humanos), el internet de las cosas cambia este paradigma al incorporar a la red objetos físicos (things) que se pueden comunicar entre ellos y/o con usuarios finales con el objetivo de crear una nueva gama de servicios y aplicaciones que mejoren la calidad de nuestras vidas, esto es posible a través de sistemas embebidos con capacidades de cómputo y comunicación que continuamente mejoran sus prestaciones.

Para que un objeto se pueda considerar parte del internet de las cosas (smart object) requiere cumplir con 3 pilares sobre los que está basado el concepto de internet de las cosas, el primero es ser identificable, el segundo es poder comunicarse y el tercer es poder interactuar ya sea con su entorno, con otros objetos o bien con usuarios finales [12].

Desde el punto de vista técnico para poder cumplir con estos 3 pilares el objeto debe cumplir con estas características:

- Ser un objeto real con características físicas como forma, tamaño, peso, etc.
- Contar con capacidades mínimas de comunicación, como lo es poder identificar y aceptar mensajes entrantes y poder responder a ellos.
- Tener un identificador único que le permita diferenciarse de otros dispositivos en la red.
- Estar asociados a un nombre (una descripción entendible por humanos) y una dirección (una cadena de texto entendible por una computadora, que es usada para poder establecer comunicación con el objeto).
- Contar con capacidades básicas de cómputo, para poder ejecutar funciones básicas como poder reconocer un mensaje entrante, administrar funciones de la red o establecer nuevas comunicaciones salientes.
- Debe contar con la capacidad de sensar fenómenos físicos como temperatura ambiental, luz, humedad, etc. o bien poder ejecutar acciones hacia el mundo real a través de actuadores.

Internet de las cosas es acerca principalmente de dispositivos físicos sensando datos del mundo real y enviándolos a la red o bien recibiendo datos de la red con el objetivo de ejecutar alguna acción hacia el mundo real, lo que contrasta con la concepción del internet tradicional donde el enfoque es solo en la comunicación de dispositivos conectados punto a punto sin importar las aplicaciones.

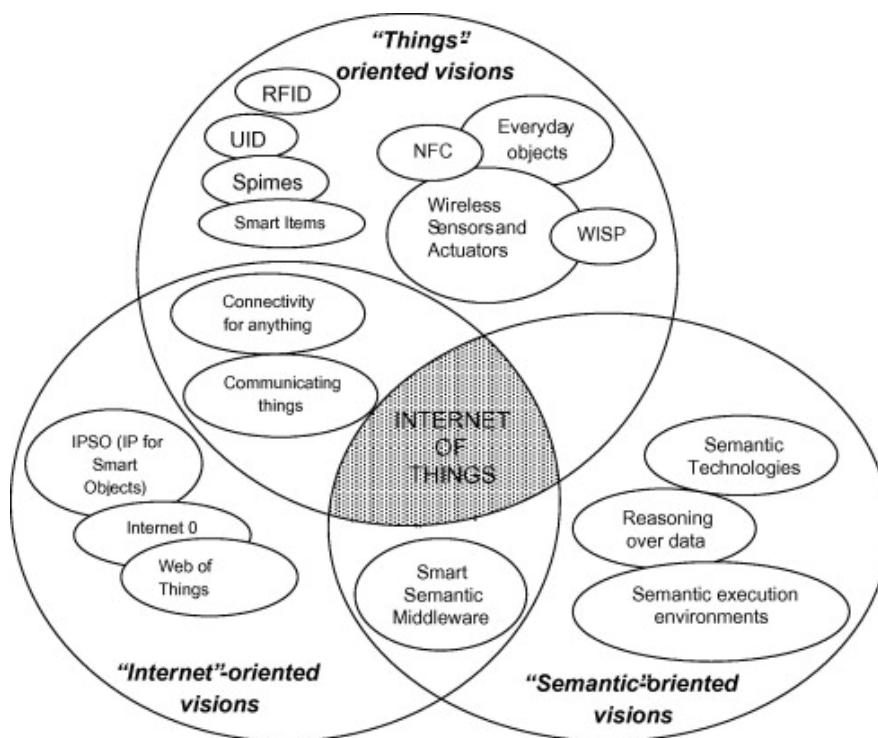


Figura 9 El internet de las cosas es la sinergia de diferentes disciplinas tecnológicas

Sin embargo, no es suficiente con tener los dispositivos conectados. Se requieren de servicios que integren y aprovechen esta infraestructura de dispositivos hacia productos que generen valor ya sea económico, académico/científico o social en uno o diferentes campos de la vida diaria, para lograr crear este tipo de servicios es necesario cumplir con ciertos objetivos a la hora de integrar los dispositivos, la primera es la capacidad de abstraer el dispositivo físico hacia el campo digital a través de representaciones estandarizadas (virtualización) de los dispositivos físicos para poder aprovechar sus características ya sea de sensado o de aplicar alguna acción hacia el mundo real, tomando en cuenta que el dispositivo envía y recibe información como una cadena de datos que debe ser diseñada para cumplir estos objetivos de forma eficiente.

Otro de los objetivos que se deben cumplir para ofrecer un servicio eficiente es la capacidad de escalar, nos referimos a que nuestro servicio debe ser capaz de poder crecer en el número de dispositivos conectados sin afectar o interferir con los dispositivos actuales así como poder trabajar en paralelo las mismas funciones en todos ellos, además de ser transparente para los usuarios al agregar un nuevo dispositivo o servicio es decir que sea sencillo evitando involucrar al usuario en los procesos técnicos que hacen que esto sea posible.

En resumen, **internet de las cosas** nos brinda nuevas oportunidades de comprender e interactuar con el mundo real, basado en grandes cantidades de información que podemos recolectar y analizar para conocer como nunca antes el mundo que nos rodea.

Es por esta razón que este trabajo tiene como interés principal diseñar un sistema de monitorización de variables ambientales que aproveche las ventajas del internet de las cosas combinado con otra tecnología emergente que nos permite tener una movilidad eficiente, nos referimos a los vehículos aéreos no tripulados (UAV por sus siglas en inglés), en combinación se espera tener un sistema de monitorización ambiental con una serie de prestaciones que aporten valor al sector que estudia la monitorización de variables ambientales.

El internet de las cosas se basa en una serie de tecnologías que lo hacen posible, que describiremos a continuación para tener un mejor entendimiento del sistema.

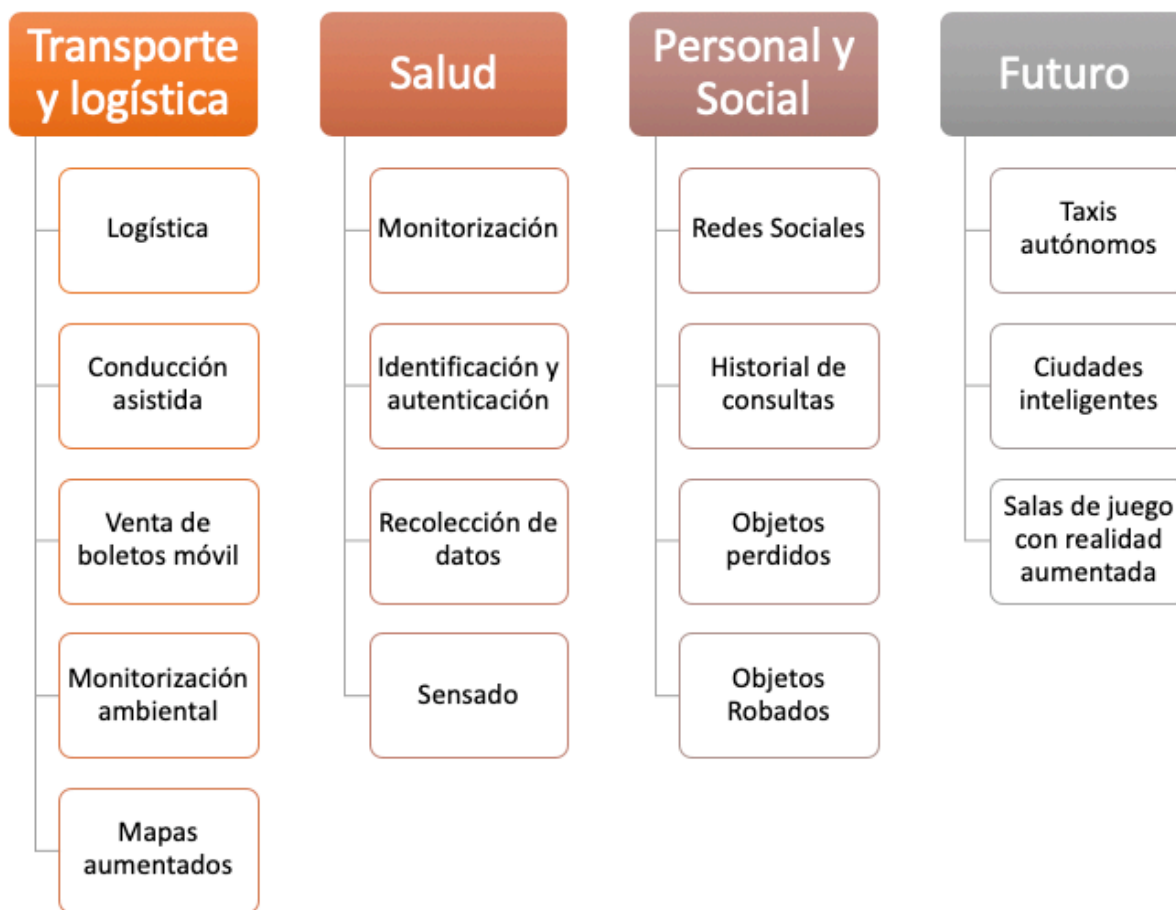


Figura 10 Campos dominantes y sus escenarios más relevantes de posibles aplicaciones del IOT

El internet de las cosas funciona sobre la misma infraestructura y protocolos que soportan el internet moderno, uno de esos pilares es el protocolo TCP/IP.

El protocolo TCP/IP es una suite de protocolos que hacen posible la comunicación entre computadoras, tiene dos componentes principales el IP (Internet Protocol) que es el más importante ya que todos los demás protocolos funcionan sobre esté, el segundo componente principal es el protocolo TCP (transmission control protocol), de este par de protocolos esta suite toma su nombre.

TCP/IP no es una tecnología nueva, sus orígenes se remontan al primer experimento de red de paquetes conmutados concebido en los años 70s dentro de la Agencia de Proyectos de investigación Avanzados de Defensa (DARPA), a esta red experimental se le denominó ARPANet esta red era administrada por el departamento de defensa de los Estados Unidos de América con fines militares, fue en este contexto donde se desarrollaron las bases de lo que actualmente es la suite de protocolos TCP/IP [13].

Posteriormente ARPANet comenzó a dejar de ser exclusivamente de uso militar y se le permitió a universidades, centros de investigación y algunas empresas unirse a la red, continuando su expansión y apertura con el paso de los años, fue en 1990 cuando ARPANet desaparece oficialmente dando paso al Internet como lo conocemos hoy.

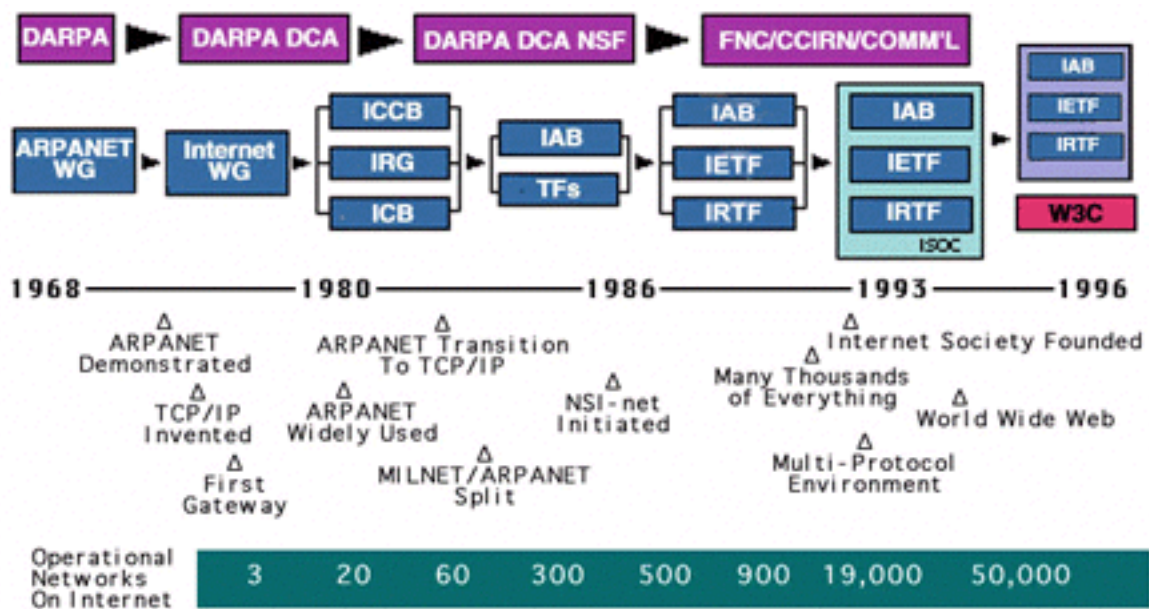


Figura 11 Línea de tiempo de la evolución de ARPANET al internet moderno.

Ahora se definiera más a fondo en que consiste esta suite de protocolos ya que actualmente está conformada por un gran número de protocolos que a continuación se describirán organizados en base a su función.

Capítulo 3. Descripción del Software

3.1 Protocolos de Internet

3.1.1 Protocolo de Internet

El protocolo de internet es la pieza medular de toda la suite (TCP/IP) ya que provee las funciones que los otros protocolos usan para definir sus propios estándares, este protocolo se encarga de crear redes en donde los paquetes puedan alcanzar su destino correspondiente dentro de la red sin importar crear una conexión previa, es decir que los equipos que desean conectarse no negocian una conexión previa y tampoco se encargan de detectar errores en la transmisión, de esto se encargan las capas de más alto nivel de la suite [13].

Actualmente existen 2 versiones del protocolo IP, IPV4 que es el más usado actualmente pero debido al crecimiento exponencial de internet se está quedando corto para poder seguir operando ante el continuo crecimiento de la red global y aun más amenazada su operatividad cuando hablamos del crecimiento en el número de dispositivos conectados a internet que se está dando con el internet de las cosas, es por ello que se diseñó un nuevo protocolo llamado IPV6 que tiene una capacidad de direccionamiento entre los dispositivos conectados muy superior a IPV4 sin embargo ambos protocolos no son compatibles entre si es por ello que la migración de IPV4 a IPV6 se está realizando gradualmente para comprometer lo menos posible la operatividad de internet.

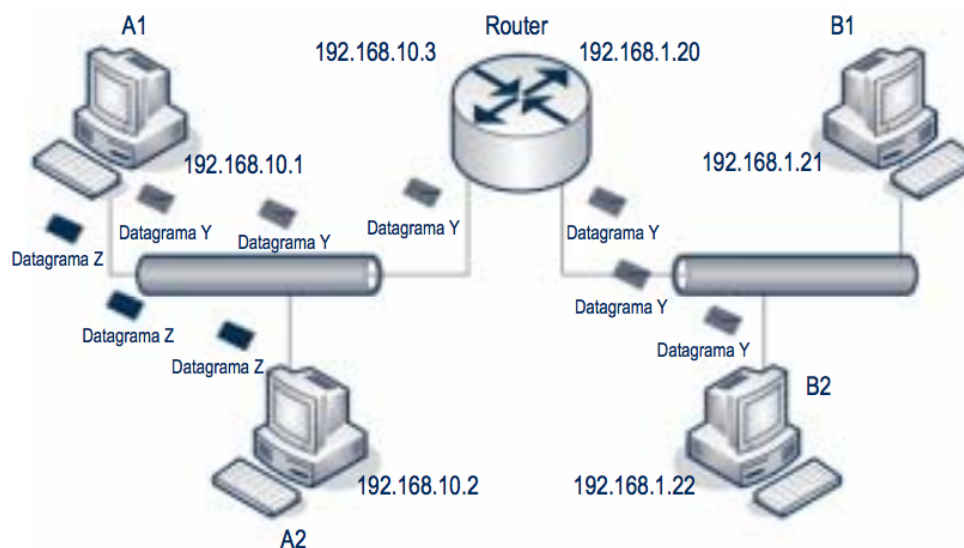


Figura 11 Diagrama general del funcionamiento del protocolo de internet

Protocolos de Transporte

La suite de protocolos tiene 2 protocolos de transporte TCP y UDP, el primero Protocolo de Control de Transmisiones tiene una función primordial que es la de proveer una capa de transporte de datos garantizada y confiable que pueda ser usada por otras aplicaciones y protocolos superiores.

UDP que significa protocolo de datagramas del usuario en contraste con TCP provee una capa de transporte más simple y rápida sin embargo delega a las aplicaciones que lo usen proveer la confiabilidad que requieren ya que UDP no se concentra en este aspecto.

Protocolos de enrutamiento

Para hacer las conexiones de red adaptativas es importante usar protocolos que puedan detectar los cambios en la topología de la red y reaccionar de forma acorde, los 2 mayores grupos de protocolos de enrutamiento que existen son IGP (Interior Gateway Procols) y EGP (Exterior Gateway Protocols), estos protocolos incluyen a otros protocolos como RIP (Routing Information Protocol) y OSPF (Open Shortest Path First Protocol).

Aplicaciones de usuario final

Finalmente encontramos la capa superior de la suite de protocolos que es donde podemos ver se funcionan una gran variedad de aplicaciones que definen sus propios protocolos sobre las capas de transporte ya descritas previamente, entre estos protocolos podemos encontrar HTTP (Hyper-Text Transfer Protocol) que es el más popular puesto que sobre este protocolo funciona la web moderna es decir todos aquellos sitios web que visitamos diariamente desde un ordenador o desde un dispositivo móvil, otro protocolo importante que podemos mencionar es FTP (File Transfer Protocol) que como su nombre lo dice es un protocolo que hace posible la transferencia de archivos entre puntos de la red, también podemos encontrar a los protocolos que hacen posible el correo electrónico como lo conocemos hoy en día nos referimos a SMTP (Simple Mail Transfer Protocol) y POP (Post Office Protocol).

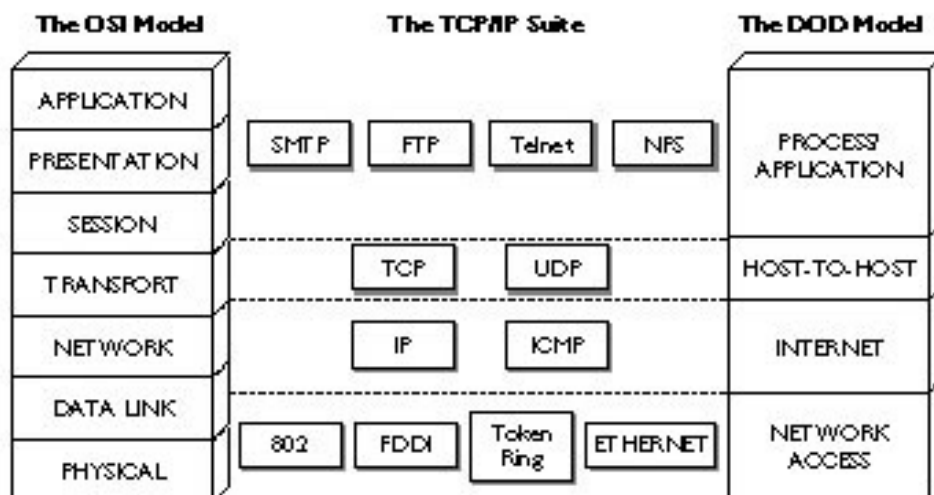


Figura 12 Descripción de la suite TCP/IP y sus modelos.

3.1.2 Protocolo HTTP

Internet por si mismo es entendido como una capa de transporte de información, sin embargo, a menudo es confundido con la WWW (world wide web), conocida comúnmente solo como Web, la cual es una aplicación más que funciona usualmente sobre la suite TCP/IP, esta confusión se deriva a que para la mayoría de los usuarios la WWW es la aplicación final con la que más a menudo tienen contacto.

LA WWW fue desarrollada por Tim Berners-Lee en 1989 mientras trabajaba en la European Organization for Nuclear Research conocida comúnmente como CERN, Tim Berners-Lee escribió el primer navegador web y las primeras aplicaciones web de servidor comenzando así una nueva industria digital que domina la internet hasta nuestros días.

Durante el desarrollo del primer navegador y primeras aplicaciones web, Tim Berners-Lee creo 2 elementos claves que sostienen la web hasta el día de hoy, nos referimos al Hyper-Text Mark-Up Language (HTML) que es lenguaje usado para la creación de páginas web y el otro elemento clave es el Hyper-Text Transfer Protocol (HTTP) que es usado en el transporte del tráfico web.

El protocolo HTTP es un protocolo basado en el modelo request/response, esto quiere decir que un cliente envía una petición al servidor incluyendo información de un método, un URI, una versión del protocolo, seguido de un mensaje MIME-like que contiene los modificadores de la petición, información del cliente y posible contenido principal a través del servidor. El servidor responderá con una línea de status, la versión del protocolo de mensajes y un código de éxito o error, seguido por un mensaje MIME-like que contiene información del servidor [14].

HTTP hace uso de una serie de métodos para definir sus peticiones a continuación se describe cada método y su función:

OPTIONS

Este método tiene como función solicitar al servidor información acerca de las opciones de comunicación disponibles en las peticiones y respuestas, le permite determinar al cliente los requerimientos asociados con un recurso o las capacidades del servidor sin tener que iniciar la solicitud de un recurso.

GET

Solicita un recurso al servidor que es identificado por una URL de petición.

HEAD

Este método es similar al método a GET a excepción que solo devuelve meta-información contenida en la cabecera de la respuesta HTTP y no en el cuerpo del mensaje.

POST

Este método es usado para solicitar que el servidor acepte información que se encuentra en el cuerpo de la petición.

PUT

Este método solicita al servidor que se almacene un recurso enviado en el cuerpo del mensaje.

DELETE

Solicita al servidor que se elimine el recurso identificado en la URL de petición.

TRACE

Este método permite al cliente determinar que está siendo recibido al otro lado de la conexión y usar la información para un diagnóstico.

3.1.3 Protocolo MQTT

MQTT es un protocolo de mensajería basado en el modelo publicar/suscribir, es ligero, de código abierto, simple de usar e implementar.

Por lo cual es una excelente opción para aplicarse en un gran variedad de proyectos, especialmente en aquellos donde se tienen limitantes definidas como recursos de computo, batería y datos, como es el caso de comunicaciones M2M (machine to machine) e IOT (Internet de las cosas).

MQTT fue desarrollado en 1999 por Andy Stanford-Clark quien era parte de IBM y por Arlen Nipper parte de Arcom hoy en día Cirrus Link, su objetivo era desarrollar un protocolo que requiriera un mínimo uso de batería y ancho de banda para conectar originalmente ductos petroleros a través de enlace satelital [15].

Se plantearon una serie de objetivos que el protocolo debía de cumplir el nuevo protocolo:

- Simple de implementar
- Proveer un sistema de calidad de entrega de los datos
- Ligero y eficiente uso del ancho de banda
- Agnóstico de datos
- Sesión continua

MQTT es un protocolo orientado a los datos en vez de documentos, ya que transfiere los datos como arrays de bytes lo que lo hace mucho más ligero que HTTP, esto es relevante para nuestra aplicación ya que significa menor uso de batería y menor uso de ancho de banda.

Otra ventaja de MQTT es que se basa en un modelo PUB-SUB, es decir los dispositivos se pueden suscribir a uno o varios “canales” para recibir los mensajes que sean enviados a este canal o bien pueden publicar a un “canal” que será escuchado por otros dispositivos suscritos a dicho canal, lo que hace posible una comunicación bidireccional eficiente y sencilla.

Para entender mejor el funcionamiento de este protocolo es necesario conocer en que consiste el modelo de comunicación en que se basa, nos referimos al modelo publicar/suscribir (pub/sub).

El patrón publicar/suscribir (pub/sub) es diferente al modelo típico cliente-servidor en donde un cliente se comunica directamente con un servidor con un enlace bidireccional, en contraparte el patrón pub/sub desacopla al cliente que está enviando un mensaje particular (publicador) de otro u otros clientes que reciben el mensaje (suscriptores), esto significa que el publicador y el suscriptor no necesitan conocer de la existencia del otro, sin embargo se tiene un tercer componente que se conoce como “broker ” que es

conocido tanto por el publicador como por el suscriptor, su tarea es filtrar todos los mensajes que pasan por él, distribuirlos hacia su destino final correspondiente.

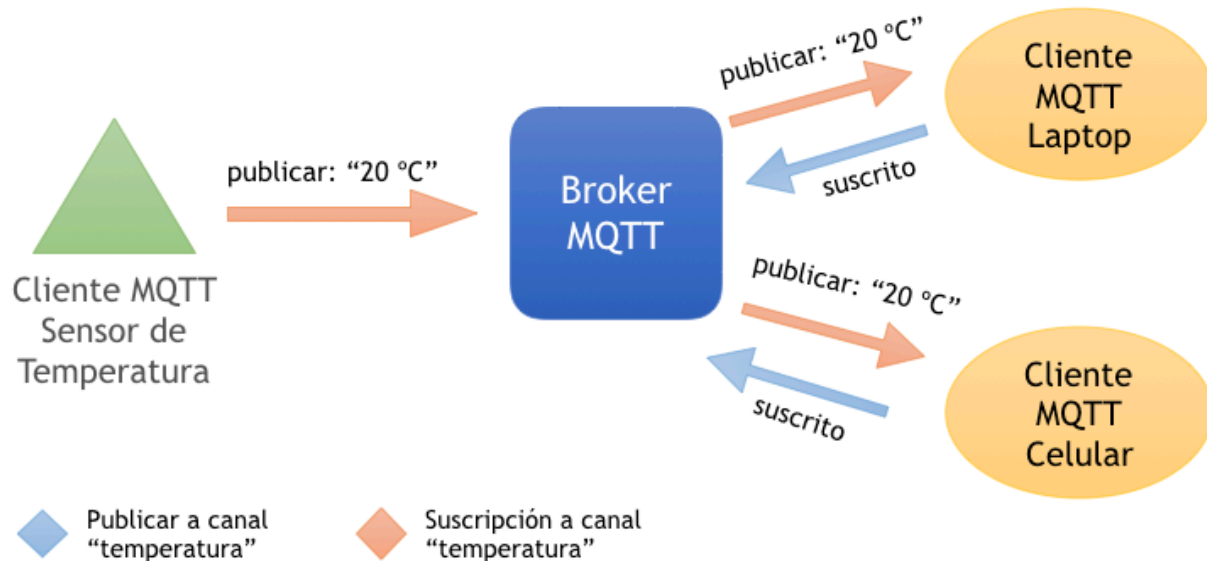


Figura 13 Patrón Publicar/Suscribir MQTT

En resumen, el concepto central del patrón pub/sub es que se desacopla el publicador del receptor del mensaje lo que da ciertas ventajas a este patrón de comunicación, este desacoplamiento tiene 3 dimensiones que se conocen como: Espacio, Tiempo y Sincronización.

- Desacoplamiento de Espacio: Se refiere a que el publicador y el suscriptor no requieren conocerse entre ellos.
- Desacoplamiento de Tiempo: El publicador y el suscriptor no necesitan ejecutarse al mismo tiempo.
- Desacoplamiento de Sincronización: Las operaciones entre los componentes no son interrumpidas durante la publicación o recepción de un mensaje.

Escalabilidad

Otra ventaja del patrón pub/sub es que provee una gran capacidad de escalamiento sobre el modelo tradicional cliente servidor, esto se debe a que las operaciones del broker pueden ser paralelizadas y procesadas bajo eventos, sin embargo, se puede convertir en un reto escalar el patrón pub/sub para millones de conexiones concurrentes pero es posible hacerlo, para esto se pueden usar clusters de brokers para distribuir la carga bajo más servidores individuales y balanceadores de carga.

Los mensajes MQTT tienen un tamaño reducido puesto que su cabecera es de solo 2 bytes y en formato binario, lo que le permite a dispositivos con recursos computacionales limitados manejar el protocolo MQTT sin problemas.

MQTT soporta 3 niveles QOS lo que permite elegir la calidad del servicio, pudiendo elegir entre una transición veloz de datos en el nivel inferior o bien una transmisión menos rápida, pero asegurando la integridad total de los mensajes enviados, HTTP no soporta algo parecido.

Ahora describiremos algunos conceptos básicos sobre el protocolo MQTT.

Cliente

En el protocolo MQTT un cliente es aquel componente que solo puede publicar mensajes, suscribirse para recibir mensajes o bien tener ambos papeles al mismo tiempo ya que el protocolo lo permite, en términos reales un cliente MQTT puede ser desde un microcontrolador hasta un servidor con grandes recursos de computo, siempre y cuando tenga la librería MQTT corriendo y esté conectado a un broker MQTT a través de cualquier tipo de red, cabe destacar que la implementación de un cliente MQTT es muy sencilla y reducida a ser funcional lo que la hace ideal para funcionar en pequeños dispositivos con recursos limitados.

Broker

El Broker es el núcleo del patrón pub/sub del protocolo MQTT, dependiendo de la implementación del broker este puede ser capaz de manejar miles de clientes concurrentes, su función principal es recibir todos los mensajes de los clientes conectados a él, filtrar los mensajes y decidir que clientes son los que están suscritos al tema del mensaje y enviarlos correspondientemente.

El broker tiene la responsabilidad de mantener la sesión de todos los clientes conectados incluyendo las suscripciones y mensajes perdidos, el broker también tiene la función de autenticar y autorizar a los clientes, para brindar seguridad al protocolo, el Broker es también flexible para poder expandir sus funciones lo que permite poder integrar sistemas personalizados de autenticación y su integración con otras aplicaciones.

Por estas razones es que se eligió MQTT como el protocolo que se usa para la comunicación entre nuestro hardware de monitorización y la plataforma web desarrollada.

3.2 Aplicación Web

En sus inicios la web solo consistía en una serie de páginas con contenido estático, es decir contenido que solo era posible leerlo o descargarlo, pero no modificarlo ni interactuar con la página web, es decir la persona que deseaba publicar algo en la web lo tenía que hacer a través de un documento HTML (Hypertextual Markup Language) y subirlo a un servidor web para ser consultado a través de una URL específica, para esto el navegador web tiene la tarea de interpretar el documento HTML para mostrar el contenido de forma gráfica entendible para los humanos, de esta forma los usuarios con acceso a dicha URL serían capaces de leer el texto u observar las imágenes que el autor hubiera integrado en el documento, esto desde cualquier parte del mundo, sin duda una gran revolución para su tiempo, sin embargo si lo comparamos con la web del día de hoy sin duda podemos decir que este modelo se encontraba muy limitado, ya que la web prácticamente era un canal de comunicación uni-direccional como otros medios de comunicación por ejemplo la televisión o la radio, si el autor del documento HTML

deseaba cambiar el contenido entonces tenía que realizar un proceso un poco complejo, modificar su documento HTML y remplazarlo en el servidor web.

Esta web estática fue evolucionando poco a poco hasta convertirse en la web moderna que conocemos hoy en día, los primeros pasos para lograr una web dinámica fue la inclusión de un método para hacer las paginas dinámicas permitiendo interactuar con las paginas desplegadas es decir técnicamente que están fuera generadas en base a la petición que realizara el usuario, este método fue nombrado CGI (Common Gateway Interface) que tenía la tarea de procesar las peticiones HTTP hacia aplicaciones externas del servidor web pero que corrían en la misma computadora, sin embargo CGIs tenían un problema y era que cada que se recibía una petición el servidor web iniciaba un nuevo proceso para correr el programa CGI y adicionalmente como los CGIs eran comúnmente escritos en lenguajes interpretados o en un lenguaje que requería de un entorno de ejecución, esto representaba una pesada carga para los servidores [16].

Hoy en día CGIs aún son usado a pesar de existir tecnologías más eficientes como las que describiremos a continuación, esto debido a que los CGIs son de fácil implementación y además son ampliamente soportados por la mayoría de los servidores web.

Debido a los problemas de CGI, se desarrollaron otras alternativas más eficientes, una de ellas fue el desarrollo de módulos que estuvieran mejor integrados a los servidores web con el fin de evitar el inicio y ejecución de múltiples procesos por cada petición HTTP, la segunda solución, fue equipar a los servidores web con un intérprete de un lenguaje de programación específico, por ejemplo PHP o VBScript permitiendo incluir dentro del documento HTML código que pudiera ser ejecutado por el servidor web y de esta forma reduciendo el tiempo de respuesta y aumentando el desempeño.

La solución más usada hasta el día de hoy para desarrollar aplicaciones web robustas, es la combinación de las dos alternativas explicadas en el párrafo anterior, es decir un lenguaje de programación integrado que le permitan al servidor web entender y ejecutar código incrustado en los documentos HTML y un sistema que permita la ejecución de programas externos pero mucho mejor integrados con el servidor para evitar problemas de desempeño, estas soluciones modernas han permitido el desarrollo de robustas aplicaciones web que hoy en día son parte de nuestras vidas y que han revolucionado muchas actividades desde comunicarnos, pasando por comprar, hasta la forma en que nos entretenemos, todo esto y mucho mas hoy lo podemos hacer a través de aplicaciones web.

Sin embargo esto tiene una explicación si nos referimos a las ventajas que tienen las aplicaciones web, entre ellas podemos encontrar que son multiusuario, es decir que una aplicación web puede ser usada por cientos e incluso millones de usuarios al mismo tiempo de acuerdo a las capacidades del servidor web que la contenga, esta es una gran ventaja ya que se elimina el problema de distribución del software, ya que la aplicación web podrá ser usada por cualquier usuario que tenga acceso a una computadora con un navegador web y con conexión a internet, sin necesidad de instalar ningún componente adicional.

Otra ventaja es que una aplicación web siempre estará actualizada puesto que solo basta con actualizar o modificar la aplicación en el servidor que la contiene para que cualquier usuario que acceda a ella pueda correr la última versión.

Además una aplicación web es accesible desde cualquier parte del mundo, el único requisito es tener acceso a un ordenador con un navegador web y una conexión a internet, teniendo al alcance de forma

remota no solo las funciones de la aplicación si no también los datos almacenados en la aplicación, siempre disponibles mientras el servidor que contenga la aplicación se encuentre operativo y si consideramos que hoy en día la mayoría de los servidores están diseñados para estar operativos por largos periodos de tiempo podemos hablar de que una aplicación web puede estar disponible 24/7.

Finalmente una aplicación web tiene la ventaja de ser escalable, es decir que si la aplicación crece en número de usuarios se puede seguir dando servicio a los anteriores y nuevos usuarios solo creciendo la infraestructura de servidores y añadiendo sistemas que permitan el crecimiento en el uso de la aplicación web conservando los datos y funciones de la aplicación original, esto sin duda es una gran ventaja ya que implica un ahorro de recursos en las fases iniciales del desarrollo de una aplicación web y creciendo los recursos conforme aumenten el número de usuarios activos de la aplicación.

3.2.1 Lenguajes comunes en la programación de aplicaciones web

Como se mencionó en párrafos anteriores, en las aplicaciones web modernas ya no se hace uso de CGI's, a su vez ahora se hace uso de lenguajes de programación dentro de los documentos HTML que pueden ser interpretados por un servidor web y lenguajes de programación que son ejecutados externamente del servidor web pero se pueden comunicar de forma eficiente, a continuación describiremos algunos de estos lenguajes que hoy en día son usados para escribir aplicaciones web modernas.

PHP

Es un lenguaje de programación del lado del servidor tipo script, es decir es un lenguaje interpretado a diferencia de los lenguajes compilados no requiere un proceso de compilación para poder correr ya que estos son interpretados línea por línea durante la ejecución, sin embargo esto puede conllevar un costo en torno a la velocidad del programa, sin embargo es compensado por otras ventajas como tener una sintaxis más limpia y fácil de entender.

PHP fue diseñado pensando en la web, es por eso que el código PHP se puede insertar dentro de los documentos HTML, este código es interpretado por el servidor web cada vez que la página web es cargada o bien una de sus funciones llamada por un método HTTP, a su vez el servidor web genera una salida en base lo solicitado por el cliente [17].

Para poder ejecutar PHP se requiere tener instalado un intérprete del lenguaje en el servidor web donde se alojará la aplicación web, PHP está disponible para diferentes sistemas operativos, como Linux, Windows y Mac OS.

PHP es un lenguaje de código abierto originalmente fue escrito por Rasmus Ledorf en 1994, posteriormente fue ampliamente adoptado por muchos desarrolladores de alrededor del mundo los cuales han contribuido al desarrollo de este proyecto open source, el cual es mantenido por una comunidad muy activa alrededor del mundo.

Es un lenguaje ampliamente usando, para junio de 2019 se estimaba que el 79% de los sitios web del mundo usaban PHP [18].

