ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

CONSTRUCCIÓN DE UN PROTOTIPO INALÁMBRICO DE MONITOREO METEOROLÓGICO CONFORMADO POR PLATAFORMAS DE HARDWARE Y SOFTWARE LIBRE, ASOCIADO AL ANÁLISIS DE SUS DATOS PARA LA PREDICCIÓN DE TEMPERATURA MEDIANTE UN MÉTODO BASADO EN MACHINE LEARNING

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

ENRIQUE DAVID BÁEZ CARRILLO
enrique.baez@epn.edu.ec

GISELA ESTEFANÍA NICOLALDE NAVARRETE
gisela.nicolalde@epn.edu.ec

DIRECTOR: MSc. ANTONIO GABRIEL VILLAVICENCIO GARZÓN
apolov@gmail.com

CODIRECTOR: MSc. JORGE EDUARDO CARVAJAL RODRÍGUEZ
jorge.carvajal@epn.edu.ec

Quito, agosto 2018

AVAL

Certificamos que el presente trabajo fue desarrollado por Enrique David Báez Carrillo y Gisela Estefanía Nicolalde Navarrete, bajo nuestra supervisión.

MSc. Antonio Gabriel Villavicencio Garzón
DIRECTOR DEL TRABAJO DE TITULACIÓN

MSc. Jorge Eduardo Carvajal Rodríguez
CODIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Nosotros, Enrique David Báez Carrillo, Gisela Estefanía Nicolalde Navarrete, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

ENRIQUE DAVID BÁEZ CARRILLO	GISELA ESTEFANÍA NICOLALDE NAVARRETE

DEDICATORIA

Este logro se lo dedico a todos mis familiares, de manera muy especial a quienes no pudieron presenciar este momento.

Kike

DEDICATORIA

Este logro lo dedico a las personas más importantes de mi vida, a los seres que me inspiran ser mejor cada día y a los que amo tanto.

A Dios, por darme la fuerza para continuar día a día, por todo su amor infinito y por todas sus bendiciones derramadas en mí.

A mi padre Wilson, por ser un ejemplo de lucha, trabajo arduo, amor incondicional, paciencia y por todo su apoyo durante mi vida estudiantil.

A mi madre Carmina, por todo su amor infinito, por todas esas palabras de aliento cuando me ha visto vencida y apoyo incondicional en todos los momentos de mi vida.

A mi hermana Caty, por todo su apoyo, cariño y aliento que siempre me brinda.

A ustedes les debo todo, este logro es nuestro logro.

Gisela E. Nicolalde N.

AGRADECIMIENTO

Agradezco a Dios, por permitirme cumplir con tan anhelada meta, por la gran familia que me ha dado y por las personas que ha puesto en mi camino.

A mis padres por su apoyo, amor y esfuerzo. Gracias por ser siempre mi ejemplo a seguir y principal soporte.

A mis hermanas, quienes han sido mis mejores amigas a lo largo de mi vida y en especial en los momentos difíciles al cursar la universidad.

Gracias Ángeles por tu cariño, por formar parte de mi vida y apoyarme siempre.

Agradecer a mis amigos Samir, Huil, Pincho y Taquito por esa amistad que empezó en las aulas, y que hizo la etapa universitaria mucho más agradable. Y a todos los amigos que me han conocido como Kike.

Finalmente, agradezco al Instituto Geográfico Militar y a todos quienes estuvieron involucrados en la realización de este proyecto, en especial al MSc. Antonio Villavicencio y mi compañera Giss. Por todo el tiempo dedicado y los conocimientos compartidos.

Kike

AGRADECIMIENTO

En primer lugar, a Dios por su amor infinito, por darme la salud, la fuerza necesaria para luchar en esta vida y por guiar mi camino.

Agradezco a mis padres por toda su amor, paciencia y confianza que siempre tuvieron en mí. Por creer siempre que llegaría hasta este punto y verme convertida en una profesional.

Hermanita querida, te agradezco por todas esas palabras de aliento, por creer en mí y por quererme tanto.

A mis amigos y amigas que conocí durante en la universidad, gracias por esa amistad, risas, enojos haciendo que la instancia en la universidad sea más llevadera.

A K.F. por tener confianza en mí y por todo tu apoyo desde el momento en el que nos conocimos.

Al Msc. Antonio Villavicencio y Msc. Jorge Carvajal por su ayuda, asesoramiento y guía durante el desarrollo de este proyecto.

Finalmente, a mi compañero de tesis por toda tu paciencia, ayuda, empeño que le pusiste para que este proyecto sea el mejor de todos y por tu amistad.

Gisela E. Nicolalde N.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ECUACIONES

ÍNDICE DE CÓDIGOS

RESUMEN

En el presente proyecto se detalla la construcción e implementación de un sistema prototipo meteorológico basado en hardware y software libre, asociado al análisis de sus datos para la predicción del valor de temperatura.

En la primera sección, se describen conceptos referentes a las variables meteorológicas de temperatura, humedad relativa y presión atmosférica, además de su influencia en las capas atmosféricas.

Posteriormente se identifican las características técnicas del hardware disponible en el mercado para la construcción del sistema prototipo tales como: sensores digitales para temperatura, humedad relativa y presión atmosférica, pantallas solares, plataformas que permitan la adquisición y procesamiento de los datos, geolocalización, transmisión inalámbrica, entre otros. Adicionalmente se presenta una breve introducción al análisis predictivo usando técnicas de *Machine Learning.*

En la segunda sección, se muestran el software y hardware seleccionado, diseño del prototipo y todas las configuraciones realizadas para el funcionamiento automático del sistema prototipo. Además, se expone un breve manual acerca de la herramienta computacional WEKA; misma que permite el tratamiento de los datos meteorológicos obtenidos para la predicción del valor de temperatura.

En la tercera sección, se realizan las pruebas de funcionamiento de todo el sistema prototipo, comparativa con una estación científica, resultados obtenidos para la predicción de la temperatura y costos referenciales del proyecto.

PALABRAS CLAVE: estación meteorológica, Machine Learning, WEKA

ABSTRACT

In the present project the construction and implementation of a meteorological prototype system based on hardware and free software, associated to the analysis of its data for the prediction of the temperature value is detailed.

In the first section, concepts related to the meteorological variables of temperature, relative humidity and atmospheric pressure are described, as well as their influence on the atmospheric layers.

Subsequently, the technical characteristics of the hardware available in the market for the construction of the prototype system are identified, such as: digital sensors for temperature, relative humidity and atmospheric pressure, solar screens, platforms that allow the acquisition and processing of data, geolocation, wireless transmission, among others. Additionally, a brief introduction to predictive analysis using Machine Learning techniques is presented.

In the second section, the selected software and hardware, prototype design and all configurations made for the automatic operation of the prototype system are shown. In addition, a brief manual about the WEKA computational tool is presented; same that allows the treatment of the meteorological data obtained for the prediction of the temperature value.

In the third section, the performance tests of the entire prototype system, comparative with a scientific station, results obtained for the prediction of temperature and reference costs of the project are carried out.

KEYWORDS: weather station, Machine Learning, WEKA

1. INTRODUCCIÓN

El Instituto Geográfico Militar (IGM) en conjunto con varias instituciones públicas y privadas han creado la Red de Monitoreo Continuo del Ecuador GNSS (REGME). Conformado por estaciones receptoras de datos satelitales GNSS, que en complemento con las estaciones meteorológicas proporcionan el marco de referencia geodésico del Ecuador. La Red al momento está conformada por 45 estaciones REGME instaladas, 31 son propias del IGM y de las cuales tan sólo 10 estaciones están acopladas a estaciones meteorológicas.

Actualmente el IGM trabaja con equipos meteorológicos PAROSCIENTIFIC MET 4A Versión 1.0 sin embargo, no todas estas estaciones han sido utilizadas para la obtención de datos meteorológicos debido al alto costo de adquisición y de funcionamiento que representan.

En este proyecto se plantea el diseño e implementación de un sistema prototipo de estación meteorológica inalámbrica que emule el funcionamiento de una estación meteorológica de la REGME, con el fin de tomar medidas de tres variables atmosféricas: temperatura, humedad relativa y presión atmosférica. Para la obtención de las medidas de las variables atmosféricas se utilizarán sensores disponibles en el mercado y plataformas de software libre como Arduino y Raspberry Pi.

El prototipo será capaz de adquirir datos con la misma resolución configurada en las estaciones REGME y presentarlos en un archivo de texto diario de 24 horas con el mismo formato que actualmente reportan las estaciones meteorológicas de la REGME.

Para la administración de los datos el prototipo contará con un servidor remoto, el mismo que estará conformado por una base de datos para el almacenamiento de los valores meteorológicos y una página web para visualización de los mismos. La conexión entre el prototipo y el servidor remoto se lo realizará mediante la red celular. Además, en el servidor se realizará el acondicionamiento de los archivos de texto producidos por el prototipo.

El IGM ha implementado un método de corrección de retraso troposférico en base a los datos obtenidos de las estaciones meteorológicas, valor que es configurado en los receptores GNSS. En el esquema actual de trabajo se emplean datos de días anteriores, pudiendo llegar a generarse fallas en el enlace debido al comportamiento variable que presenta el valor de temperatura. Por lo cual, los valores meteorológicos obtenidos por el prototipo serán empleados para generar un método que permita la predicción del valor de temperatura, basado en *Machine Learning*.

1.1 Objetivos

El objetivo general de este estudio técnico es desarrollar un prototipo de estación meteorológica inalámbrica, que maneje tres parámetros (temperatura, humedad relativa y presión del aire) asociados a la creación de un modelo de predicción para el valor de temperatura.

Los objetivos específicos de este estudio técnico son:

- Diseñar un prototipo con sensores y plataformas que permitan la emulación de una estación meteorológica de la REGME.
- Verificar el funcionamiento del prototipo al ser expuesto a las condiciones climáticas.
- Comparar los datos obtenidos por el prototipo con los datos provenientes de una estación meteorológica de la REGME.
- Monitorear los datos remotamente en un servidor a través de una tecnología celular.
- Desarrollar un modelo de predicción para el valor de temperatura en base a los datos obtenidos previamente, empleando *Machine Learning*.

1.2 Alcance

Se presentará un prototipo de estación meteorológica construido en base a plataformas en software libre y sensores disponibles en el mercado. Dicha estación debe ser capaz de tomar datos correspondientes a temperatura, humedad relativa y presión del aire, y determinar un valor promedio de cada uno por minuto, almacenándolos en un archivo de texto. Además, la estación poseerá conexión a la red celular para reportar los datos obtenidos hacia un servidor remoto.

El servidor estará formado por una base de datos, la cual permitirá el almacenamiento de los valores obtenidos y una visualización de los mismos en una interfaz web. Consecuentemente, en el servidor será posible realizar el acondicionamiento de los archivos de texto producidos por la estación y emplearlos para generar un método que permita la predicción del valor de temperatura, basado en *Machine Learning*.

1.3 Marco Teórico

1.3.1 Estaciones Meteorológicas.

La estación meteorológica es un equipo que permite medir y registrar distintas variables meteorológicas acorde a las necesidades de la aplicación en la que sea empleada [1]. Una estación meteorológica está conformada por tres subsistemas: sensores, consola y software de almacenamiento, como se muestra en la Figura 1.1.

El subsistema de sensores son todos los dispositivos de alta precisión, en su gran mayoría digitales capaces de realizar la toma de medidas de distintas variables meteorológica como temperatura, presión, humedad relativa, velocidad del viento, etc.; para luego enviarlos a subsistema de consola.

Para controlar posibles errores que se pueden producir en las medidas de valores de los sensores de temperatura o el excesivo calentamiento en los mismos se utilizan escudos de protección solar, tales como: pantalla de Stevenson, escudos con ventilación natural y escudos con ventilación forzada [2].

El subsistema de consola recepta los valores tomados por los sensores y los envían al software de almacenamiento. Este subsistema es opcional debido a que los valores tomados por los sensores pueden ser enviados directamente al software de almacenamiento.

El subsistema de software posee la capacidad de almacenamiento de los valores meteorológicos para luego representarlos en una interfaz gráfica propia del sistema operativo ya instalado para su monitoreo o con la posibilidad de conectarse a un ordenador para descargar los datos [1].

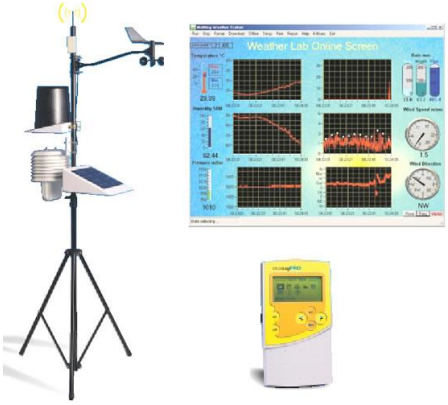


Figura 1.1. Sensores, consola y software de una estación meteorológica [3]

1.3.2 Conceptos meteorológicos

- **Temperatura del aire**

La guía de la WMO (*World Meteorological Organization*, Organización Mundial de Meteorología) define a la temperatura como una cantidad física que caracteriza la media aleatoria de movimiento de moléculas en un cuerpo físico [4]. Para propósitos meteorológicos, la temperatura es medida en varios escenarios como son temperatura del suelo, del mar, mínima de césped, y entre otras la más común que es la temperatura del aire.