Algunas características que hacen de PHP una buen opción para el desarrollo de aplicaciones es que cuenta con soporte nativo de diferentes tipos de bases de datos tanto open source como comerciales, como MySQL, PostgreSQL, Oracle, MongoDB, SQLite y usando el estándar ODBC es posible conectarse a bases de datos que soportan este estándar como son algunas bases de datos de Microsoft, también PHP cuenta con una serie de librerías nativas especialmente orientadas a la web que permiten implementar algunas funciones complejas con unas pocas líneas de código, por ejemplo la conexión a servicios web, parse XML y JSON, envío de email, cookies, entre otros.

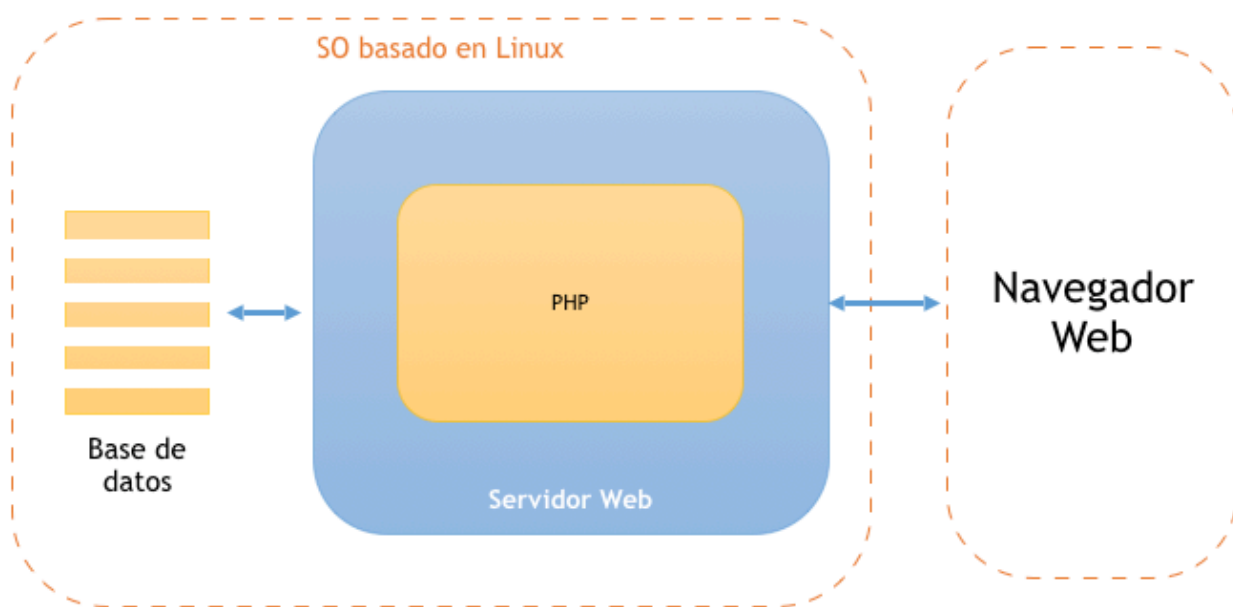


Figura 14 Arquitectura de una aplicación web LAMP (Linux, Apache, Mysql, PHP)

Python

Este lenguaje fue desarrollado por Guido Van Rossum en 1989 mientras trabajaba en para el Centro de Matemáticas en Amsterdam, antes Guido había sido uno de los desarrollares de ABC, un lenguaje considerado impopular entre desarrolladores ya que era poco flexible para crecer y su sintaxis era atípica.

Conociendo estas debilidades de ABC, durante las vacaciones de Navidad decidió desarrollar un lenguaje que solventara los problemas de ABC, buscando un lenguaje fácil de entender y eliminando la sintaxis innecesaria.

Pensando en atraer a los hackers de UNIX/C que utilizaban ABC para desarrollar extensas pruebas de usabilidad además de buscar que su nuevo lenguaje fuera adecuado tanto para programadores principiantes como para los más experimentados, entre otras características Python heredo su conocida estructura de bloques identados del lenguaje ABC. Tiempo después Python gano popularidad entre desarrolladores alrededor del mundo y fue usado por pequeñas y grandes organizaciones que su vez se

convirtieron en parte de la comunidad que ayuda desarrollar y mantener Python, lo que lo ha convertido en lenguaje adecuado para organizaciones de diferentes tamaños y requerimientos [19].

Algunas características de Python son las siguientes:

Lenguaje Interpretado.

Python es un lenguaje interpretado al igual que PHP, por lo cual no se requiere de un compilador lo que hace sencillo desplegar un programa escrito en Python, sin embargo su interpretación tiene algunas ventajas frente a otros lenguajes tipo scripting entre ellas se encuentra que el intérprete si tiene los permisos necesarios producirá un archivo compilado para los módulos importados lo que resulta en un incremento en la velocidad en que las librerías son cargadas y ejecutadas, esto no aplica para el programa principal.

Operaciones Interactivas.

El intérprete de Python es interactivo, es posible trabajar con Python sin necesidad de un archivo ya que nos proporciona una consola donde podemos tener acceso ejecutar expresiones, líneas o bloques de código y tendremos una salida a las expresiones que se ejecuten e incluso los bloques de código pueden crear definiciones de clases y funciones que podemos usar de forma interactiva, esta característica es muy útil para poder probar nuevo código en tiempo real.

Representaciones de datos flexibles.

Python cuenta con un amplio y flexible conjunto de tipos de datos, por ejemplo, los números pueden ser enteros, reales o flotantes las cadenas pueden estar disponibles en ASCII o Unicode.

En Python existen tipos de datos particulares como lo son los diccionarios, tuplas y listas, que son tipos de datos que nos permiten almacenar diferentes tipos de datos en una misma variable y de forma organizada.

Parte de la flexibilidad que brinda Python es que las listas pueden crecer o decrecer de forma dinámica y los diccionarios permiten la búsqueda flexible de cualquier tipo de dato.

También en Python las variables no tienen que ser declaradas, son escritas dinámicamente en vez de estáticas.

Orientado a objetos.

Los tipos de datos complejos en Python como lo son los diccionarios, son objetos, es decir que comparten operaciones y propiedades comunes.

Amplio rango de funciones predefinidas y métodos.

Python incluye diversas funciones predefinidas que pueden ahorrar considerable tiempo de desarrollo para ejecutar tareas comunes pero que pueden llegar a ser un tanto complejas. Algunas de estas funciones están asociadas con un tipo de dato en particular que son implementadas como métodos del tipo de dato. Por ejemplo, puedes ordenar listas, aplicar funciones sobre secuencias, obtener el tipo de dato de un valor ligado a una variable en específico y otras funciones más.

Soporte de librerías.

Python incluye más de 150 módulos para ejecutar diferentes tipos de tareas, por ejemplo, podemos encontrar módulos que nos brinde soporte para HTML, una red de sockets, creación y manipulación de archivos comprimidos como ZIP, manejo de bases de datos y muchas funciones más, incluso al contar con una gran comunidad de desarrolladores se pueden encontrar cada día nuevas librerías contribuidas por la comunidad a través de repositorios públicos.

JAVA

Java es uno de los lenguajes más usados hasta el día de hoy, fue desarrollado por un pequeño equipo de Sun Microsystems denominado “Green Team” liderado por James Gosling, el objetivo inicial era desarrollar un lenguaje de programación multiplataforma es decir que pudiera ser portado fácilmente hacia otras plataformas independientemente del entorno donde fue escrito el programa, incluso el lema de James Gosling con el que definía los objetivos del lenguaje era “Write Once, Run Anywhere” (Escribe una vez, ejecuta en cualquier lugar), para lograr esto además del lenguaje Java se desarrolló un entorno de ejecución JVM (Máquina Virtual Java) disponible para diversas plataformas, la función de JVM es ejecutar el bytecode Java sin importar donde fue compilado y siempre generando el mismo resultado sin adaptaciones para cada plataforma [20].

En 1995 Sun Microsystems hizo pública la primera implementación de esta plataforma conocida como Java 1.0a, sin embargo, no fue hasta 1996 cuando se lanzó la versión 1.1 con soporte para diversas plataformas como Linux, Solaris, Mac y Windows, desde entonces Java ha continuado su desarrollo integrando constantemente nuevas características y mejoras al entorno de desarrollo, hoy en día Java lo podemos encontrar en diferentes tipos de implementaciones como aplicación independiente, aplicación web, aplicaciones corporativas, aplicaciones móviles y en diferentes lugares como consolas de videojuegos, electrodomésticos, sistemas embebidos, robots, computadores de escritorio, celulares, etc.

Sin embargo, lo que nos concierne en este trabajo es la orientación de Java al desarrollo de aplicaciones web, para ello existe la versión Java EE orientada a servidores y que contiene una serie de módulos destinados para el desarrollo de este tipo de aplicaciones como son los Servlets que en resumen se encargan de generar respuestas a peticiones realizadas por un cliente a través del protocolo HTTP.

En general Java EE está enfocado en el desarrollo de aplicaciones corporativas previendo un modelo para implementar aplicaciones de múltiples capas, es decir en donde las funciones de la aplicación se encuentran separadas en diferentes áreas que se conocen como niveles “tiers” [21].

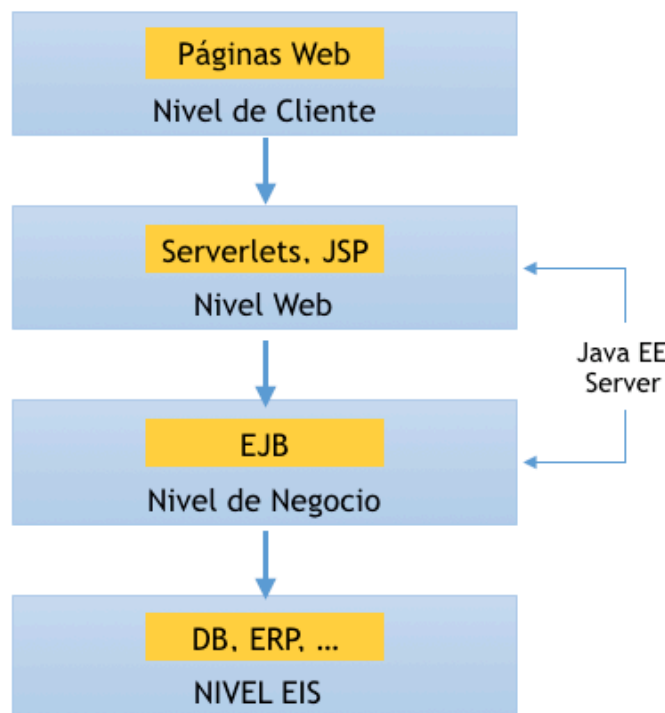


Figura 15 JAVA - Arquitectura Multinivel

El nivel de cliente.

El nivel de cliente es el nivel más alto en la arquitectura de Java EE, básicamente consiste en aplicaciones clientes que hacen peticiones al servidor Java EE que puede encontrarse localmente o remotamente (que es lo más común), al recibir la petición el servidor procesa la petición y envía una respuesta al cliente, el cliente puede ser una aplicación independiente o bien un navegador web.

El nivel Web.

El nivel web consiste en una serie de componentes del servidor Java EE que se encargan de manejar la interacción entre un cliente y la lógica de la aplicación, al recibir una petición de un cliente el nivel web realiza una serie de procesos para brindar una respuesta al mismo cliente, primero captura la entrada enviada por el cliente, controla el flujo de pantallas o páginas del cliente, mantiene los datos del cliente como una sesión, procesa una respuesta para el usuario en base a la lógica de la aplicación y finalmente genera contenido dinámico para devolver al usuario.

El nivel de Negocio.

Este nivel consiste una capa que contiene componentes que se encargan de solucionar específicamente la lógica de un tipo de negocio específico, por ejemplo bancos, retail, finanzas, etc, es manejada por “enterprise beans” ya sea en el nivel de negocio o en el nivel web. Un “Enterprise Bean” obtiene datos desde el almacenamiento y los procesa si es necesario para posteriormente enviar de vuelta al nivel de cliente.

El nivel EIS (Enterprise Information System Tier).

Este nivel tiene la tarea de manejar el software EIS e incluye sistemas para el manejo de infraestructura corporativa como puede ser un ERP (Enterprise resource planning), sistemas de bases de datos y otros sistemas heredados de información.

ASP.NET

ASP es el acrónimo de Active Server Pages que es una tecnología desarrollada por Microsoft para el desarrollo de páginas web dinámicas y en combinación con .NET el cual es un framework para el desarrollo de todo tipo de aplicaciones tanto de escritorio como de servidor, conforman ASP.NET una plataforma de desarrollo enfocada a la creación de Web Services que permiten crear robustas aplicaciones web bajo un enfoque orientado a objetos [22].

Unas de las novedades de ASP.NET a comparación de ASP puro es como el código es interpretado, versiones anteriores de aplicaciones ASP eran escritas con un lenguaje tipo script que era interpretado en tiempo de rutina, sin embargo ahora las aplicaciones ASP.NET son aplicaciones compiladas, esto además de ser aplicaciones más eficientes, representa una ventaja para los desarrolladores la cual es que solo deben desplegar los archivos compilados al servidor donde correrá la aplicación en vez de todo el código fuente, lo cual hace más sencillo y ligero el despliegue en producción, además de que representa una ventaja comercial puesto que los desarrolladores pueden vender las aplicaciones y ponerlas en producción en los servidores de sus clientes sin tener que ceder su código fuente original.

ASP.NET usa un protocolo llamado SOAP para interactuar con los web services, la principal característica de SOAP es que usa XML como el formato para representar los objetos, propiedades y datos que son enviados y recibidos por los web services, lo cual hace posible no solo recibir peticiones de clientes a través de navegadores web si no también poder comunicarse con otras aplicaciones de forma directa.

Para desarrollar aplicaciones en ASP.NET se usa el lenguaje C# el cual es un lenguaje desarrollado por Microsoft para el entorno Windows, la sintaxis de C# es similar y robusta como la de C y C++, sin embargo ha sido simplificada para ayudar a los desarrolladores a construir aplicaciones complejas en un corto periodo de tiempo, C# es totalmente orientado objetos y además arregla algunas de las complejidades de C++, como eliminar la necesidad de inicializar las variables puesto que estas son automáticamente inicializadas en 0 y las variable tipo string como una cadena vacía, otra mejora es que se eliminan los archivos de cabeceras que eran un problema mantenerlos en C++ debido al surgimiento del problema de referencia circular que podía surgir en proyectos extensos.

Otra ventaja de ASP.NET es que tiene soporte nativo para manejar eventos, es decir cuando un usuario realiza alguna acción en el lado del cliente por ejemplo presionar un botón, un evento es disparado y se ejecuta su código en el servidor, lo cual hace posible el desarrollo de aplicaciones más dinámicas y por lo tanto más cercanas al desarrollo de aplicaciones de escritorio.

NODE.JS

Node.js no es un lenguaje de programación, más bien es una tecnología del lado del servidor para desarrollo de aplicaciones web, basada en el motor de Javascript V8 de Chrome, lo cual por si mismo significa que Node.js nos permite escribir el backend de una aplicación web en el mismo lenguaje que se usa en el frontend nos referimos a Javascript, lo cual por si mismo represento una gran noticia para los desarrollares cuando fue anunciado en 2009 por Ryan Dahl, pero esto no es lo único interesante de Node.js, otro punto importante a destacar es que está basado en un modelo I/O (entrada salida) no bloqueante [23].

Node.js es diferente a otras tecnologías de desarrollo de aplicaciones web debido a 2 características que explicaremos a continuación, estas características de acuerdo al tipo de aplicación puede representar una ventaja frente a otras tecnologías, la primera es que Node.js es manejando por eventos “event driven” y por lo tanto funciona de forma asíncrona, esto quiere decir que a diferencia de la arquitectura tradicional donde el código de las aplicaciones se ejecuta de forma secuencial del inicio al final del código, en Node.js la ejecución del código responde a eventos que pueden ser acciones o peticiones que realiza el cliente o bien evento activados por otra aplicación, para lograr esto, el núcleo de Node.js tiene un mecanismo que monitorea los procesos en busca de eventos y llamadas a funciones así como su callback para devolver una respuesta tan pronto se termine el proceso en cuestión, esto es relevante puesto que nos permite construir aplicaciones en donde es primordial obtener una respuesta tan pronto se solicita y poder manejar esa respuesta tan pronto esté disponible, el tipo de aplicaciones que funcionan cercanas al tiempo real, como podría ser un chat, notificaciones push, juegos, etc.

Como se mencionó previamente Node.js es I/O (input/output) no bloqueante, esto quiere decir que puede ejecutar operaciones de entrada/salida sin tener que esperar a que termine una operación para poder continuar con la siguiente que es lo que sucede con otras tecnologías, para lograr esto Node.js posee un manejador de procesos I/O llamado Event Loop, a diferencia de otras tecnologías que son multi hilo en donde se crea un proceso nuevo por cada cliente conectado a la aplicación y donde se destinan recursos virtuales para que cada proceso pueda ejecutarse, Node.js es de un solo hilo es decir que el Event Loop se encarga de manejar todas las entradas y salidas de todos los clientes de la aplicación para que estos sean procesados y las repuestas lleguen al cliente adecuado cada vez, es por esta razón que se evita el bloqueo de la aplicación ya que a pesar de que el evento loop es un solo hilo de procesamiento este solo se encarga de recibir las peticiones y canalizarlas hacia otros hilos de trabajo en un nivel más bajo y son ellos los que ejecutan la tarea solicitada (puede ser leer un archivo, una consulta a base de datos, un procesamiento matemático, etc.) mientras el Event Loop se encarga de manejar otras peticiones y respuestas, una vez que la tarea esta lista el hilo de trabajo encargado del proceso notifica al Event loop y este devuelve el resultado al cliente que solicito el proceso.

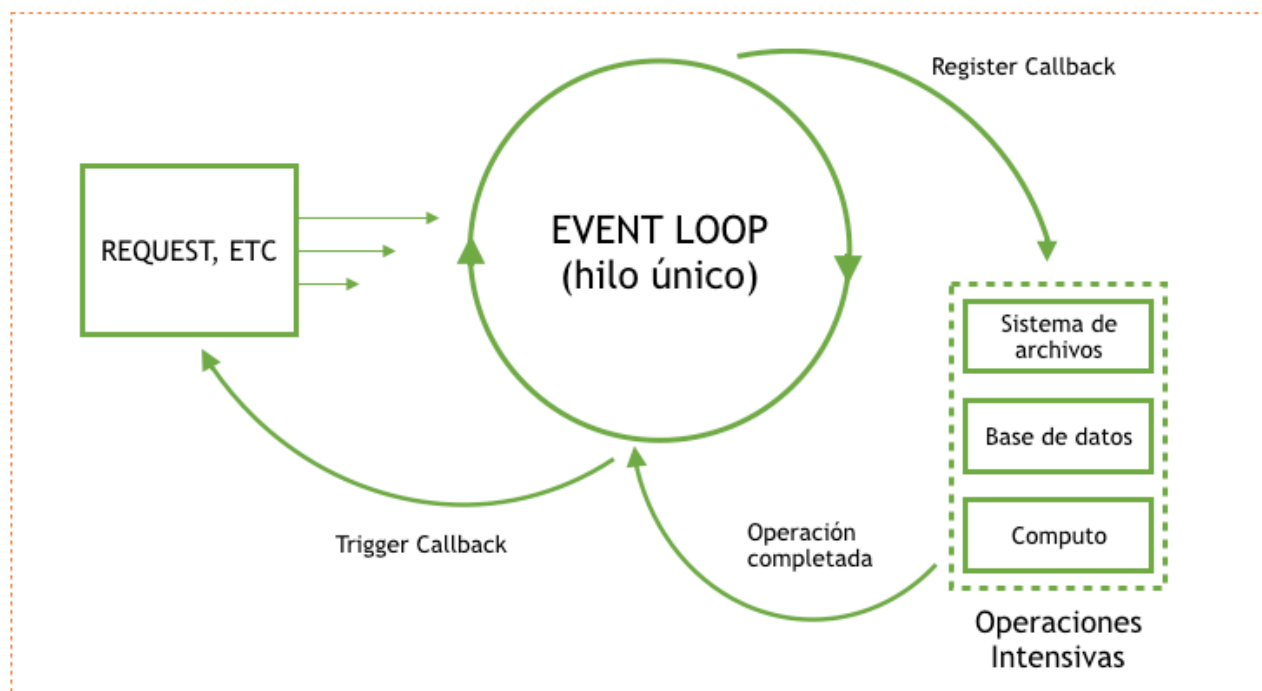


Figura 16 Arquitectura y flujo de procesos en NODE.JS

3.2.2 Elección de lenguaje de programación para el desarrollo de la plataforma web de monitorización.

Para el desarrollo de la plataforma web de monitorización se optó por utilizar Python como lenguaje de programación del lado del servidor, esta decisión se basó principalmente en que se trata de un lenguaje orientado a objetos, flexible, con sintaxis simple y por lo tanto con una curva de aprendizaje menor a otros lenguajes, en cuestión de rendimiento se encuentra en un punto medio entre un lenguaje compilado (como JAVA) y un lenguaje totalmente interpretado tipo scripting como PHP, es decir se obtiene la ventaja de escribir aplicaciones en un menor tiempo sin sacrificar demasiado el rendimiento de la aplicación, otro punto a favor de Python es que cuenta con una gran comunidad que se encuentra constantemente mejorando el lenguaje y brindando soporte a nuevas funciones a través de módulos y frameworks que facilitan el desarrollo de aplicaciones sobre Python, uno de estos frameworks es Django que provee un andamiaje para el desarrollo de aplicaciones web, es decir aplicaciones Cliente-Servidor basadas en el protocolo HTTP, en párrafos siguientes que se describirá con más detalle este framework.

3.2.2 DJANGO (Framework para desarrollo de aplicaciones web)

Django es un Framework para el desarrollo de aplicaciones web basado en Python, según su sitio oficial se describe como “Django es un web framework de alto nivel que alienta el desarrollo limpio y veloz con un diseño pragmático. Desarrollado por expertos, toma en cuenta gran parte de lo molesto que tiene el desarrollo web, para que puedas enfocarte en escribir tu aplicación sin tener la necesidad de reinventar la rueda, es gratuito y código abierto”

Por lo cual Django es un conjunto de módulos prefabricados que contienen lo necesario para el desarrollo de una aplicación web, es decir brinda funciones necesarias que toda aplicación web usa evitando que se tenga desarrollar estas funciones que son comunes entre aplicaciones desde cero, ahorrando trabajo, evitando errores y aumentando la seguridad de la aplicación web en general al usar métodos que ya fueron probados por miles de usuarios y con actualizaciones constantes de seguridad.

Al ser un Framework para el desarrollo de aplicaciones web, muchos de sus módulos se centran en proveer lo necesario para poder desarrollar bajo el protocolo HTTP, esto incluye manejo de rutas (URLS), formularios, templates HTML, interprete de métodos HTTP como GET y POST, además de otros elementos que hacen posible el desarrollo de aplicaciones de alto nivel orientadas a objetos para esto Django hace uso de la arquitectura Modelo Vista Controlador (MVC) que básicamente provee un andamiaje para desarrollar siguiendo este patrón que permite obtener una aplicación eficiente, escalable y fácil de mantener a largo plazo, más adelante se abordará con más detalle este importante patrón de desarrollo de aplicaciones [24].

Algunas de las ventajas de usar Django son:

Funciones incluidas.

Incluye muchas funciones precargadas de funciones comunes en el desarrollo web que permiten ahorrar trabajo, por ejemplo sistema y autenticación de usuarios, administrador de contenidos, mapas de sitio.

Seguridad.

Django toma en cuenta y resuelve muchos de los problemas de seguridad comunes en aplicaciones web, además de ayudar a los desarrolladores evitar muchos de los errores comunes de seguridad como inyección SQL, cross-site scripting, cross-site request y clickjacking, además de que el sistema de usuario incluido es una forma segura de manejar a los usuarios y passwords.

Escalable.

Django fue desarrollado pensando en la eficiencia, actualmente muchos de los sitios más concurridos en internet usando Django, logrando hacer frente de forma rápida y flexible a grandes cargas de tráfico.

Versátil.

Muchas compañías, organizaciones y gobiernos han usado Django para construir todo tipo de aplicaciones, desde sistemas de gestión de contenido, redes sociales hasta plataformas de cómputo científico.

3.2.3 MVC (Modelo Vista Controlador)

Como se mencionó anteriormente para el desarrollo de esta aplicación se realizará bajo el patrón de desarrollo de software MVC (Modelo Vista Controlador) en el cual se basa Django.

El patrón de diseño Modelo Vista Controlador fue desarrollado en 1979 en el Xerox PARC por Trague Reenskaug, desde entonces se ha convertido en el estándar para el desarrollo de aplicaciones con una GUI (Graphical User Interface), se basa en el concepto de que una aplicación está compuesta por un modelo de datos, un control de la información y una presentación de la información, cada uno de estos elementos deben de estar separados en diferentes objetos [25].

A continuación, se describen con detalle estos 3 elementos que conforman el patrón de diseño MVC:

Modelo.

Es la representación de la información sobre la que la aplicación funciona y también es la lógica de la aplicación es decir el elemento que le da sentido a los datos, se encarga de estructurar los datos en objetos que tengan significado para el usuario, así como realizar operaciones con los mismos datos, por ejemplo, calcular un saldo, cálculo de inventarios, etc.

Muchas veces se confunde el modelo con el almacenamiento de datos que puede ser una base de datos o algún otro medio de almacenamiento persistente, sin embargo, no debe ser entendido de esta forma ya que el patrón MVC no especifica un medio específico para almacenamiento de los datos, si no que este se encuentra implícito y encapsulado en el modelo.

Vista.

Se encarga de desplegar el modelo en un elemento adecuado para su interacción con los usuarios, comúnmente es una interfaz, sin embargo, no se debe entender la vista como la parte visible gráficamente al usuario sino más bien como el elemento que se encarga de estructurar y procesar los datos que se reciben del modelo para ser enviados a la interfaz gráfica.

Controlador.

Este elemento del patrón MVC se encarga de responder a eventos que regularmente son acciones del usuario sobre la interfaz, el controlador se encarga de manejar estas interacciones de los usuarios hacia el modelo o la vista según corresponda.

Ventajas del patrón MVC

Soporte multi-vista.

Debido a que la vista se encuentra separada del modelo y no existe una dependencia directa del modelo hacia la vista, la interfaz de usuario puede desplegar múltiples vistas con los mismos datos al mismo tiempo.

Persistente al cambio.

Regularmente los requerimientos de una interfaz de usuario tienden a cambiar más rápidamente que las bases de la aplicación, debido a que el modelo no depende de las vistas es posible agregar nuevas vistas a la aplicación sin afectar al modelo, por lo cual los cambios frecuentemente pueden limitarse al nivel de la vista.

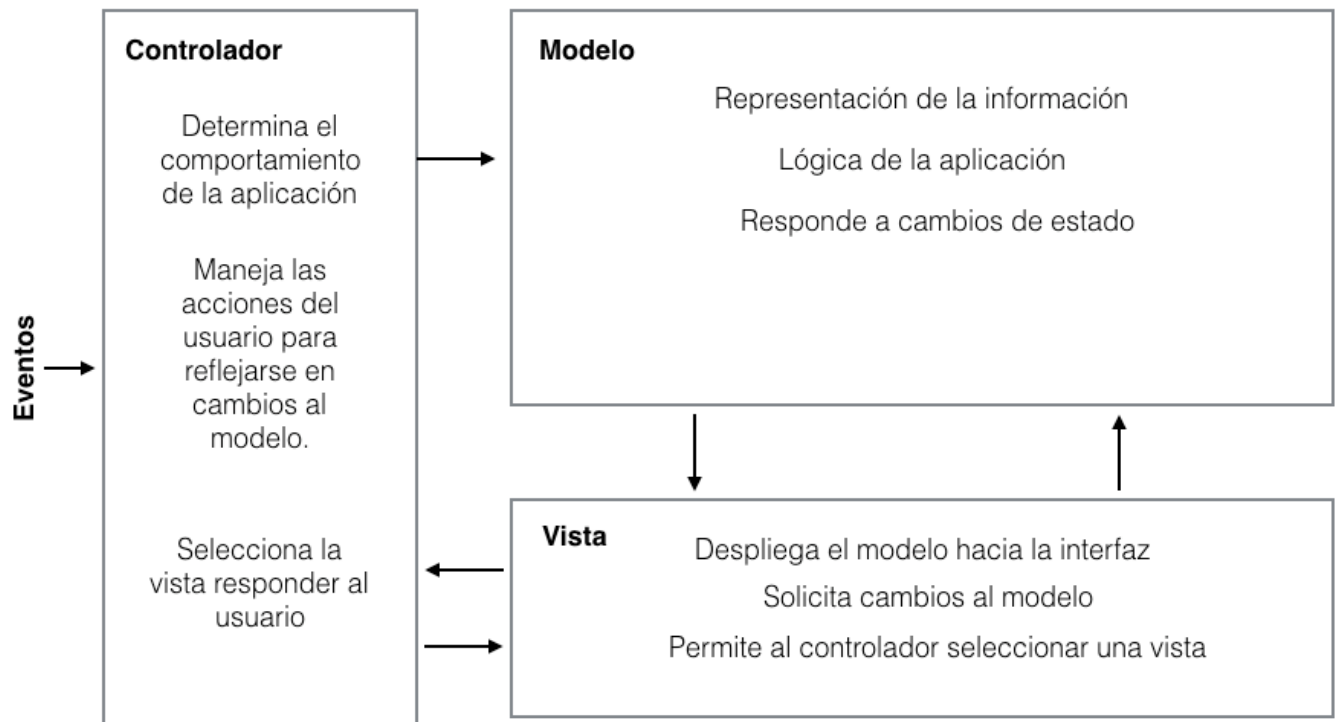


Figura 17 Patrón Modelo Vista Controlador

A continuación, se describe el flujo general del patrón MVC:

1. El usuario dispara un evento interactuando con la interfaz de usuario, por ejemplo envía un formulario, presiona un botón, etc.
2. El controlador interpreta el evento disparado por el usuario a través de la interfaz de usuario.
3. El controlador accede al modelo y ejecuta la acción apropiada que dé respuesta a la acción del usuario, por ejemplo, actualizar el modelo, obtener un dato en específico, etc.
4. Una vista usa la notificación del modelo para generar una interfaz de usuario adecuada a la ejecución del modelo, así mismo obtiene los datos correspondientes desde el modelo para mostrarlos al usuario.
5. Se actualiza la interfaz del usuario y queda en espera de nuevas interacciones por parte del usuario para iniciar un nuevo ciclo.

3.2.4 API REST

REST proviene de la abreviación de Representational State Transfer (en español: Transferencia de Estado Representacional), es un estilo de arquitectura para el desarrollo de servicios web, fue presentado por primera vez por Roy Fielding en el año 2000.

REST es una arquitectura muy usada debido a su simplicidad ya que al encontrarse construida sobre otras tecnologías ya existentes como lo es el protocolo HTTP aprovecha muchos de sus estándares y ventajas, lo cual se traduce en una curva de aprendizaje pequeña para todos aquellos desarrolladores que conozcan el protocolo HTTP.

Otra ventaja de REST es que es una arquitectura que puede ser desarrollada y consumida sobre una gran cantidad de lenguajes de programación la cual la vuelve independiente de las tecnologías usadas, lo único que se requiere es que el lenguaje base sea capaz de interactuar con el protocolo HTTP, de esta forma REST es soportada por una gran variedad de lenguajes como: Python, Ruby, PHP, Java, Javascript, C#, etc.

A continuación, se describe de forma general la arquitectura que debe tener una aplicación REST para ser considerada como tal [26]:

1.-Orientación Cliente-Servidor.

Una aplicación REST solo se concentra en la interacción Cliente-Servidor, delegando la capa de interfaz de usuario a las aplicaciones que consuman el servicio REST, esto permite que REST se concentre en el modelado y almacenamiento de la estructura de datos lo que se refleja en una mayor eficiencia en el manejo de los datos y permite una gran escalabilidad a futuro, además esta orientación permite una gran portabilidad entre múltiples plataformas pudiendo ser utilizada en ambientes web, móviles y de escritorio, todo esto de forma sencilla.

2.-Protocolo sin estado (Stateless).

REST al estar basada sobre el protocolo HTTP mantiene su carácter de utilizar un protocolo sin estado, es decir que cada petición al servidor es independiente de las peticiones previas, de igual forma las respuestas del servidor al cliente solo dependiente de la petición que la genere, esto mantiene la gestión de la sesión de usuario totalmente del lado del cliente.

3.-Cacheable.

Una aplicación REST debe poder ser cacheada, lo que permite que un cliente cache pueda reutilizar una respuesta dada previamente cuando se solicite una petición equivalente, esto permite reducir la carga del servidor y mejorar la eficiencia de la aplicación.

4.-Interfaz Uniforme.

Siguiendo el principio de ingeniería de software de la generalidad a la interfaz del componente que permite obtener una interfaz uniforme, REST está definida por 4 restricciones de la interfaz: identificación de recursos, manipulación de recursos a través de representaciones, mensajes auto-descriptivo e hipertexto como el motor de la arquitectura Stateless.

5.- Sistema basado en capas.

El modelo basado en capas de REST permite desarrollar una arquitectura en donde cada recurso está restringido a interactuar solo con la capa para la que fue diseñado, lo que permite un mejor control y comportamiento de los recursos de la aplicación.

En resumen un API REST es un modelo de arquitectura para crear aplicaciones Cliente-Servidor bajo un estándar conocido y compatible entre múltiples plataformas, lo cual se traduce en una aplicación que pueda ser consumida por cualquier aplicación con interés en nuestra aplicación.

3.2.5 Servidor Web

La aplicación web que se desarrolló debido a su naturaleza, se basa en el protocolo HTTP y por ende en la arquitectura Cliente-Servidor, el servidor puede entenderse de diferentes formas, la más común es referente al hardware computacional en donde se ejecutara la aplicación, lo cual es correcto, sin embargo en estos párrafos la orientación es diferente, por Servidor Web nos referimos al Software que hace posible que nuestra aplicación web este expuesta y disponible de forma persistente a través de Internet es decir basado en el protocolo HTTP [27].

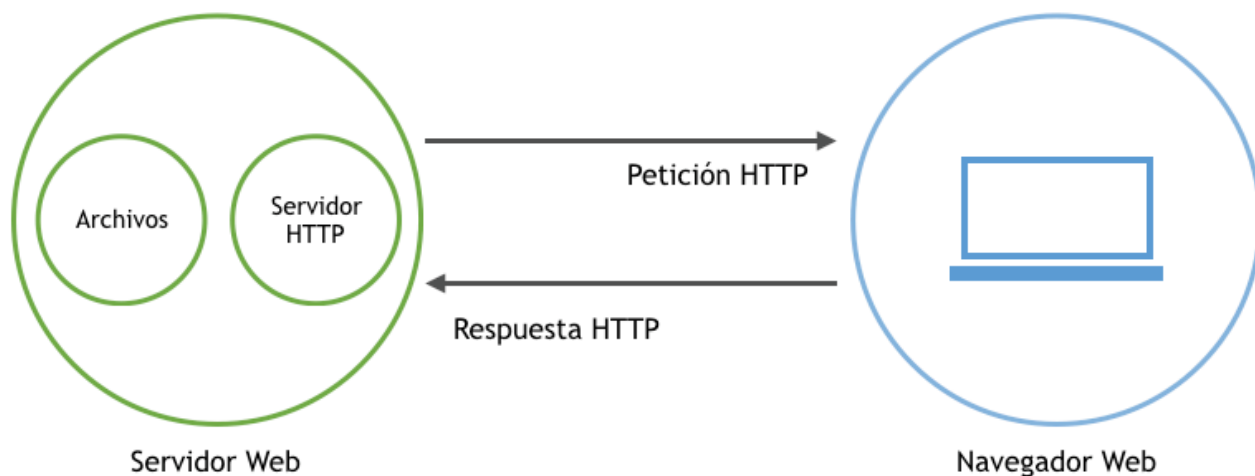


Figura 18 Arquitectura Cliente-Servidor

Existen diferentes software de servidores web, debido a la naturaleza OpenSource de este proyecto y en base a las ventajas que se describen a continuación el servidor web elegido fue NGINX.

NGINX es un software de código abierto para funcionar como servidor web, proxy inverso, cache, balanceador de carga, streaming de contenido multimedia entre otras aplicaciones, fue diseñado para lograr una máxima eficiencia y estabilidad, desde sus inicios la meta principal de NGIX fue crear el servidor web más rápido, hoy en día constantemente supera constantemente a uno de sus principales competidores Apache Web Server en diversas mediciones en el desempeño de servidores web.