La temperatura del aire es definida como “la temperatura indicada por un termómetro expuesto al aire en un lugar aislado de la radiación solar directa” [4]. Siendo esta definición útil para la mayoría de las aplicaciones.

- **Unidades**

Para la mayoría de los propósitos meteorológicos la temperatura es expresada en grados Celsius (°C); existiendo también unidades aceptadas como el grado Fahrenheit (°F) y el Kelvin (K).

- **Instrumentos de medición**

El instrumento empleado para medir la temperatura es el termómetro. Para observaciones de rutina aún es común el uso de termómetros de líquido en vidrio, donde se usa la expansión diferencial de un líquido puro con respecto a un contenedor de vidrio para indicar la temperatura [4]; el líquido usado depende del rango de temperatura implicado, generalmente contienen mercurio o alcohol etílico.

Los instrumentos eléctricos han tenido una aceptación en la meteorología gracias a su característica de generar una señal idónea para el registro, almacenamiento, o transmisión de sus datos. En cuanto a sensores de temperatura los más usados son termómetros basados en semiconductores, resistencia eléctrica y termocuplas.

Los termómetros de resistencia eléctrica basan su funcionamiento en la medida de la variación de resistencia eléctrica de un material conocido, y que guarda relación con la temperatura a la que se encuentra el material. Para la fabricación de estos termómetros se emplean metales puros; debido a que su incremento de resistencia es proporcional al cambio de la temperatura, en escenarios donde existen pequeños cambios en la temperatura [4].

Termómetros de semiconductor están conformados por un elemento resistivo llamado termistor, que es un semiconductor con un coeficiente de resistencia relativo mayor al de los materiales empleados en los termómetros de resistencia eléctrica. Por lo que para alcanzar la misma sensibilidad requieren de un menor valor de voltaje aplicado a través del material resistivo [4].

Las termocuplas son los instrumentos eléctricos más utilizados a nivel industrial para la medición de temperatura. Está conformado por dos alambres de distintos materiales unidos en un extremo, cuya temperatura refleja un voltaje en el otro extremo.

- **Requerimientos**

En la guía de observación y de instrumentos meteorológicos de la WMO se muestran los rangos mínimos aceptables de calibración y errores para termómetros que cubran un rango de medición típico:

- **Rango de medidas:** -40 - 40 °C
- **Incertidumbre:** 0,3 °C
- **Resolución:** 0,1 °C
- **Constante de tiempo de medida del sensor:** 20 s

Sin embargo, para la guía emitida por la WMO es aceptable que el rango de los termómetros sea escogido para reflejar un rango climático local. Al igual que para termómetros de menor costo, se recomienda realizar correcciones a sus medidas de ser necesarios.

Para fines meteorológicos, la temperatura del aire observada debe representar las condiciones del aire a una altura de entre 1,25 y 2 metros sobre el nivel del suelo [4]. De igual manera se debe evitar la cercanía a árboles, construcciones y cualquier elemento que obstruya la estación con el entorno.

- **Pantallas de protección de radiación**

La radiación proveniente del sol, nubes, suelo u otros objetos pasan a través del aire sin reflejar cambios apreciables en su temperatura; pero un termómetro expuesto a esta radiación puede ser afectado con cambios considerables de hasta 25 °C en condiciones extremas [4]. Razón por la cual las medidas de la temperatura del aire deben provenir de un termómetro protegido de la radiación por una pantalla o un escudo y a la vez evitando perder el contacto con el aire.

Existen varios tipos de pantallas y escudos diseñados para proteger los instrumentos de medición, básicamente por su diseño se clasifican en pantallas de ventilación natural y de ventilación artificial.

- **Pantallas de ventilación natural**

Los diseños de estas pantallas están conformados en su superficie por capas, con ranuras entre las mismas para permitir la circulación del aire. El paso del aire a través de las pantallas permite que la temperatura interior se adapte a los cambios del ambiente.

Típicamente estas pantallas han sido fabricadas de madera, como en el diseño original de la pantalla de Stevenson; modelos más recientes emplean plástico o metal para su fabricación presentando mejores resultados debido a su diseño cilíndrico [5].

En la Figura 1.2. se muestra una pantalla de Stevenson convencional y una pantalla moderna de ventilación natural.

Figura 1.2. Pantalla de Stevenson de madera y pantalla de radiación de ventilación natural cilíndrica [6] [7].

- **Pantallas de ventilación artificial**

También conocidas como de ventilación forzada, estas pantallas mantienen un aislamiento superior a las de ventilación natural, pues no necesitan que el aire cruce a través de ellas. Sino que el aire es ingresado hacia la estructura mediante un ventilador instalado dentro de la pantalla.

En la Figura 1.3. se muestra una pantalla de ventilación artificial comercial, se puede apreciar que el diseño permite el flujo del aire de manera forzada.

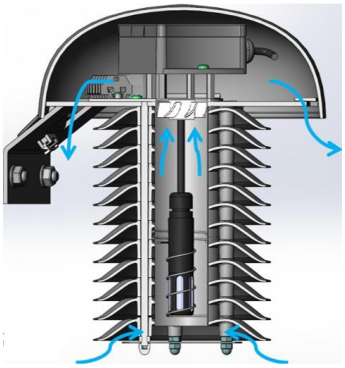


Figura 1.3. Flujo de aire a través de una pantalla de radiación de ventilación artificial [8]

- **Presión Atmosférica**

La guía de la WMO define a la presión atmosférica como la fuerza ejercida por unidad de superficie a consecuencia del peso de la atmósfera que se encuentra sobre el objeto de medición [4]. Es decir, es igual al peso de la columna vertical de aire sobre la proyección de la unidad de superficie.

➢ **Unidades y escalas**

La unidad de medida para la presión atmosférica es el Pascal (Pa) o Newton por metro cuadrado (N/m^2), pero para fines meteorológicos se utiliza el prefijo “hecto”, haciendo referencia a que un hectopascal (hPa) equivale a 100 Pa. En algunos casos se suele presentar a la presión atmosférica en unidades de milibar (mbar) el mismo que es equivalente a un hectopascal.

➢ **Requerimientos**

Las mediciones de presión atmosférica deben ser muy precisas, con baja tolerancia a error y con procedimientos de medición de acuerdo con lo expresado por la WMO.

Los requerimientos mencionados por la WMO son:

- ***Rango de medidas:*** 500 – 1 080 hPa (engloba la presión de la estación y la presión al nivel del mar)
- ***Incertidumbre:*** 0,1 hPa
- ***Resolución:*** 0,1 hPa
- ***Constante de tiempo de medida del sensor:*** 20 s
- ***Tiempo promedio de salida:*** 1 minuto

Adicional a los requerimientos anteriores, el sensor de presión atmosférica debe estar montado sobre un soporte sólido donde no exista cambios abruptos de temperatura, ni exposición directa al sol.

- **Instrumentos de medición**

El instrumento que se utiliza para realizar las mediciones de presión atmosférica es el barómetro, existen algunos barómetros disponibles en el mercado, aptos para procesos meteorológicos y accesibilidad económica, entre ellos están: barómetros de mercurio, barómetros electrónicos, etc.

En la actualidad los barómetros de mercurio ya no son muy utilizados debido a que el vapor de mercurio es altamente tóxico, la geometría de estos instrumentos es grande en comparación a otros sensores, son difíciles de realizar mantenimiento, además que las lecturas y correcciones deben realizarse manualmente.

Los barómetros electrónicos se caracterizan por usar un transductor el mismo que trasforma la respuesta de salida del sensor en cantidades eléctricas relacionadas a la presión en forma de señales analógicas o digitales, también pueden contar con protocolos de comunicación de datos estándar como RS232, RS422 [4]. Estos dispositivos poseen mejor estabilidad, mayor precisión e incluso poseen compensación automática de temperatura debido a que cuenta con sensores de temperatura montados internamente.

Dependiendo de las necesidades y exactitud requerida, se han creado algunos barómetros electrónicos como: transductores de desplazamiento Aneroid, barómetros resonadores cilíndricos y barómetros digitales piezoresistivos.

- **Humedad relativa**

Se define a la humedad como el vapor de agua contenido en la atmósfera. La humedad ha sido clasificada en: humedad específica, humedad absoluta y humedad relativa (HR).

La guía de WMO define a la humedad relativa como la humedad que contiene una masa de aire, en relación con la máxima humedad absoluta que podría admitir sin producirse condensación, conservando las condiciones de temperatura y presión atmosférica [4].

- **Unidades**

La unidad estandarizada para la humedad relativa es el porcentaje (%).

- **Requerimientos**

- ***Rango de medidas:*** 5 – 100 %
- ***Incertidumbre:*** (margen de error) menor 3%
- ***Exactitud:*** 5%(si HR <= 50%) y 3%(si HR > 50%)
- ***Resolución:*** 1%

Existe requerimientos adicionales a los antes mencionados, mismos que tienen muchas semejanzas con los requerimientos del sensor de temperatura. Además, el sensor no debe estar expuesto directamente a la luz del sol, lluvia y viento. La estructura de montaje para el sensor de humedad no debe ser de madera o de material sintético, dado que absorben o expulsan el vapor de agua de acuerdo con la humedad atmosférica.

1.3.3 Presentación de datos meteorológicos

La visualización de los datos obtenidos por las estaciones meteorológicas generalmente se realiza a través de un software dinámico y de fácil manejo para los usuarios. Adicionalmente, se obtienen registros de archivos en los cuales se observan los datos, siguiendo un formato propietario o estandarizado acorde al tipo de estación meteorológica. Los equipos utilizados por el IGM tanto para receptores GPS como para los datos meteorológicos generan archivos de formato RINEX para representar sus datos.

- **Archivos RINEX**

RINEX (*Receiver Independent Exchange Format,* Formato de Intercambio Independiente del Receptor) es un formato de intercambio de información GPS. Que fue presentado en 1989 en el 5to. Simposio Geodésico Internacional en Posicionamiento por Satélites [9] y que a partir de ese año ha sido recomendado como el formato estándar de archivos GPS.

Este formato de archivos ha sido diseñado para que los datos binarios propios de cada tipo de receptor se puedan transformar a formato ASCII7 al momento de ser descargados; pese a que el funcionamiento de los equipos sea diferente, el proveedor de software GPS adapta los datos al formato para ser descargados.

- **Versiones de RINEX**

A partir de su presentación en 1989 el formato se ha ido adaptando a los requerimientos de usuario, teniendo revisiones a lo largo del tiempo. A partir de julio del 2015 se encuentra disponible la versión 3.03 del formato [10] sin haber sufrido revisiones hasta la actualidad.

De entre las actualizaciones más importantes destaca la versión 2 ya que desde este punto en adelante el formato RINEX se compone de la creación de cuatro tipos de archivos los cuales son [11]:

1. Archivo de los datos de observación.
2. Archivo con el mensaje de navegación.
3. Archivo de datos meteorológicos.
4. Archivo del mensaje de navegación del sistema GLONASS.

Pese a haber mantenido varias revisiones con el tiempo las últimas versiones del formato RINEX 2 siguen siendo muy utilizadas, el Instituto Geográfico Militar actualmente trabaja con equipos GPS funcionales con la versión de RINEX 2.11.

- **Archivo de datos meteorológicos**

Cumple con simplificar la exportación y el procesamiento de datos meteorológicos tomados por una estación. Contiene datos como la presión atmosférica, humedad relativa y temperaturas.

- **Composición del archivo**

Cada archivo está compuesto por una cabecera y una sección de datos separados por el texto “END OF HEADER”, el formato limita cada línea del archivo a un máximo de 80 caracteres [9].

La cabecera del archivo contiene la información general del mismo, como la referente a la estación o al receptor, versión de RINEX, tipo de datos presentes en el archivo, registro de acceso, entro otros. La sección de datos de meteorología está relacionada a cada sesión y lugar.

De la versión 2 en delante se tiene la posibilidad de incluir cabeceras adicionales para nuevos registros que resulten útiles para el entendimiento del usuario, valiéndose de que los últimos caracteres de cada línea (desde el carácter 61 al 80) en la cabecera contienen una descripción del registro [11]. Además, que dentro de cada cabecera se pueden incluir comentarios.