Desde la liberación de NGINX la WWW se ha transformado significativamente, de hospedar sitios web con solo contenido HTML estático hacia sitios web dinámicos y con una gran variedad de contenido, es decir contribuyo de forma significativa a la Web que conocemos hoy en día donde prácticamente cualquier sitio es una aplicación web dinámica, lo que se conoce como web 2.0.

Actualmente NGINX es uno de los mejores servidores web para aplicaciones web, es usado por sitios de alto tráfico como Dropbox, Netflix, Zynga y más de 358 millones de sitios web en toda la WWW, incluyendo la mayoría de los 100,000 sitios más visitados de toda la WWW.

3.2.6 Base de datos

Toda aplicación web moderna debe contar con un método para almacenar sus datos de forma persistente, el patrón MVC que se usara para este desarrollo no especifica qué tipo de base de datos se debe usar pero implícitamente considera que el modelo encapsula a la base de datos.

Debido a la naturaleza de la aplicación es necesario usar una base de datos relacional como se describe a continuación.

Base de datos relacional

Una base de datos relacional es un tipo de base de datos que almacena y provee acceso a datos que se encuentran relacionados unos de otros, las bases de datos relacionales se basan en un modelo relacional una forma intuitiva y sencilla de representar datos en tablas, es decir cada fila en una base de datos relacional es un registro con un ID único (Primary Key), las columnas de la tabla almacenan atributos del objeto de datos y cada registro usualmente tiene un valor para cada atributo lo facilita realizar relaciones entre los datos de diferentes tablas, esto permite a las bases de datos relacionales almacenar y consultar grandes cantidades de datos de forma eficiente.

Para trabajar con bases de datos relacionales se utiliza el lenguaje SQL (Structured Query Language), continuación se describen algunos conceptos generales para el diseño y trabajo con bases de datos relacionales.

Llave Primaria (Primary Key)

En un modelo relacional una tabla no puede contener filas duplicadas ya que esto ocasionaría respuestas ambiguas al solicitar datos de la base de datos, para asegurar respuestas únicas cada tabla debe tener una columna que almacene un ID único para cada fila esta columna se le conoce como primary key [28].

Por ejemplo, una columna llamada dispositivo ID puede almacenar los ID únicos de una tabla Dispositivos, en donde cada fila almacenara un “Dispositivo” identificado por un ID único.

Para seleccionar la columna que se asignara como Primary Key se debe tomar en cuenta las siguientes consideraciones:

Los valores deben ser únicos, es decir no debe existir cabida a valores duplicados, por ejemplo, una columna llamada “dispositivo Nombre” que almacene los nombres de dispositivos no sería apropiado utilizarla como Primary Key de la tabla “Dispositivos” ya que los valores de Nombre pueden no ser únicos si dos o más dispositivos se llaman igual.

El Primary Key siempre debe contener un valor es decir nunca puede contener un valor Null.

Los valores del Primary Key no deberían cambiar ya que esta columna será usada para referenciar a otras tablas ya que si se cambian los valores las referencias se pueden perder o incluso ocasionar relaciones erróneas, por lo cual una columna sujeta a cambios no debería ser Primary Key.

El Primary Key se recomienda sea un dato entero ya que esto hará más eficiente las consultas a la base de datos.

El Primary Key se recomienda sea una sola columna aunque puede ser un valor compuesto de diferentes columnas que cumplan los requisitos previamente descritos, sin embargo lo recomendable es mantenerlo único o con el menor número de columnas posible, para aumentar la eficiencia de la base de datos.

Relaciones entre tablas

Una base de datos con una serie de tablas independientes no tendría mucha utilidad ya que sería una colección de datos muy similar a una hoja de cálculo, el verdadero poder de las bases de datos relacionales radica en sus relaciones con otras tablas, por lo cual el punto fundamental en el diseño de una base de datos relacional se encuentra en definir las relaciones entre tablas, a continuación, se describen los tipos de relaciones fundamentales:

One-to-Many.

Es una relación uno a muchos, es decir un elemento representado en una tabla de la base de datos puede relacionarse con varios elementos de otra tabla de la base de datos, para entenderlo mejor podemos imaginar algunas relaciones cotidianas que se adaptan a este modelo, por ejemplo, en una escuela un profesor puede tener asignado diferentes grupos de alumnos, pero un grupo de alumnos solo se encuentra relacionado a un solo grupo de alumnos, esta es una relación “One to Many”.

Many-to-Many.

Este tipo de relación muchos a muchos, es usada para representar relaciones donde un elemento de la base de datos (elementos A) puede estar relacionado a uno o muchos elementos de otra tabla (elementos B) y a la vez este elemento relacionado (elemento B) puede estar relacionado a otros elementos (elemento A), para entender mejor este tipo de relación podemos tomar como ejemplo el siguiente, en una escuela los alumnos pueden estar inscritos a diferentes cursos y a su vez cada curso a su vez puede relacionar a varios alumnos inscritos.

One-to-One.

Esta relación uno a uno se usa para representar la relación de un elemento (elemento A) con otro elemento de otra tabla de la base de datos (elemento B), pero este (elemento B) solo se encuentra relacionado con el elemento A.

Considerando que usaremos Django como framework para el desarrollo de la aplicación se tiene que elegir dentro de la base de datos soportadas, por lo cual la elección es MYSQL.

3.2.7 Interfaz Gráfica (GUI)

Hasta ahora hemos tratado a detalle el funcionamiento interno de una aplicación web o también conocido como “Backend”, pero nos falta tratar una parte fundamental de las aplicaciones web que sin ella no sería posible comunicarnos con los usuarios de nuestra aplicación nos referimos a la interfaz de la aplicación la cual se encarga de decir a los usuarios que pueden hacer con nuestra aplicación, interactuar con ella y recibir las respuestas que procedemos en el “Backend”.

Una interfaz puede ser tan sencilla como una simple línea de texto y un campo para ingresar un dato, en donde se le indique al usuario que tiene que hacer o que tiene que esperar de nuestra aplicación y donde pueda ingresar la información a procesar por la aplicación, sin embargo con el tiempo las interfaces han evolucionado con el objetivo de hacer más sencillo para los usuarios la interacción con las aplicaciones, para esto es común que se hagan uso de elementos gráficos que faciliten la comunicación con los usuarios a través de la imitación de interfaces o mandos comunes en la vida real, elementos como botones, paneles, punteros, etc.

Para el caso de las aplicaciones web, estas interfaces se construyen a través de 3 tecnologías base que son HTML, CSS y JS, debido a que son estas tecnologías las que se han estandarizado para ser soportadas por todos los navegadores web modernos, al día de hoy estas 3 tecnologías han evolucionado de forma simultánea y constante al punto de que nos permiten construir robustas interfaces para nuestras aplicaciones web que incluso pueden llegar a ser tan interactivas y amigables como la interfaz de una aplicación nativa de escritorio o móvil.

A continuación, explicaremos cada uno de estos componentes y su papel en el desarrollo de la interfaz gráfica de una aplicación web que el mundo informático recibe el nombre de “Frontend”.

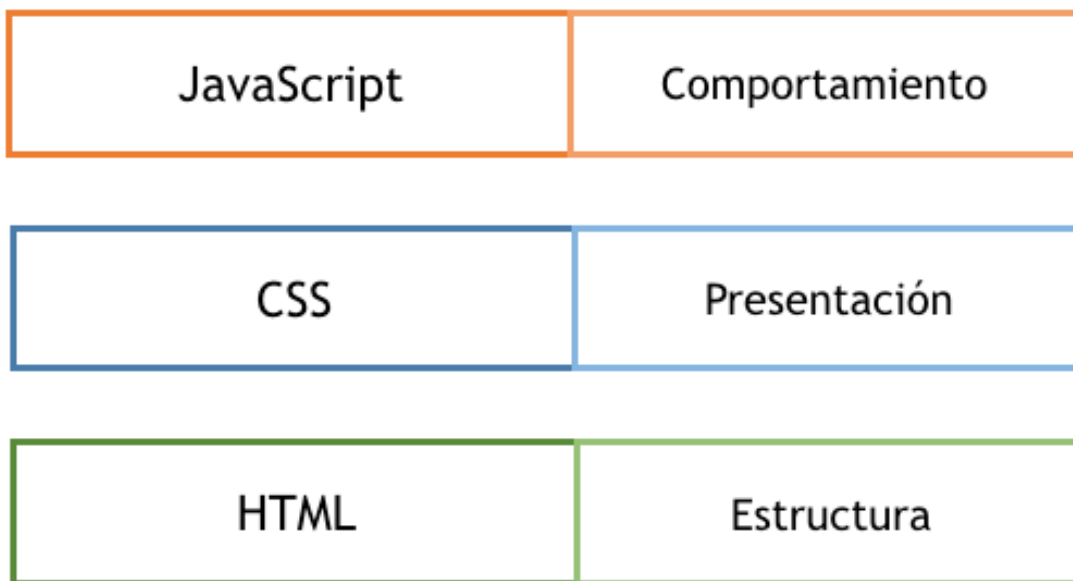


Figura 19 Esquema de los componentes de la interfaz de una aplicación web

HTML

HTML es la abreviatura de Hipertextual Markup Lenguaje, esta tecnología estuvo presente desde los inicios de la web y ha evolucionado la web misma, el desarrollo de HTML así como sus implementaciones está regulado por un grupo de expertos, programadores, autores, y académicos que conforman el W3C (World Wide Web Consortium), se encargan de crear los estándares que todos los desarrolladores de navegadores web deben seguir, con el objetivo de que un sitio web se vea y funcione igual en cualquier navegador web, sin embargo los estándares que emite la W3C son solo recomendaciones por lo cual depende de cada desarrollado de un navegador web seguir o no estas

recomendaciones, la buena noticia es que al día de hoy la mayoría de los navegadores web más usados a nivel mundial tratan de apegarse los más posible a las recomendaciones de la W3C [29].

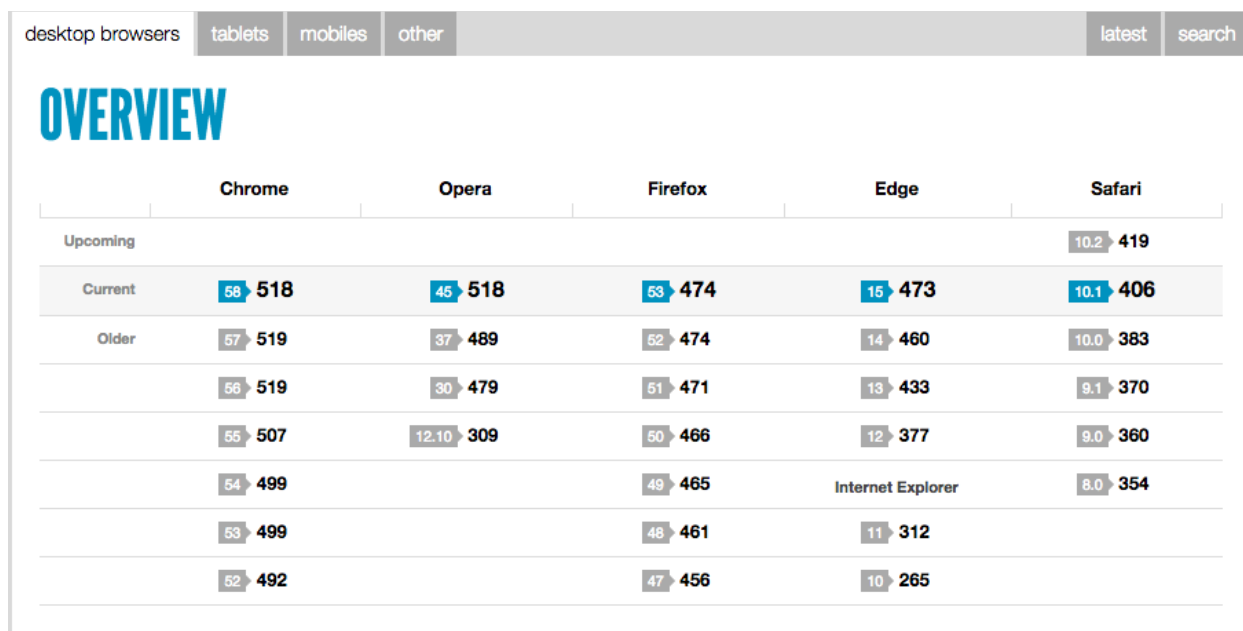


Figura 20 Cumplimiento de los estándares de la W3C por parte de los diferentes navegadores del mercado [30]

En resumen, HTML es el lenguaje que especifica cómo deben ser desplegadas las páginas web por cualquier navegador ya sea de una computadora de escritorio, laptop, tablet, móvil o cualquier otro dispositivo que posea un navegador web.

Para entender cómo funciona HTML basta con explicar sus componentes, Hipertextual viene de Hipertexto que hace referencia a simple texto pero que en lugar de encontrarse en papel se encuentra en el llamado Hiperespacio que es otra forma de llamarle al internet, básicamente este texto contiene la información de una página web y las instrucciones de como deber ser desplegada por un navegador web.

El markup, como el nombre lo indica es una serie de marcadores formados por símbolos y códigos que especifican a los navegadores web como debe ser estructurada la página web, por ejemplo, los encabezados, párrafos, listas, tablas, etc.

En los inicios de la web la única información que podía ser transmitida era texto por lo cual no se requería de un formato o estructura especial al texto, sin embargo cuando fue posible transmitir otro tipo de contenidos como imágenes, sonido y video, fue necesario agregar marcadores para poder dar formato las páginas web con este nuevo contenido multimedia, con el paso del tiempo cuando el contenido de las páginas web se volvió más extenso y rico elementos multimedia el lenguaje de marcadores se volvió ineficiente para definir el formato de cada uno de los elementos, fue así como se desarrolló posteriormente CSS del cual hablaremos a continuación.

```
<HTML>
  <HEAD>
    <TITLE>Titulo de este Documento</TITLE>
  </HEAD>
  <BODY>
    <H1>Encabezado del Documento</H1>
    <a href="https://buap.mx">Link</a>
    <p>Texto plano</p>
  </BODY>
</HTML>
```

Figura 21 Estructura básica de un documento HTML

CSS

CSS es el acrónimo de Cascading Style Sheets (Hojas de estilo en cascada), esta tecnología tiene la función de especificar a través de reglas CSS llamados estilos como se debe ser la presentación visual de un documento HTML.

Los estilos que nos permite definir CSS son demasiados desde tipos de fuente, tamaños de fuente, colores, márgenes, bordes, imágenes de fondo, sombras e incluso en la versión más reciente CSS3 se incluyen animaciones, lo que nos da una gran flexibilidad para construir interfaces visualmente ricas [31].

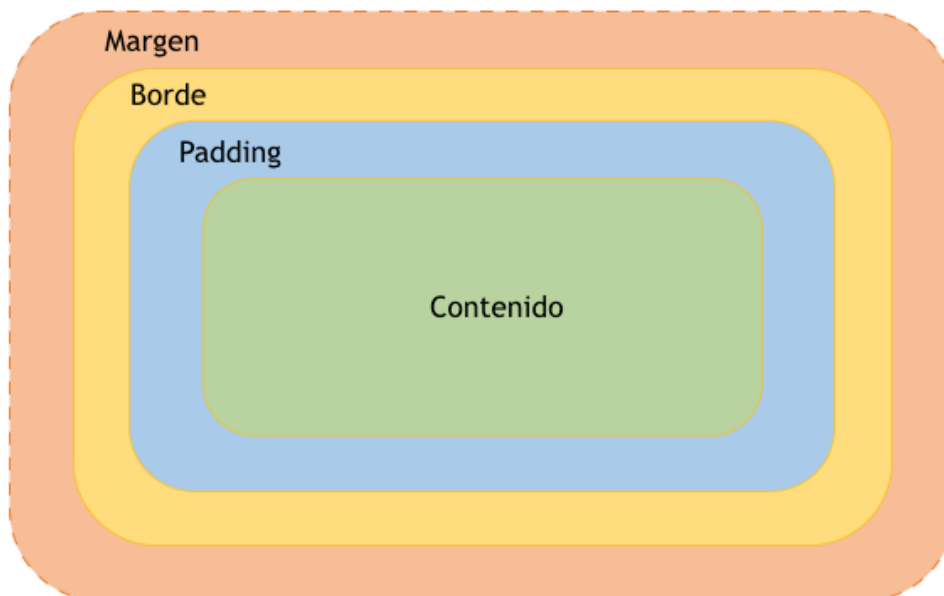


Figura 22 CSS Box Model

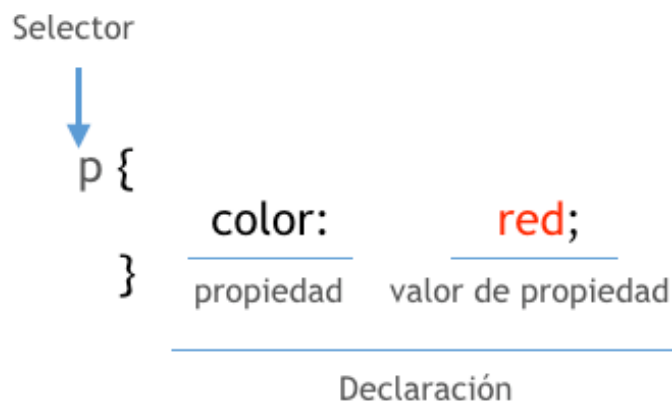


Figura 23 Estructura de una declaración CSS

CSS se puede integrar a través de documento independiente o bien en el documento HTML directamente, definiendo los estilos para cada marcador, o bien integrando un bloque CSS dentro del mismo archivo HTML para definir estilos generales del documento, sin embargo lo más recomendable es hacer uso del documento independiente con terminación .css, mantener los archivos de estilos separados del código HTML es una buena práctica de desarrollo que nos permite desarrollar proyectos más limpios y ordenados.

JavaScript

JavaScript es un lenguaje de alto nivel, orientado a objetos e interpretado, es decir que no requiere ser compilado para poder ser ejecutado por lo cual las instrucciones son interpretadas directamente durante el tiempo de ejecución [32].

JavaScript surgió de la necesidad de dotar a las páginas web de interactividad ya que hasta antes de su uso solo tenían una presentación visual estática, por lo cual se requería que las páginas web pudieran responder a las acciones que tuviera el usuario dentro de la página web en tiempo real, con el objetivo de brindar una experiencia de usuario mucho más amigable y sencilla de usar.

Hoy en día JavaScript es uno de los 3 elementos claves que hacen posible las páginas web modernas, HTML, CSS y JavaScript, prácticamente la totalidad de los navegadores web modernos incluye un intérprete de JavaScript integrado, por ejemplo Chrome integra el motor V8, Firefox integra SpiderMonkey, Edge hace uso de ChakraCore y Safari usa SquirrelFish, aunque cada navegador integra su propio motor JavaScript esto no afecta la ejecución de un mismo código JavaScript en diferentes navegadores debido a que JavaScript se rige por su propia especificación llamada ECMAScript, por lo cual se sigue un standard para el diseño de cada motor JavaScript.

A pesar de que JavaScript fue diseñado originalmente para correr del lado del cliente (dentro del navegador web) hoy en día también es usado del lado del servidor, una de estas tecnologías es Node.js que hemos tratado en párrafos anteriores, en esta sección solo se tratará su uso en desarrollo de interfaces web.

Javascript puede interactuar como cualquier elemento de la página web, con el usuario y con el servidor web, a continuación se listan algunos ejemplos de acciones que se pueden llevar a cabo con Javascript:

- Insertar nuevo HTML a la página o cambiar el HTML actual.
- Modificar estilos CSS
- Interactuar con acciones del usuario, como clics, posición del cursor, presionado de teclas e incluso en navegadores móviles o pantallas táctiles reaccionar a gestos “touch”.
- Mostrar mensajes o hacer preguntas al usuario
- Obtener y establecer Cookies, así como almacenamiento de variables locales dentro del navegador.
- Enviar y recibir peticiones al servidor a través de tecnologías como AJAX

Escribir una rutina Javascript es sencillo, se puede integrar el script directamente dentro del código HTML o bien escribirlo en un archivo separado con extensión .js y mandarlo a llamar dentro del código HTML.

Basado en Objetos

Javascript sigue el paradigma orientado a objetos sin embargo lo hace a partir de programación basada en prototipos en donde los objetos no son creados desde cero a través de instancias de clases si no a través de la clonación de otros objetos ya existentes, por lo cual se dice que es un prototipo para crear nuevos objetos.

Manejado por Eventos

Se dice que Javascript es manejado por eventos debido a que es capaz de reaccionar ante eventos que sucedan en cualquier momento durante la interacción del usuario con la página web, esto es importante debido a que a diferencia de una estructura de ejecución secuencial en donde el programa se ejecuta siempre siguiendo la secuencia con la cual fue programado, Javascript al ser manejado por eventos puede responder ante una acción del usuario en cualquier momento y de forma inmediata, lo que es vital para su objetivo el cual es hacer una de la web un ambiente interactivo.

3.3 Desarrollo del Software

3.3.1 Diagrama general del sistema de monitorización.

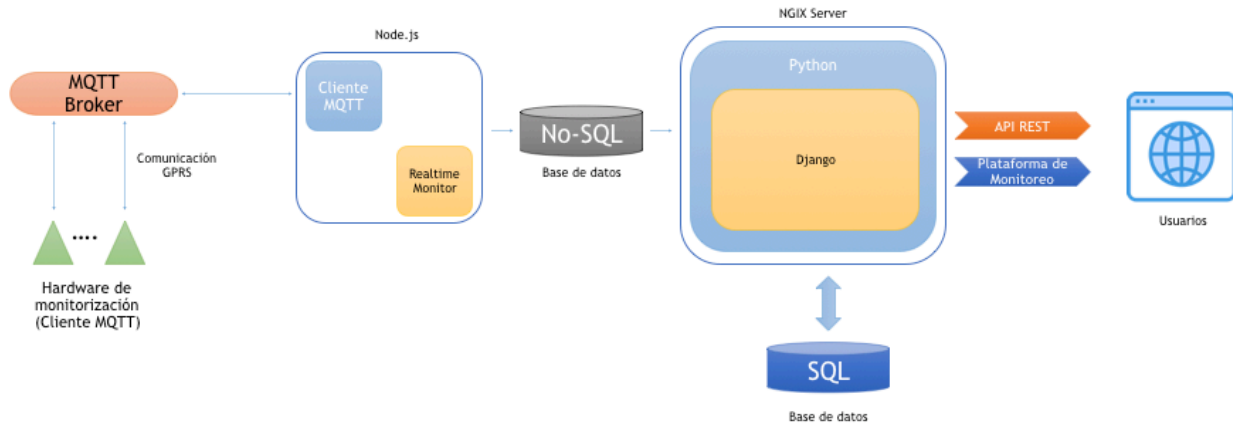


Figura 24 Diagrama general del sistema de monitorización

Para lograr el objetivo propuesto en esta tesis el sistema de monitorización se puede dividir en 2 partes fundamentales, el hardware de monitorización que consiste en un sistema de adquisición de datos ambientales habilitado para comunicarse a través de la red celular GPRS hacia plataforma de monitorización que es la segunda parte fundamental que consiste en el software del sistema.

En resumen, el funcionamiento integrado del sistema de monitorización se puede definir como sigue:

El hardware de monitorización tiene la función de adquirir datos ambientales a partir de sus sensores integrados, además de enviarlos en tiempo real hacia el software de la plataforma de monitorización, otro requisito importante que debe cumplir el hardware de monitorización es que debe ser inalámbrico con comunicación de largo alcance y lo suficientemente portable para poder instalarse en un UAV, para lograr esto se usa comunicación GPRS a través de la red celular, el protocolo de comunicación usado es MQTT un protocolo de mensajería que como se describió en capítulos anteriores es ligero y fue especialmente diseñado para ser utilizado en sistemas embebidos.

El software de la plataforma de monitorización debe ser capaz de comunicarse con el hardware de monitorización en tiempo real y de forma eficiente, esto se logra a través del protocolo MQTT, además debe de contar con un módulo para visualizar los datos en tiempo real y en adición ser recolectados y organizados para su posterior consulta, además debe cumplir con el objetivo de ser una plataforma con una interfaz de usuario fácil de operar sin tener que involucrar al usuario en cuestiones técnicas que dificulten su operación.

3.3.2 Desarrollo de la plataforma de monitorización.

Se compone de un conjunto de vistas que tienen la función de gestionar los usuarios de la plataforma, dispositivos de monitorización y sus variables a medir, también gestionar las zonas de monitorización, la programación de vuelos y un API REST.

Los datos son almacenados en la base de datos de acuerdo a su zona de monitorización y vuelo correspondiente, la información recolectada se puede desplegar de forma cronológica por días o intervalos de tiempo basados en horas, días, semanas, mes y año, también se puede consultar por zona de monitorización para poder realizar consultas con representaciones en tablas de datos que es posible exportar la información a formatos estándar como CSV y PDF, también es posible sobre un mapa poder visualizar la posición exacta donde se tomó cada muestra de información.

La plataforma está diseñada para que sea sencillo agregar nuevos sistemas de monitorización, para que puedan comenzar a recolectar datos de forma inmediata, se podrá configurar que variable medirán, así como sus unidades correspondientes.

Finalmente se cuenta con una capa de acceso a las funciones y datos de la plataforma a través de un API REST, para que puedan ser consultados por aplicaciones externas.

A continuación, se describe cada módulo que compone la plataforma:

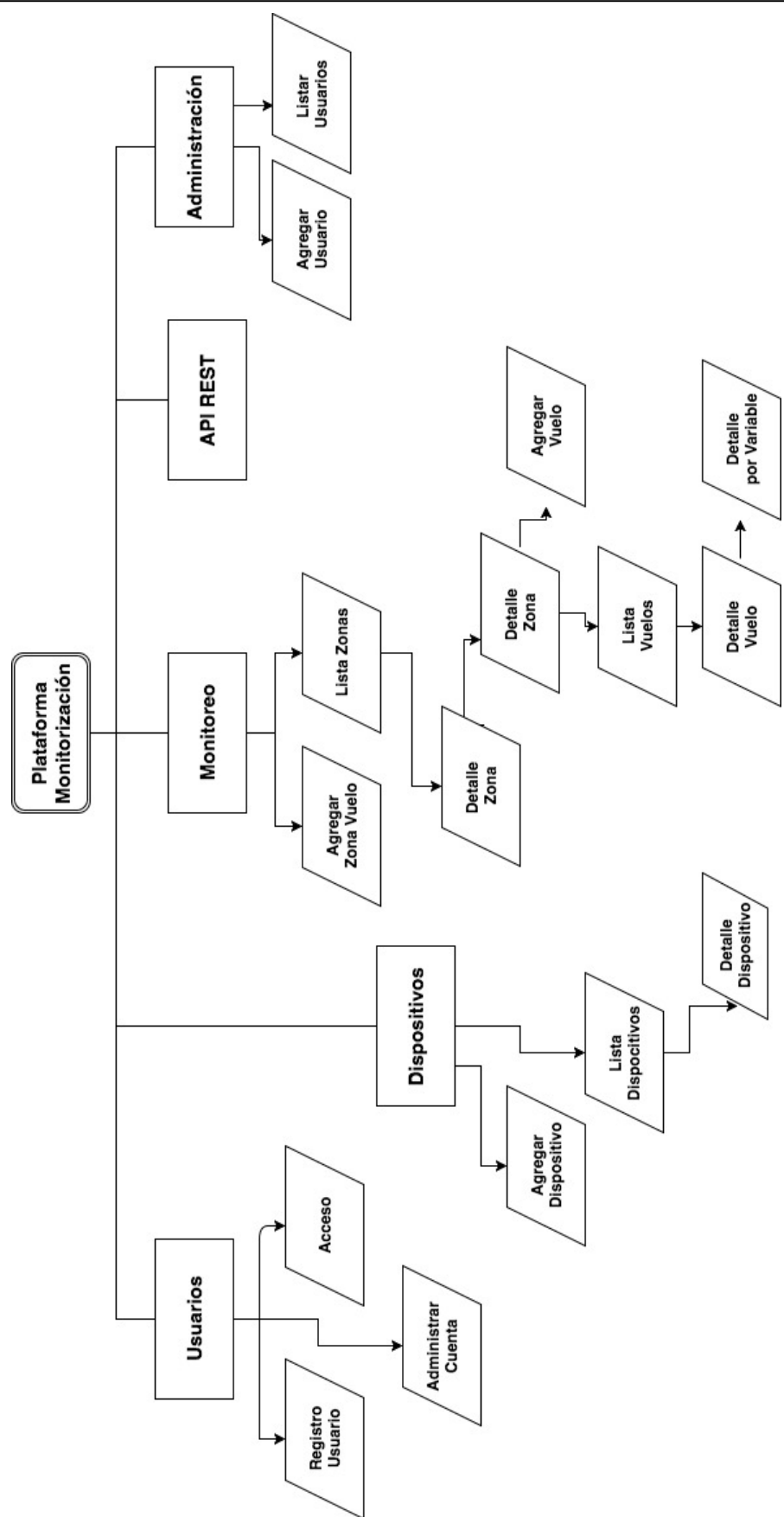


Figura 25 Diagrama de módulos y componentes de la plataforma de monitorización

Usuarios

Este módulo se encarga de gestionar todo lo relacionado los usuarios de la plataforma, desde registrarse, iniciar sesión y editar información de la cuenta.

Django como framework incluye un sistema de usuarios y autenticación con algunas funciones esenciales ya implementadas, como cuentas de usuarios, grupos, permisos y sesiones basadas en cookies, este sistema implementa autenticación y autorización de usuarios, en resumen el primero se encarga de verificar que un usuario es quien dice ser a través de solicitar un usuario y contraseña cada vez que solicite iniciar sesión en la plataforma, el segundo se encarga de determinar qué acciones tiene permitido realizar el usuario dentro de la plataforma.

Una ventaja de este sistema de usuarios y autenticación que incluye Django es que puede ser extendido y personalizado con el objetivo de manejar información adicional para cada usuario o bien implementar funciones que mejoren la seguridad de los usuarios.

Los campos que incluye el modelo Usuario que incluye Django por default son:

- **username:** es un nickname corto con el cual se identifica al usuario y al mismo tiempo sea fácilmente recordable por el usuario para un fácil inicio de sesión.
- **password:** las contraseñas no se guardan como texto plano, para mayor seguridad estas son encriptadas y desencriptadas a través de un algoritmo PBKDF2 que usa un hash SHA25 y es esta cadena encriptada la que se almacena en este campo.
- **email:** almacena el correo electrónico del usuario
- **first_name:** almacena el nombre del usuario
- **last_name:** almacena los apellidos del usuario

Para nuestra plataforma requerimos contar con información extra de nuestros usuarios, por lo cual se extendió el modelo Usuario a través de un nuevo modelo Perfil con algunos campos extras, los cuales son:

edad: campo tipo entero, almacena la edad del usuario

telefono: campo tipo entero, almacena el teléfono del usuario

celular: campo tipo entero, almacena el celular del usuario

direccion: campo tipo texto, almacena la dirección del usuario

email_verif: campo tipo booleano, este campo almacena el estado de verificación del email, si esta ya fue verificada o no.

Monitoreo

Este módulo tiene la función de administrar toda información de monitoreo, se diseñó de tal forma que sea fácil e intuitivo de usar, además de que los datos estén organizados para su fácil acceso y uso.

Quedando de la siguiente forma:

Zona de vuelo

Tiene como objetivo representar a la zona geográfica de interés de estudio, es el eslabón superior donde concurren todos los datos de monitorización recolectados.

El modelo de Zona de vuelo se compone de los siguientes campos:

- **nombre:** campo tipo texto, este campo nos sirve para identificar fácilmente la zona de vuelo
- **descripción:** campo tipo área de texto, este almacena una breve descripción de la zona de vuelo
- **usuario:** campo tipo ForeignKey, este campo permite relacionar la zona de vuelo con el usuario que la creo.
- **fecha:** campo tipo Fecha, nos permite almacenar la fecha de creación de la zona de vuelo
- **latitud:** campo tipo carácter, nos permite almacenar la latitud geográfica de un punto de referencia de la zona de vuelo.
- **longitud:** campo tipo carácter, nos permite almacenar la longitud geográfica de un punto de referencia de la zona de vuelo.

Vuelos

Un vuelo es una representación de un recorrido de monitorización realizado dentro de una Zona de vuelo, por lo cual se encuentra relacionado a está haciendo posible que todos los datos recolectados en un vuelo queden ligados a la zona geográfica que es de interés de estudio.

Una zona de vuelo puede tener múltiples vuelos, en diferentes periodos de tiempo y con un dispositivo de monitoreo independiente.

El modelo de Vuelo se compone de los siguientes campos:

- **descripcion:** campo tipo área de texto, almacena una breve descripción del vuelo de monitorización.
- **fecha:** campo tipo Fecha, nos permite almacenar la fecha del vuelo
- **dispositivo:** campo tipo ForeignKey, nos permite relacionar un vuelo con un dispositivo de monitorización
- **zona:** campo tipo ForeignKey, nos permite relacionar el vuelo con una zona de vuelo
- **usuario:** campo tipo ForeignKey, nos permite relacionar el vuelo con el usuario que lo creo

Datos de Vuelo

Durante cada vuelo de monitoreo será posible recolectar diferentes tipos de datos, cada tipo de dato perteneciente a una variable a medir diferente, estas variables dependerán de los sensores disponibles en el dispositivo de monitorización, estos datos recolectados quedarán ligados al vuelo en que fueron recolectados.

El modelo Datos de Vuelo se compone de los siguientes campos:

- **fecha:** campo tipo Fecha y hora, nos permite almacenar la fecha y hora exacta en que la medición fue tomada.
- **latitud:** campo tipo carácter, nos permite almacenar la latitud geográfica del punto donde fue tomada la medición.
- **longitud:** campo tipo carácter, nos permite almacenar la longitud geográfica del punto donde fue tomada la medición.
- **altitud:** campo tipo carácter, nos permite almacenar la altitud geográfica del punto donde fue tomada la medición.
- **vuelo:** campo tipo ForeignKey, nos permite relacionar el dato con un vuelo.
- **zona:** campo tipo ForeignKey, nos permite relacionar el dato con una zona de vuelo.
- **variable:** campo tipo ForeignKey, nos permite determinar qué tipo de variable es el dato.
- **data:** campo tipo carácter, es el valor del dato enviado por el dispositivo de monitorización.

Variables

Siguiendo el objetivo de contar con una plataforma de monitorización flexible y fácil de usar, se optó por dejar abierto la asignación de variables a cada dispositivo de monitorización, de esta forma el usuario será capaz de asignar las variables soportadas por su dispositivo cada una con su unidad de medida correspondiente.

El modelo Variable se compone de los siguientes campos:

nombre: campo tipo carácter, nos permite identificar el tipo de variable.

unidad: campo tipo carácter, nos permite asignar la unidad de medida de la variable.

Dispositivos

Un dispositivo es una unidad de monitorización que consta de diferentes tipos de sensores que le permiten recolectar información durante un vuelo de monitorización, su representación se diseñó de tal forma que el usuario pueda asignar las variables soportadas por su dispositivo y pueda usarlo en diferentes vuelos e incluso con diferentes zonas de vuelo.

El modelo Dispositivo se compone de los siguientes campos:

ide: campo tipo carácter, es el identificador único del dispositivo físico de monitoreo, a través de este identificador único es posible vincular los datos provenientes del dispositivo físico con su respectivo objeto virtual.

nombre: campo tipo carácter, nos permite asignar un nombre al dispositivo fácil de recordar para el usuario.

descripcion: campo tipo área de texto, almacena una breve descripción del dispositivo.

usuario: campo tipo ForeignKey, relaciona el dispositivo con el usuario que dio de alta.

variables: campo tipo ManyToManyField, este campo nos permite asignar las variables que soporta el dispositivo de monitorización físico.

Mi cuenta

El usuario podrá mantener actualizados los datos de su cuenta a través de esta sección, a continuación se describen las funciones disponibles:

Editar datos de cuenta

El usuario podrá editar datos básicos de su cuenta como:

Nombre

Apellido

Correo

Editar información de Perfil:

El usuario podrá editar datos adicionales de su cuenta como:

Edad

Teléfono

Dirección

Descripción

Cambiar contraseña

El usuario puede usar esta función como medida de seguridad para actualizar su contraseña por una nueva.

Cerrar cuenta

El usuario puede optar por cerrar su cuenta, todos sus datos serán eliminados de la base de datos.

3.3.3 API REST

La plataforma cuenta con un API REST que permite consumir recursos de la aplicación de una forma sencilla y al alcance de cualquier desarrollador, esto es posible al usar el estándar REST.

Los recursos REST disponibles son los siguientes:

- Consulta de zonas de monitoreo: Obtener todas las zonas de monitoreo de un usuario.
- Consulta de vuelos por zona de monitoreo: Obtener todos los vuelos de una zona de monitoreo
- Consulta de variables de un vuelo: Obtener los datos de un vuelo.