En la Figura 1.4 se muestra la sección de la cabecera de un archivo de datos meteorológicos, generado al descargar un archivo de una estación de la REGME correspondiente a la versión 2.11.

```
     2.11           METEOROLOGICAL DATA                     RINEX VERSION / TYPE
teqc  2013Mar15     ALBERTO CHAVEZ      20160329 16:01:18UTCPGM / RUN BY / DATE
Linux2.4.20-8|i386|gcc|Win32-MinGW32|=                      COMMENT
ALEC (COGO code)                                            COMMENT
ALEC                                                        MARKER NAME
ALEC                                                        MARKER NUMBER
     7    PR    TD    HR    WS    WD    RI    HI           # / TYPES OF OBSERV
                                                 0.0    PR SENSOR MOD/TYPE/ACC
                                                 0.0    TD SENSOR MOD/TYPE/ACC
                                                 0.0    HR SENSOR MOD/TYPE/ACC
                                                 0.0    WS SENSOR MOD/TYPE/ACC
                                                 0.0    WD SENSOR MOD/TYPE/ACC
                                                 0.0    RI SENSOR MOD/TYPE/ACC
                                                 0.0    HI SENSOR MOD/TYPE/ACC
        0.0000        0.0000        0.0000        0.0000 PR SENSOR POS XYZ/H
                                                           END OF HEADER
```

Figura 1.4. Cabecera de un archivo de datos meteorológicos

Los registros presentes en la cabecera corresponden con la etiqueta presente al final de cada línea, cada uno de los registros que han sido empleados en el archivo generado por el Instituto Geográfico Militar se detalla a continuación [9]:

- **RINEX VERSION / TYPE:** Versión de RINEX y tipo de archivo.
- **PGM / RUN BY / DATE:** Nombre del programa creando el actual archivo, nombre de la agencia o responsable de la creación del archivo, fecha de creación del archivo.
- **COMMENT:** Comentario.
- **MARKER NAME:** Nombre de la estación
- **MARKER NUMBER:** Número de la estación.
- **# / TYPES OF OBSERVATION:** Número y tipo de observaciones guardados en el archivo.
- **SENSOR MOD/TYPE/ACC:** Descripción del sensor, modelo, tipo y precisión.
    - **PR:** Presión (mbar)
    - **TD:** Temperatura seca (°C)
    - **HR:** Humedad Relativa (%)
    - **WS:** Velocidad del viento (m/s)
    - **WD:** Azimut del viento (grados)
    - **RI:** Incremento de lluvia (1/10 mm)
    - **HI:** Incremento de granizo
- **SENSOR POS XYZ/H:** Posición aproximada del sensor de la estación.
- **END OF HEADER:** Último registro en la sección de cabecera.

La sección de los datos de un archivo meteorológico de RINEX generado por una estación de la REGME luce como la Figura 1.5., en donde los datos obtenidos corresponden a 6

valores separados por espacios que representan año, mes, día, hora, minuto y segundo leyéndolos de izquierda a derecha. Los valores “0.0” representan que ese sensor está desconectado al momento de obtener ese dato.

```
                                                            END OF HEADER
16  3 25  0  0  1  767.9   16.4   91.3    0.0    0.0    0.0    0.0
16  3 25  0  1  1  768.0   16.4   91.7    0.0    0.0    0.0    0.0
16  3 25  0  2  1  768.0   16.2   92.6    0.0    0.0    0.0    0.0
16  3 25  0  3  1  768.1   16.1   93.6    0.0    0.0    0.0    0.0
16  3 25  0  4  1  768.1   16.0   94.5    0.0    0.0    0.0    0.0
16  3 25  0  5  1  768.1   16.0   95.1    0.0    0.0    0.0    0.0
```

Figura 1.5. Sección de datos de un archivo de datos meteorológicos

La nomenclatura para el archivo con este formato tiene la estructura “SSSSdddf.yyT”, que corresponden a los siguientes parámetros:

- **SSSS:** Identificador de la estación.
- **ddd:** El día del año.
- **f:** Número de sesión.
- **yy:** Últimos dos dígitos del año.C
- **T:** Tipo de archivo (N: navegación GPS, O: observación, G: navegación GLONASS, M: meteorológica).

❖ **Formato de datos de geolocalización**

Para el formato RINEX los valores de posición de la estación pueden requerir ser transformados de coordenadas geodésicas a coordenadas cartesianas XYZ.

Para esta transformación es necesario mirar a la Tierra como un elipsoide de revolución, en donde cualquier punto sobre la superficie terrestre queda definido por latitud, longitud y altura en coordenadas geodésicas geocéntricas, tal como se muestra en la Figura 1.6.

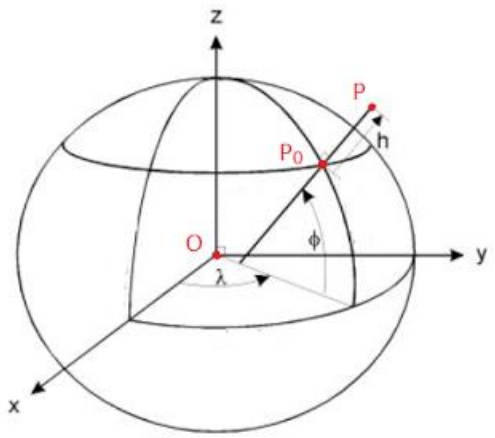


Figura 1.6. Sistema de coordenadas geodésicas y XYZ [12]

Para el sistema de coordenadas cartesianas se considera que el centro del elipsoide coincide con el geocentro terrestre. En base a esta primicia y la Figura 1.6 se obtienen las Ecuaciones 1.1 y 1.2 mismas que permiten la conversión de coordenadas geodésicas-cartesianas.

$$X = (\mathrm{N} + h)\cos\phi * \cos\lambda$$

$$Y = (N + h)\cos\phi * \sin\lambda$$

$$Z = (N * \left(\frac{b^2}{a^2}\right) + h) * \sin\phi$$

Ecuación 1.1. Transformación de coordenadas geodésicas-cartesianas [13]

$$N = \frac{a^2}{\sqrt{a^2 cos^2\phi + b^2 sin^2\phi}}$$

Ecuación 1.2. Radio de curvatura de la Tierra [13]

Donde:

a: radio ecuatorial, 6 378 137 m

b: radio polar, 6 356 752,3 m

h: altura elipsoidal

ϕ: latitud geodésica

λ: longitud geodésica

1.3.4 Estaciones GNSS

La tecnología GNSS (*Global Navigation Satellite System,* Sistema Global de Navegación por Satélite) hace referencia a un conjunto de tecnologías de sistemas de navegación por satélite, desarrollado a partir de GPS (*Global Positioning System,* Sistema de Posicionamiento Global) para proveer posicionamiento geoespacial de manera autónoma con una cobertura global.

GNSS abarca varios sistemas como son el GPS, el sistema ruso GLONASS (*Global Navigation Satellite System,* Sistema Global de Navegación por Satélite) y de manera más reciente Galileo, lo cual le ha permitido cumplir con su consigna de funcionamiento en cualquier parte del planeta [14].

- **Trayectoria de la señal GPS**

La señal GPS que transmite un satélite es enviada a través de señales de radio, dichas señales son ondas electromagnéticas que deben atravesar tres zonas: el vacío, la ionósfera y la tropósfera, antes de llegar al receptor ubicado en la superficie terrestre.

Es así como durante este proceso de viaje de la información la señal puede ser atenuada, reflejada, refractada y difractada. Estos efectos están influenciados por varias variables como son: composición de masas en el aire, temperatura, humedad, altura, hora, entre otros.

Por la manera en que se propagan las ondas de radio, la atmósfera es caracterizada en una subdivisión de ionósfera y atmósfera neutra. Donde la atmósfera neutra está conformada por tres capas que son: tropósfera, estratósfera y mesósfera; a menudo este conjunto de capas es referido simplemente como tropósfera, debido a la influencia de esta capa sobre las ondas [15].

- **Ionósfera**

La ionósfera es la capa que se encuentra por sobre los 80 km hasta los 1000 km de altitud; debe su nombre al efecto de ionización provocado por los rayos ultravioleta y solares. El efecto de ionización afecta a una porción de las moléculas gaseosas presentes en la capa, liberando electrones y creando un medio dispersivo; es decir, que la velocidad de propagación de la onda dependerá de su frecuencia [16].

- **Tropósfera**

Esta capa se encuentra en contacto con la superficie de la tierra alcanzando unos 16 km para el Ecuador y 8 km para regiones polares. Aquí se experimenta un cambio de temperatura en función de la altura (la temperatura disminuye al aumentar la altura), concentrando la mayor cantidad de oxígeno y vapor de agua, produciéndose los distintos fenómenos meteorológicos [16].

Además, la tropósfera es considerada como un medio no dispersivo, es decir, la velocidad de propagación de la onda no va a depender de la frecuencia, sino que depende de los fenómenos atmosféricos variables en esta capa. Produciéndose así retrasos en las señales electromagnéticas.

El retraso producido por la tropósfera se debe a dos componentes: retraso hidrostático y el retraso troposférico.

- **Retraso hidrostático.** Producido por gases inertes presentes en la atmósfera, dependiente de la presión atmosférica y la altura, retraso estable causante de un 90% del retardo total.

- **Retraso troposférico.** Se basa en el contenido de vapor de agua en la atmósfera, puede cambiar rápidamente debido a las condiciones atmosféricas. El retraso troposférico se expresa en unidades de distancia, mismas que pueden ser fácilmente transformadas a unidades de tiempo utilizando la velocidad de la luz.

1.3.5 Modelos Empíricos del retraso Troposférico

En las últimas décadas, varios modelos han sido reportados en la literatura científica; para representar el retraso troposférico, algunos usan información de la presión en la superficie y temperatura para derivar sus estimaciones. Aunque, la mayoría de los modelos requieren hacer ciertas suposiciones acerca de la atmósfera que cubre la estación.

Entre los modelos más empleados para el cálculo del retraso troposférico se encuentran Saastamoinen (1972), Hopfield (1969) y Hopfield modificado (Goad y Goodman 1974) [16]. En este documento se amplía información sobre el primer modelo debido a que es el modelo utilizado por el IGM por su precisión y practicidad.

- **Modelo de Saastamoinen**

El modelo de Saastamoinen (SAAS) es uno de los más populares debido a su precisión; este modelo asume que la atmósfera se encuentra en equilibrio hidrostático, por lo que la presión local provee la fuerza para balancear el peso atmosférico por unidad de área. Además, características como la presión de vapor de agua y la temperatura son consideradas como variables que van decreciendo linealmente con la altura. La Ecuación 1.3 representa el modelo en condiciones normales y latitudes medias.

$$T_k^P = 0{,}002277 \sec z \left[P_0 + \left(\frac{1255}{T_0} + 0{,}05 \right) e_0 - \Omega \tan^2 z \right]$$

Ecuación 1.3. Retraso troposférico basado en el modelo de Saastamoinen [16]

Donde:

T_k^P: retraso troposférico, del satélite P al receptor k

z: distancia cenital

T_0: temperatura en la superficie de la estación en ^{0}C

P_0: presión local en mbar

e_0: presión de vapor de agua

Ω: coeficiente

La Ecuación 1.3 se compone de dos partes, un primer término que está en función de la presión en la superficie (representa la componente seca) y un segundo término la componente húmeda, ya que una implicación del modelo es tratar a la atmósfera como la mezcla de dos gases ideales, el aire seco y el vapor de agua [16].

La distancia cenital es el ángulo comprendido entre el punto imaginario sobre la estación hasta la posición del satélite, como se puede apreciar en la Figura 1.7, siendo z la distancia cenital.

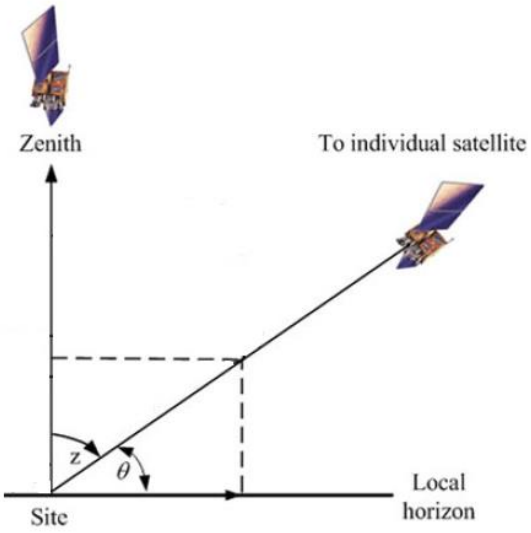


Figura 1.7. Representación de la distancia cenital [15]

La presión de vapor de agua (e_0), puede calcularse a partir de la humedad relativa y de la temperatura de la estación mediante la Ecuación 1.4.

$$e_0 = 0{,}0611RH^{\frac{7{,}5(T_0-273{,}15)}{237{,}3+T_0-273{,}15}}$$

Ecuación 1.4. Presión de vapor de agua [17]

Donde:

RH: humedad relativa de la estación

T_0: temperatura en la superficie de la estación

1.3.6 Requerimientos para construcción de una estación meteorológica

- **Sensores**

En la actualidad, se cuenta con la existencia de “sensores inteligentes”, los mismos que facilitan el tratamiento de las medidas de las variables meteorológicas a un sistema microcontrolador [18]. En base a esta premisa se detalla la siguiente información.

- **Sensor de temperatura y humedad**

Para la detección de valores de temperatura y humedad se requieren sensores de alta precisión, amplio rango de medición y fácil procesamiento de la señal. Es así como los sensores DHT11 y DHT22 fueron considerados para formar parte del sistema prototipo.

DHT11 y DHT22 son sensores que realizan mediciones de temperatura y humedad de manera simultánea, además poseen un procesador interno el cual permite obtener una salida digital y facilita la compatibilidad con otros dispositivos digitales como Arduino [19] [20].

A continuación, se muestra una tabla comparativa entre los dos sensores resaltando algunas de sus características, como se visualiza en la Tabla 1.1.

Tabla 1.1. Comparativa entre los sensores DHT11 y DHT22

Parámetros	DHT-11	DHT-22
Alimentación	3,3 – 5 V DC	3,3 – 6 V DC
Rango de medida Temperatura	0 a 50^{0}C	-40 a 80^{0}C
Precisión Temperatura	<+ -2^{0}C	<+ -0,5^{0}C
Resolución Temperatura	1^{0}C	0,1^{0}C
Rango de medida Humedad	20 a 80% RH	0 a 100% RH
Precisión Humedad	5% RH	2% RH
Resolución Humedad	1% RH	0,1% RH
Frecuencia de muestreo	1 Hz	2 Hz
Disponibilidad en el mercado	Disponible	Disponible

- **Sensor de presión atmosférica**

La obtención de valores de presión atmosférica requiere de sensores de alta precisión, amplio rango de medidas, margen de error mínimo y facilidad de tratamiento de la señal. A ello se consideró el estudio de los sensores BMP180 y BMP085.

BMP180 y BMP085 son sensores digitales diseñados para la conexión directa a un microcontrolador mediante su interfaz TWI facilitando la lectura de los datos. Además, realizan mediciones de presión atmosférica y temperatura simultáneamente [21] [22].

Se manejan bajo la tecnología piezoresistiva permitiendo que sea un sensor robusto en cuanto a la estabilidad a largo plazo y alta precisión.

A continuación, en la Tabla 1.2. se indican algunas características adicionales sobre estos sensores.

Tabla 1.2. Comparativa entre los sensores BMP085 y BMP180

Parámetros	BMP-085	BMP-180
Alimentación	1,8 – 3,6 V DC	1,8 – 3,6 V DC
Rango de medida Presión	300 a 1100 hPa	300 a 1100 hPa
Precisión absoluta	0,03 hPa	0,02 hPa
Disponibilidad en el mercado	No disponible	Disponible

- **Consola y software de almacenamiento**

➢ **Plataforma Raspberry Pi**

Raspberry Pi es una computadora de bajo costo y tamaño reducido con capacidad de conectarse a un monitor o TV, y ser controlado por teclados y ratones estándar. La funcionalidad de los dispositivos Raspberry Pi puede variar desde tareas de ofimática simple, navegación en Internet, reproducción de video, hasta tareas de programación e interacción con otros dispositivos a través de sus pines y puertos dedicados.

Los modelos que han sido lanzados bajo el nombre de Raspberry Pi se caracterizan por estar conformados por una placa con sus puertos, pines y procesador fijos; con soporte para el uso de tarjeta SD o MicroSD como disco de arranque y de memoria en general, la Figura 1.8 muestra el primer modelo lanzado por la marca.

Figura 1.8. Raspberry Pi 1 Model A

Los dispositivos Raspberry Pi cuentan con soporte para varias distribuciones de Linux que se acoplan a sus capacidades y llegando a existir distribuciones desarrolladas para cumplir

con funciones específicas como reproductores multimedia o IoT (*Internet of Things*, Internet de las cosas). De manera oficial por Raspberry Pi se distribuyen de forma abierta dos distribuciones: Noobs, que es más sencillo y limitado; y Raspbian, el sistema operativo oficial de Raspberry Pi que presenta mayores capacidades debido a su derivación del sistema operativo Debian.

Las versiones disponibles de Raspbian ya incluyen Python y Scratch con un número considerable de librerías incorporadas; existiendo información disponible basada en estos lenguajes para configuraciones o funcionamiento especial de sus dispositivos. Dicha documentación procede de la propia marca Raspberry o de desarrolladores independientes.

- **Modelos disponibles de Raspberry Pi**

Actualmente la marca Raspberry Pi distribuye varios modelos directamente en su página web y a través de otras compañías como es el caso de Adafruit, esta variedad de modelos busca suplir la mayor cantidad de requerimientos del usuario y a la vez mantener sus características distintivas de bajo costo y tamaño reducido.

A continuación, se presentan algunas características de las revisiones de hardware más representativas de cada generación de hardware de la compañía.

- **Raspberry Pi 1 Model A+**

Actualmente listada en la tienda en línea de la marca Raspberry, este modelo es una variante de Raspberry Pi de bajo costo, que pasó a reemplazar al modelo A original en noviembre del 2014 [23], tanto el modelo A, como el modelo A+ carecen de puerto Ethernet. Este último modelo está basado en un chip BCM2835 con un CPU ARM a 700 MHz, memoria RAM de 256 MB; además que fue la versión de hardware que generalizó la presencia de los pines GPIO (*General Purpose Input Output,* Entradas y Salidas de Propósito General) de 40 pines y la tarjeta Micro SD como disco de arranque [24].

Cuenta con un solo puerto USB, un conector para salida de audio de 3,5 mm, salida HDMI de tamaño estándar tipo A para audio y video, al igual que soporte para conectar cámara CSI (*Camera Serial Interface,* Interfaz Serial de Cámara) y en el lateral un puerto DSI (*Display Serial Interface,* Interfaz Serial de Pantalla) [24] que permite la conexión de una pantalla táctil directamente como se pueden apreciar en la Figura 1.9.

Figura 1.9. Raspberry Pi 1 Model A+ [23]

Esta versión de hardware resalta por mejorar el consumo eléctrico con respecto a las versiones A y B+ de primera generación, esta última ya contando con puerto Ethernet y 4 puertos USB pese a haber sido lanzada al mercado antes que el modelo A+.

- **Raspberry Pi 2 Model B**

La segunda generación de Raspberry Pi, en su modelo B reemplazó al modelo B+ de la primera generación en febrero del 2015; presentando un CPU Cortex-A7 Quad Core de 900 MHz basado en la arquitectura ARM y con 1 GB de RAM [24].

En cuanto a sus conexiones e interfaces presenta una configuración que se volvería común a partir de la versión B+ de primera generación contando con un puerto Ethernet, 4 puertos USB y manteniendo las interfaces GPIO , CSI, HDMI y de audio tal como se aprecia en la Figura 1.10.

Figura 1.10. Raspberry Pi 2 Model B [25]

Este modelo de la segunda generación aún se encuentra listado en la tienda de la marca, pese a que existe una versión superior que mejora la velocidad de su procesador.

- **Raspberry Pi 3 Model B**

Se trata de la versión estándar de Raspberry Pi de tercera generación, lanzada en febrero del 2016 [26], reemplazando a la Raspberry Pi 2 con ciertas mejoras principalmente en su procesador, usando un CPU Quad Core 1,2 GHz Broadcom BCM2837 de 64 bits [26] y manteniendo 1 GB de RAM.

- **Entradas y salidas**

Por la parte frontal se encuentra una conexión micro USB para energizar el dispositivo, y por la parte posterior una bandeja de tarjeta microSD que hace la función de disco de arranque de la Raspberry Pi. El Modelo B de tercera generación mantiene como su predecesora un puerto HDMI de tamaño estándar tipo A para la salida de video y audio, y un conector estándar de 3,5 mm capaz de entregar video compuesto y audio estéreo. Por lo que la configuración de audio y video puede ser mixta [26] [27].

Además, en la parte frontal cuenta con un puerto 10/100 Ethernet colocado junto a los 4 puertos USB 2.0 para la conexión de periféricos como mouse y teclado, memorias flash u otros dispositivos soportados. Entre el conector HDMI y el conector de 3,5 mm se encuentra un puerto CSI que facilita la conexión de una cámara al procesador Broadcom BCM2835; y en la zona lateral el puerto DSI [26] [28] [29], dichas conexiones se pueden apreciar en la Figura 1.11.

Figura 1.11. Raspberry Pi 3 Model B [26]

Cuenta con cabezal de expansión 40 pines, basado en el chip BCM2835 [30] incorporado en la placa de la Raspberry Pi; de estos pines 26 son GPIO puros [30] y de los 14 restantes se subdividen en conexiones de tierra, 3,3 VDC, 5 VDC y de acceso EEPROM, tal como se muestra en la Figura 1.12.

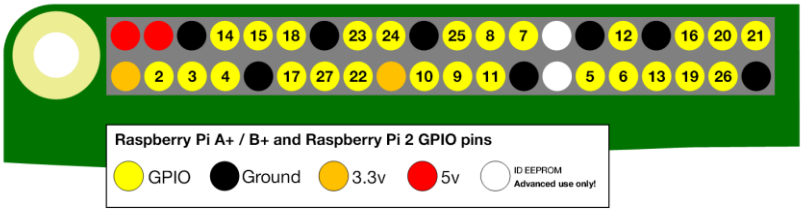


Figura 1.12. Pines GPIO Raspberry Pi 3 Model B [30]

El cabezal de 40 pines se ha vuelto un común denominador de las placas Raspberry Pi; ofertando en la actualidad esta versión en lugar de la de 26 pines, incluso en las versiones más básicas de la placa como es el caso de Raspberry Pi Zero [31].

- **Alimentación eléctrica**

La alimentación eléctrica es realizada mediante el conector micro USB, con una entrada de 5 VDC, en cuanto a la demanda de corriente esta depende de los dispositivos conectados al dispositivo.

Típicamente el modelo B consume entre 700 y 1000 mA, estos requerimientos aumentan al hacer uso de las interfaces; ya que los puertos GPIO puede usar hasta 50 mA, el puerto HDMI 50 mA, la cámara requiere de 250 mA, periféricos como mouse y teclado pueden llegar a requerir de hasta 100 mA [32].

- **Memoria**

Al igual que su versión predecesora mantiene una memoria RAM de 1 GB LPDDR2; en cuanto al disco de arranque cuenta con una ranura microSD compatible hasta con clase 10 de 64 GB [26].

- **Comunicación**

Cuenta con 4 puertos USB 2.0, que como ya se indicó brindan la posibilidad de conectar periféricos, y además pueden ser usados para establecer comunicación con otros dispositivos como placas electrónicas.

Este modelo comparte con los modelos anteriores un puerto serial UART TTL en los pines GPIO 14 (RX) y GPIO 15 (TX) [33]. Mediante este puerto se puede recibir y enviar información, al igual que acceder a consola del dispositivo.

El modelo B de tercera generación además del puerto ethernet que ya incluía el anterior modelo de la marca, añade la posibilidad de establecer conexiones inalámbricas por “Bluetooth” o “Wi-Fi”.

- **Plataforma Arduino**

Arduino es una plataforma electrónica de código abierto que ha basado el desarrollo de su hardware y software en la facilidad de uso [34], haciéndolo simple y accesible. Las placas de Arduino proporcionan la capacidad de leer entradas, sensores, señales y responder a estas de igual manera; todo esto a través de sus pines de entrada y salida.

En la Figura 1.13 se muestra un Arduino Leonardo, que es una placa con mejoras sutiles frente al modelo más básico que es el Arduino Uno.

Figura 1.13. Placa Arduino Leonardo [35]

Arduino cuenta con su propio lenguaje de programación basado en C++, por lo cual comandos estándar de este último son aceptados también en la programación; resultando útil para la expansión de librerías a través de C++.

Arduino ofrece de manera libre su software llamado Arduino IDE compatible con los sistemas operativos Mac, Windows y Linux [34], cumpliendo con su objetivo de facilidad de uso y a la vez no deja de ser una opción útil para usuarios más avanzados.

Los planos de diseño de placas Arduino han sido liberados por el propio fabricante [36], permitiendo que cualquier persona las pueda construir, facilitando la existencia de una distribución no oficial a precios más accesibles que los manufacturados propiamente por Arduino.

- **Tipos de placas Arduino disponibles**

Existen un sin número de placas Arduino disponibles en el mercado, concentrándose principalmente en suplir todas las necesidades de los usuarios, marcando diferencias en tamaños, capacidad de memoria y precios.

A continuación, se describen las principales características de tres placas Arduino más utilizadas en el mercado y con mejores prestaciones.