Para poder acceder a los recursos se usa una autenticación por token como se describe a continuación:

Todos los recursos se encuentran para el tipo de contenido: Content-Type: application/json

Recurso: /api/get_auth_token/

Método: POST

Parámetros aceptados:

username: nombre de usuario

password: contraseña del usuario

Respuesta exitosa:

```
{“token”:“37c0c8f966f9380a98c66b0bf46d7d86ed7d5108”}
```

Respuesta

errónea:

```
{“non_field_errors”: [“No puede iniciar sesión con las credenciales proporcionadas.”]}
```

Este token deberá ser enviado en la cabecera de cada petición que se realice para acceder cada recurso de la aplicación.

• Consulta de zonas de monitoreo

Recurso: /api/get_zones/

Método: GET

Parámetros aceptados:

ninguno

Respuesta exitosa:

```
{“count”:2,“next”：“/api/get_zones? page=2”,“previous”:null,
“results”:
[ {“id”:1,“nombre”:“Zona 1”, “descripcion”:“Descripción Zona 1”, “latitud”: “19.041377”,
“longitud”:”-98.214623
” },
{“id”:2,“nombre”:“Zona 2”, “descripcion”:“Descripción Zona 2”, “latitud”: “19.041377”, “longitud”:”-
98.214623
” }
]}
```

Descripción de respuesta:

count: número de elementos por página
next: url del recurso de la página siguiente en caso de existir
previous: url del recurso de la página anterior en caso de existir
results: elementos de la respuesta por página

id: id único de la zona
nombre: nombre de la zona
descripción: descripción de la zona
latitud: latitud de la zona
longitud: longitud de la zona

Respuesta errónea:

```
{“status”:”error”, “message”: “404 not found”}
```

• Consulta de vuelos por zona de monitoreo

Recurso: /api/get_vuelos/{id_zona}

Método: GET

Parámetros aceptados:

id_zona: id de la zona de vuelo

Respuesta exitosa:

```
{“count”:1,“next":"/api/get_vuelos? page=2”,“previous”:null,
“results”:
[{"id":1, “descripcion”:”Vuelo 1”, “fecha”:”2018-10-27T02|58|14.927728”, “dispositivo”:1,
“variables”:[{"id”:1, “nombre”:”temperatura”, “unidad”:”°C”}, {"id”:2, “nombre”:”presión”,
“unidad”:”mb”}, {"id”:3, “nombre”:”calidad del aire”, “unidad”:”ppm”}, {"id”:4, “nombre”:”radiación
UV”, “unidad”:”Índice UV”}]}
]}
```

Descripción de respuesta:

count: número de elementos por página
next: url del recurso de la página siguiente en caso de existir
previous: url del recurso de la página anterior en caso de existir
results: elementos de la respuesta por página

id: id único del vuelo
descripción: descripción del vuelo
fecha: fecha de alta del vuelo
dispositivo: id del dispositivo de vuelo
variables: array de variables disponibles en el vuelo:
 id: id único de la variable de vuelo
 nombre: nombre de la variable de vuelo
 unidad: unidad de medida de la variable de vuelo

Respuesta errónea:

```
{“status”:”error”, “message”: “404 not found”}
```

• **Consulta de variables de un vuelo**

Recurso: `/api/get_data_variable/{id_vuelo}/{id_variable_vuelo}`

Metodo: GET

Parametros aceptados:

id_vuelo: id del vuelo a consultar

id_variable_vuelo: id de la variable a consultar

Respuesta exitosa:

```
{“count”:10,“next":"/api/get_data_variable/{id_vuelo}/{id_variable_vuelo}
? page=2”,“previous”:null,
“results”:
[{{“id”:1, “latitud”: “19.041377”, “longitud”:”-98.214623
”, “altitud”:5, “fecha”:”2018-10-27T02|58|14.927728”, “dato”: 23.45}
}]
```

Descripción de respuesta:

count: numero de elementos por página

next: url del recurso de la página siguiente en caso de existir

previous: url del recurso de la página anterior en caso de existir

results: elementos de la respuesta por página

id: id único de la medición

fecha: fecha y hora en la que se tomo la medición

latitud: latitud donde se toma la medición

longitud: longitud donde se toma la medición

altitud: altitud relativa donde se toma la medición

dato: dato de la medición, la unidad de medida es la correspondiente a la variable

Respuesta errónea:

```
{“status”:”error”, “message”: “404 not found”}
```

3.3.4 API REST para agregar datos de vuelo

Este recurso tiene la función de recibir y guardar los datos de vuelo en la base datos transmitidos por el hardware de monitoreo.

Recurso: /datosadd/?device={id_device}&?vuelo={vuelo_id}

Metodo: POST

Parametros aceptados:
variables

Ejemplo:

```
{  
  "variables": [  
    { "var_name": "temp", "data": 22 },  
    { "var_name": "presion", "data": 1025 } ],  
  
  "latitud": "19.041377",  
  "logitud": "98.214623",  
  "altitud": "50"  
}
```

Respuesta exitosa:

```
{ "status": "success" }
```

Respuesta errónea:

```
{ "status": "error" }
```

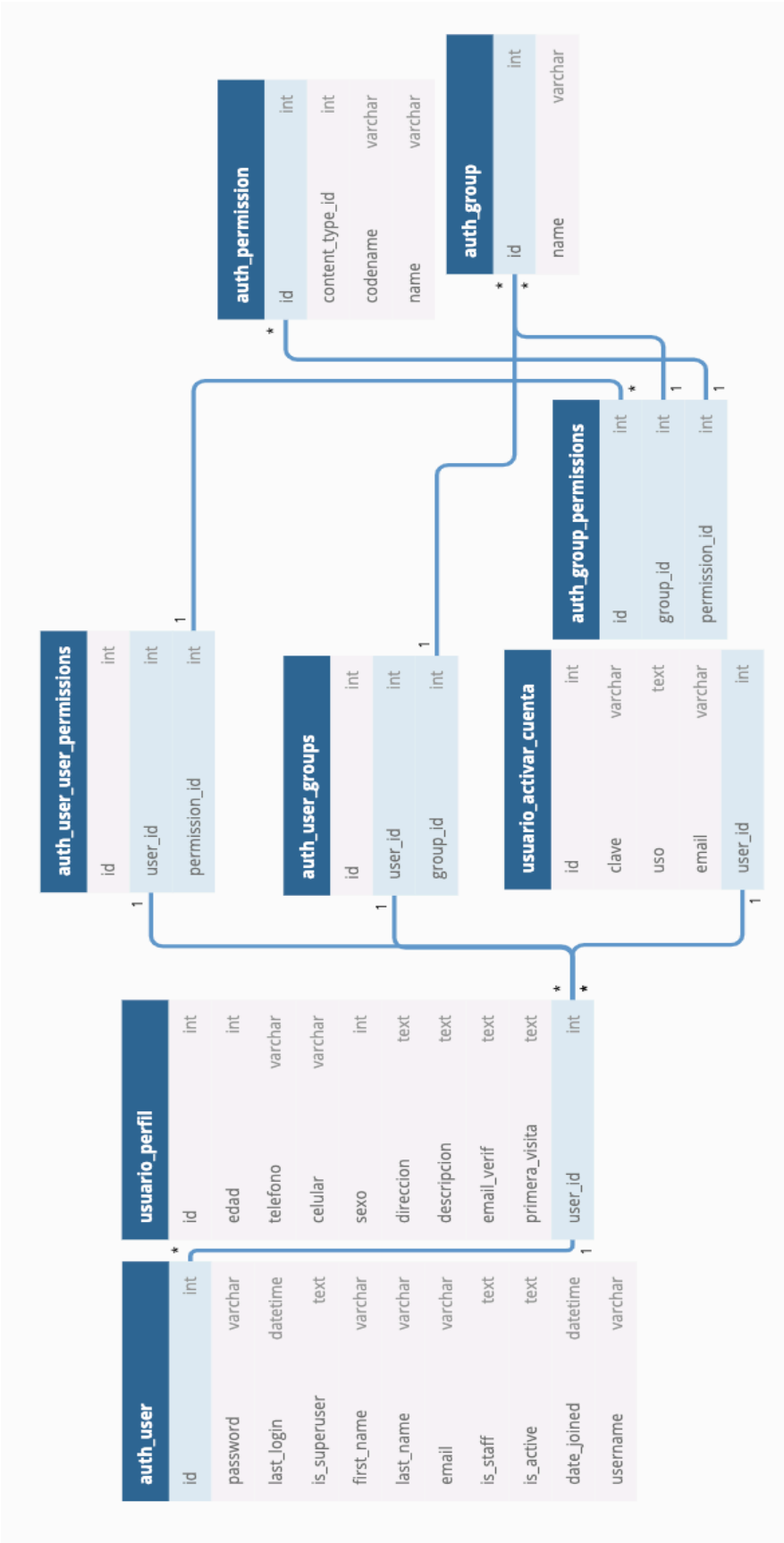


Figura 26 Diagrama entidad relación de la base de datos, parte 1

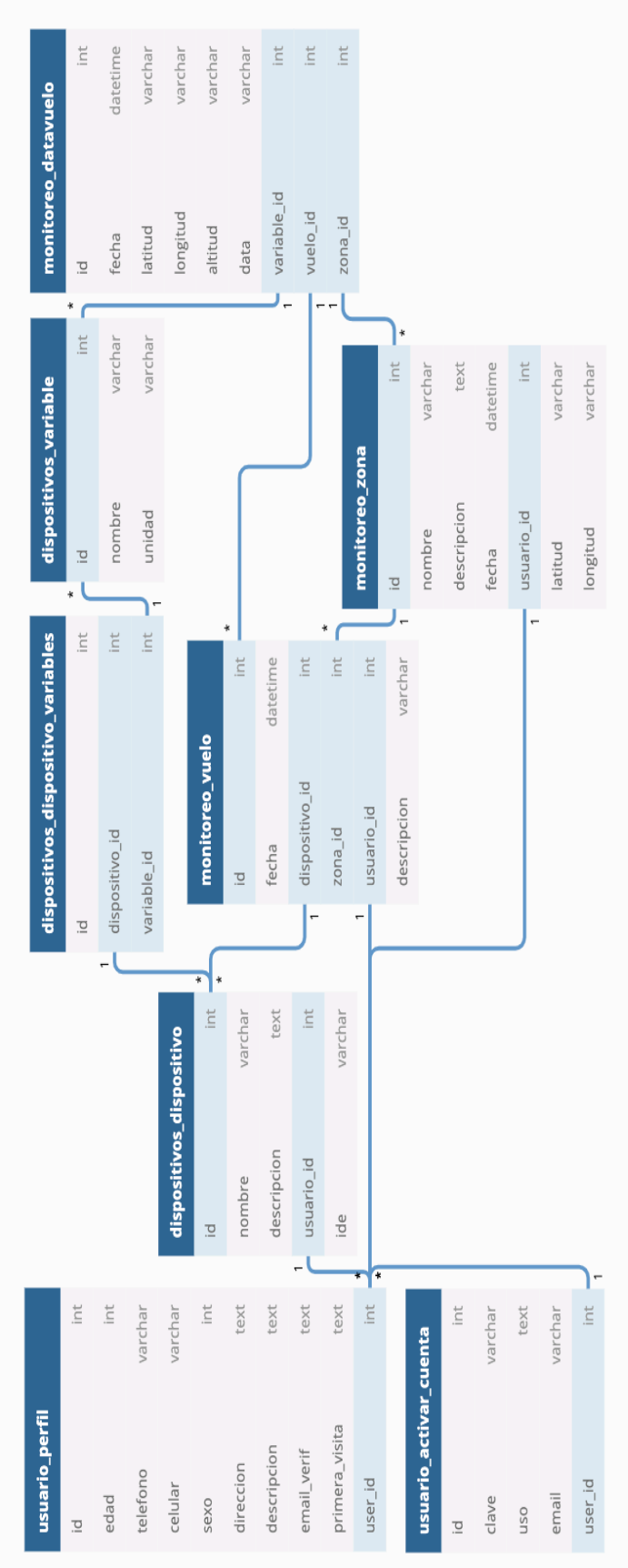


Figura 27 Diagrama entidad relación de la base de datos, parte 2

A continuación, se puede observar la interfaz grafica de la plataforma desarrollada, en el [Apéndice A](#) se muestra a detalle el funcionamiento de la misma.

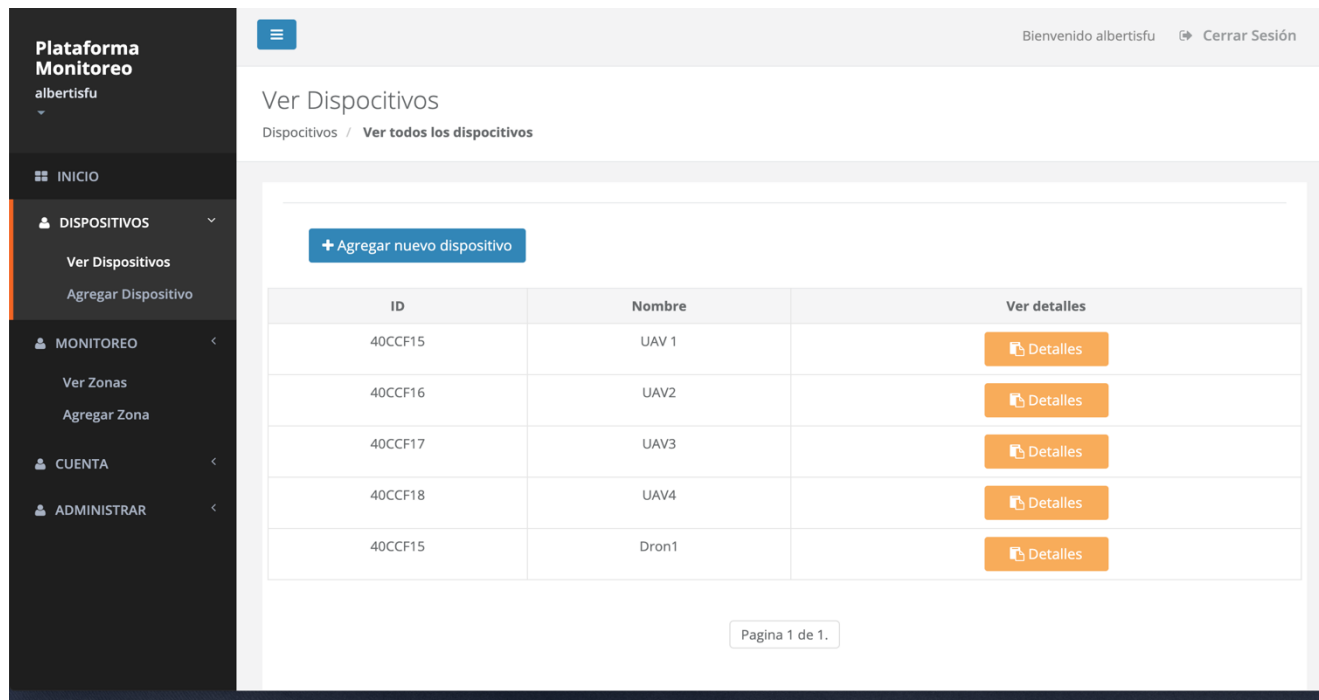


Figura 28 Pantalla - Ver Dispositivos de monitoreo disponibles.

3.4 Panel de monitorización en Tiempo Real

El panel de monitoreo tiene como función tener una salida visual en tiempo real de los datos enviados por el hardware de monitoreo, para poder ser analizados durante el tiempo de vuelo y toma de datos, para poder tomar decisiones inmediatas, detectar posibles errores.

Los datos son mostrados a través de una interfaz visual con instrumentos virtuales que reflejan las mediciones de cada una de las variables en tiempo real.

El desarrollo de este panel de monitoreo se basó en Node-Red un proyecto de software libre desarrollado a iniciativa de IBM, que se describe como “Programación basada en flujo para el Internet de las cosas”, consiste en una aplicación web que al estar construida Node.js es ideal para trabajar con tiempo real, consta de una librería de módulos pre construidos y una hoja de trabajo en donde es posible interconectar entradas y salidas de los diferentes módulos para construir flujos de programación de forma gráfica, lo cual facilita la programación de rutinas.

Node-Red cuenta con módulos pre construidos como: cliente MQTT, cliente HTTP, funciones lógicas e incluso a servicios externos como Email y Twitter, en nuestro caso el módulo de nuestro interés es el cliente MQTT ya que es el protocolo de comunicación que usa nuestro hardware de monitoreo.

Al ser un software libre y con documentación abierta, es posible desarrollar nuevos módulos para ser usados dentro del ambiente de trabajo de Node-Red, lo cual es una gran ventaja para este proyecto ya que es posible modelar los sensores de nuestro hardware de monitoreo como un módulo y ser usado solo cuando es requerido, además de poder definir nuevos sensores y variables ambientales de forma relativamente sencilla.

Por lo cual se procedió al desarrollo de 2 tipos de módulos a la medida de las necesidades del proyecto, sensores y monitor.

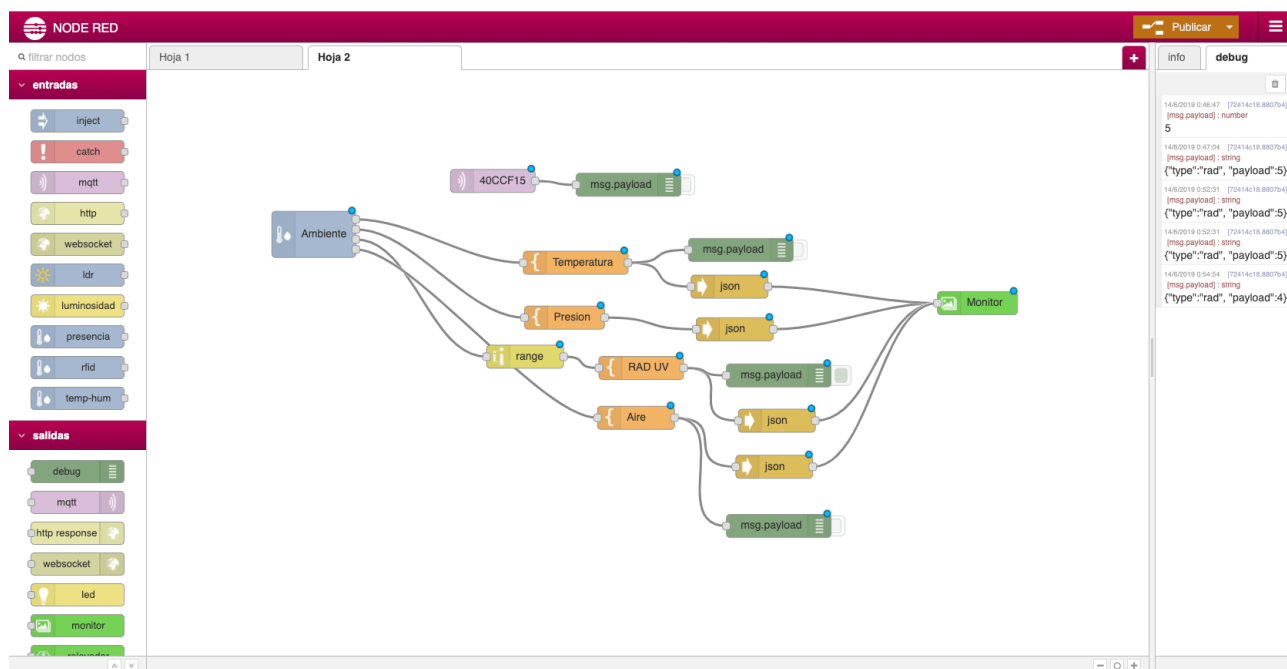


Figura 29 Pantalla - Interfaz Node-Red, diagrama de aplicación

3.4.1 Módulo Sensores

El módulo de sensores consiste en un módulo con una o varias salidas de datos que se pueden interconectar con otros módulos para interactuar con los datos de los sensores, el componente principal de este módulo es un cliente MQTT el cual permite configurar sus datos de conexión como lo es el Broker MQTT y el canal MQTT al cual se debe suscribir para recibir los datos del hardware de monitoreo.

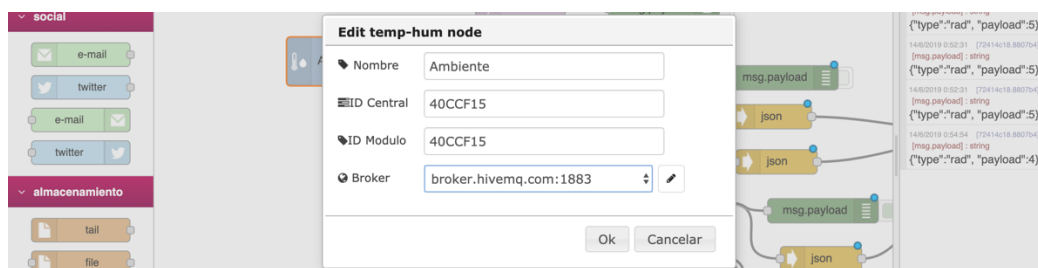


Figura 30 Pantalla - Configuración de módulo Sensor

Internamente este módulo contiene la programación necesaria para obtener la cadena de datos del hardware de monitoreo y traducirla en un dato entero que sea útil a la salida del módulo.

3.4.2 Módulo Monitor

El módulo monitor es un módulo que solo recibe entradas de datos, es diseñado para recibir datos enteros que puedan ser representados de forma gráfica, para lograr este objetivo este módulo genera una aplicación web basada en Node.js, se usa la librería Express.js para la generación de un servidor web y a través de Websockets se envían los datos en tiempo real hacia la interfaz gráfica del panel de monitoreo, este panel de monitoreo al estar basado en Web puede ser consultado desde cualquier parte del mundo en tiempo real.

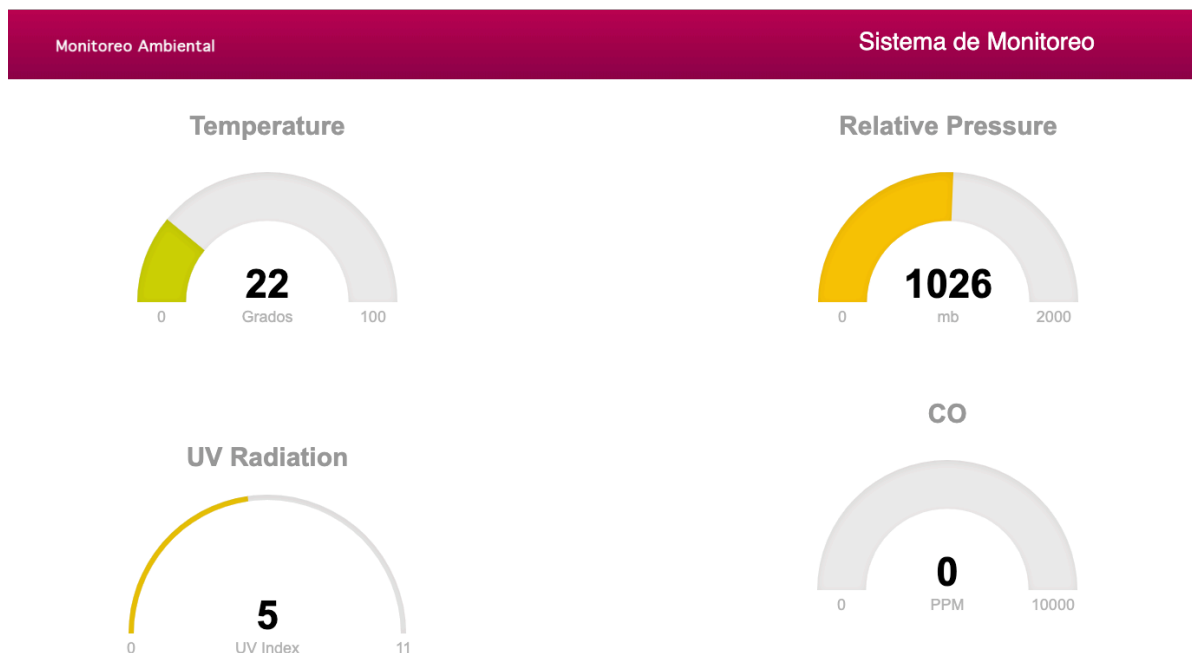


Figura 31 Pantalla - Panel de Monitorización en tiempo real

Capítulo 4. Descripción del Hardware

Como se comentó en párrafos anteriores el hardware de monitorización debe cumplir con los siguientes requisitos:

- Medición de datos ambientales, específicamente: Temperatura, Presión atmosférica, Índice de radiación UV y Calidad del aire, además de recolectar las coordenadas geográficas donde se toma cada medición.
- Inalámbrico de largo alcance, para poder comunicarse a la plataforma de monitorización durante el vuelo del UAV, por lo cual debe contar con un sistema de comunicación a través de la red celular GPRS.
- Portable, debe ser lo suficientemente pequeño para poder ser transportado en vuelo por un UAV.

4.1 Selección de módulos y sensores.

Para cumplir estos requisitos se analizaron diferentes alternativas de sensores, módulos de comunicación y microcontroladores, para seleccionar los más adecuados que nos permitieran cumplir con el objetivo propuesto, a continuación se describen los componentes seleccionados:

4.1.1 Adafruit FONA - Mini Cellular GSM

El primer componente que se seleccionó debido a su importancia dentro de los requisitos que debía cumplir el hardware de monitorización fue el módulo de comunicación celular (GPRS) por lo cual fue la base para la selección del resto de componentes.

El módulo de comunicación además de brindar conexión a internet GPRS debía ser pequeño y con un bajo consumo energético, otro punto de peso que se tomó en cuenta fue que contará con librerías para soportar el protocolo de comunicación MQTT fundamental para contar un hardware de monitorización rápido, estable y con un consumo de datos pequeño.

Por lo cual se seleccionó el módulo Adafruit FONA - Mini Cellular GSM que es un módulo que integra el chip de comunicación GSM SIM800 con soporte en redes celulares 2G en las bandas 850, 900, 1800 y 1900 MHZ, el módulo cuenta con soporte GPRS para enviar y recibir datos a través de los protocolos TCP/IP y HTTP y sus derivados como lo es MQTT [33].

El soporte MQTT para este módulo es través de la librería MQTT_Library disponible para el ambiente de desarrollo Arduino, por lo cual la elección del resto de componentes se tomó bajo la premisa de ser compatibles con el ambiente Arduino.

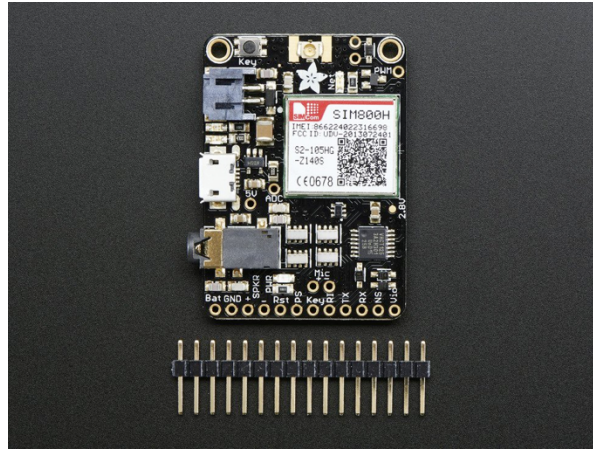


Figura 32 Módulo Adafruit FONA - Mini Cellular GSM

4.1.2 Microcontrolador Atmega328P

Tomada la decisión de trabajar bajo el ambiente de desarrollo Arduino, el siguiente paso fue elegir el microcontrolador para trabajar ya que debido a otro de los requisitos del hardware es que debía ser lo más pequeño posible por lo cual se tomó la decisión de construir un PCB a la medida donde se pudieran integrar los módulos y sensores necesarios en el menor espacio posible, en vez de usar una placa de desarrollo Arduino comercial, para esto era necesario seleccionar un microcontrolador compatible con el ambiente Arduino que se pudiera integrar en nuestro PCB desarrollado a la medida.

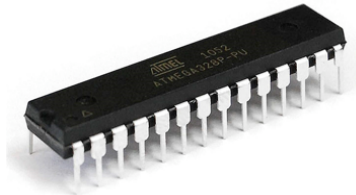


Figura 33 Microcontrolador Atmega328P

Dentro de las opciones disponibles se eligió el Microcontrolador Atmega328P debido su fácil acceso y bajo costo, además de cumplir con los requisitos técnicos necesarios para el proyecto, el Atmega328P es un microcontrolador desarrollado por Atmel de alto rendimiento basado en un microcontrolador RISC, combinando 32 KB ISP flash una memoria con la capacidad de leer-mientras-escribe, 1 KB de memoria EEPROM, 2 KB de SRAM, 23 líneas de E/S de propósito general, 32 registros de proceso general, tres temporizadores flexibles/contadores con modo de comparación, interrupciones internas y externas, programador de modo USART, una interfaz serial orientada a byte de 2 cables, SPI puerto serial, 6-canales 10-bit Conversor A/D, "watchdog timer" programable con oscilador interno, y cinco modos de ahorro de energía seleccionables por software [34].

PARÁMETROS	VALORES
Flash	32 Kbytes
SRAM	2 Kbytes
Cantidad Pines	28
Frecuencia máxima de operación	20 MHz
CPU	8-bit AVR
Pines máximos de E/S	23
Interrupciones internas	24
Canales ADC	8
Resolución de ADC	10
Eeprom	1K
Canales PWM	6
Voltaje de operación	1.8-5.5 v
Timers	3

Figura 34 Especificaciones técnicas Microcontrolador Atmega328P

4.1.3 Gy-GPS6MV2

Otro módulo importante es el GPS que tendrá como función obtener las coordenadas geográficas de donde se tomen las mediciones ambientales, para esto se eligió el módulo Gy-GPS6MV2 el cual cuenta con comunicación serial, bajo consumo energético y buena precisión [35].

Sus características técnicas son:

- Receptor: Ublox NEO 6M
- Voltaje de alimentación: 3V a 5V
- Interfaz: UART, comunicación asíncrona



Figura 35 Módulo GPS Gy-GPS6MV2

4.1.4 BMP180

Para tomar mediciones de precisión atmosférica y temperatura se eligió usar sensor de presión barométrica BMP180 el cual posee una precisión y bajo consumo de energético, además ofrece un rango de medición que va de 300 a 1100 hPa, con una precisión absoluta de hasta 0,03 hPa, su forma de comunicación es a través del protocolo I2C [36].



Figura 36 Módulo BMP180

Sus características técnicas son:

- Voltaje de Operación: 3.3V - 5V DC
- Interfaz de comunicación: I2C (3.3V)
- Rango de Presión: 300 a 1100 hPa
- Resolución: 1 Pa
- Precisión absoluta: 1 hPa
- Medición de temperatura incluida
- Resolución de temperatura: 0.1°C
- Precisión Temperatura: 1°C
- Frecuencia de Muestreo: 120 Hz (máx.)
- Ultra-bajo consumo de energía
- Completamente calibrado

4.1.5 SI1145

Para realizar mediciones del índice de radiación UV se utilizó el sensor SI1145 el cual es un sensor de bajo consumo energético con comunicación I2C, este sensor es capaz de detectar fotones visibles, infrarrojos y luz ultravioleta, por lo cual en combinación con una serie de algoritmos es capaz de proporcionarnos una medición confiable de conocido índice de radiación UV [37].



Figura 37 Módulo SI1145

Sus características técnicas son:

- Espectro infrarrojo: 550nm-1000nm
- Espectro de luz visible: 400nm - 800nm
- Voltaje: 3-5V
- Comunicación I2C
- Temperatura de operación: -40°C - 85°C

4.1.6 MQ7

Para medir la calidad del aire, nos centramos en medir la concentración de Monóxido de carbono, por lo cual se usó el sensor MQ7 el cual es un sensor de gas usado comúnmente para la detección de Monóxido de carbono, su comunicación es vía serial [38].



Figura 38 Módulo MQ7

Sus características técnicas son:

- Tipo de sensor: Semiconductor
- Encapsulado: Plástico
- Detección de gas: Monóxido de carbono
- Concentración 10 - 10,000 ppm CO
- Voltaje: $\leq 10V$ DC

4.1.7 Baterías

Para alimentar el hardware de monitorización se usan 2 baterías, la primera es una batería LI-PO de 7.4v con 250 mAh que alimenta la tarjeta principal que incluye el microcontrolador y los diferentes sensores descritos, la segunda es una batería LI-ION de 3.7v con 500 mAh que alimenta especialmente el módulo de comunicación FONA - Mini Cellular GSM.

4.2 Diagrama general del Hardware.

Una vez elegidos los módulos y microcontrolador, se realizó el diseño del diagrama electrónico que se puede ver a continuación.

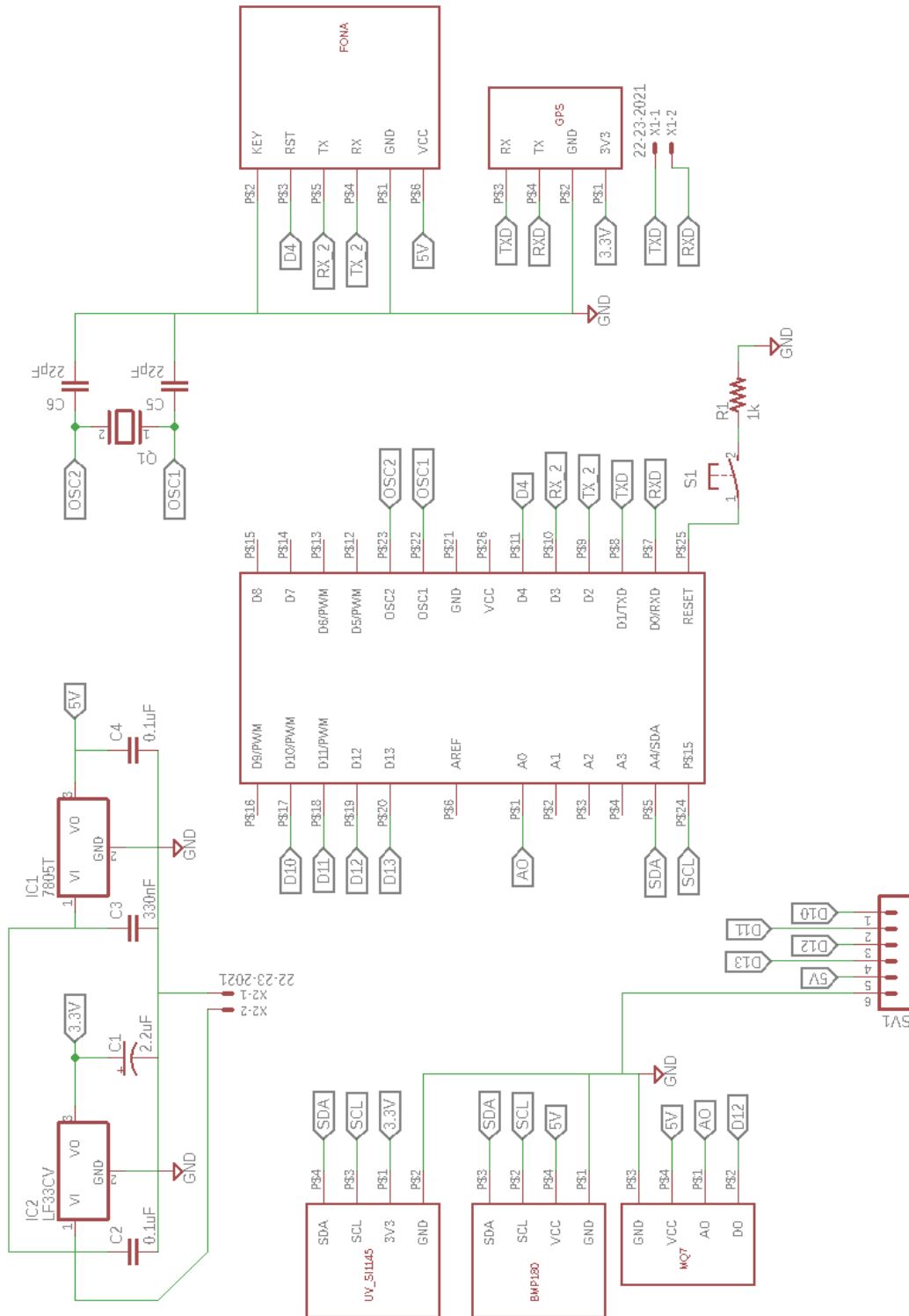


Figura 39 Diagrama electrónico del Hardware de Monitorización

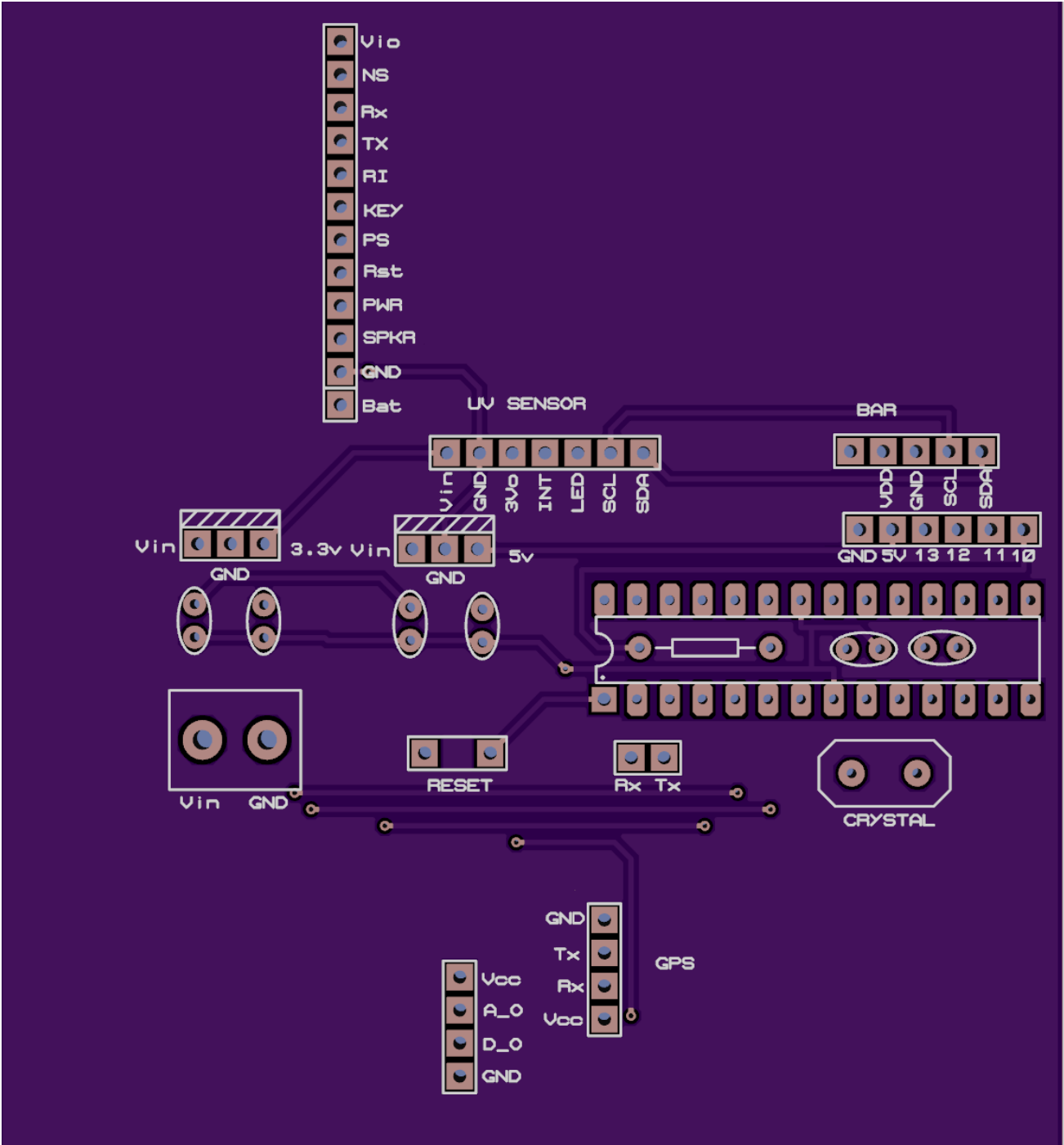


Figura 40 Diseño del PCB

4.3 Fotos del hardware de monitorización montado sobre el UAV

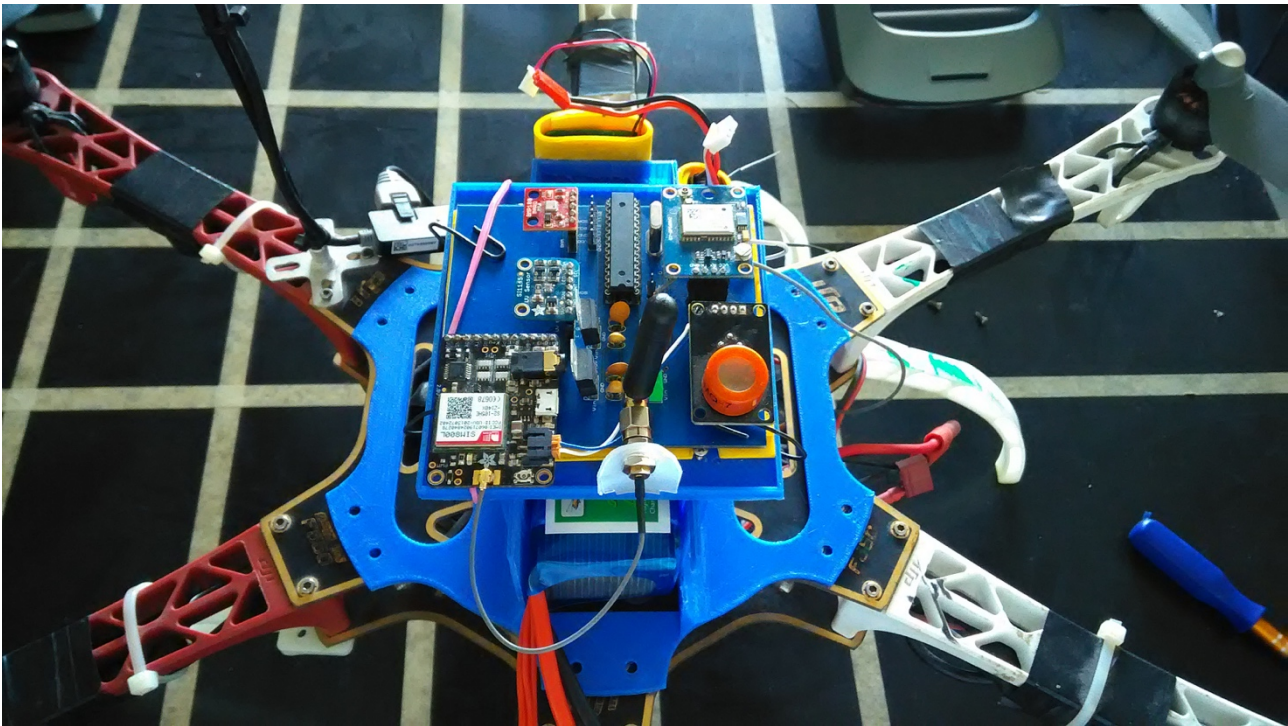


Figura 41 Foto 1 - Hardware de Monitorización sobre UAV

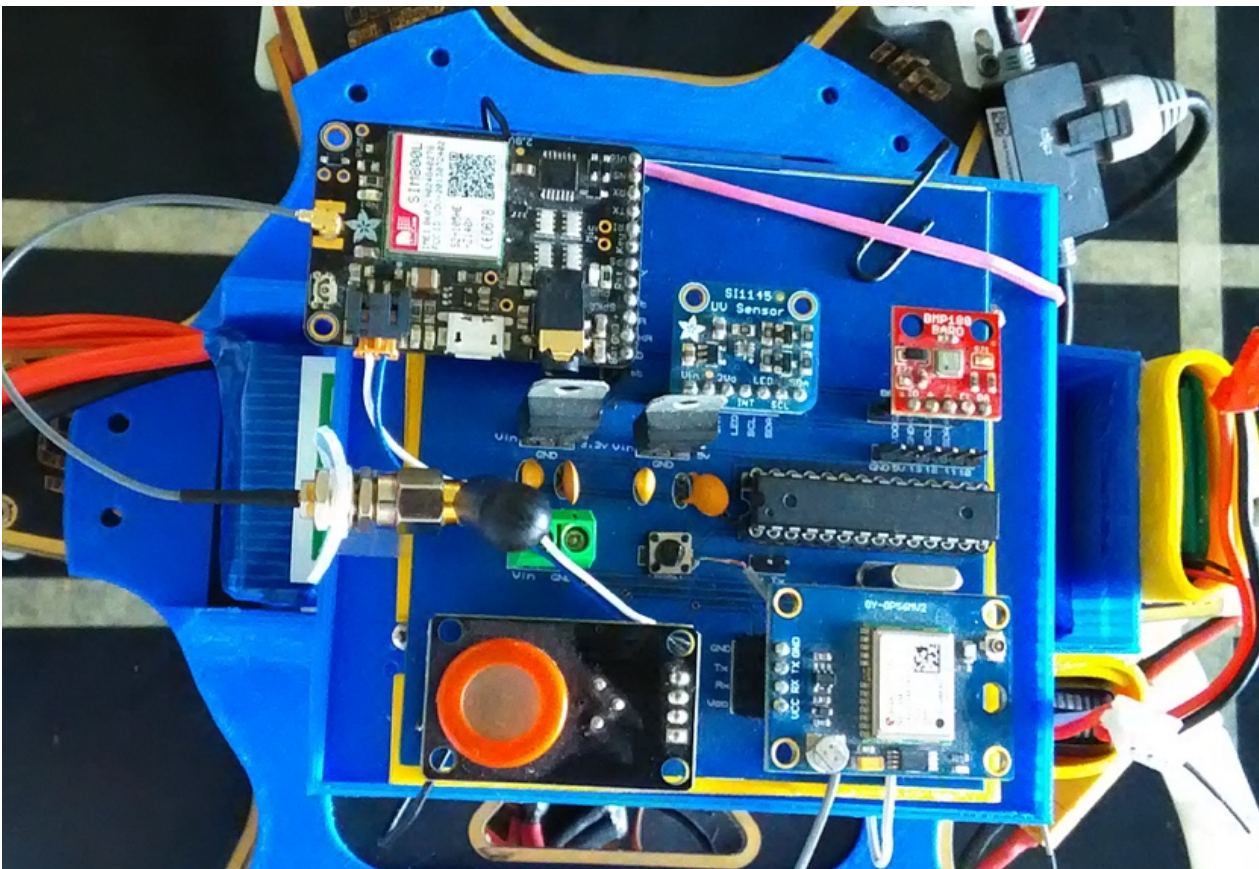


Figura 42 Foto 2 - Hardware de Monitorización sobre UAV

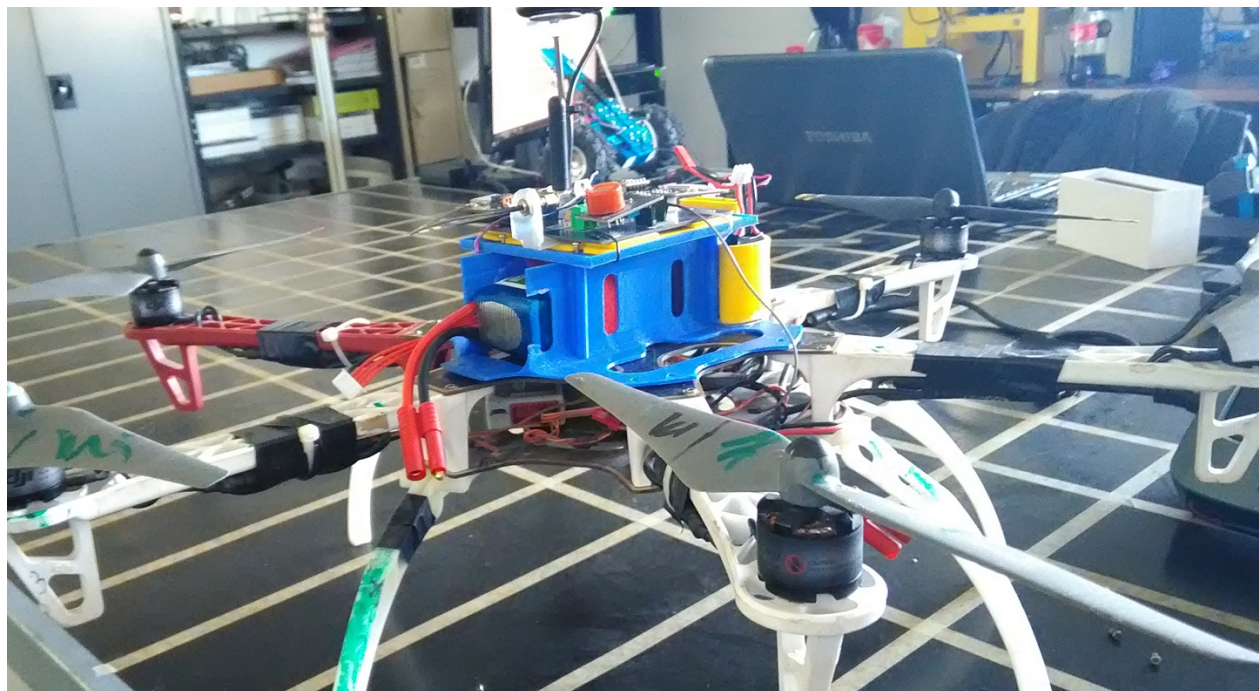


Figura 43 Foto 3 - Hardware de Monitorización sobre UAV

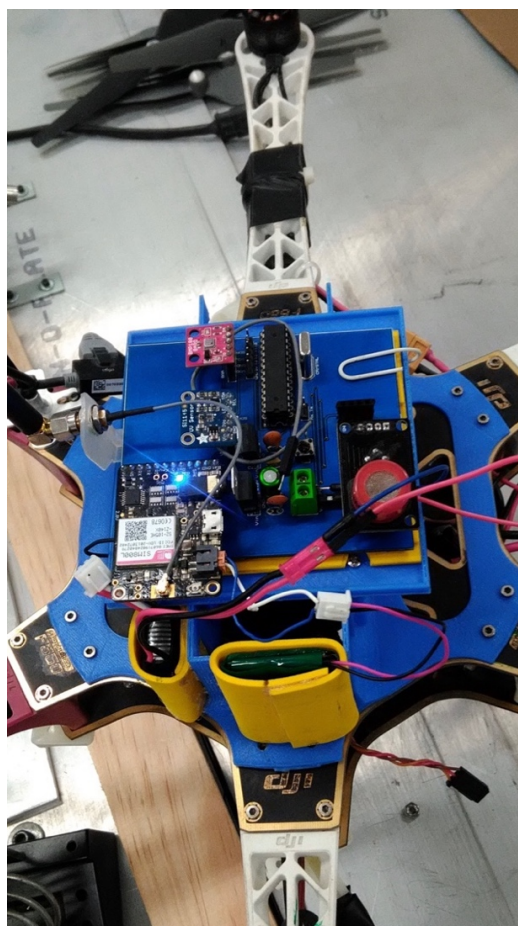


Figura 44 Foto 4 - Hardware de Monitorización sobre UAV

4.4 Integración con la plataforma de monitorización

Realizar la conexión del hardware de monitorización con la plataforma de monitorización es una parte fundamental para cumplir con el objetivo de este trabajo, contar con un sistema integral de monitorización basado en el paradigma del internet de las cosas, para lograrlo nuestra plataforma de monitorización deberá ser capaz de recibir en tiempo real a través de internet las mediciones adquiridas por el hardware de monitorización.

Como se mencionó anteriormente, para realizar la comunicación entre el hardware de monitorización y la plataforma web de monitorización se usa el protocolo MQTT el cual es parte fundamental de esta integración, a continuación se describen cada uno de los elementos principales dentro de esta integración:

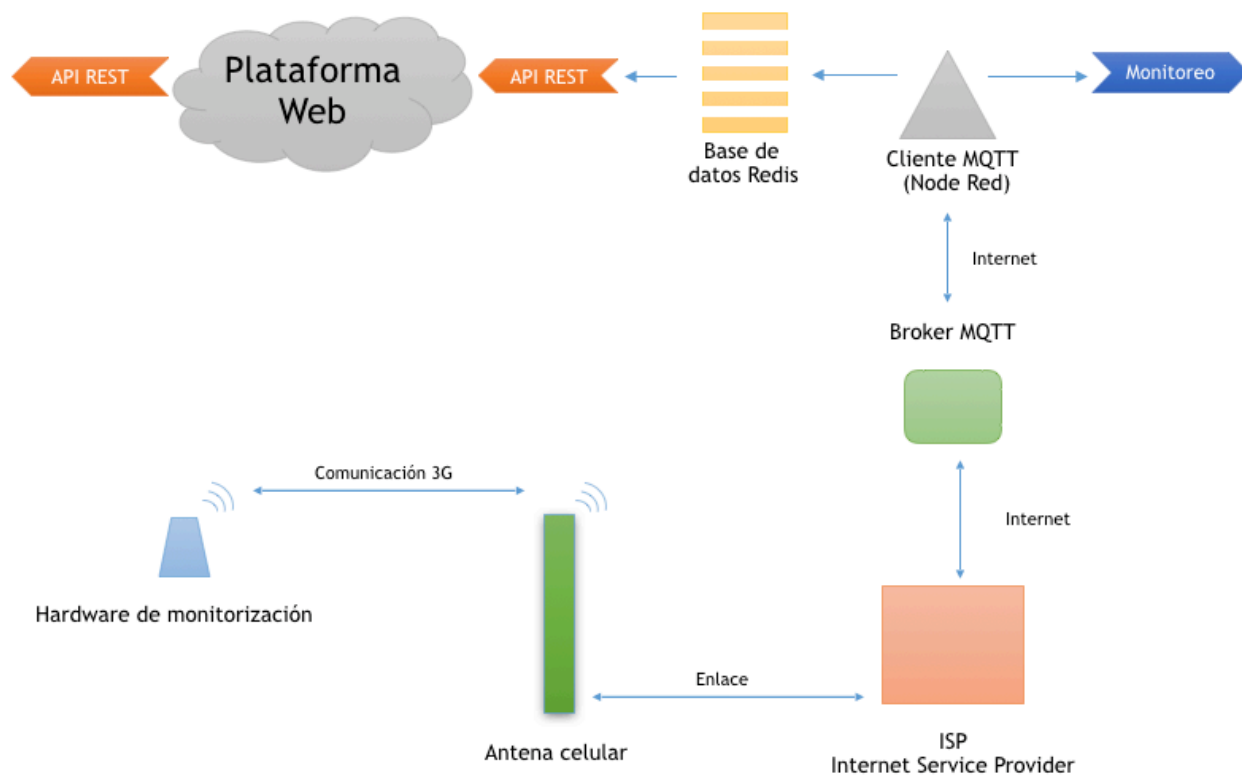


Figura 45 Diagrama de integración de la plataforma de monitorización

Para poder completar esta integración se requiere un puente entre los datos transmitidos por el hardware de monitoreo vía MQTT y la plataforma de monitorización, en la sección 3.4 se abordó la conexión con el panel de monitorización en tiempo real, sin embargo, en este caso el método de conexión con la plataforma de monitorización requiere algunos requerimientos adicionales.

La plataforma de monitorización como se explicó en capítulos anteriores es una aplicación web basada en Python con el framework Django y una base de datos relacional (MySQL), por lo cual su naturaleza la vuelve una plataforma no optimizada para trabajar con datos en tiempo real, si bien los objetivos de esta plataforma no involucra trabajar con los datos en tiempo real si es de suma importancia implementar

un método de conexión eficiente que evite pérdidas de datos o bien que pueda llegar a saturar y colapsar el servidor que hospeda esta plataforma, para lograr este objetivo se optó por el siguiente enfoque.

Consiste en un cliente MQTT basado en Node.js con el objetivo de ser eficiente en la recepción de los datos del hardware de monitoreo en tiempo real, los datos recibidos se almacenan en una base de datos no relacional que debido a su naturaleza es eficiente para guardar datos a grandes velocidades, en nuestro caso se usó Redis, el objetivo de esta base de datos es almacenar todos los datos de monitoreo en tiempo real y puedan ser enviados de forma más lenta y segura hacia la plataforma de monitorización.

Después del almacenamiento de los datos en Redis, se extraen los datos de la misma siguiendo un enfoque FIFO (First in, First out) es decir los datos son enviados a la plataforma de monitorización según su orden de llegada, para enviarlos a la plataforma de monitorización se hace uso del API “Agregar Datos” de la plataforma de monitorización, descrita en la sección 3.3.4.

Capítulo 5. Pruebas

Los objetivos de las pruebas del sistema de monitorización son dos, el primero es contar con medidas de referencia para poder comparar la precisión de nuestro sistema de monitorización con un sistema de monitoreo ambiental oficial y en operación, el segundo objetivo es realizar un vuelo de un UAV que lleve a bordo del sistema de monitorización para comprobar su correcto funcionamiento durante un vuelo real.

5.1 Pruebas en la estación de monitoreo ambiental “Ninfas”

Para el primer objetivo de las pruebas se solicitó una colaboración con el departamento de Calidad del Aire y Cambio Climático de la Secretaría de Desarrollo Rural, Sustentabilidad y Ordenamiento Territorial del Estado de Puebla, con el objetivo de tener acceso físico a una estación de monitoreo ambiental en la ciudad de Puebla, así como a los datos de las mediciones para poder realizar la comparación.

La estación de monitoreo ambiental a la cual se tuvo acceso fue la llamada “Ninfas” que se ubica en el parque homónimo en la ciudad de Puebla, su dirección es 15 Sur, Los Volcanes, 72410 Puebla, Puebla.



Figura 46 Foto 5 - Estación de Monitoreo Atmosférico "Ninfas"



Figura 47 Figura 57 Foto 6 - Estación de Monitoreo Atmosférico "Ninfas"

La metodología seguida fue realizar mediciones in situ para obtener datos en las mismas condiciones que la estación “Ninfas”, estas mediciones se realizaron por 7 días en diferentes horarios con el objetivo de obtener datos en diferentes condiciones del día.

5.1 Cronograma

A continuación se muestra el cronograma de monitorización con los días y horarios en los cuales se tomaron las mediciones, como se puede observar se intentó tener datos de 3 horas diferentes cada día, durante 7 días.

	Cronograma de monitoreo en estación Las Ninfas						
	Lunes 24/09/18	Martes 25/09/18	Miércoles 26/09/18	Jueves 27/09/18	Viernes 28/09/18	Lunes 01/10/18	Martes 02/10/18
9:00 - 9:59							
10:00 - 10:59							
11:00 - 11:59							
12:00 - 12:59							
13:00 - 13:59							
14:00 - 14:59							
15:00 - 15:59							
16:00 - 16:59							
17:00 - 17:59							
18:00 - 18:59							
19:00 - 19:59							
20:00 - 20:59							

Figura 48 Cronograma de monitoreo "Estación Ninfas"

5.2 Fotos

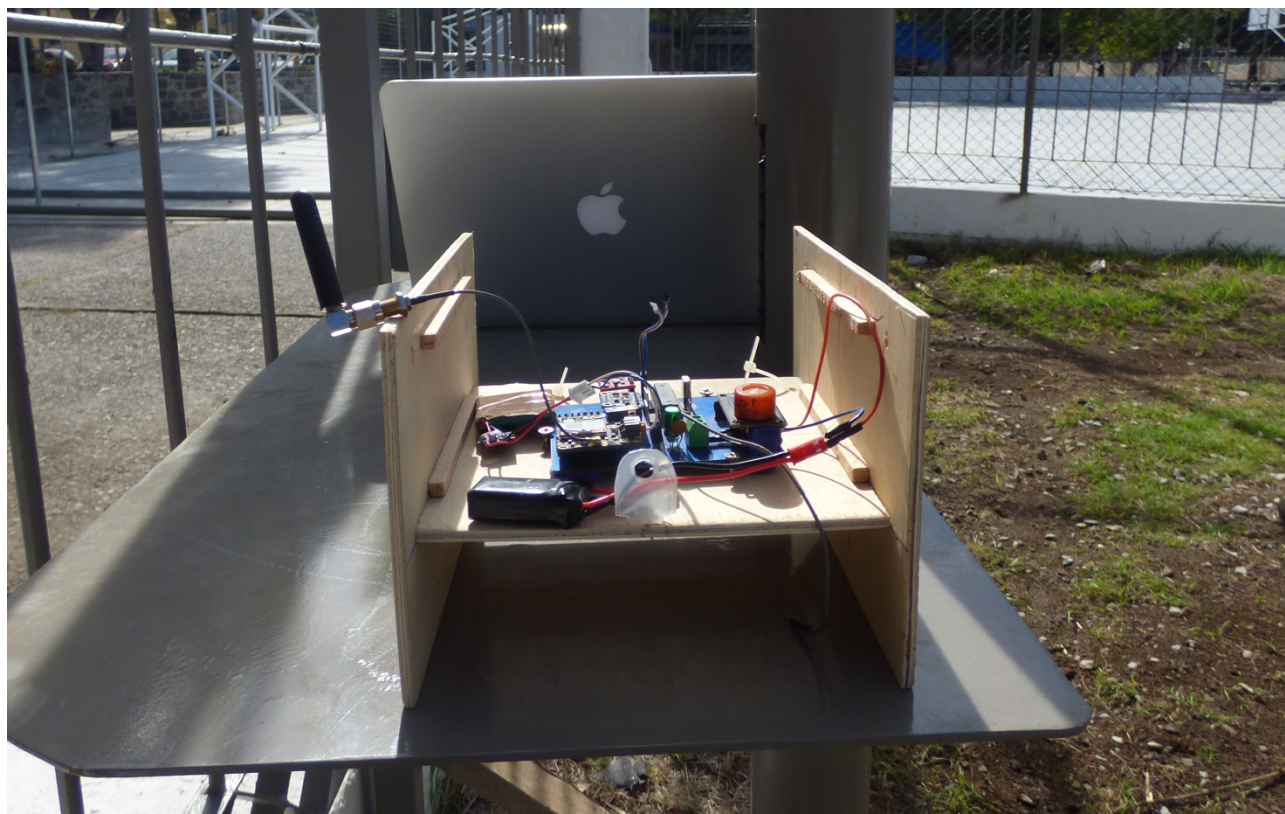


Figura 49 Foto 7 - Monitoreo en Estación "Ninfas"



Figura 50 Figura 60 Foto 8 - Monitoreo en Estación "Ninfas"

5.3 Resultados

Las primeras mediciones realizadas fueron el 24 de septiembre de 2018 en un horario de 10 am a 12 pm, donde se obtuvieron los siguientes datos en bruto.

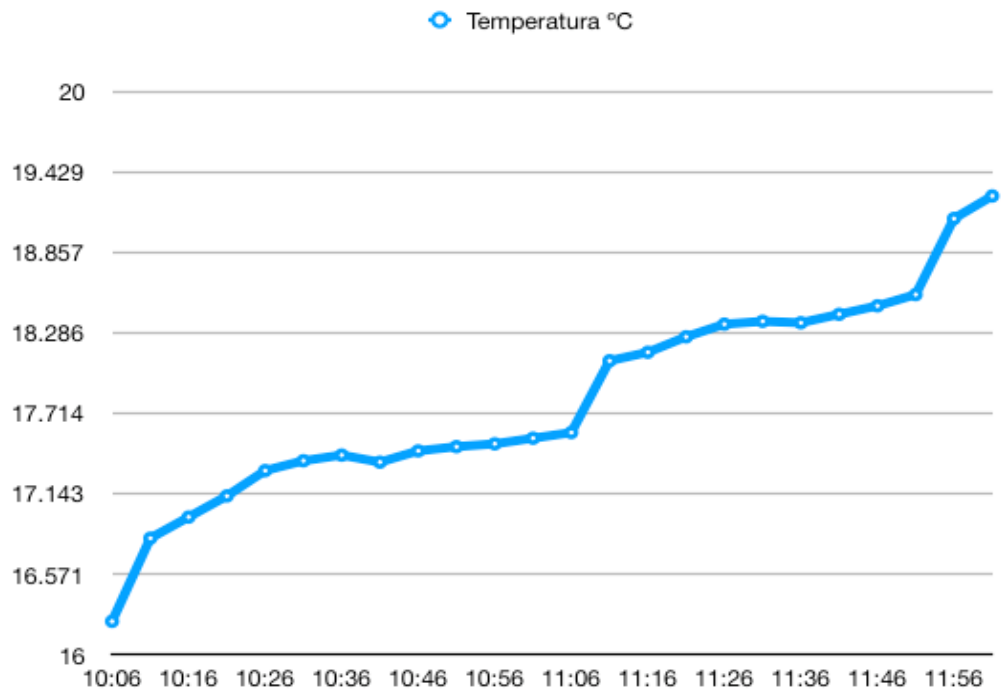


Figura 51 Grafica - Medición de temperatura en el 24 de septiembre de 2018

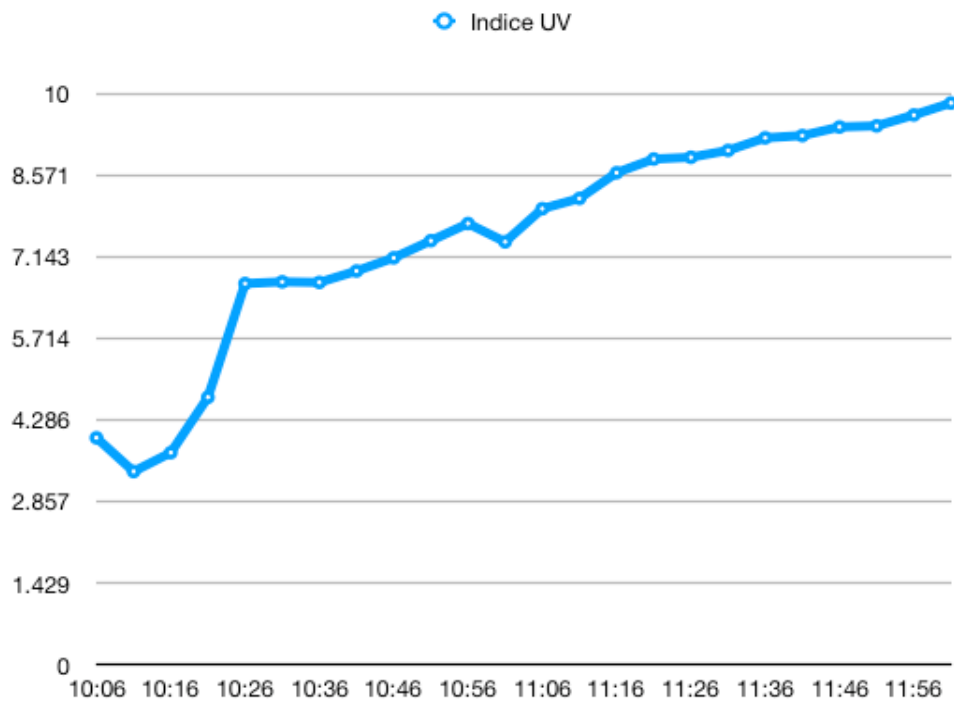


Figura 52 Grafica - Medición de Índice de Radiación UV el 24 de septiembre de 2018

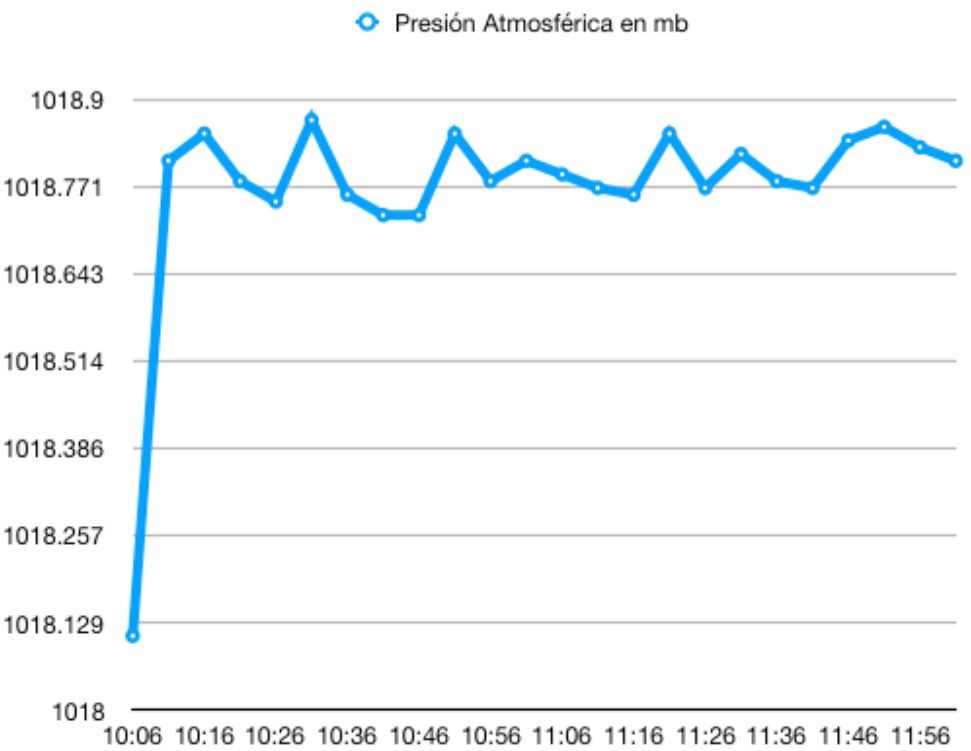


Figura 53 Grafica Medición de Presión atmosférica el 24 de septiembre de 2018



Figura 54 Grafica Medición de Monóxido de carbono el 24 de septiembre de 2018

Como se puede observar en las gráficas anteriores se obtuvieron datos de temperatura, presión atmosférica e Índice de radiación UV, sin embargo, no se obtuvieron datos de Monóxido de carbono.

Posteriormente al concluir las mediciones y recibir los datos de la estación de Monitorización Ninfas provistos por el departamento de Calidad del Aire y Cambio Climático de la Secretaría de Desarrollo Rural, Sustentabilidad y Ordenamiento Territorial del estado de Puebla, se procedió a realizar una comparación de los mismos.

Tenemos en primer lugar que existe una diferencia significativa de frecuencia en las mediciones, ya que la estación Ninfas provee datos cada hora y nuestro sistema de monitorización se configuro para obtener datos cada 10 minutos, lo que nos dejó un intervalo pequeño de comparación en donde los datos coincidieran en horario.

A continuación se puede observar la comparación de datos entre la estación Ninfas y nuestro sistema de monitorización, para horarios similares.

Estación Ninfas							Hardware de monitorización				
Date&Time	CO	Out	Hi	Low	Average	mb	Index	CO	mb	Index	
	PPM	Temperatura (°C)				Presión	UV	PPM	Temperatura (°C)	Presión	UV
24/09/2018 10:00	1.152	16.5	16.5	15.1	15.8	1014.1	1.5	0	16.24	1018.11	3.98
24/09/2018 11:00	1.058	18.6	18.6	16.5	17.55	1014.4	3.6	0	17.54	1018.81	7.41
24/09/2018 12:00	0.921	20.6	20.7	18.6	19.65	1014.2	4.3	0	19.24	1018.83	9.63

Figura 55 Tabla Comparación de datos con Estación "Ninfas"

Revisando los datos se puede observar que la concentración de Monóxido de carbono medido por la estación Ninfas se encuentra en un rango de 0.921 - 1.152 ppm, es por esta razón que nuestro sistema no fue capaz de medir la concentración de CO ya que el sensor utilizado MQ7 tiene un rango de entre 10 y 10,000 ppm.

Por lo cual se descartó el análisis de los datos de concentración de CO y solo se procedió a analizar los datos de temperatura, presión atmosférica e Índice de radiación UV.

Como se puede observar en los datos obtenidos, a simple vista existen diferencias significativas entre los datos de la estación ninfas y los de nuestro sistema de monitorización, por lo cual se procedió a calcular un coeficiente que nos permitiera ajustar la curva de los datos de nuestro sistema de monitorización para que los datos se acercarán lo mejor posible a los de la estación Ninfas, el proceso fue el siguiente:

- 1.- Se agruparon todas las mediciones por variable y se calculó la diferencia entre cada medición de nuestro sistema de monitorización y la estación ninfas.
- 2.-Por variable se calculó un promedio de todas las diferencias calculadas en el paso previo, para obtener un coeficiente de ajuste.

Los coeficientes obtenidos fueron los siguientes:

Temperatura: 1.07644005157512

Presión Atmosférica: 1.00445896601359

Índice de Radiación UV: 1.92980576458759

3.-Se ajustaron las mediciones de nuestro sistema de monitoreo para cada una de las variables a través del coeficiente obtenido en el paso previo.

En esta tabla se muestran las variables medidas (Temperatura, Presión atmosférica e Índice UV), tanto de la estación Ninfas, comparado con nuestro sistema de monitorización y el coeficiente de ajuste obtenido.

24/09/2018					
Temperatura					
Hora	Ninfas	Sistema de Monitorización	Ajuste Sistema		
10:00	15.8	16.24	No aplica		
11:00	17.55	17.54	No aplica		
12:00	19.65	19.24	No aplica		
Presión					
Hora	Ninfas	Sistema de Monitorización			
10:00	1014.1	1018.11	1013.59043469997		
11:00	1014.4	1018.81	1014.28732727964		
12:00	1014.2	1018.83	1014.3072384962		
Índice UV					
Hora	Ninfas	Sistema de Monitorización			
10:00	1.5	3.98	2.0623837243281		
11:00	3.6	7.41	3.83976467268122		
12:00	4.3	9.63	4.99013951388935		

Figura 56 Tabla Mediciones del día 24/09/18

En esta grafica se puede observar la comparación de temperatura medida por la estación Ninfas y el sistema de monitorización desarrollado.