- **Arduino Mega**

La placa Arduino Mega basa su funcionamiento en el microcontrolador ATmega2560 preprogramado, es decir que se pueda cargar un programa sin la necesidad de un hardware externo, pero es posible obviar esta característica dado que permite programar directamente en el controlador mediante su puerto ICSP (*In Circuit Serial Programming,* Programación Serial En Circuito). Posee una capacidad de memoria flash de 256 KB para almacenamiento de código, 8 KB de SRAM y 4 KB de EEPROM [37].

Otra característica importante es su tamaño, dado que posee 54 pines con lo que necesita un espacio considerable para su implementación. En la Figura 1.14 se observa la placa Arduino Mega.

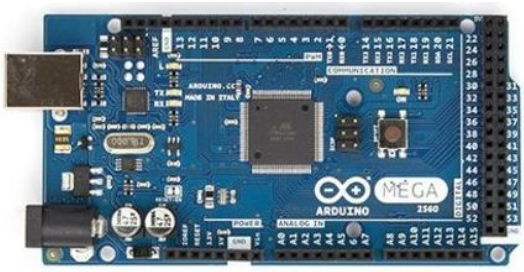


Figura 1.14. Placa Arduino Mega [37]

- **Arduino Ethernet**

La placa Arduino Ethernet posee un microcontrolador ATmega328 que trabaja a 16 MHz. Su memoria flash es de 32 KB, 2 KB de SRAM y 1 KB de EEPROM. Incorpora un módulo para transmisión de datos de hasta 100 metros de alcance, gracias a un controlador W5100 TCP/IP embebido y posibilidad de conectar tarjetas de memorias MicroSD [37] . Está conformada por 20 pines, reduciendo el tamaño necesario para su implementación, tal como se observa en la Figura 1.15. Comúnmente utilizado para aplicaciones en las que se necesita controlar un objeto a distancia sin la necesidad de utilizar cables.

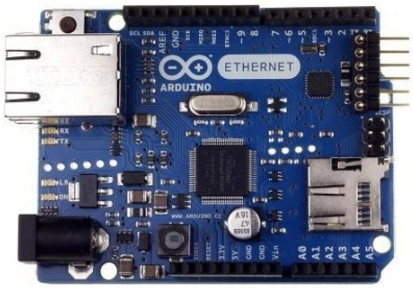


Figura 1.15. Placa Arduino Ethernet [37]

- **Arduino Uno**

La placa Arduino Uno en su versión más actual que se la identifica como Rev3 basa su funcionamiento en el microcontrolador ATmega328P preprogramado de tal manera que sea posible cargar nuevo código a la placa sin necesidad de hardware adicional; esta placa que trabaja con un cristal de 16 MHz se caracteriza por ser la más simple y robusta, lo que la convierte en la más usada y documentada de la familia de Arduino. En la Figura 1.16 se muestra la placa que es distribuida por Arduino.

Figura 1.16. Placa Arduino UNO

- **Alimentación eléctrica**

La placa Arduino Uno puede funcionar con una conexión USB tipo B de 5 VDC o con una fuente de poder externa. Para la conexión externa se cuenta con dos opciones, la primera es mediante una conexión DC de 2,1 mm de centro positivo en el puerto especializado del Arduino, la segunda opción emplea los pines Vin y GND que pueden ser conectados directamente en la placa.

La placa puede operar con un suministro externo de 6 a 20 VDC, pero se recomienda trabajar en un rango de 7 a 12 VDC [38], con el fin de evitar que la placa presente inestabilidad; esto no sucede con la conexión USB ya que esta proviene de una fuente regulada a 5 VDC [39].

- **Memoria**

La memoria para los programas a ser cargados en la placa está limitada a la existente en el microcontrolador ATmega328P, que cuenta con 32 KB [39] de los cuales 0,5 KB se encuentran ocupados por el gestor de arranque.

En cuanto a SRAM Arduino Uno dispone de 2 KB y un 1 KB de memoria EEPROM, la cual es accesible mediante una librería proporcionada directamente por Arduino [39].

- **Entradas y salidas**

Dispone de 14 pines digitales que pueden ser utilizados como entradas o salidas, que operan a 5 voltios y pueden recibir o proveer un valor máximo recomendado de corriente de 20 mA [39].

En cuanto a las entradas analógicas, son 6 identificadas desde A0 hasta A5; cada una de las entradas provee una resolución de 10 bits. Por defecto se miden estas entradas desde tierra (GND) hasta 5 voltios, sin embargo, el rango superior puede ser extendido con el pin AREF.

Algunos pines tienen funciones especializadas como se detalla a continuación:

- **Serial:** pin 0(RX) y pin 1(TX). Empleado para recibir y transmitir datos TTL de manera serial [39].
- **Interrupciones externas:** pin 2 y pin 3. Estos pines pueden ser configurados para activar una **interrupción** en un valor bajo, flanco ascendente o descendente, o cambio de valor [39].
- **PWM:** pines 3,5,6,9,10,11. Proveen una salida PWM de 8 bits [39].
- **SPI:** pines 10 (SS), 11 (MOSI), 12 (MISO), 13(SCK). Proporcionan soporte de comunicación SPI [39] (*Serial Peripheral Interface,* Interfaz Periférica Serial), que es un estándar de comunicaciones empleado entre circuitos integrados [40].
- **LED:** pin 13. Aprovecha el LED incorporado en la placa para responder al estado del pin 13, si este pin se encuentra en valor alto el Led se enciende, de no ser así se apaga [39].
- **TWI:** pin A4 (SDA) y pin A5 (SCL). Proporciona soporte de comunicación TWI (*Two Wire Interface,* Interfaz de Dos Hilos) [39].
- **AREF:** Voltaje superior de referencia para entradas digitales [39].
- **Reset:** Reinicia el funcionamiento del microcontrolador [39].

En la Figura 1.17 se detalla la ubicación de cada uno de los pines mencionados anteriormente.

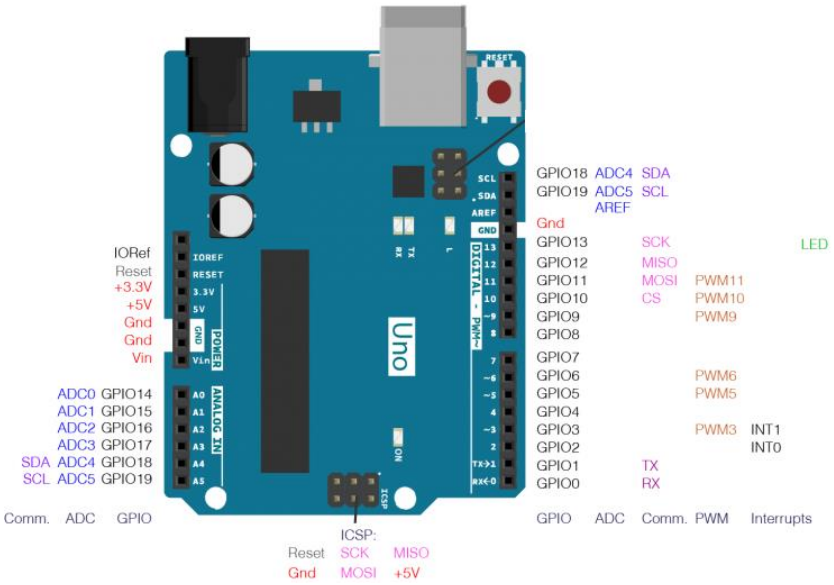


Figura 1.17. Diagrama de pines de Arduino UNO [41]

- **Comunicación**

Arduino Uno proporciona facilidades de comunicación tanto con una computadora, otra placa Arduino, u otros microcontroladores; debido a los microcontroladores ATmega328P y ATmega16U2 incorporados en la placa, con el primero hace uso de la comunicación serial UART TTL (5V) conectándose a los pines 0 (RX) y 1(TX), y con el segundo brinda soporte para comunicarse de manera serial a través de puerto USB de la placa [39]. Además, el microcontrolador ATmega328P soporta comunicaciones TWI y SPI [38].

- **Módem 3G**

Un módem 3G es un dispositivo que permite acceder a internet a través de la tecnología 3G, mediante la red del operador de telefonía móvil al que se encuentre suscrito. Además, permite conectarse a cualquier computador fijo o portátil independientemente del sistema operativo con su puerto USB [42].

Figura 1.18. Módem 3G [42]

El avance tecnológico y los innumerables requerimientos presentados por los usuarios ante la navegación en internet en dispositivos móviles han permitido que la red celular tenga un avance significativo, mejorando la experiencia de navegación de internet con altas tasas de velocidad.

En un inicio la red 3G permitía usar la tecnología UMTS (*Universa*l *Mobile Telecommunications System*) especificada en el Release 99, que alcanza velocidades teóricas de hasta 2 Mbps para *downlink* y 384 Kbps para *uplink*. Consecuentemente se desarrolló la red 3.5G usando la tecnología HSDPA (*High Speed Downlink Packet Access*) especificada en el Release 5, alcanzando velocidades teóricas para *downlink* de hasta 14,4 Mbps. Luego apareció la red 3.6G especificada en el Release 6, usando la tecnología HSPA (*High Speed Packet Accecc*) alcanzando velocidades de hasta 5,7 Mbps *uplink*. Finalmente se desarrolló la red 3.75G con la tecnología HSPA+ (*Envolved HSPA*) especificada en el Release 7 y 8, alcanzando velocidades teóricas de 84 Mbps *uplink* como 22 Mbps *downlink* [43].

Actualmente ya se ha incorporado la red 4G con la tecnología LTE (*Long Term Evolution*) especificada en el Release 8 y 9, con velocidades teóricas de 50 Mbps *uplink* y 100 Mbps *downlink* [43].

Para medir el nivel de cobertura para la zona en dónde se realizará el monitoreo, se utiliza la página web OpenSignal, en la cual se observan los niveles de cobertura 2G/3G y 4G para las operadoras Movistar, CNT EP y Claro. Sin embargo, para el presente proyecto se trabajará bajo la tecnología 3G, debido a que presenta mayores niveles de cobertura en la zona mencionada.

A continuación, se observan los mapas de cobertura para las operadoras antes mencionadas, todas operando en las tecnologías 2G/3G, tal como se muestra en la Figura 1.19, Figura 1.20 y Figura 1.21.

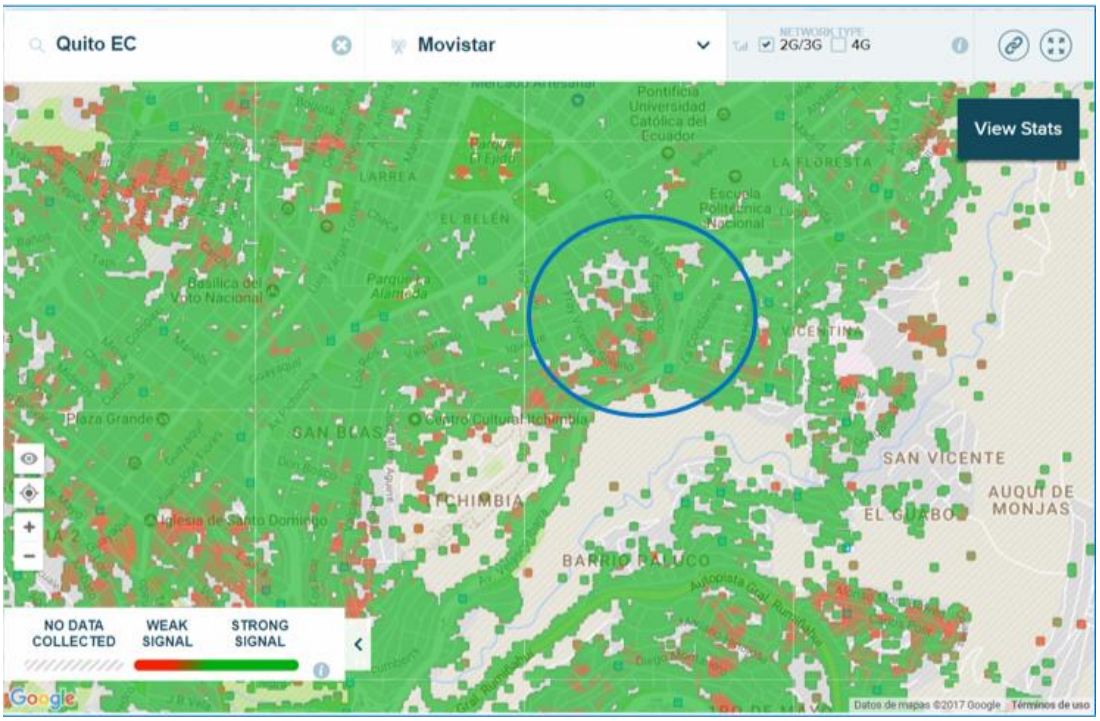


Figura 1.19. Mapa de cobertura para la operadora Movistar [44]