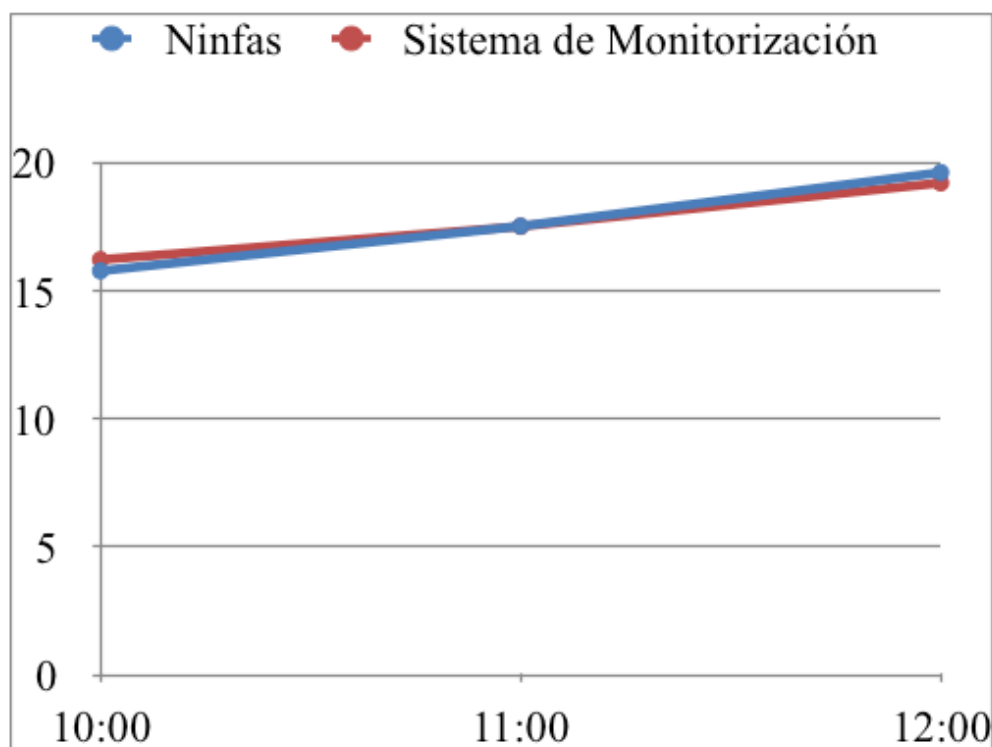


Figura 57 Grafica Temperatura (°C): 24/09/18

En esta grafica se puede observar la comparación de presión atmosférica medida por la estación Ninfas y el sistema de monitorización desarrollado, así como su ajuste de curva a través del coeficiente calculado.

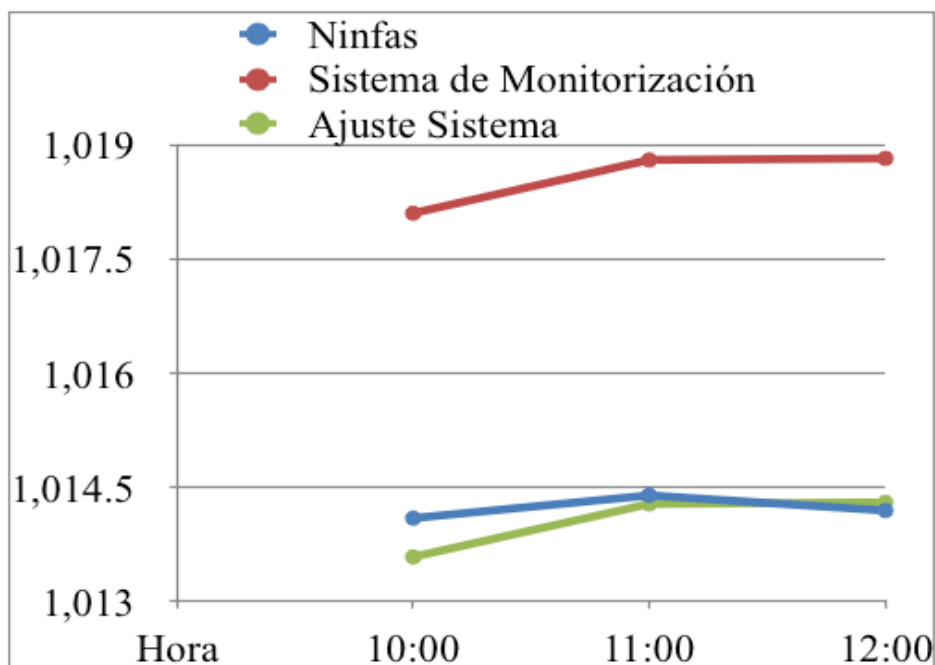


Figura 58 Grafica Presión Atmosférica (mb): 24/09/18

En esta grafica se puede observar la comparación del Índice de radiación UV medido por la estación Ninfas y el sistema de monitorización desarrollado, así como su ajuste de curva a través del coeficiente calculado.

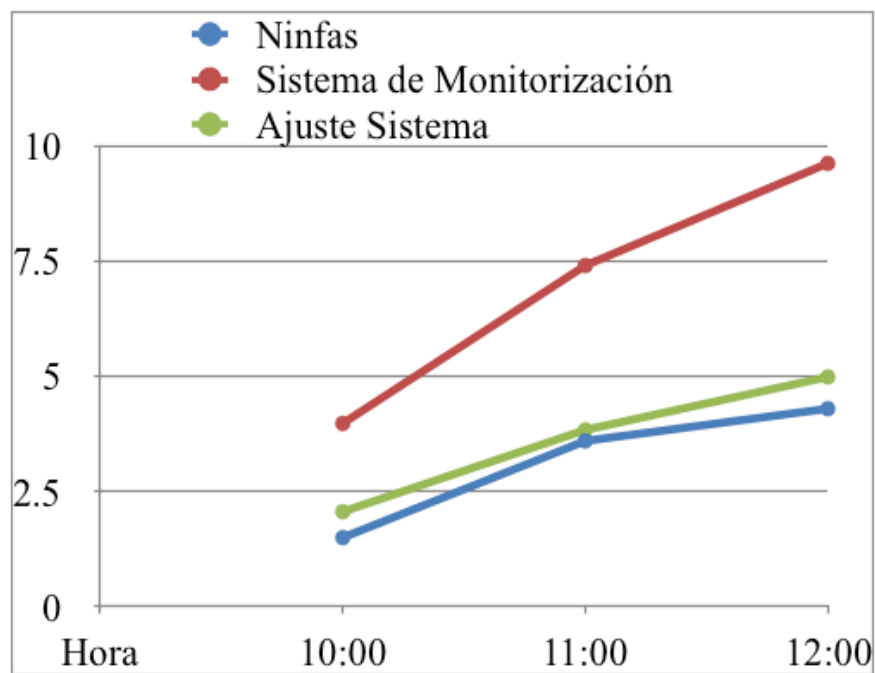


Figura 59 Grafica Índice de radiación UV: 24/09/18

Capítulo 5. Pruebas

Las siguientes tablas y gráficas, contienen los mismos datos descritos anteriormente, para cada uno de los días que se realizaron mediciones.

25/09/2018

Temperatura				
Hora	Ninfas	Sistema de Monitorización	Ajuste Sistema	
12:00	21.3	23.45	No aplica	
13:00	22.25	28.82	No aplica	
14:00	23.1	28.16	No aplica	
Presión				
Hora	Ninfas	Sistema de Monitorización		
12:00	1013.0	1017.52	1013.00305381139	
13:00	1011.9	1017.26	1012.74420799608	
14:00	1010.4	1016.34	1011.82829203423	
Índice UV				
Hora	Ninfas	Sistema de Monitorización		
12:00	1.9	5.54	2.87075523436625	
13:00	2.3	5.94	3.07802998052988	
14:00	2.7	6.23	3.22830417149851	

Figura 60 Tabla Mediciones del día 25/09/18

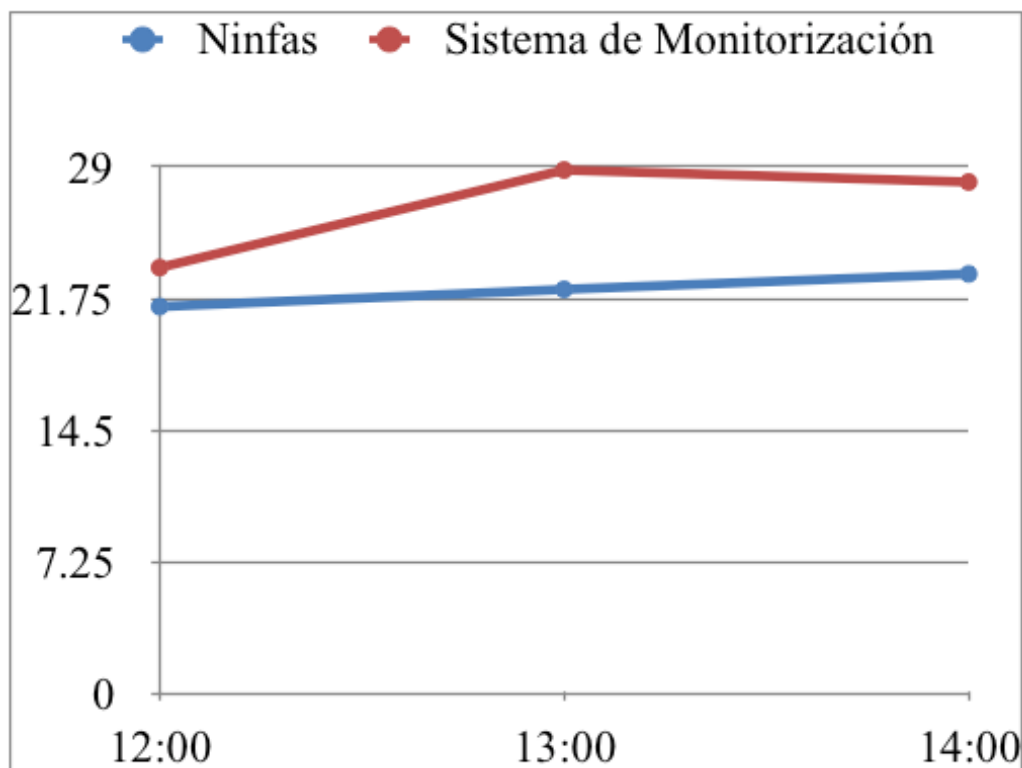


Figura 61 Gráfica Temperatura (°C): 25/09/18

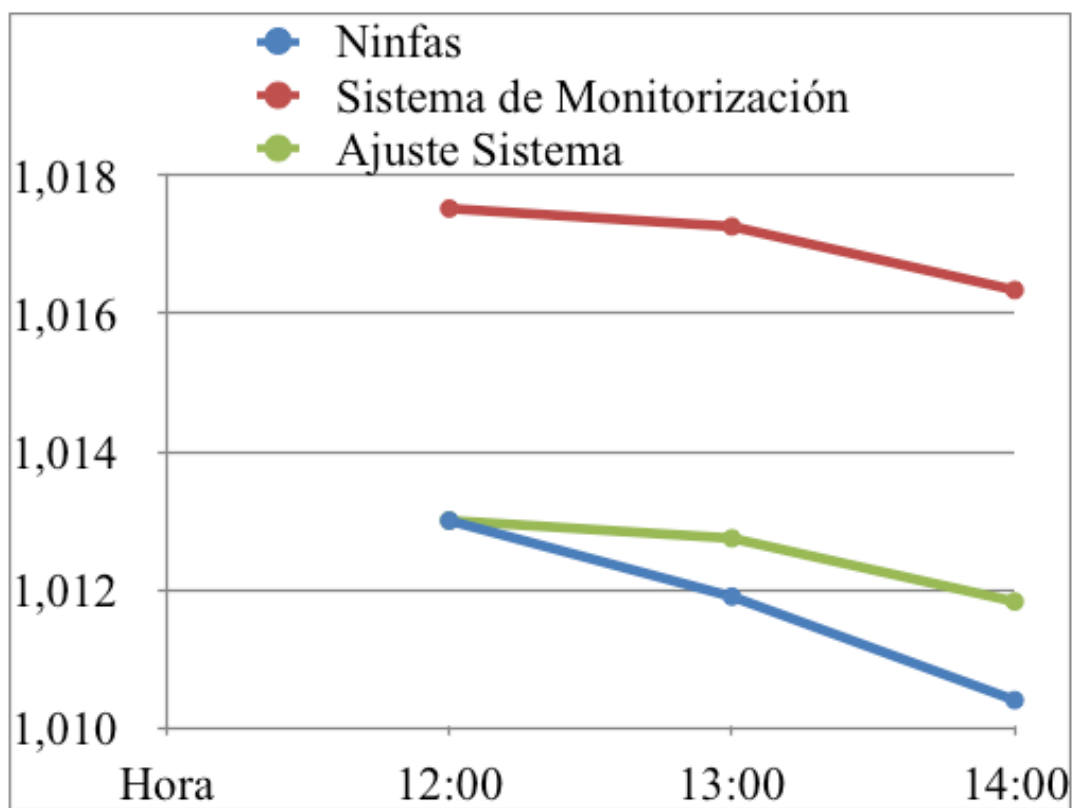


Figura 62 Grafica Presión Atmosférica (mb): 25/09/18

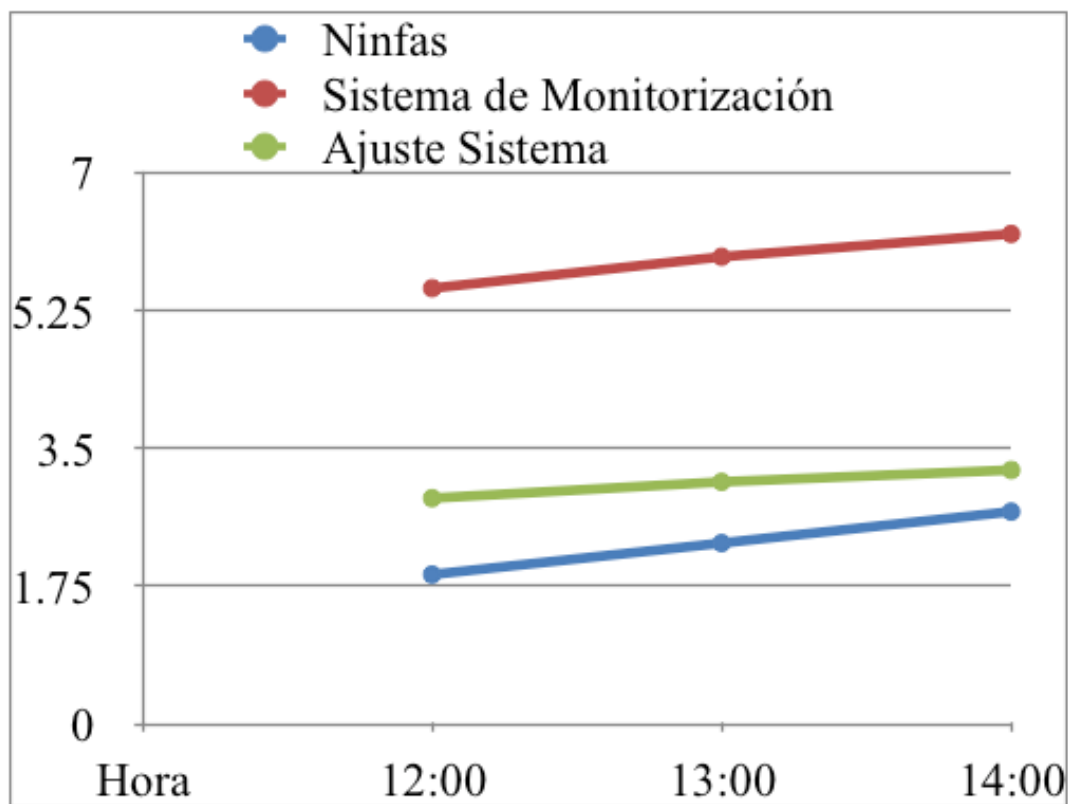


Figura 63 Grafica Índice de radiación UV: 25/09/18

26/09/2018

Temperatura				
Hora	Ninfas	Sistema de Monitorización	Ajuste Sistema	
14:00	22.85	24	22.2957144384228	
15:00	23.9	24	22.2957144384228	
16:00	24.1	25	23.2247025400237	
Presión				
Hora	Ninfas	Sistema de Monitorización		
14:00	1013.1	1017.32	1012.80394164577	
15:00	1012.4	1016.66	1012.14687149922	
16:00	1011.6	1017.18	1012.66456312983	
Indice UV				
Hora	Ninfas	Sistema de Monitorización		
14:00	5.1	8.24	4.26985977097075	
15:00	2.2	7.56	3.91749270249258	
16:00	5.6	8.46	4.38386088136074	

Figura 64 Tabla Mediciones del día 26/09/18

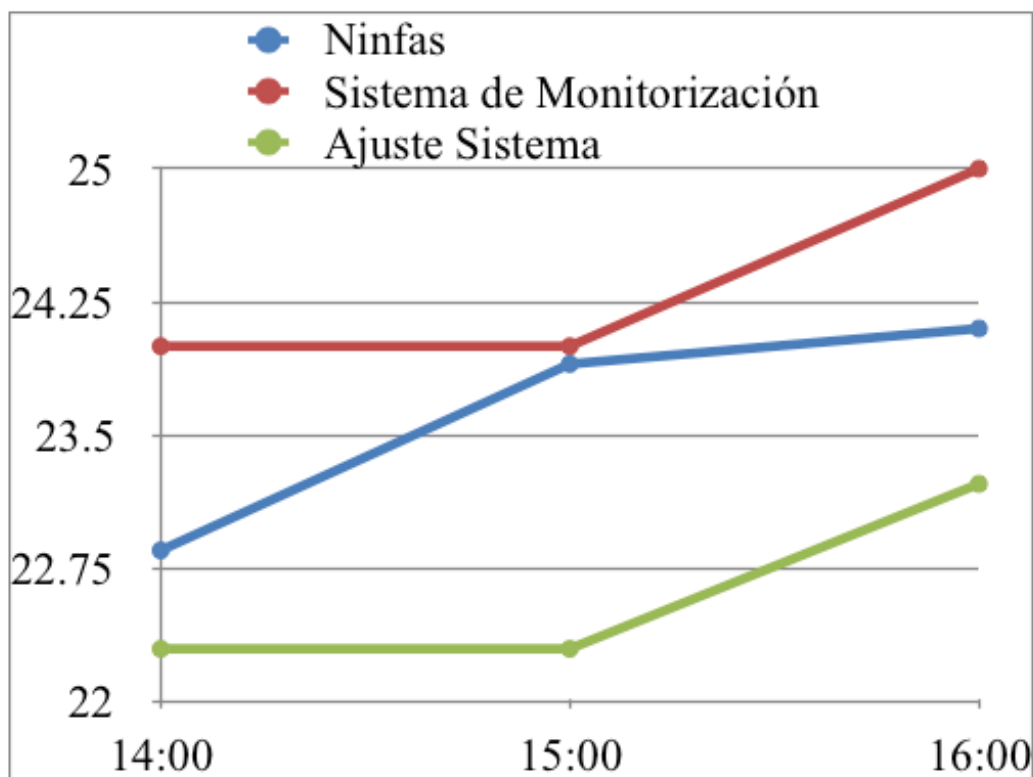


Figura 65 Grafica Temperatura (°C): 26/09/18

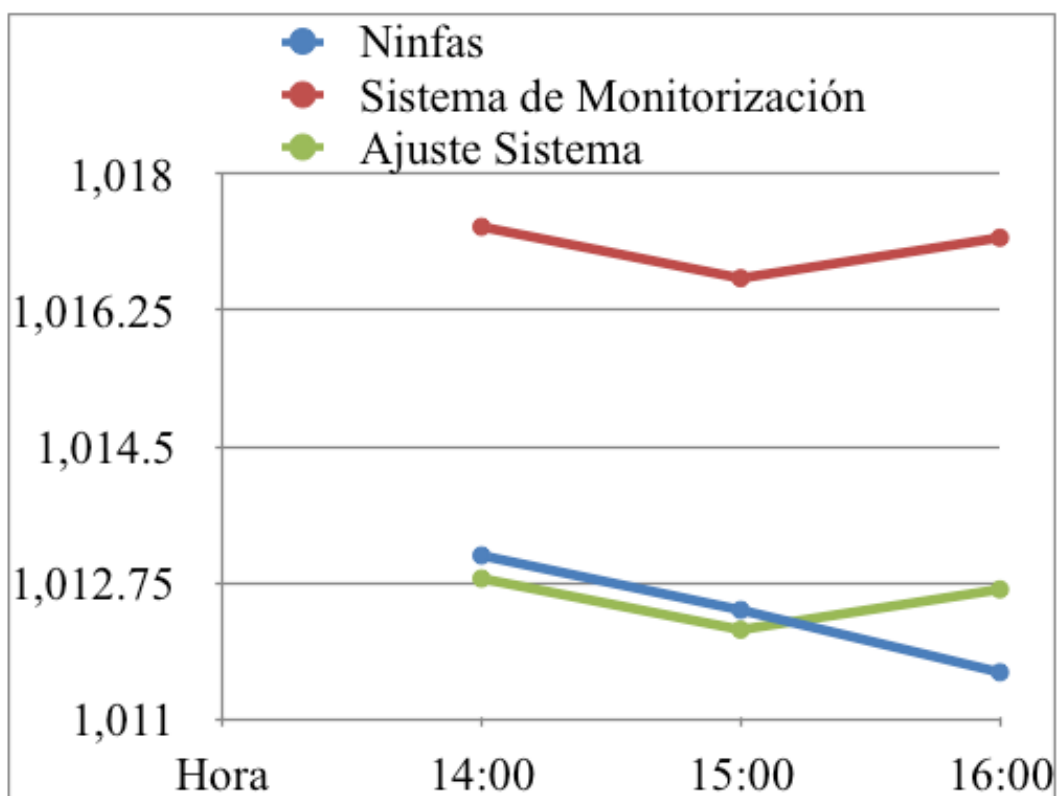


Figura 66 Gráfica Presión Atmosférica (mb): 26/09/18

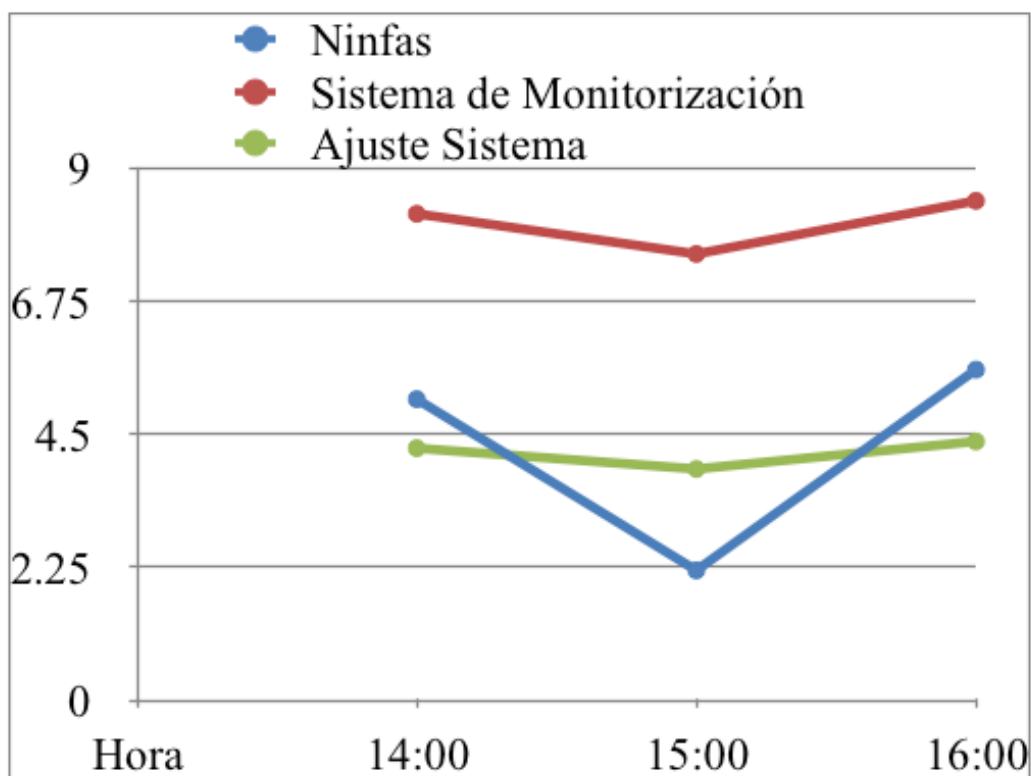


Figura 67 Gráfica Índice de radiación UV: 26/09/18

27/09/2018

Temperatura				
Hora	Ninfas	Sistema de Monitorización	Ajuste Sistema	
18:00	20.75	22	20.4377382352209	
19:00	20.65	22	20.4377382352209	
20:00	19.8	21	19.5087501336199	
Presión				
Hora	Ninfas	Sistema de Monitorización		
18:00	1009.6	1015.26	1010.75308633988	
19:00	1010.7	1015.74	1011.23095553737	
20:00	1011.4	1016.35	1011.83824764251	
Indice UV				
Hora	Ninfas	Sistema de Monitorización		
18:00	1.0	0.08	0.0414549492327257	
19:00	0.0	0.07	0.036273080578635	
20:00	0.0	0.06	0.0310912119245443	

Figura 68 Tabla Mediciones del día 27/09/18

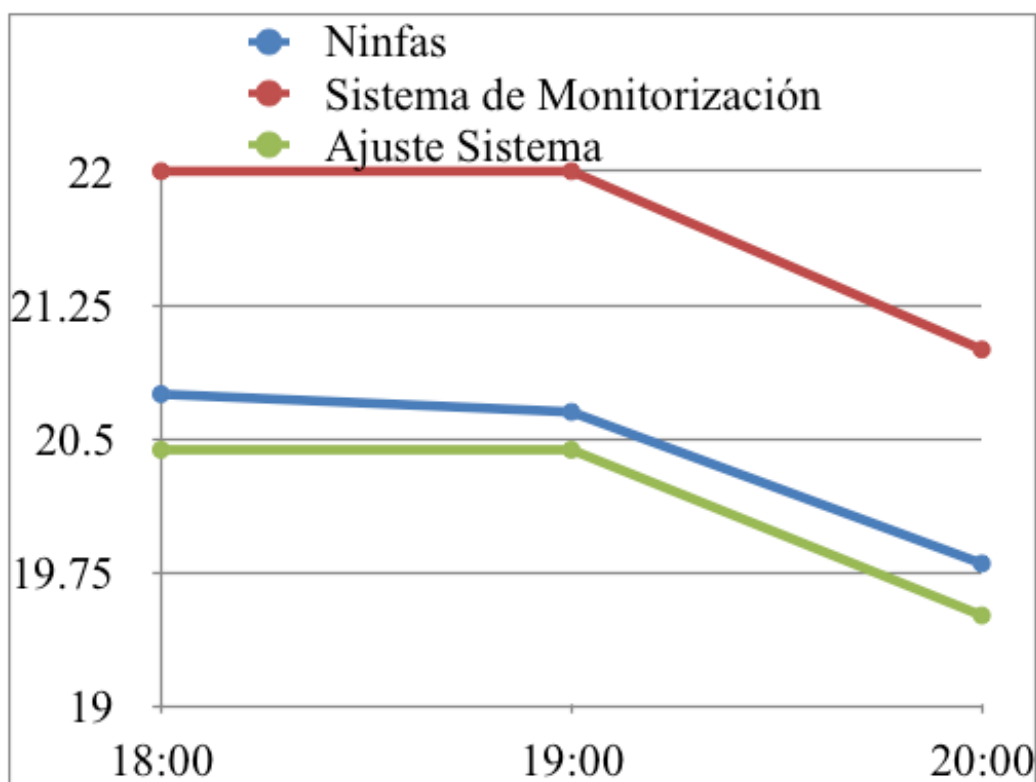


Figura 69 Grafica Temperatura (°C): 27/09/18

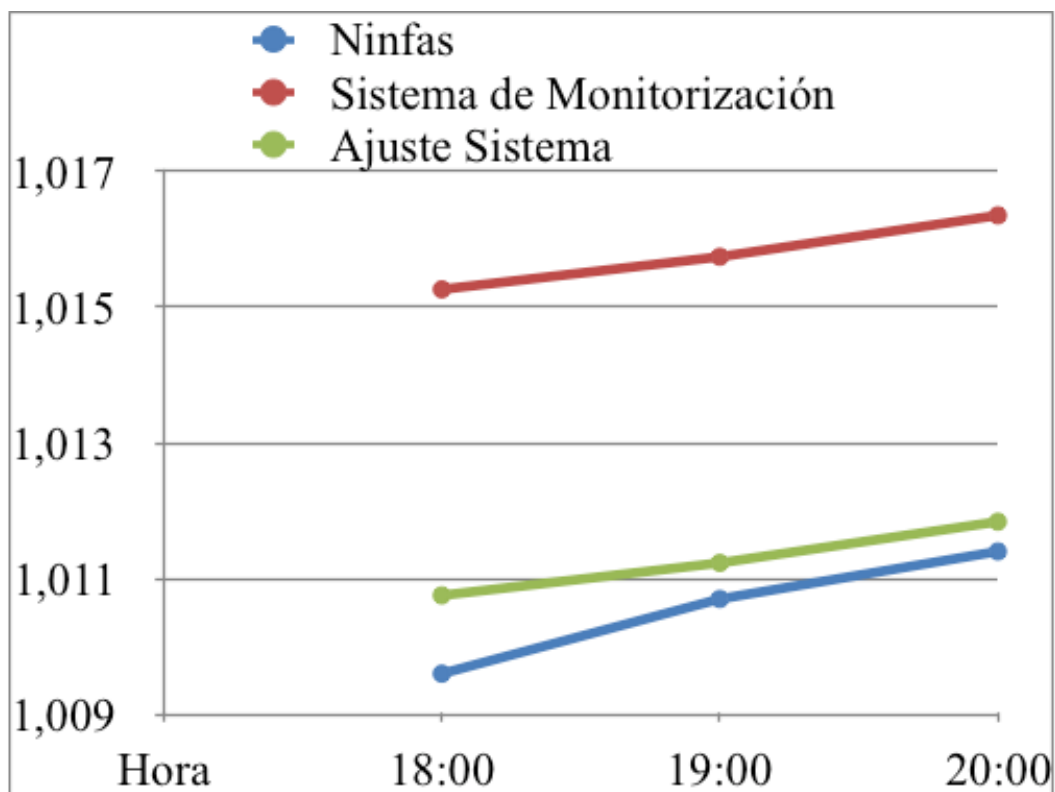


Figura 70 Grafica Presión Atmosférica (mb): 27/09/18

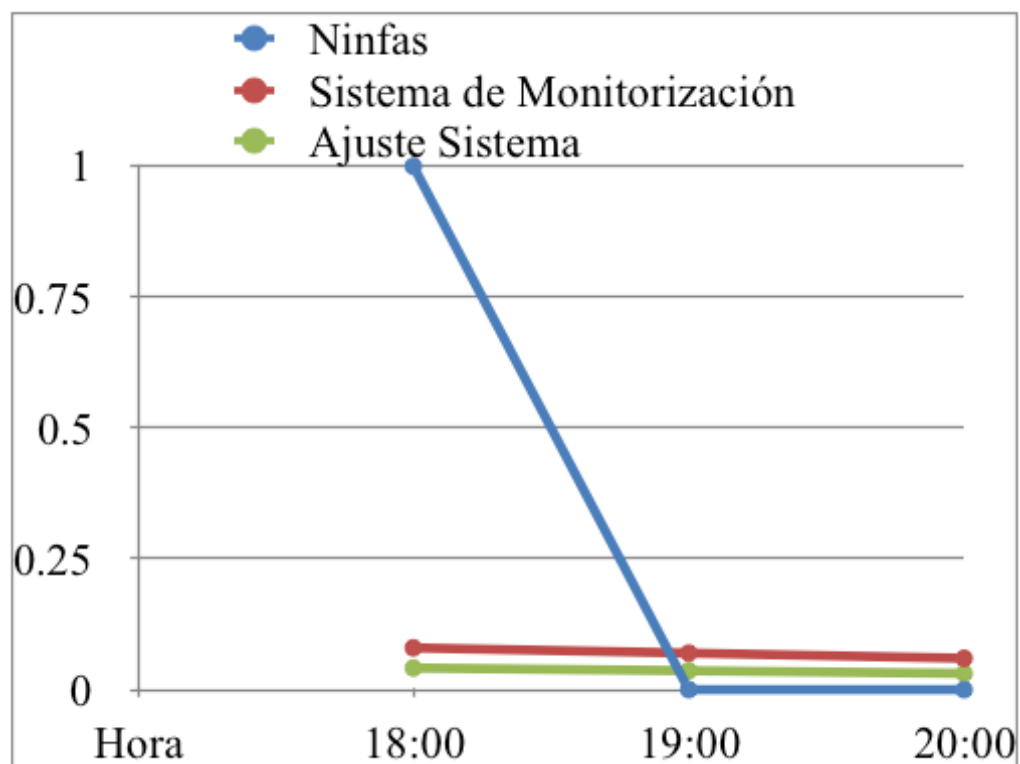


Figura 71 Grafica Índice de radiación UV: 27/09/18

28/09/2018

Temperatura				
Hora	Ninfas	Sistema de Monitorización	Ajuste sistema	
17:00	19.5	21	19.5087501336199	
18:00	19.1	21	19.5087501336199	
19:00	19	21	19.5087501336199	
Presión				
Hora	Ninfas	Sistema de Monitorización		
17:00	1012.1	1016.16	1011.64909108517	
18:00	1012.2	1016.63	1012.11700467438	
19:00	1012.6	1016.87	1012.35593927312	
Indice UV				
Hora	Ninfas	Sistema de Monitorización		
17:00	0.8	1	0.518186865409071	
18:00	0.0	0.06	0.0310912119245443	
19:00	0.0	0.04	0.0207274746163628	

Figura 72 Tabla Mediciones del día 28/09/18

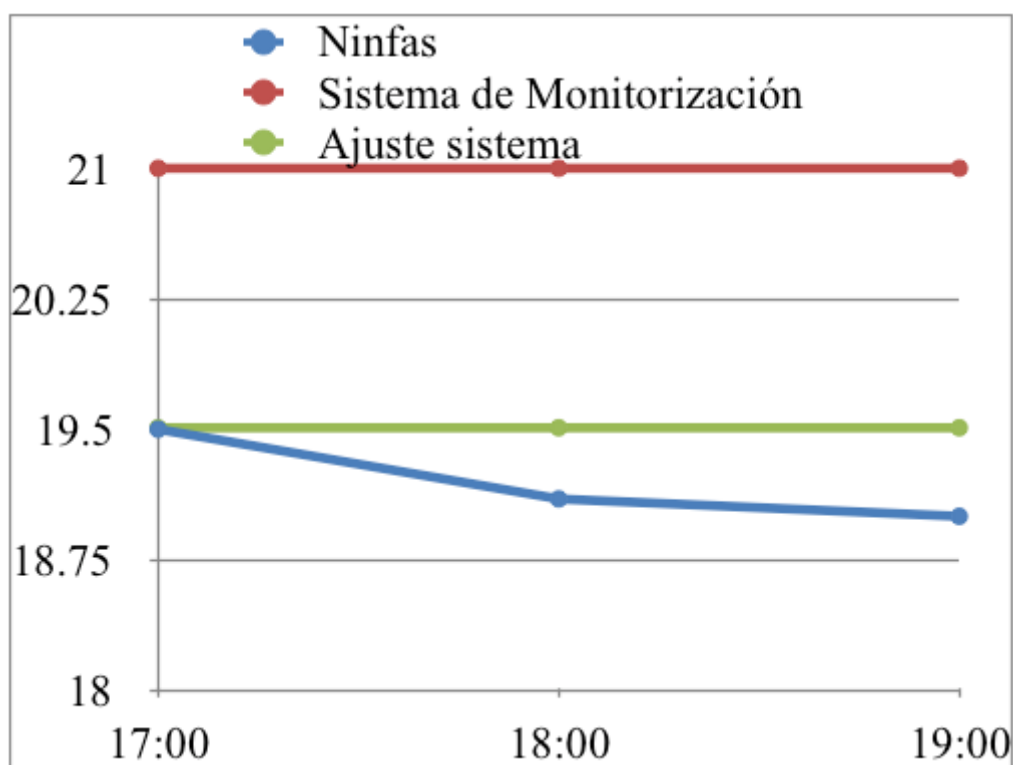


Figura 73 Grafica Temperatura (°C): 28/09/18

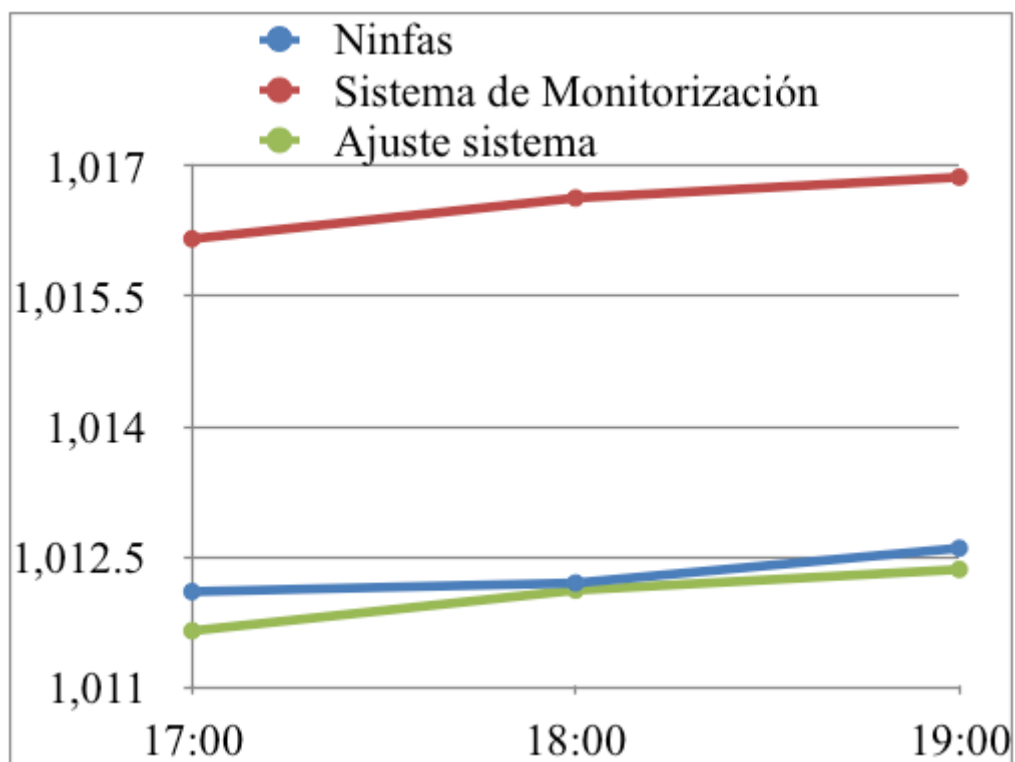


Figura 74 Grafica Presión Atmosférica (mb): 28/09/18

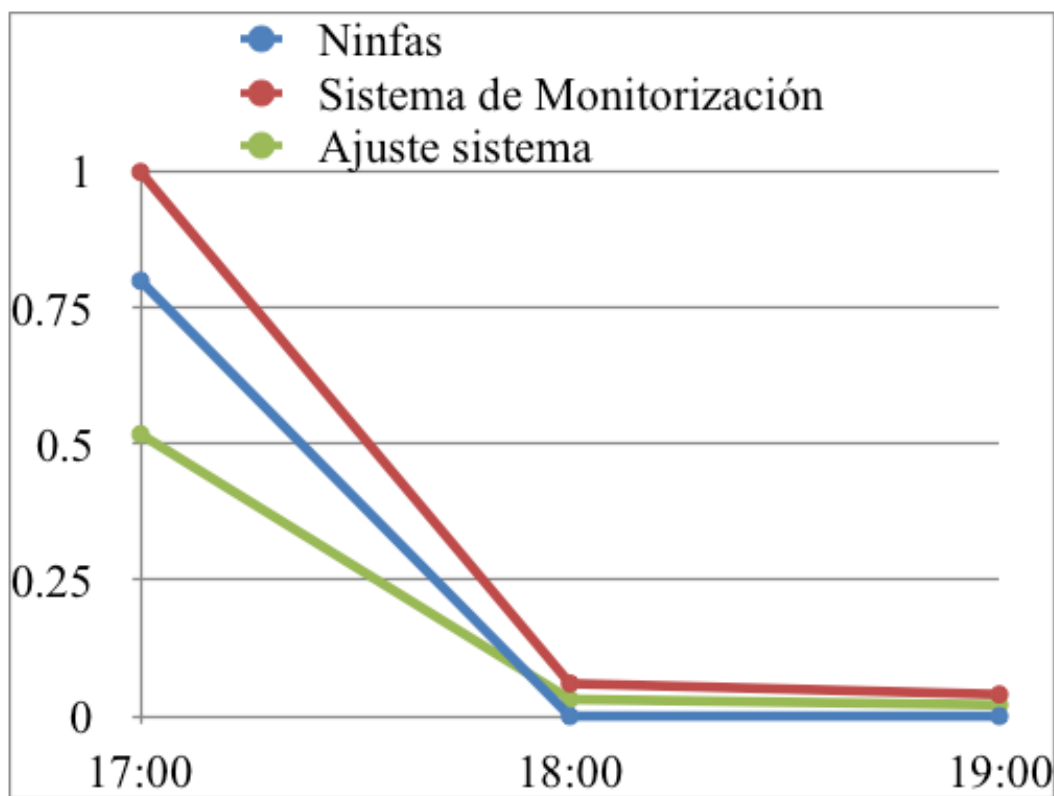


Figura 75 Grafica Índice de radiación UV: 28/09/18

01/10/2018

Temperatura				
Hora	Ninfas	Sistema de Monitorización	Ajuste sistema	
9:00	15.05	17	15.7927977272161	
10:00	16.8	19	17.650773930418	
11:00	17.9	20	18.579762032019	
Presión				
Hora	Ninfas	Sistema de Monitorización	Ajuste sistema	
9:00	1015.1	1019.36	1014.8348857351	
10:00	1015.5	1019.21	1014.68555161088	
11:00	1015.6	1018.84	1014.31719410449	
Indice UV				
Hora	Ninfas	Sistema de Monitorización	Ajuste sistema	
9:00	0.5	2.56	1.32655837544722	
10:00	1.7	3.39	1.75665347373675	
11:00	3.4	7.49	3.88121962191394	

Figura 76 Tabla Mediciones del día 01/10/18

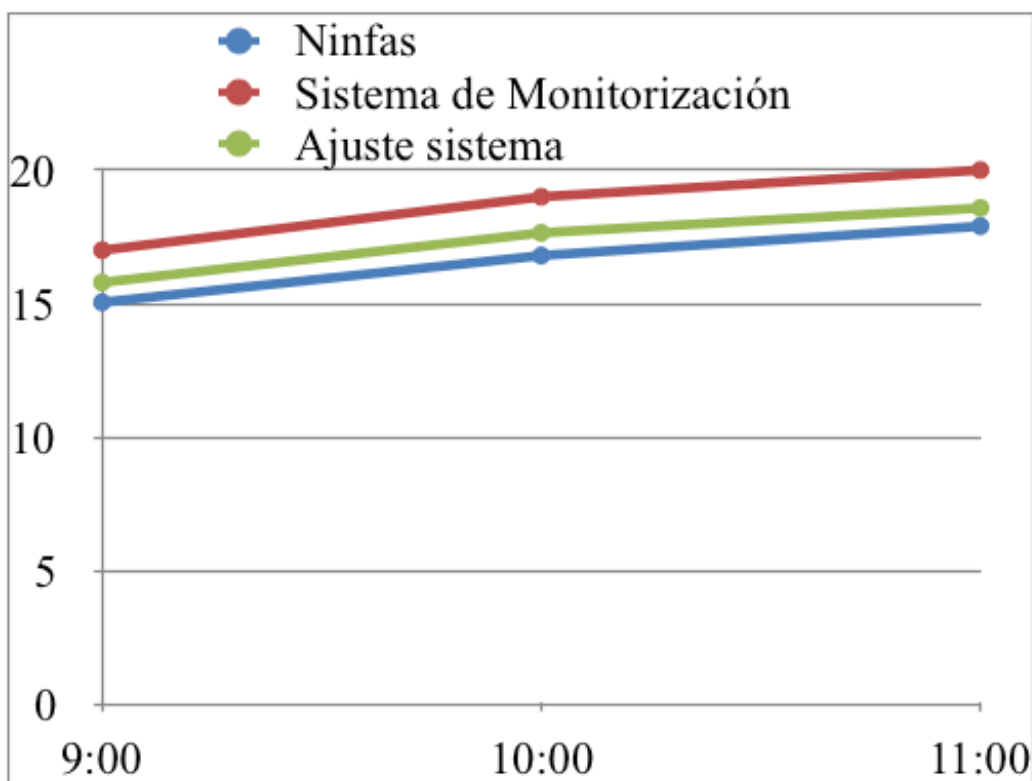


Figura 77 Grafica Temperatura (°C): 01/10/18

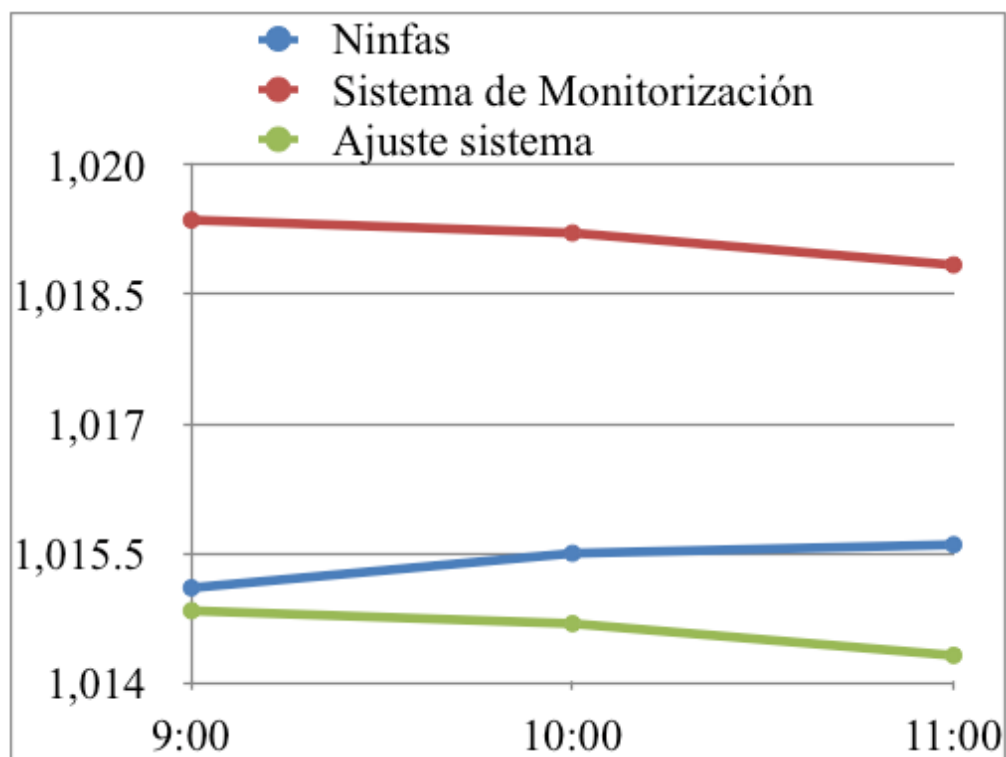


Figura 78 Grafica Presión Atmosférica (mb): 01/10/18

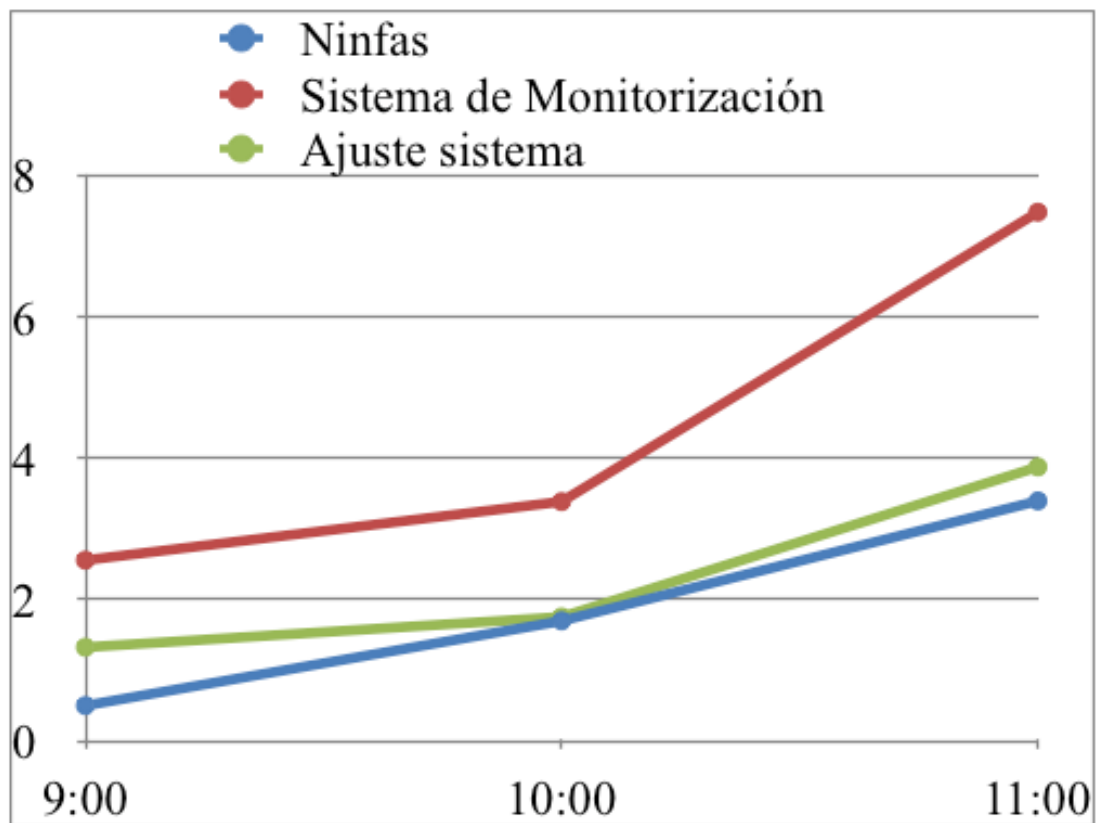


Figura 79 Grafica Índice de radiación UV: 01/10/18

02/10/2018

Temperatura				
Hora	Ninfas	Sistema de Monitorización	Ajuste sistema	
14:00	22.7	25	23.2247025400237	
15:00	22.95	24	22.2957144384228	
16:00	22.6	24	22.2957144384228	
Presión				
Hora	Ninfas	Sistema de Monitorización		
14:00	1013.0	1016.46	1011.9477593336	
15:00	1012.4	1016.73	1012.21656075719	
16:00	1011.4	1015.87	1011.36037844502	
Indice UV				
Hora	Ninfas	Sistema de Monitorización		
14:00	2.2	5.45	2.82411841647944	
15:00	2.2	6.78	3.5133069474735	
16:00	2.5	7.50	3.88640149056803	

Figura 80 Tabla Mediciones del día 02/10/18

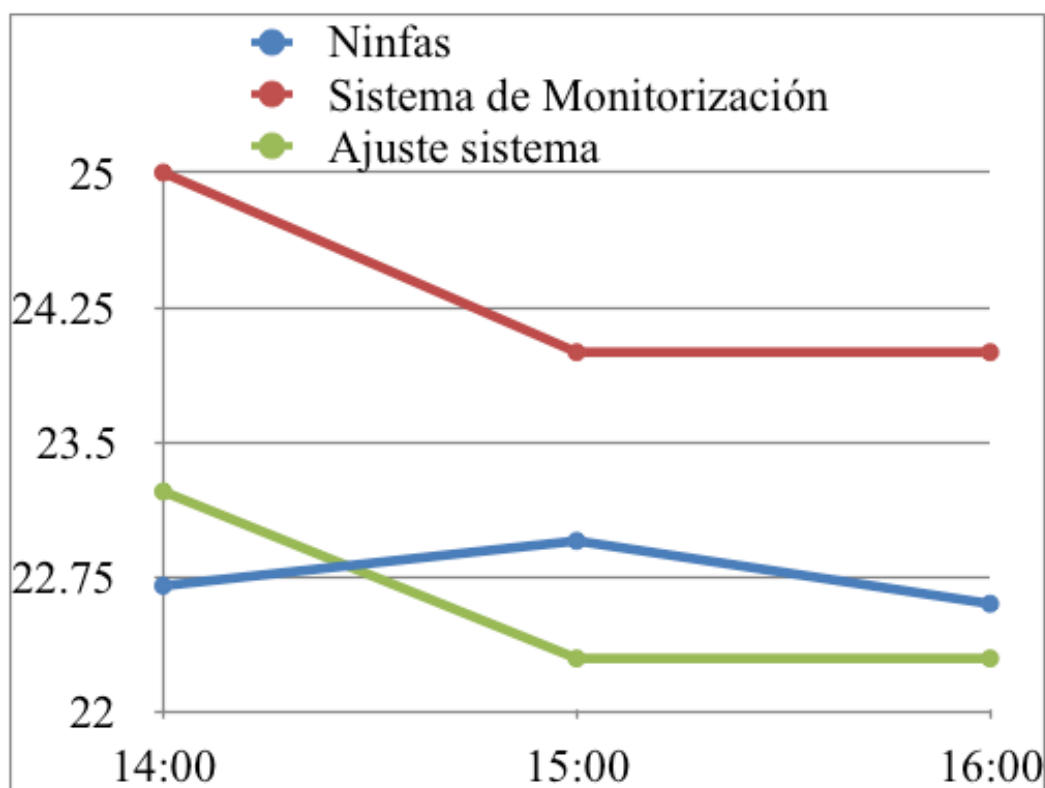


Figura 81 Grafica Temperatura (°C): 02/10/18

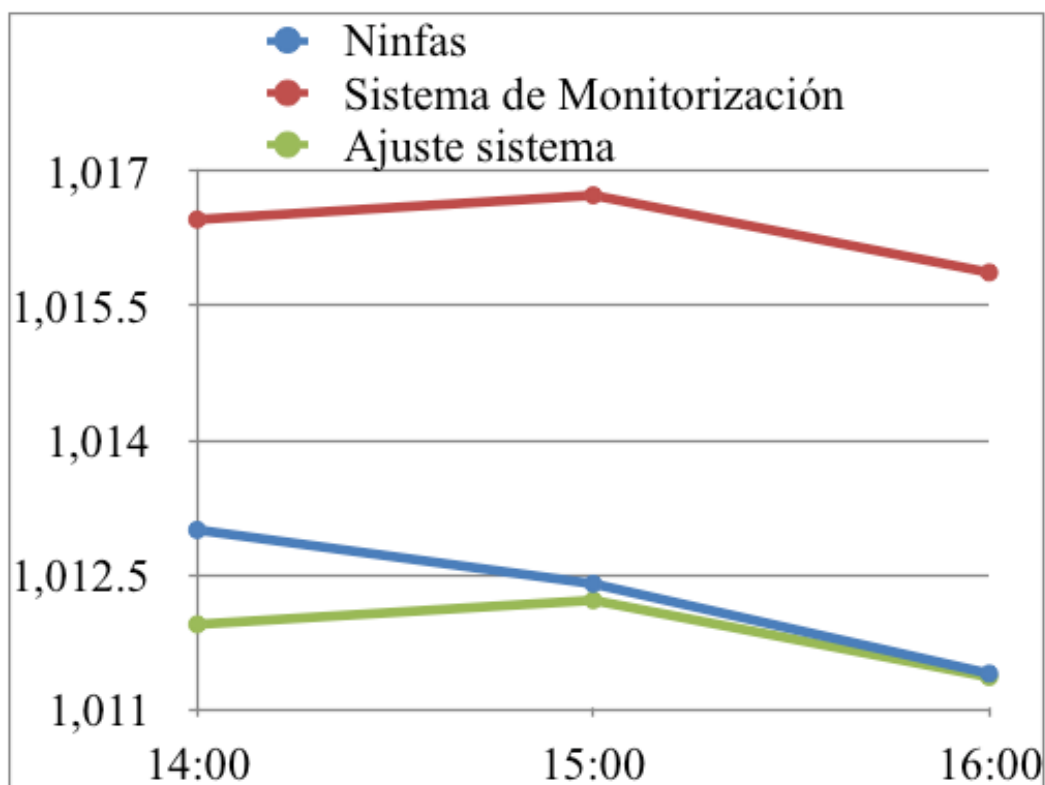


Figura 82 Grafica Presión Atmosférica (mb): 02/10/18

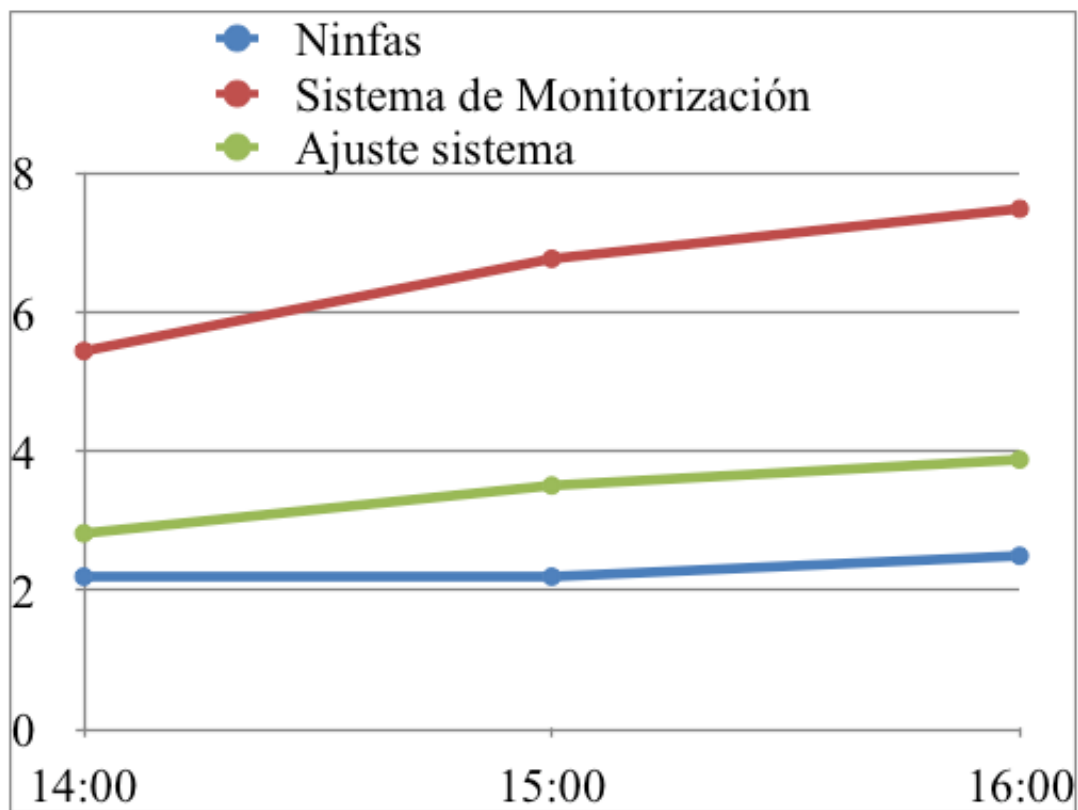


Figura 83 Grafica Índice de radiación UV: 02/10/18

5.2 Prueba de vuelo sobre UAV

Esta prueba se realizó con el objetivo de comprobar el correcto funcionamiento del sistema de monitorización a bordo de un UAV en pleno vuelo, en resumen, que los sensores se comportarán de forma correcta, el sistema de comunicación GPRS enviará los datos sin pérdidas y que no existirá algún tipo de interferencia con los sistemas del UAV en pleno vuelo.

Para esta prueba el requerimiento principal del UAV a usar es que fuera capaz de cargar el peso del hardware de monitorización y no poner en riesgo la integridad del sistema y del UAV, para ello se usó un UAV comercial lo suficientemente robusto para poder llevar a bordo el hardware de monitorización, se trató del Phantom 4 Advanced que tiene las siguientes especificaciones:

Peso (batería y hélices incluidas): 1368 g
Velocidad máxima de ascenso: 6 m/s
Velocidad máxima de descenso: 4 m/s
Velocidad máxima horizontal: 72 km/h
Resistencia al viento máxima: 10 m/s
Tiempo de vuelo máximo: 30 minutos aproximadamente
Sistema de posicionamiento por satélite: GPS/GLONASS
Rango de temperatura de funcionamiento: De 0 a 40 °C
Batería: 5870 mAh

Debido a la estructura del UAV, se tuvo que construir una base de soporte externa al UAV para poder llevar a bordo el hardware de monitorización de forma segura, se construyó de madera tipo triplay, se fijó al UAV en la parte inferior.



Figura 84 Vista del UAV con el sistema de monitorización acoplado

Debido a la posición del hardware de monitorización en la parte inferior del UAV se omitió el sensor de índice de radiación UV ya que al no tener contacto con la luz solar no brindaba datos relevantes.

En las pruebas preliminares de vuelo se encontró que la carga extra en el UAV, aumentaba el grado de dificultad en el control del mismo, así como también aumentaba el consumo de la batería, por lo tanto tomando medidas de precaución el vuelo se realizó dentro del campo de béisbol de ciudad universitaria de la BUAP, por seguridad se elevó a una altura máxima de 62 metros y siempre se permaneció dentro del perímetro del campo, el vuelo duro 9 minutos.

Una vez comprobado con éxito el funcionamiento correcto del sistema de monitorización, se finalizó el vuelo con un 41% de batería restante.



Figura 85 Despegue durante la prueba de vuelo



Figura 86 UAV y sistema de monitorización durante la prueba de vuelo



Figura 87 Vista de Ciudad Universitaria BUAP, durante la prueba de vuelo

A continuación se pueden observar los resultados obtenidos.

Los datos de altura de la siguiente grafica se obtuvieron del registro de vuelo del UAV debido a que su sistema de navegación es robusto la altura registrada es de alta precisión, además de tener una referencia de datos externa al sistema de monitorización.

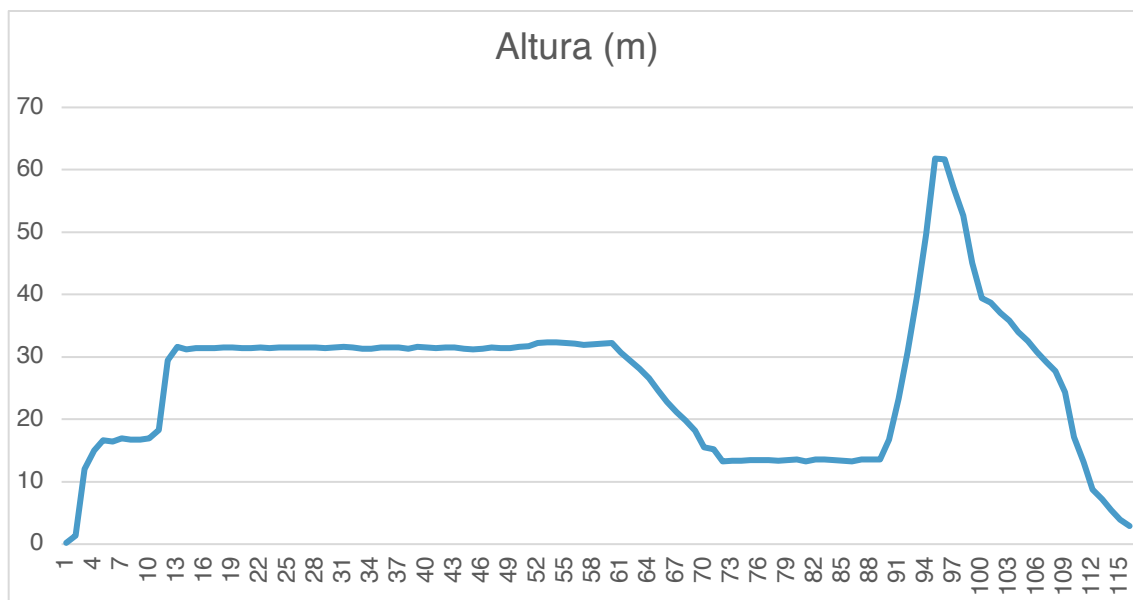


Figura 88 Grafica de la Altura durante el vuelo de prueba

Como se sabe la presión atmosférica es inversamente proporcional a la altitud, esta correlación se puede observar comparando la gráfica anterior y la siguiente, por lo cual se puede comprobar que el registro de presión atmosférica fue correcto.

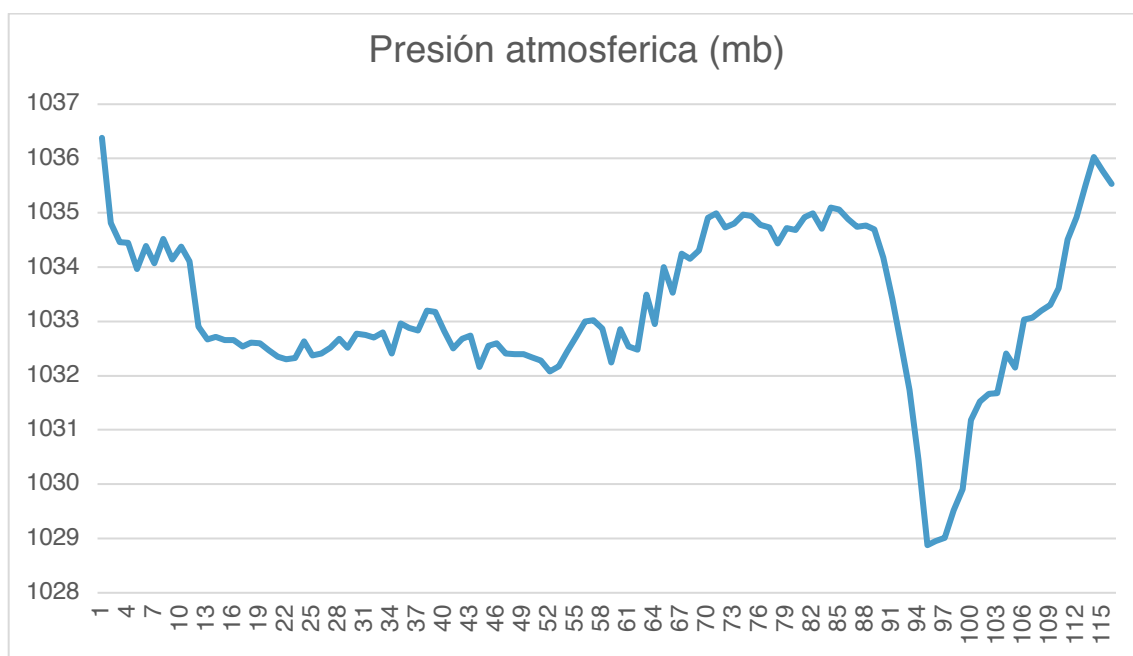


Figura 89 Grafica de la presión atmosférica durante la prueba de vuelo

A continuación se puede observar la gráfica de temperatura durante el vuelo, según la ley de los gases a un volumen constante la temperatura es proporcional a la presión, si se compara con la gráfica anterior de presión atmosférica se puede observar esta tendencia sin embargo no es tan evidente como la comparación altura vs presión atmosférica, esto se debe a que la temperatura en el medio ambiente no está sujeta 100% a la ley de gases, es afectada muchos otros factores como la radiación solar o corrientes de aire.

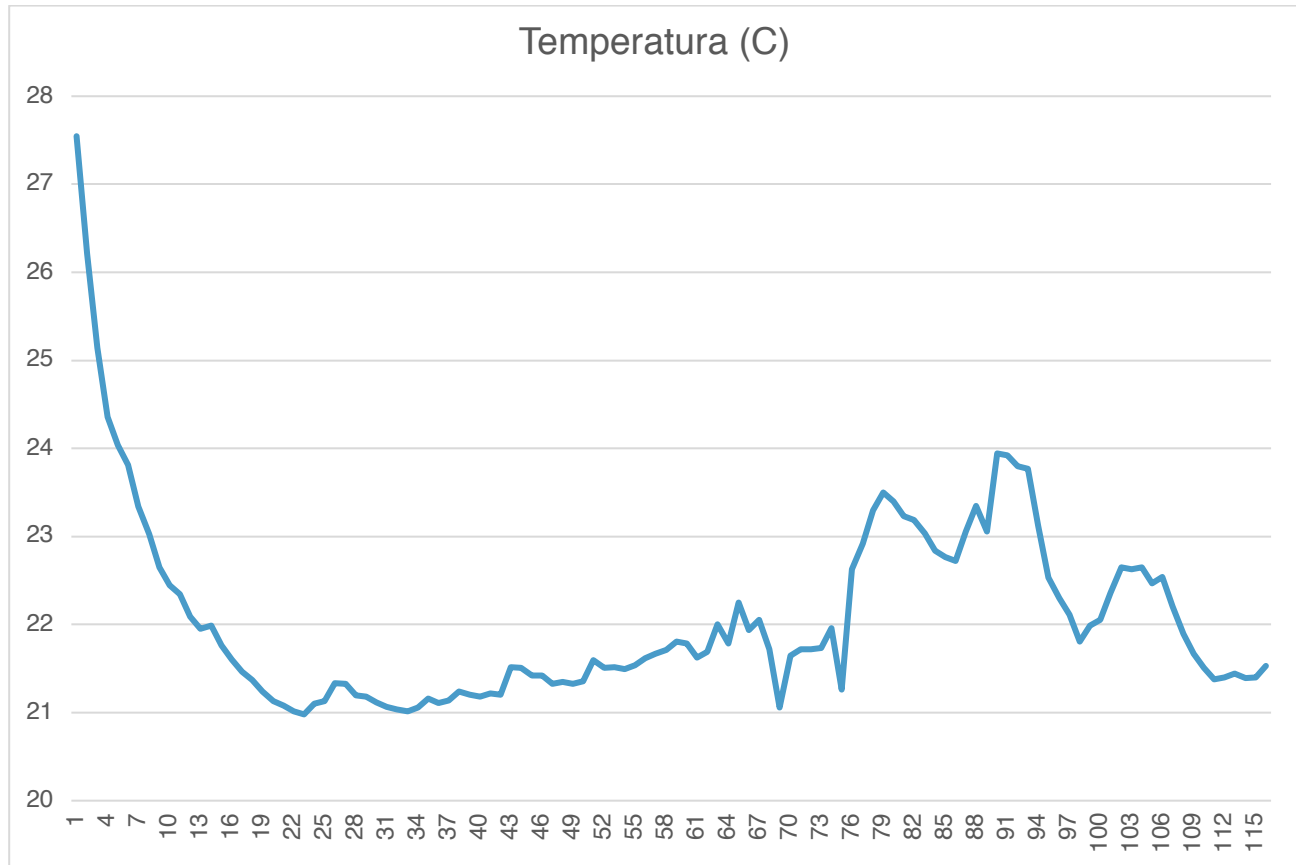


Figura 90 Grafica de la Temperatura durante la prueba de vuelo

Como se puede observar en las gráficas anteriores, es muy ilustrativo poder observar las variables medidas durante todo el transcurso del vuelo para poder observar su comportamiento, es por ello que se decidió ampliar la plataforma web de monitorización con la capacidad de graficar directamente en el detalle de cada variable el valor leído vs tiempo.

A continuación, se pueden observar las gráficas de presión atmosférica y temperatura generadas directamente dentro de la plataforma:



Figura 91 Detalle de variable de vuelo (Presión atmosférica), se puede observar el mapa de muestras y grafica vs tiempo



Figura 92 Detalle de variable de vuelo (Temperatura), se puede observar el mapa de muestras y grafica vs tiempo

Capítulo 6. Conclusiones

Podemos confirmar que se logró cumplir el objetivo principal que era el diseño de un sistema de monitorización ambiental basado en el internet de las cosas lo suficientemente compacto para poder ser utilizado sobre un vehículo aéreo no tripulado (UAV), el hardware de monitorización tiene la capacidad para recolectar datos ambientes y transmitirlos en tiempo real hacia internet (conexión celular GPRS), ser monitoreados, organizados y almacenados a través de la plataforma de monitorización.

El sistema en conjunto, hardware y plataforma web de monitorización tienen la ventaja de ser un sistema de fácil configuración y operación, por lo cual puede ser aprovechado por cualquier usuario sin la necesidad de contar con conocimientos de programación.

Con la prueba de vuelo del UAV, se logró confirmar el correcto funcionamiento del sistema de monitorización para ser operado a bordo de un UAV en pleno vuelo, además de que fue muy interesante observar el comportamiento de las variables medidas con respecto a la altura, la cual es una de las ventajas de este sistema de monitorización con respecto a un sistema de monitorización estático.

Si bien el sistema no fue capaz de recolectar datos de la calidad del aire por la limitación del sensor utilizado para medir la concentración de CO en la atmósfera, si se logró comprobar su funcionamiento y utilidad al medir otras variables como fueron Temperatura, Presión atmosférica e Índice de Radiación UV.