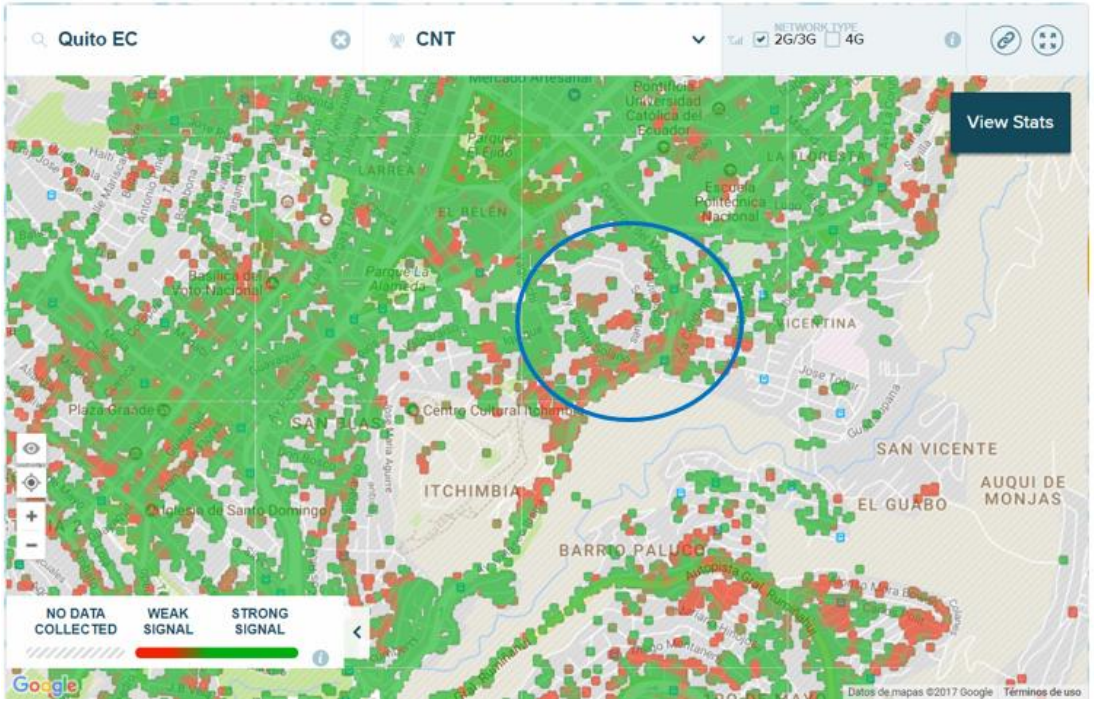


Figura 1.20. Mapa de cobertura para la operadora CNT EP [44]

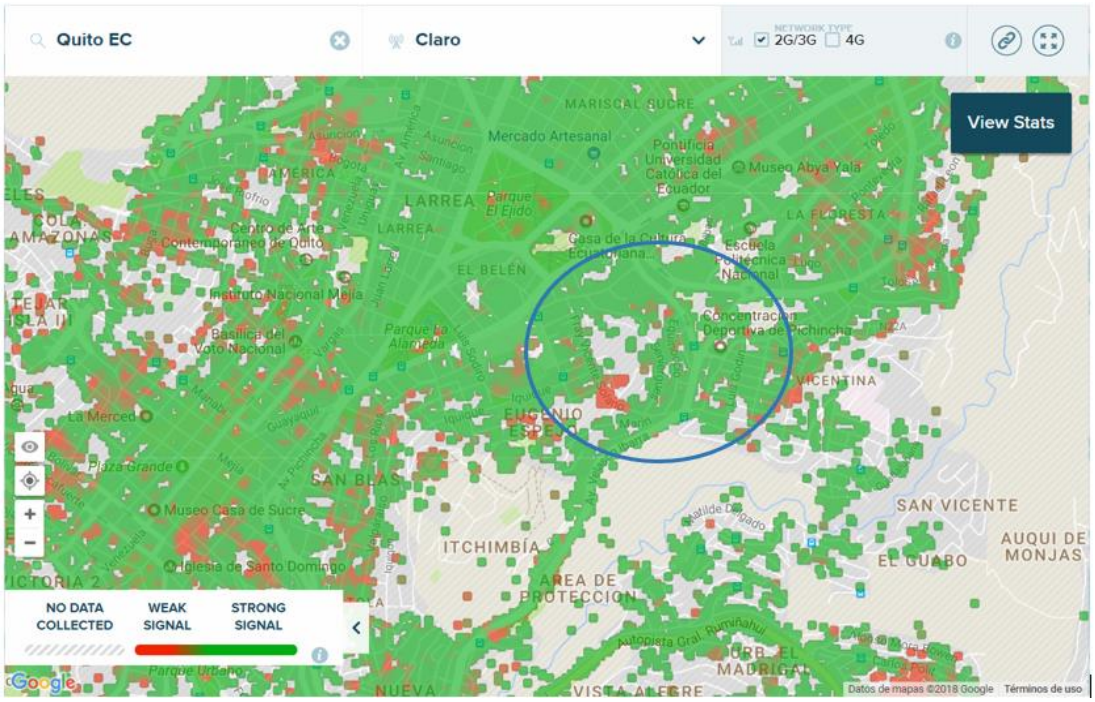


Figura 1.21. Mapa de cobertura para la operadora Claro [44]

Como se observa, el nivel de señal celular es mayor para la operadora Movistar en comparación a las dos otras operadoras, específicamente en el sector en el que se pretende instalar el sistema prototipo. Por lo cual se decide trabajar con un módem 3G que opere bajo la plataforma de la red celular de Movistar.

- **Módem 3G disponibles**

Dependiendo de la aplicación, fabricante y costos; existe un sin número de módems 3G disponibles en el mercado. Para este proyecto se ha optado por utilizar un módem 3G tipo USB stick, debido a que maneja el principio de "*Plug and Play*" para su funcionamiento, facilitando la interacción con el usuario. En base a esta premisa se ha escogido el módem Huawei E3531 HSPA+ USB Stick, cuyas características de muestran en la Tabla 1.3.

Tabla 1.3. Características del módem Huawei E3531s-6 HSPA+ USB Stick [45]

Fabricante	Huawei
Modelo	E3531s – 6
Tipo	USB Stick
Sistema de Comunicación	HSPA+/HSUPA/HSDPA/UMTS (2100/1900/850 MHz) GSM / GPRS /EDGE (850/900/1800/1900 MHz)
Velocidad HSPA+	21 Mbps (DL), 5,76 Mbps(UL)
Ranura MicroSD de expansión	Soportada
Antena Externa	No soportada
Interfaz USB	USB tipo A estándar
Dimensiones	Atura: 84,5 mm, Ancho: 27 mm, Profundidad: 10,5 mm

- **Módulo GPS**

Existe una gran variedad de módulos GPS disponibles en el mercado, diferenciándose unos de otros según el fabricante, aplicación y costos. Dependiendo de las aplicaciones a las que esté destinado el módulo GPS, se evidencia un cambio notorio en sus principales características como: niveles de sensibilidad, potencia, etc., permitiendo escoger el que más se acople a las necesidades de los usuarios.

A continuación, se describen dos módulos GPS disponibles en el mercado.

- **Módulo GPS Ublox Neo 6M**

Dispositivo de alta precisión, tamaño reducido y bajo costo, considerándose el más aplicado para sistemas portátiles.

Contiene un chip Ublox NEO 6M, una placa electrónica con un indicador visual de recepción de la señal GPS y la comunicación de placas de desarrollo de la realiza a través de los pines de transmisión y recepción serial [46]. Además, la antena que posee opera en la frecuencia central de 1575 $\pm$ 3 MHz con impedancia de 50 ohmios y polarización circular derecha. Adicionalmente, es compatible con varias plataformas electrónicas, destacando su capacidad de trabajar con una Raspberry Pi a través de los pines GPIO.

Figura 1.22. Módulo GPS Ublox NEO 6M [46]

Las principales características de este módulo GPS se observan en la Tabla 1.4.

Tabla 1.4. Características Módulo GPS u-blox Neo 6M [46]

Modelo	Módulo u-blox Neo 6M GPS
Voltaje de alimentación	3 – 5 VDC
Consumo	55 mA
Tasa de actualización	5 Hz
Frecuencia receptora	L1 (1575,42 MHz)
Satélites	22
Canales de Adquisición	66
Sensibilidad	De Adquisición: -148 dBm De Seguimiento: -161 dBm
Precisión de Posición:	3 metros

Modelo	Módulo u-blox Neo 6M GPS
Máximo de altura medible	18000 metros
Sistema de Coordenadas	WGS 84
Interfaz	UART
Baud Rate	9600 bps
Dimensiones	Módulo: 22 x 30 mm Antena: 22 x 22 x 13 mm

❖ **Módulo Adafruit Ultimate GPS breakout**

Este módulo GPS es un dispositivo de alta precisión y consumo mínimo de energía. Usa un chip MKT3339, incluye indicador visual de recepción de señal y en la placa tiene un conector U.FL para conexión de antenas externas [47]; en la Figura 1.23 se muestra el módulo GPS de la marca Adafruit.

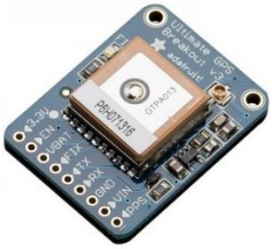


Figura 1.23. Módulo Ultimate GPS breakout [47]

Las principales características de este módulo GPS se observan en la Tabla 1.5.

Tabla 1.5. Características del Módulo Adafruit Ultimate GPS breakout [47]

Modelo	Módulo Adafruit Ultimate GPS breakout
Voltaje de alimentación	3 – 5 VDC
Consumo	20 mA
Tasa de actualización	10 Hz
Satélites	22
Canales de Adquisición	66
Sensibilidad	De Adquisición: -145 dBm De Seguimiento: -165 dBm
Precisión de Posición:	3 metros
Máximo de altura medible	18000 metros
Interfaz	UART
Baud Rate	9600 bps
Dimensiones	25,5 x 35 mm

- **UPS (*Uninterruptible Power Supply*, Sistema de alimentación Ininterrumpida)**

UPS es un dispositivo electrónico que sirve como fuente de energía eléctrica para alimentar a un equipo en caso de existan fallas en la red de alimentación. Cuenta con una batería

que suministra electricidad por cierto tiempo hasta que se pueda reparar la falla eléctrica [48].

- **Raspi UPS HAT Board**

UPS creado específicamente para Raspberry Pi, cuenta con una placa de dimensión estándar compatible con los puertos GPIO y una batería extraíble de litio recargable [49], tal como se muestra en la Figura 1.24.

Figura 1.24. Raspi UPS HAT Board [49]

En la Tabla 1.6 se observan las principales características de Raspi UPS HAT Board.

Tabla 1.6. Características de Raspi UPS HAT Board [49]

MODELO	Raspi UPS HAT Board
Alimentación	Máxima salida: 5V/2A Máxima entrada: 5V/2ª
Tipo de batería	Batería extraíble a elección del usuario voltaje 3,7V
Protección	Circuito de protección para evitar sobrecargas
Dimensión	2,8 x 2 x 0,7 pulgadas
Carga	Permite cargar y suministrar energía a la Raspberry Pi al mismo tiempo
Indicador energía	Posee 4 leds para indicar el nivel de energía

1.3.7 *Machine Learning*, Aprendizaje Automático

Machine Learning, se considera como una derivación de la inteligencia artificial, mismo que busca obtener un aprendizaje automático de los sistemas a partir de observaciones o experiencias. Este aprendizaje automático inicia con la búsqueda de patrones en los datos, para luego construir un algoritmo que pueda recibir datos de entrada y predecir valores de salida aceptables [50] [51].

Existen dos tipos de algoritmos en *Machine Learning*: supervisados y no supervisados.

Algoritmos supervisados. Aplica el conocimiento de los datos ya obtenidos, conocimiento a priori, para emplearlos en los nuevos datos mediante etiquetas o clasificaciones y así producir una función inferida para predecir eventos futuros [51].

Algoritmos no supervisados. No requiere de conocimiento a priori de los datos, dado que infiere una función para eventos futuros a partir de la exploración de los datos [51].

Conforme a lo antes mencionado y a las necesidades de este proyecto, se utilizarán algoritmos de aprendizaje automático supervisados, debido a que se trabajará con un histórico de datos meteorológicos para la predicción del valor de temperatura.

Entre los algoritmos más utilizados y capaces de predecir un valor numérico se tienen: algoritmos de regresión, algoritmos de clasificación y algoritmos resultantes de la combinación de los dos antes mencionados.

Algoritmos de regresión. Se usan para predecir una salida con un valor continuo, entre los más comunes se tienen: Regresiones Lineales, SVR (*Support Vector Regression*), etc.