En la siguiente grafica se puede observar los resultados de la temperatura (°C), en color azul la temperatura medida en la estación Ninfas, en color verde la temperatura medida por el sistema de monitorización y en color naranja la temperatura medida por el sistema de monitorización aplicando el coeficiente de ajuste calculado para la temperatura (1.07644005157512), aplicando este coeficiente en todas las mediciones se puede concluir que la medición de temperatura por el sistema de monitorización tiene una precisión de ± 1.60 °C, lo cual se puede considerar aceptable.

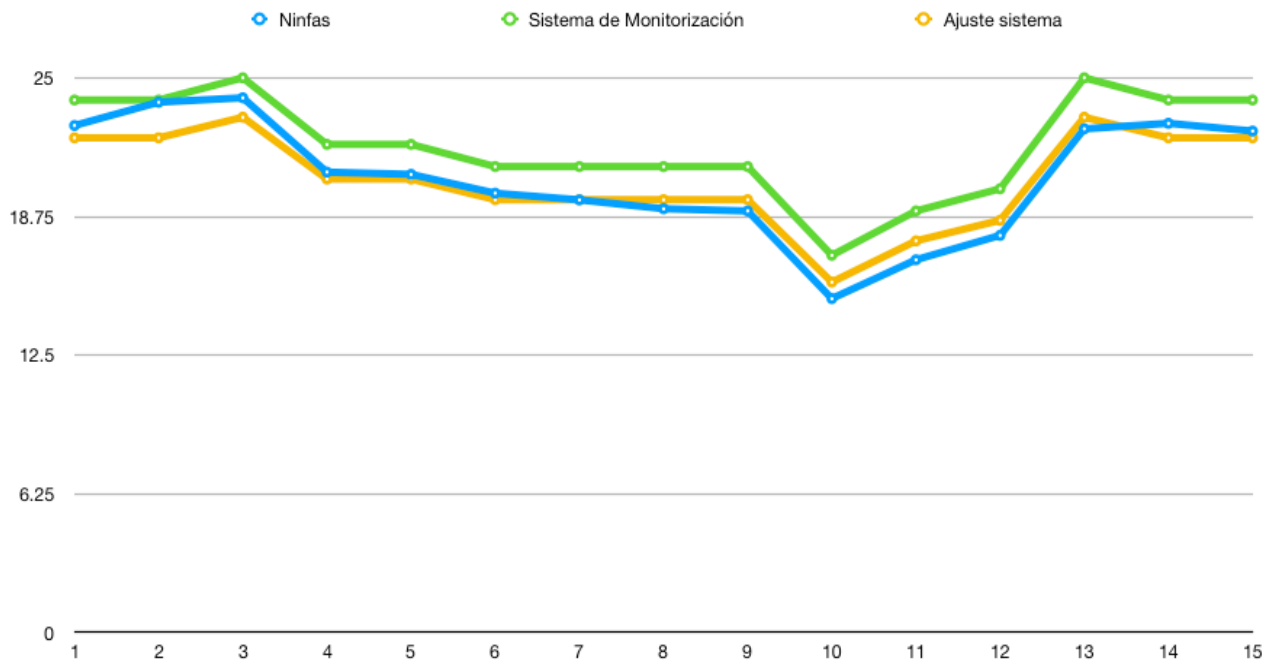


Figura 93 Grafica unificada Temperatura °C

En la siguiente grafica se puede observar los resultados de presión atmosférica (mb), en color azul la presión atmosférica medida en la estación Ninfas, en color verde la presión atmosférica medida por el sistema de monitorización y en color naranja la presión atmosférica medida por el sistema de monitorización aplicando el coeficiente de ajuste calculado para la presión atmosférica (1.00445896601359), aplicando este coeficiente en todas las mediciones se puede concluir que la medición de presión atmosférica tiene una precisión de ± 1.42 mb

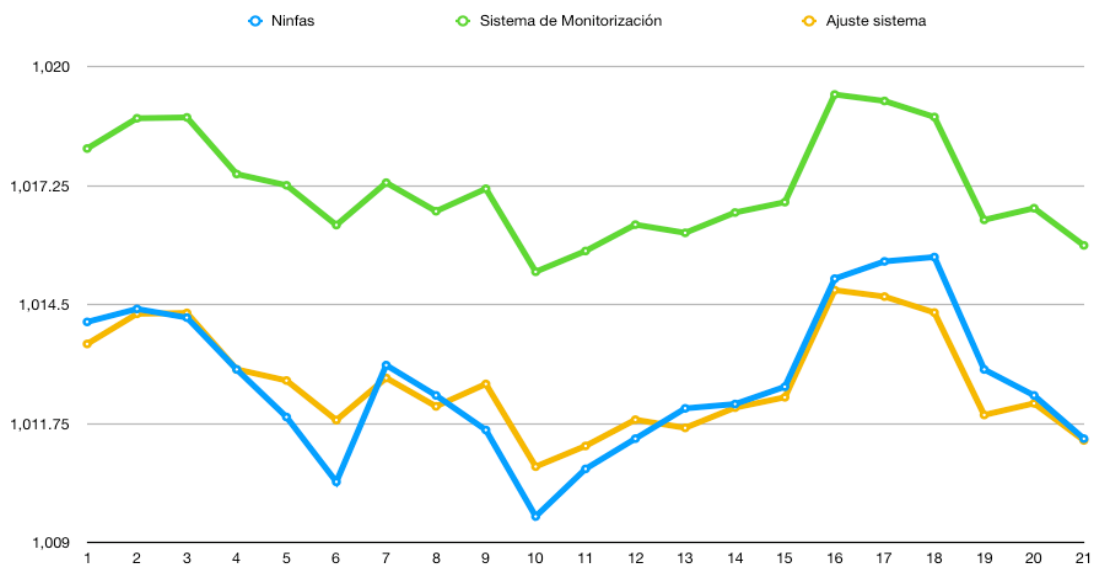


Figura 94 Grafica unificada Presión atmosférica mb

Capítulo 6. Conclusiones

En la siguiente grafica se puede observar los resultados de índice de radiación UV, en color azul el índice de radiación UV medido en la estación Ninfas, en color verde índice de radiación UV medido por el sistema de monitorización y en color naranja el índice de radiación UV medido por el sistema de monitorización aplicando el coeficiente de ajuste calculado para el índice de radiación UV (1.92980576458759), aplicando este coeficiente en todas las mediciones se puede concluir que la medición de presión atmosférica tiene una precisión de ± 1.71 unidades de índice de radiación UV.

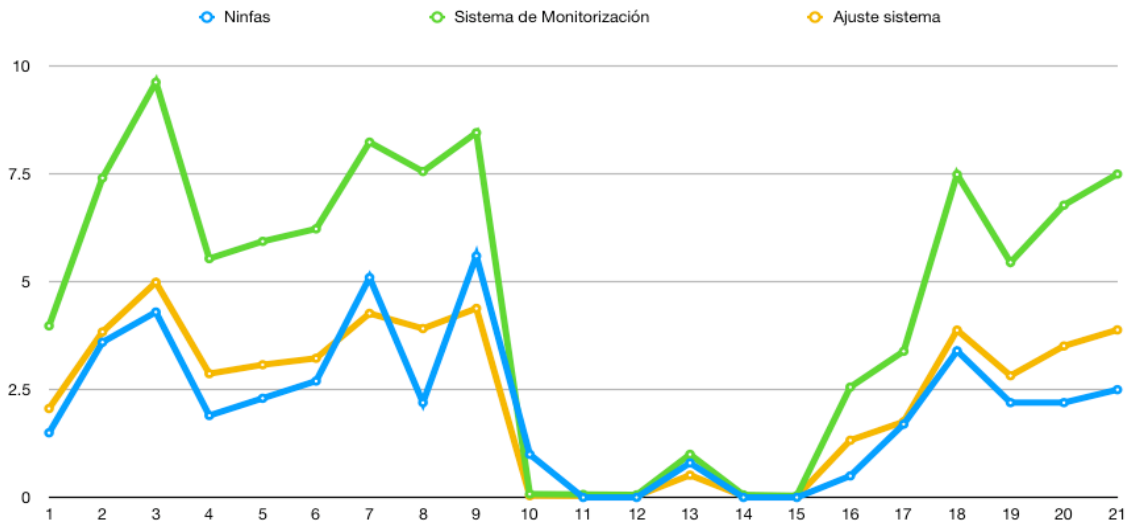


Figura 95 Grafica unificada Índice de radiación UV

Por lo cual en el ámbito de funcionamiento y utilidad del sistema se puede concluir que se logró desarrollar un sistema capaz de medir diferentes tipos de variables de forma eficiente siempre y cuando se cuente con el sensor correcto y adecuadamente calibrado.

Debido a su arquitectura modular y abierta tanto en el hardware y software, se abre la posibilidad de que esté sistema pueda ser utilizado para una gran variedad de aplicaciones donde los principales requerimientos sean contar con uno o múltiples sistemas de monitoreo remoto, compactos y accesibles con la capacidad de transmitir los datos en tiempo real y ser recibidos desde cualquier parte del mundo con acceso a internet, por lo tanto además del uso en el monitoreo de datos ambientales se puede usar desde la monitorización de invernaderos u otras aplicaciones agrícolas, hasta su aplicación de Smart Cities.

6.1 Trabajo a futuro

Como principal objetivo a futuro es lograr integrar al sistema un sensor que sea capaz de medir de forma fiable algún contaminante que nos permita tener datos sobre la calidad del aire, como puede ser el CO o bien partículas PM2.5 o PM10, el principal impedimento que se tiene para lograr este objetivo a futuro es que esta clase de sensores utilizan métodos físico-químicos para poder determinar de forma fiable la concentración de un contaminante en específico ya sea gaseoso o partícula, esto significa que esta clase de sensores son de tamaños grandes a comparación de las dimensiones actuales de nuestro sistema y usan requerimientos de energía mucho mayores, por lo cual el principal reto es encontrar un sensor que permita lograr este objetivo sin sacrificar demasiado la movilidad de nuestro sistema que es una de sus principales ventajas.

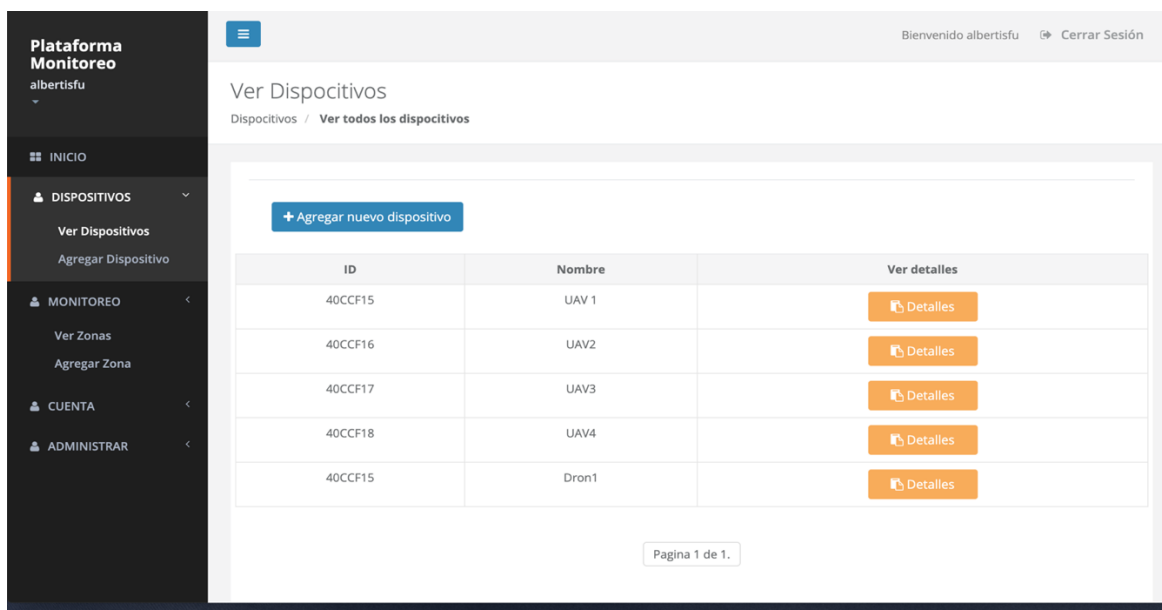
También como parte del trabajo a futuro con este sistema es usarlo en diferentes aplicaciones de la vida real, aunque se encuentren fuera del ámbito ambiental pero donde se requiera de un sistema como el nuestro para obtener datos de forma eficiente y darle sentido a los datos obtenidos con el objetivo de aportar valor agregado a través de soluciones innovadoras basadas en datos en los diferentes ámbitos que se pueda aplicar este sistema, ya que la aplicación de este sistema puede ser una excelente forma de obtener datos de forma masiva y organizados para ser procesados posteriormente aplicando tecnologías innovadoras como Big Data, Machine Learning e Inteligencia Artificial campos que tendrán un gran auge en los próximos años y en donde su éxito dependerá de contar con datos de calidad y suficientes para poder alimentar estas tecnologías que logren brindar soluciones que mejoren la calidad de vida en las ciudades en que vivimos.

Apéndices

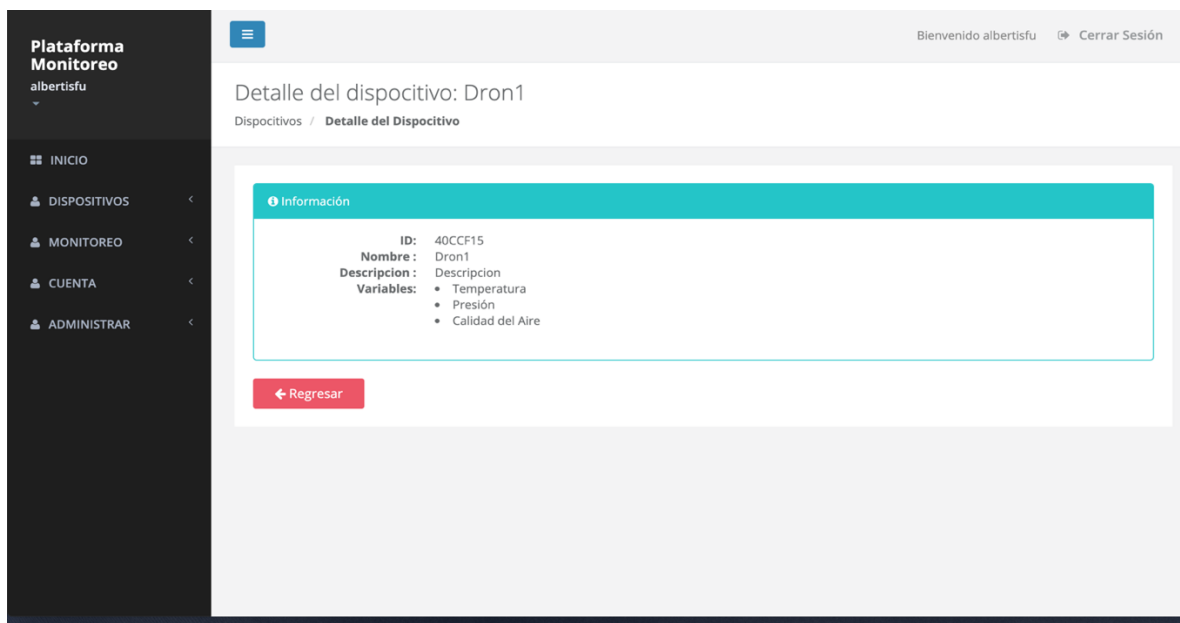
Apéndice A: Funcionamiento de la plataforma de monitorización Web.

La plataforma web de monitorización como se describió en el capítulo 3.3, cuenta con diferentes módulos que a continuación se describen:

El primer módulo es la administración de dispositivos, aquí se pueden consultar los dispositivos de monitoreo registrados o bien agregar uno nuevo.



Al dar clic en “Detalle” de un dispositivo se puede consultar los datos del mismo, como ID, Nombre, Descripción y Variables que mide.



Apéndices

Para agregar un nuevo dispositivo, se debe hacer clic en el botón “Agregar nuevo dispositivo”, se muestra la siguiente pantalla.

Aquí se debe agregar el ID de dispositivo, que es el identificador único de cada dispositivo de monitorización (hardware) para poder vincularlo con la plataforma.

Plataforma Monitoreo
albertisfu

Bienvenido albertisfu Cerrar Sesión

Agregar Dispositivos

Dispositivos / Agregar nuevo dispositivo

Datos de nueva Dispositivo

ID: 4

Nombre del dispositivo: Dron 5

Descripción: UAV 5

Variables:

- Temperatura
- Presión
- Calidad del Aire
- Radiación UV

Acciones:

Cancelar

Guardar nuevo dispositivo

Otro modulo es la gestión de zonas de monitorización, en la siguiente pantalla se puede observar el listado de zonas disponibles o bien se puede agregar una nueva zona de interés para monitorear.

Plataforma Monitoreo
albertisfu

Bienvenido albertisfu Cerrar Sesión

Ver zonas

Zonas / Ver todas las zonas

+ Agregar nueva zona

Nombre	Ver detalles
Ciudad Universitaria	Detalles
Puebla Centro	Detalles
INTEL	Detalles
Estación Ninfas	Detalles

Pagina 1 de 1.

Para agregar una nueva zona de monitorización, solo se debe agregar su nombre y una descripción corta.

Plataforma Monitoreo
albertisfu

Bienvenido albertisfu Cerrar Sesión

Agregar zonas

Zonas / Agregar nueva zona

Datos de nueva zona

Nombre de la zona:

Descripción de la zona:

Acciones:

[Cancelar](#)

[Guardar nueva zona](#)

Al consultar el detalle de una zona de monitorización se puede ver un mapa con la ubicación de la zona y una lista de los vuelos por día vinculados a esta zona.

Plataforma Monitoreo
albertisfu

Bienvenido albertisfu Cerrar Sesión

Detalle de Zona: Estación Ninfas

Zonas / Detalle de la Zona

Información

Nombre de la zona: Estación Ninfas

Descripción de la Zona: Estación Ninfas

[+ Agregar nuevo vuelo](#)

Vuelos por día:

2018/10/02 2018/10/01 2018/09/28 2018/09/27

2018/09/26 2018/09/25 2018/09/24

Página 1 de 1.

Últimos Vuelos:

Apéndices

Al hacer clic en un día se puede observar la siguiente pantalla donde se muestran las variables asociadas al dispositivo de vuelo correspondiente para poder consultar su detalle.

Detalle de Zona: Estación Ninfas - 2018/09/25
Zonas / Vuelos del

Información

Nombre de la zona: Estación Ninfas
Descripción de la Zona: Estación Ninfas

Mapa: Google Maps showing the location of Estación Ninfas, including landmarks like Parque de las Ninfas and Hermanos Serdan.

Variables:

- Temperatura
- Presión
- Calidad del Aire
- Radiación UV

Vuelos

Descripción	Fecha	Ver detalles
25 septiembre		Detalles

Si se desea agregar un nuevo vuelo, solo se debe introducir la descripción y seleccionar el dispositivo que se usara para el vuelo.

Plataforma Monitoreo
albertisfu

Bienvenido albertisfu [Cerrar Sesión](#)

Agregar vuelo
Zonas / Agregar nuevo vuelo

Datos de nuevo vuelo

Descripción:

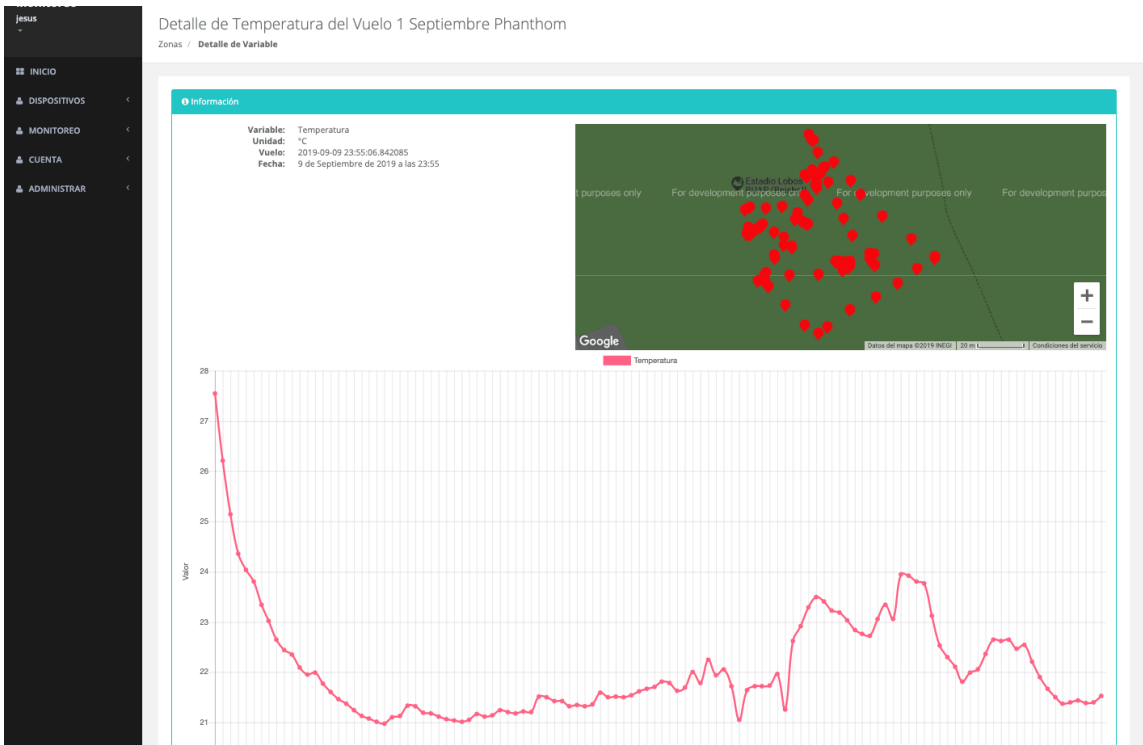
Dispositivo:

Acciones:

- [Cancelar](#)
- [Guardar nuevo vuelo](#)

Apéndices

Al hacer clic en el detalle de una variable de vuelo se puede observar una pantalla similar a la siguiente, en donde se muestra detalles de la variable, un mapa con el recorrido de vuelo y una gráfica con los datos de la variable contra el tiempo de vuelo.



También se puede consultar la tabla de los datos de vuelo, que incluye la posición y altitud de la muestra, fecha, hora y el valor de la muestra, además se puede exportar esta tabla en formato CSV, Excel y PDF.

Datos

Show 25 entries

Showing 1 to 25 of 116 entries

Search: CSV Excel PDF

ID	Latitud	Longitud	Altitud	Fecha	Dato
43793	18.99856061	-98.20244776	0.2	1 de Septiembre de 2019 a las 13:54	27.5461 °C
43794	18.9985634	-98.20246731	1.4	1 de Septiembre de 2019 a las 13:54	26.2132 °C
43795	18.99855597	-98.20245848	12	1 de Septiembre de 2019 a las 13:55	25.14564 °C
43796	18.99851375	-98.20244427	15	1 de Septiembre de 2019 a las 13:55	24.35984 °C
43797	18.9985262	-98.202457	16.6	1 de Septiembre de 2019 a las 13:55	24.03691 °C
43798	18.99854335	-98.20246677	16.4	1 de Septiembre de 2019 a las 13:55	23.81085 °C
43799	18.9986341	-98.20253976	17	1 de Septiembre de 2019 a las 13:55	23.37722 °C
43800	18.99869886	-98.20257444	16.8	1 de Septiembre de 2019 a las 13:55	23.02505 °C
43801	18.99876122	-98.20260365	16.8	1 de Septiembre de 2019 a las 13:55	22.6483 °C
43802	18.9988405	-98.20263792	17	1 de Septiembre de 2019 a las 13:55	22.44378 °C
43803	18.99890095	-98.20265963	18.3	1 de Septiembre de 2019 a las 13:55	22.3469 °C
43804	18.99890362	-98.20265234	29.5	1 de Septiembre de 2019 a las 13:55	22.08855 °C
43805	18.99887296	-98.20267165	31.6	1 de Septiembre de 2019 a las 13:55	21.94861 °C
43806	18.99882965	-98.2026903	31.2	1 de Septiembre de 2019 a las 13:55	21.99167 °C
43807	18.99884211	-98.20268962	31.4	1 de Septiembre de 2019 a las 13:55	21.76562 °C
43808	18.99883603	-98.20268616	31.4	1 de Septiembre de 2019 a las 13:56	21.60415 °C
43809	18.99882008	-98.2026864	31.4	1 de Septiembre de 2019 a las 13:56	21.46421 °C
43810	18.99877186	-98.20272414	31.5	1 de Septiembre de 2019 a las 13:56	21.36734 °C
43811	18.99869515	-98.20277529	31.5	1 de Septiembre de 2019 a las 13:56	21.23816 °C
43812	18.99862656	-98.20282177	31.4	1 de Septiembre de 2019 a las 13:56	21.13052 °C
43813	18.9985575	-98.20286236	31.4	1 de Septiembre de 2019 a las 13:56	21.0767 °C
43814	18.99847725	-98.20290297	31.5	1 de Septiembre de 2019 a las 13:56	21.01211 °C
43815	18.99845093	-98.20290074	31.4	1 de Septiembre de 2019 a las 13:56	20.97982 °C
43816	18.99843329	-98.20288687	31.5	1 de Septiembre de 2019 a las 13:56	21.09823 °C
43817	18.99835897	-98.20281475	31.5	1 de Septiembre de 2019 a las 13:56	21.13052 °C

Previous 1 2 3 4 5 Next

Apéndices

En el módulo de Cuenta se puede administrar datos personales de la cuenta de usuario, así como reiniciar la contraseña o eliminar la cuenta.

albertisfu

Cuenta de: albertisfu

Mi cuenta / Configuración

Información general de cuenta

Datos de cuenta:

Nombre completo:

Email:
albertisfu@gmail.com

[Editar datos de cuenta](#) [Cambiar contraseña](#) [Cerrar cuenta](#)

Resumen datos de perfil:

Edad:

Telefono:

Celular:

Dirección:

Descripción:

[Editar información de perfil](#)

Apéndice B: Glosario de términos.

API: del inglés Application Programing Interface, en español Interfaz de programación de aplicaciones, es un conjunto de rutinas que provee acceso a funciones de un determinado software.

API REST: es un API bajo la arquitectura REST (del inglés: representational state transfer) es un estilo de arquitectura *software* para sistemas hipermedia distribuidos como la Web moderna.

ARDUINO: Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre que incorpora un microcontrolador re-programable y una serie de pines hembra, los que permiten establecer conexiones entre el microcontrolador y los diferentes sensores y actuadores de una manera sencilla.

BACKEND: es la capa de acceso a datos de un software o cualquier dispositivo, que no es directamente accesible por los usuarios, además contiene la lógica de la aplicación que maneja dichos datos.

CSS: en español Hojas de estilo en cascada, es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado.

CSV: son un tipo de documento en formato abierto sencillo para representar datos en forma de tabla, en las que las columnas se separan por comas y las filas por saltos de línea.

DJANGO: Django es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como Modelo–vista–controlador.

FIRMWARE: el firmware o soporte lógico inalterable es un programa informático que establece la lógica de más bajo nivel que controla los circuitos electrónicos de un dispositivo de cualquier tipo.

FRAMEWORK: es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

FRONTEND: Frontend es la parte de un programa o dispositivo a la que un usuario puede acceder directamente. Son todas las tecnologías de diseño y desarrollo web que corren en el navegador y que se encargan de la interactividad con los usuarios.

GPRS: en inglés General Packet Radio Service, fue creado en la década de los 80, una conexión GPRS está establecida por la referencia a su nombre de punto de acceso, permite mandar y recibir paquetes de datos usando la red de telefonía por satélite.

GPS: es un sistema que permite determinar en toda la Tierra la posición de cualquier objeto con una precisión de hasta centímetros, aunque lo habitual son unos pocos metros de precisión.

HTML: siglas en inglés de HyperText Markup Language, hace referencia al lenguaje de marcado para la elaboración de páginas web.

IP: la dirección IP es un número que identifica, de manera lógica y jerárquica, a una Interfaz en red de un dispositivo que utilice el protocolo o, que corresponde al nivel de red del modelo TCP/IP.

JAVASCRIPT: es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

JSON: es un formato de texto sencillo para el intercambio de datos. Se trata de un subconjunto de la notación literal de objetos de JavaScript, aunque, debido a su amplia adopción como alternativa a XML, se considera un formato independiente del lenguaje.

MQTT: Traducción del inglés-MQTT es un protocolo de mensajería basado en publicación-suscripción estándar ISO. Funciona sobre el conjunto de protocolos TCP / IP. Está diseñado para conexiones con ubicaciones remotas donde se requiere una "huella de código pequeña" o el ancho de banda de la red es limitado.

MYSQL: es un sistema de gestión de bases de datos relacional, está considerada como la base de datos de código abierto más popular del mundo.

NGINX: Nginx es un servidor web/proxy inverso ligero de alto rendimiento y un proxy para protocolos de correo electrónico.

NODE.JS: Node.js es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor basado en el lenguaje de programación ECMAScript, asíncrono, con I/O de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google.

PYTHON: es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional.

REDIS: Redis es un motor de base de datos en memoria, basado en el almacenamiento en tablas de hashes pero que opcionalmente puede ser usada como una base de datos durable o persistente.

UAV: un vehículo aéreo no tripulado, UAV o comúnmente dron, que también se denomina RPA, es una aeronave que vuela sin tripulación.

UNIX: Unix es un sistema operativo portable, multitarea y multiusuario; desarrollado en 1969 por un grupo de empleados de los laboratorios Bell de AT&T.

Bibliografía

- [1] D. H. Liu y B. G. Liptak, Air Pollution, CRC Press, 1999, pp. 3-4.
- [2] T. J. B. J. L. D. P. T. R. A. O. H. Isakov V, «Combining Regional- and Local-Scale Air Quality Models with Exposure Models for Use in Environmental Health Studies,» *Journal of the Air & Waste Management Association* , 2009.
- [3] World Meteorological Organization, «Guide to Meteorological Instruments and Methods of Observation,» World Meteorological Organization, 2008.
- [4] World Health Organization, «GLOBAL SOLAR UV INDEX, A Practical Guide,» Geneva, Switzerland, 2002.
- [5] H. Jing-Shiang y C. Chang-Chuan, «Effects of Air Pollution on Daily Clinic Visits for Lower Respiratory Tract Illness,» *American Journal of EPIDEMIOLOGY*, vol. 155, nº 1, pp. 1-3, Enero 2002.
- [6] L. Jen-Hao, C. Yu-Fan, L. Tzu-Shiang y L. Da-Wei, «Developed Urban Air Quality Monitoring System Based on Wireless Sensor Networks,» de *Fifth International Conference on Sensing Technology*, Taipei, Taiwan, 2011.
- [7] World Health Organization, «7 million premature deaths annually linked to air pollution,» 25 Marzo 2014. [En línea]. Available: <http://www.who.int/mediacentre/news/releases/2014/air-pollution/en/>. [Último acceso: 5 Enero 2016].
- [8] Clear Air Institute, «La Calidad del Aire en América Latina: Una Visión Panorámica,» Clean Air Institute, Washington D.C, 2013.
- [9] Instituto Mexicano para la Competitividad A.C., «¿Cuánto nos cuesta la contaminación del aire?,» IMCO, Ciudad de México, 2013.
- [10] DIRECCIÓN GENERAL DEL CENTRO NACIONAL DE INVESTIGACIÓN Y CAPACITACIÓN AMBIENTAL; INSTITUTO NACIONAL DE ECOLOGÍA; SECRETARÍA DE MEDIO AMBIENTE Y RECURSOS NATURALES, «PROGRAMA NACIONAL DE MONITOREO ATMOSFÉRICO,» Ciudad de México, 2008.
- [11] Dirección General del Centro Nacional de Investigación y Capacitación Ambiental, «Sistema Nacional de Información de la Calidad del Aire,» [En línea]. Available: <http://sinaica.ine.gob.mx/>. [Último acceso: 7 Enero 2016].
- [12] Z. & A. H. & M. B. M. Hassan, «Internet of Things (IoT): Definitions, Challenges, and Recent Research Directions,» *International Journal of Computer Applications*, 2015.
- [13] D. T. B. C. D. J. F. W. L. C. M. N. R. Lydia Parziale, TCP/IP Tutorial and Technical Overview, IBM, 2006.
- [14] «Hypertext Transfer Protocol -- HTTP/1.1,» [En línea]. Available: <https://www.w3.org/Protocols/rfc2616/rfc2616-sec1.html#sec1>. [Último acceso: 2019].
- [15] «MQTT Essentials,» [En línea]. Available: <http://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt>. [Último acceso: 2019].
- [16] D. Parsons, Dynamic Web Application Development Using XML and Java, Cengage Learning EMEA, 2008, p. 5.
- [17] «What is PHP?,» [En línea]. Available: <https://www.php.net/manual/en/intro-what-is.php>. [Último acceso: Junio 2019].
- [18] «Usage statistics of PHP for websites,» [En línea]. Available: <https://w3techs.com/technologies/details/pl-php/all/all>. [Último acceso: Junio 2019].
- [19] D. M. B. Steve Holden, Python Web Programming, Sams Publishing, 2002, pp. 4-7.
- [20] «The History of Java Technology,» [En línea]. Available: <https://www.oracle.com/technetwork/java/javase/overview/javahistory-index-198355.html>. [Último acceso: Junio 2019].

- [21] Oracle, «Java Platform, Enterprise Edition: The Java EE Tutorial,» [En línea]. Available: <https://docs.oracle.com/javaee/7/tutorial/overview003.htm>. [Último acceso: Junio 2019].
- [22] «ASP Overview,» [En línea]. Available: <https://docs.microsoft.com/en-us/previous-versions/iis/6.0-sdk/ms524929%28v%3dvs.90%29>. [Último acceso: Junio 2019].
- [23] D. Herron, Node.js Web Development, Packt Publishing Ltd, 2016, pp. 1-9.
- [24] Mozilla, «Introducción a Django,» [En línea]. Available: <https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Introducci%C3%B3n>. [Último acceso: 2019].
- [25] T. Reenskaug, «The Model-View-Controller (MVC) Its Past and Present,» *University of Oslo*, 2003.
- [26] R. T. Fielding, «Architectural Styles and the Design of Network-based Software Architectures,» *UNIVERSITY OF CALIFORNIA, IRVINE*, p. CHAPTER 5 Representational State Transfer (REST), 2000.
- [27] «What Is an Application Server vs. a Web Server?,» [En línea]. Available: <https://www.nginx.com/resources/glossary/application-server-vs-web-server/>. [Último acceso: 2019].
- [28] «A Quick-Start Tutorial on Relational Database Design,» [En línea]. Available: https://www.ntu.edu.sg/home/ehchua/programming/sql/relational_database_design.html. [Último acceso: 2019].
- [29] «HTML 5.2,» W3C Recommendation, [En línea]. Available: <https://www.w3.org/TR/html52/introduction.html#background>. [Último acceso: 2019].
- [30] «How well does your browser support html5?,» [En línea]. Available: <https://html5test.com/results/desktop.html><https://html5test.com/results/desktop.html>. [Último acceso: 2019].
- [31] «What is CSS?,» w3, [En línea]. Available: <https://www.w3.org/standards/webdesign/htmlcss#whatcss>.
- [32] Mozilla, «JavaScript Guide,» [En línea]. Available: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction>. [Último acceso: 2019].
- [33] «Adafruit FONA,» [En línea]. Available: <https://learn.adafruit.com/adafruit-fona-mini-gsm-gprs-cellular-phone-module/overview>. [Último acceso: 2019].
- [34] Microchip, «ATmega328P,» [En línea]. Available: <https://www.microchip.com/wwwproducts/en/ATmega328p>. [Último acceso: 2019].
- [35] «NEO-6 u-blox 6 GPS Modules,» [En línea]. Available: [https://www.openimpulse.com/blog/wp-content/uploads/wpsc/downloadables/GY-NEO6MV2-GPS-Module-Datasheet.pdf](https://www.openimpulse.com/blog/wp-content/uploads/wp-content/uploads/wpsc/downloadables/GY-NEO6MV2-GPS-Module-Datasheet.pdf). [Último acceso: 2019].
- [36] Adafruit, «Bosch BMP085 Breakout Board,» [En línea]. Available: <https://learn.adafruit.com/bmp085/overview>. [Último acceso: 2019].
- [37] Adafruit, «Adafruit SI1145 Breakout Board - UV index / IR / Visible Sensor,» [En línea]. Available: <https://learn.adafruit.com/adafruit-si1145-breakout-board-uv-ir-visible-sensor/overview>. [Último acceso: 2019].
- [38] HANWEI ELECTRONICS CO, «TECHNICAL DATA MQ-7,» [En línea]. Available: <https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf>. [Último acceso: 2019].