Algoritmos de clasificación. Se usan para clasificar y determinar a qué clase es parte un punto de datos (valor discreto). Sin embargo, también permiten predecir valores numéricos, entre los algoritmos más importantes están: Clasificación Bayesiana, Redes Neuronales y Árboles de Decisión.

2. METODOLOGÍA

En esta sección se describen las etapas necesarias para el desarrollo de este estudio técnico.

Primero, se analizan los requerimientos del sistema prototipo; necesarios para la adquisición de los datos meteorológicos, procesamiento de los datos, almacenamiento, visualización y transferencia de datos a un servidor remoto a través de la red celular y finalmente tratamiento de los datos para la predicción del valor de temperatura.

Segundo, se detalla la selección de componentes que se utilizarán en el sistema prototipo. Estos componentes serán seleccionados bajo el criterio de utilización de hardware disponible en el mercado ecuatoriano y herramientas de software libre.

Tercero, se describe el diseño y funcionamiento del sistema prototipo, que para mejor desempeño se ha subdividido en dos subsistemas: transmisor y receptor. El subsistema transmisor será el encargado de la adquisición, lectura, escritura y transmisión de datos al subsistema receptor. Mientras que el subsistema receptor será el encargado de recibir los datos para almacenarlos, visualizarlos y procesarlos en un servidor remoto.

Finalmente, se explica la instalación de paquetes y librerías necesarias para la configuración de los dispositivos pertenecientes al subsistema transmisor. Además de la configuración de servicios que formarán parte del subsistema receptor.

2.1 Requerimientos del sistema prototipo

En la Figura 2.1 se observa los procesos generales que realizará el sistema prototipo.

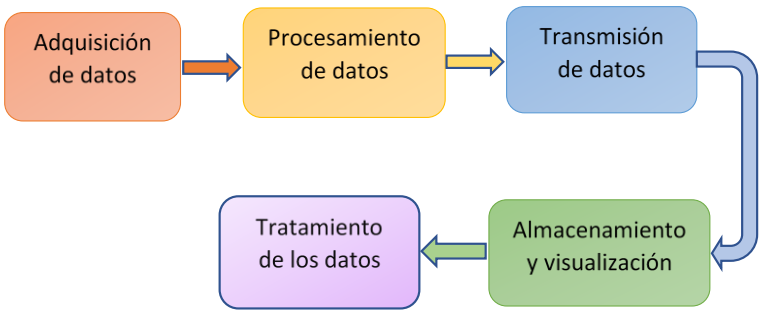


Figura 2.1. Diagrama general de procedimientos para el sistema prototipo

A continuación, se detalla los requisitos que necesita el sistema prototipo.

a) Adquirir datos correspondientes a valores de temperatura, humedad relativa y presión atmosférica.

b) Lectura y escritura de los datos obtenidos.

c) Acondicionar los datos en un formato adecuado.

d) Transmitir los datos al receptor de manera inalámbrica.

e) Receptar y procesar la información alojada en el receptor.

f) Almacenar los datos.

g) Permitir visualizar los datos procesados.

h) Realizar un tratamiento de los datos para obtener el valor de temperatura usando *"Machine Learning".*

2.2 Selección de componentes

La elección de cada uno de los componentes del sistema prototipo, deben basarse en el principio de software y hardware libre, disponibilidad en el mercado y precios accesibles.

2.2.1 Selección de Hardware

En base al análisis mostrado en el capítulo 1, se seleccionan los dispositivos que conformaran el hardware del sistema prototipo, enfocándose en el acoplamiento de estos para un adecuado desarrollo del proyecto. A continuación, se mencionarán las plataformas y dispositivos electrónicos que serán parte del sistema prototipo.

Raspberry Pi 3 B: para el proyecto se requiere un sistema que pueda procesar los datos, organizarlos y enviarlos de manera automática; es decir que además de poder conectarse con otros dispositivos debe tener la capacidad de procesamiento suficiente como para la generación y compartición de los archivos de texto donde constan los reportes de los sensores. Razón por la cual se ha considerado entre las opciones un dispositivo con capacidad de programación y de memoria semejantes a los de una computadora, pero que a su vez su tamaño no resulte contraproducente para el diseño del prototipo. Dentro de las opciones en el mercado, destacan los modelos de Raspberry Pi 3 y 2 ambos cumpliendo con los requerimientos del sistema. Pese a que ambos modelos son fáciles de encontrar, el de tercera generación se encuentra distribuido más ampliamente.

Arduino UNO: debido a las necesidades del proyecto, para la etapa adquisición de datos se necesita un dispositivo que sea compatible con los sensores de temperatura, humedad y presión; y que además pueda comunicarse con otro dispositivo que realice el tratamiento de los datos. Bajo esta premisa, de las placas de Arduino existentes en el mercado

cualquiera resultaría capaz de realizar estas tareas, por lo que se ha considerado el modelo más básico como una opción.

Sensor de temperatura y humedad DHT-22: dispositivo electrónico con amplio rango de medición, alto grado de precisión y resolución en comparación al sensor DHT11. Su salida digital facilita el acoplamiento con microprocesadores de alto nivel.

Sensor de presión atmosférica BMP-180: dispositivo electrónico de alta precisión, amplio rango de medición, mínimo margen de error. Además, destaca su interfaz bus de comunicación TWI, facilitando la conexión con sistemas microprocesados.

Módem USB 3G: referente a la tecnología para la conexión a internet del prototipo y basándose en la disponibilidad en el mercado ecuatoriano se ha escogido un modelo de módem USB *stick* (Huawei E3531 HSPA+ USB Stick), compatible con la plataforma Raspberrry Pi 3, en específico con la versión de hardware y software empleada para el prototipo.

Módulo GPS: basados en su facilidad de conexión, tamaño reducido, alta precisión, compatibilidad con la plataforma Raspberry Pi 3 y reducido costo; se ha optado por la utilización del módulo GPSu-blox Neo 6M para la implementación de este proyecto.

2.2.2 Selección de software

Uno de los objetivos del proyecto es la utilización de herramientas de software *open source,* distribución libre y gratuita, razón por la cual se decide utilizar el sistema operativo Linux. En base a esta premisa se presentan cada uno de los sistemas operativos que formarán parte del desarrollo del sistema prototipo.

Raspbian: sistema operativo basado en Debian, de descarga gratuita en la página oficial de Raspberry Pi [52]. En el proyecto se utilizará la versión Raspbian Jessie with PIXEL, mejorando la experiencia de utilización de la plataforma Raspberry Pi.

Python: lenguaje de programación gratuito, independiente de plataforma y orientado a objetos, de descarga gratuita en la página oficial de Python [53]. Es muy utilizado por su cantidad de librerías que soportan varios tipos de datos y funciones incorporadas en el propio lenguaje.

LAMP: para la instalación del servidor remoto se manejará el paquete LAMP (Linux, Apache, MySQL y PHP), el cual está formado por un conjunto de programas de código abierto que corren sobre cualquier distribución de Linux. A continuación, se explica sobre cada uno de los programas antes mencionados:

- ***Linux:*** sistema operativo de distribución libre y gratuita, utilizado para desarrollar varias aplicaciones (páginas web, juegos, etc.) y ejecutar programas. Debido a que es software libre se han creado varias distribuciones dependientes de las diferentes aplicaciones, las principales son: Debian, Ubuntu y Centos [54]. Para el desarrollo de este sistema prototipo se decide utilizar Ubuntu, dado que es menos complicado para la creación de servidores.

- ***Apache:*** es un proyecto desarrollado por The Apache Software Foundation, formado por un servidor web de código abierto y multiplataforma que implementa el protocolo HTTP (*HyperText Transfer Protocol,* Protocolo de Transferencia de Hipertexto) [55].

- ***MySQL (My Structures Query Language):*** es un sistema de gestión de base de datos desarrollado por Oracle Corporation. Se basa en un lenguaje de consulta estructurado (SQL) y permite el almacenamiento y gestión de información de manera robusta mediante una base de datos [56].

- ***PHP (Hypertext Preprocessor,*** **Preprocesador de Hipertexto*):*** lenguaje de programación de código abierto principalmente utilizado para la creación de páginas web dinámicas. Su principal característica es la ejecución de líneas de código en el lado del servidor y no en el navegador del usuario, accediendo a recursos como base de datos, archivos, videos, etc [57].

FTP (*File Transfer Protocol*, Protocolo de Transferencia de Archivos): permite el transporte seguro de archivos de un remitente a un receptor desde un servidor a un usuario a través del internet. El servidor FTP que se ha utilizado para el proyecto es VSFTPD (*Very Secure FTP Daemon*) ampliamente considerado como el servidor FTP más rápido y seguro para los sistemas UNIX.

2.3 Arquitectura del sistema prototipo

Tomando en cuenta el software y el hardware a emplear se ha diseñado el prototipo separándolo en dos subsistemas.

El primero es el subsistema transmisor, encargado de la adquisición, procesamiento y envío de los datos tomados por los sensores. El segundo subsistema es el receptor, encargado de la recepción, visualización y análisis de los datos.

2.3.1 Diseño del subsistema transmisor

El subsistema transmisor está conformado por las plataformas Arduino Uno, Raspberry Pi 3 B, los sensores y módulos que se conectan a las mismas. A su vez el subsistema ha sido separado en dos bloques, cada uno asociado a una plataforma de las escogidas anteriormente.

Los bloques que forman parte del subsistema transmisor son:

a. Bloque de adquisición de datos meteorológicos.

b. Bloque de lectura, escritura y transmisión de datos.

En la Figura 2.2 se observa la estructura del subsistema transmisor, y los componentes físicos de sus bloques.

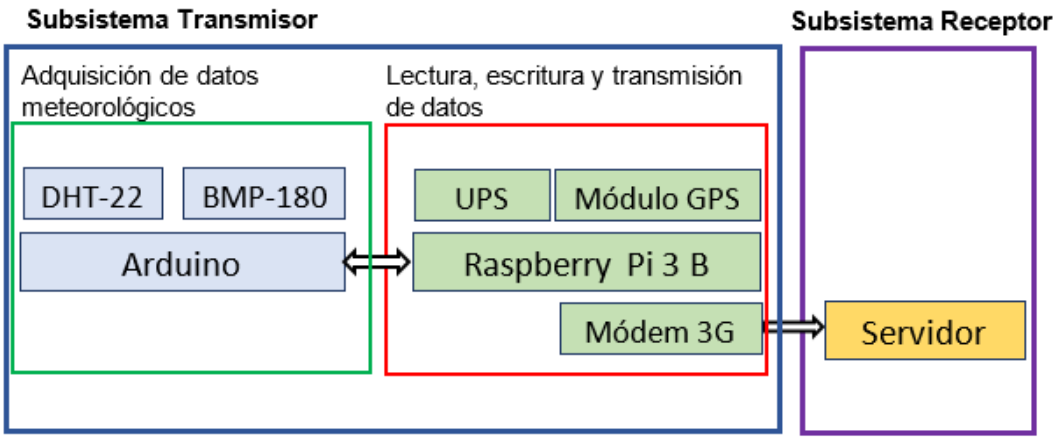


Figura 2.2. Arquitectura del sistema prototipo

- **Bloque de adquisición de datos meteorológicos**

Está conformado por el Arduino UNO, sensor de humedad y temperatura (DHT-22) y sensor de presión atmosférica (BMP-180), este bloque se encarga de tomar la información de los sensores mencionados y enviarla cada vez que sea requerida por la Raspberry Pi.

- **Bloque de lectura, escritura y transmisión de datos.**

Está basado en el funcionamiento de la plataforma Raspberry Pi 3 B, además de la plataforma Raspberry comprende al módulo GPS, módem 3G y batería. Este bloque realiza los siguientes procesos:

a. Lectura y escritura de datos en un archivo de texto.

b. Almacenamiento de datos en una base de datos.

c. Transmisión de datos en formato RINEX.

2.3.2 Diseño del subsistema receptor

El subsistema receptor está formado por un servidor remoto ubicado en la Facultad de Eléctrica y Electrónica de la Escuela Politécnica Nacional. Este servidor ha sido desarrollado en la plataforma de software libre Linux, con la distribución Ubuntu 16.04.4.

En el servidor remoto se encuentran alojados servicios que permiten la administración y visualización detallada de los datos, mismos que han sido implementados con los servicios disponibles en el paquete LAMP de Linux. Además, se realizará un tratamiento a los datos obtenidos para la predicción del valor de temperatura mediante un algoritmo basado en *"Machine Learning".*

Para un adecuado desempeño del sistema receptor, a este se lo ha dividido en bloques:

a. Bloque de almacenamiento de datos

b. Bloque de visualización y administración de datos

c. Bloque de tratamiento de datos

En la Figura 2.3 se visualizan los diferentes servidores que conforman al subsistema receptor.

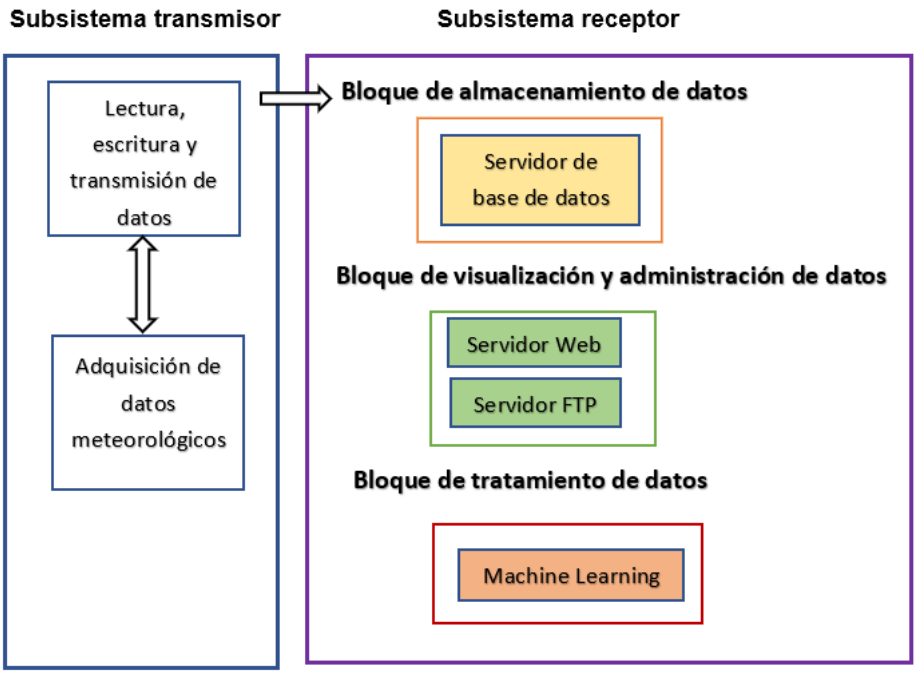


Figura 2.3. Servicios que conforman al subsistema receptor

- **Bloque de almacenamiento de datos**

Este bloque permite guardar la información de los datos obtenidos del subsistema transmisor hacia un servidor de almacenamiento de datos, a través del gestor de base de datos MySQL y su gestor gráfico PhpMyAdmin. Garantizándose la integridad y disponibilidad de los datos adquiridos.

- **Bloque de visualización y administración de datos**

Este bloque es el encargado de permitir la visualización de los datos almacenados en la base de datos a través de una página web. Se accederá a la página web mediante la ruta *"http:// 172.31.33.36"* en el explorador de preferencia. La interfaz web se ejecutará bajo la implementación del servicio Apache y el lenguaje de programación PHP.

Adicionalmente, en este bloque se contará con el servicio FTP, mismo que permitirá una óptima administración de los archivos generados por el sistema prototipo. Este servicio se ejecutará bajo el desarrollo del servidor VSFTP, disponible para Ubuntu. Cabe mencionar que, para acceder a este servicio desde una red externa a la Escuela Politécnica Nacional, se utilizará la dirección IP pública "190.96.111.183".

- **Bloque de tratamiento de datos**

En este bloque se realizará el tratamiento de los datos generados por el sistema prototipo, para desarrollar un método de predicción del valor de temperatura en base a algoritmos utilizados en *"Machine Learning".* El desarrollo del modelo predictivo se llevará a cabo en el software de distribución libre WEKA (*Waikato Environment for Knowledge Analysis*).

2.4 Instalación y configuración de los dispositivos que conforman al subsistema transmisor

2.4.1 Bloque de adquisición de datos meteorológicos

- **Arduino UNO**

- **Sensor de temperatura y humedad relativa**

El sensor utilizado para la adquisición de valores de temperatura y humedad relativa (sensor DHT-22), posee una distribución de pines conforme a la Figura 2.4.

Figura 2.4. Distribución de pines del sensor DHT-22

La conexión entre el sensor DHT-22 y la plataforma Arduino Uno se observa en la Figura 2.5.

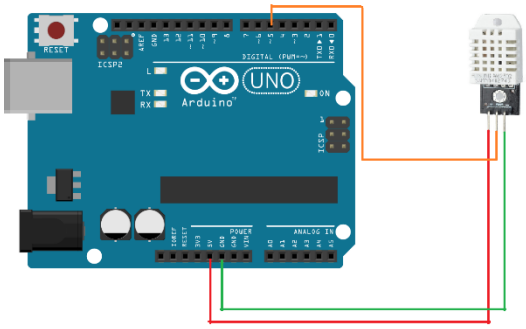


Figura 2.5. Diagrama de conexión entre el sensor DHT-22 y la plataforma Arduino UNO

Para el funcionamiento del sensor DHT-22 es necesario instalar la librería DHT.h en el IDLE de Arduino, misma que se encuentra entre las especificaciones de Adafruit.

- **Sensor de presión atmosférica**

En la Figura 2.6 se observa la distibución de los cinco pines que posee el sensor de presión atmosférica BMP-180.

Figura 2.6. Distribución de pines del sensor BMP-180

Para la comunicación, el sensor de presión atmosférica ocupa su interfaz TWI, mismo que utiliza dos líneas de transmisión: SDA (datos serie) y SCL (reloj serie) para su funcionamiento.

El diagrama de conexión entre el sensor de presión atmosférica BMP-180 y la plataforma Arduino UNO se observa en la Figura 2.7.

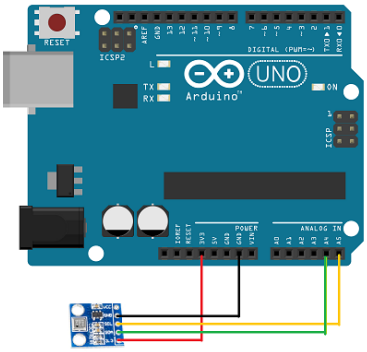


Figura 2.7. Diagrama de conexión entre sensor BMP-180 y la plataforma Arduino UNO

Para el funcionamiento del sensor BMP-180 es necesario instalar en el IDLE de Arduino la librería SFE_BMP180.h, misma que permite el manejo del sensor de presión atmosférica.

2.4.2 Bloque de lectura, escritura y transmisión de datos

- **Raspberry Pi 3 B**

➢ **Sistema operativo**

Para el funcionamiento de las tarjetas Raspberry Pi es necesario instalar el sistema operativo en una memoria MicroSD, cuya capacidad debe ser de al menos 8 GB. El sistema operativo es descargable de manera gratuita en la página oficial del fabricante, las últimas versiones disponibles para descarga al momento de la realización del proyecto son las mostradas en la Figura 2.8.

Para el desarrollo del proyecto se ha instalado el sistema operativo “Raspbian Jessie With Pixel”, debido a que cuenta con software preinstalado útil para el desarrollo de proyectos como: Python, Scratch, Sonic Pi, Java, Navegador Web, entre otros [52].

Figura 2.8 Imagen del sistema operativo "Raspbian Jessie With Pixel" [52]

La imagen del sistema operativo descargado debe ser instalado en una tarjeta MicroSD [58]; para lo cual se ha empleado el software sin costo Etcher, disponible para Windows, Linux y MacOS [59]. En la Figura 2.9 se muestra la interfaz del software empleado para cargar el sistema operativo.

Figura 2.9. Instalación de la imagen del sistema operativo usando Etcher

- **Python**

Debido a que la versión utilizada del sistema operativo ya cuenta con un paquete de Python (Python 2.7) y que esta versión cuenta con gran variedad de librerías preinstaladas, será necesario instalar solamente una librería adicional.

- **Instalación de librerías**

Para el proyecto la única librería adicional de Python que debe ser instalada es la que permite la conexión con la base de datos. Esta librería es "mysql.connector" que es descargada desde el terminal con el comando "*sudo apt-get install python-mysql.connector".*

- **Arduino UNO**

La conexión física entre ambas plataformas es mediante un cable USB como se muestra en la Figura 2.10, los conectores del cable son tipo A y tipo B, para la Raspberry Pi y el Arduino respectivamente. Esta conexión permitirá tanto la comunicación serial, como la alimentación del Arduino.

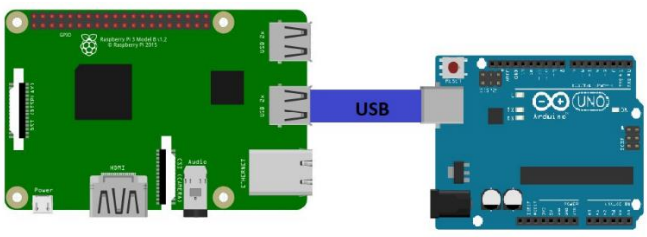


Figura 2.10. Diagrama de conexión Arduino Uno y Raspberry Pi 3

- **Módem 3G**

El módem 3G estará conectado directamente a la Raspberry Pi mediante uno de sus puertos USB, como se muestra en la Figura 2.11.

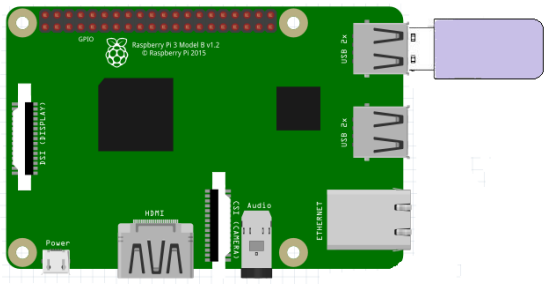


Figura 2.11. Diagrama de conexión módem 3G y Raspberry Pi 3

- **Instalación del Módem 3G**

Al conectar el módem 3G sin haber instalado ningún paquete adicional a los que vienen con la distribución de Raspbian utilizada en el proyecto, el dispositivo debería ser montado y reconocido automáticamente como una memoria flash.

El identificador que está asociado al módem 3G con su modo de funcionamiento como unidad de almacenamiento, puede ser observado revisando el archivo /var/log/messages a partir de que es conectado el dispositivo. El comando: *"tail -f /var/log/messages"* mostrará las entradas actualizadas de este archivo, en la Figura 2.12. se muestra la parte de interés donde el dispositivo es detectado como una unidad de almacenamiento USB. De aquí que el identificador sea ID 12D1:1f01, conformado por la identificación del fabricante y la del producto separadas por ":".

```
Nov 13 15:19:56 raspberrypi kernel: [  635.392513] usb 1-1.2: New USB device found, idVendor=12d1, idProduct=1f01
Nov 13 15:19:56 raspberrypi kernel: [  635.392529] usb 1-1.2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
Nov 13 15:19:56 raspberrypi kernel: [  635.392537] usb 1-1.2: Product: HUAWEI Mobile
Nov 13 15:19:56 raspberrypi kernel: [  635.392545] usb 1-1.2: Manufacturer: HUAWEI
Nov 13 15:19:56 raspberrypi kernel: [  635.392552] usb 1-1.2: SerialNumber: FFFFFFFFFFFFFFFF
Nov 13 15:19:56 raspberrypi kernel: [  635.485952] usb-storage 1-1.2:1.0: USB Mass Storage device detected
Nov 13 15:19:56 raspberrypi kernel: [  635.486319] scsi host0: usb-storage 1-1.2:1.0
```

Figura 2.12. Resultado del comando tail -f /var/log/messages correspondiente al módem 3G

Varios dispositivos USB como el que se está empleando en el prototipo para la conexión a internet no cuentan con los drivers de instalación para su funcionamiento en sistemas operativos como Raspbian. El paquete USB_ModeSwitch suple esta falencia dando soporte a gran cantidad de dispositivos, facilitando así tanto el manejo como la configuración de estos.

Para instalar esta herramienta, desde el terminal se procede a escribir el comando: *"sudo apt-get install usb-modeswitch".* Después de que hayan sido instalados los paquetes, al listar los dispositivos USB con el comando: "*lsusb*" se podrá observar que el módem ha cambiado su identificador (ID 12d1:14dc), tal como se muestra en la Figura 2.13; siendo este ID el que representa su modo de funcionamiento como módem y no como memoria flash.

```
root@raspberrypi:/home/pi# lsusb
Bus 001 Device 005: ID 1a2c:0021 China Resource Semico Co., Ltd Keyboard
Bus 001 Device 004: ID 0458:00e3 KYE Systems Corp. (Mouse Systems)
Bus 001 Device 010: ID 12d1:14dc Huawei Technologies Co., Ltd.
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Figura 2.13. Listado de dispositivos USB conectados a la Raspberry Pi 3

Con el dispositivo USB ya reconocido por el sistema operativo como un módem 3G, no es necesario realizar más configuraciones; ya que automáticamente la interfaz se configurará al conectarse el dispositivo a cualquier de las ranuras USB de la Raspberry y el sistema le asignará una dirección IP interna para que pueda funcionar.

- **Módulo GPS**

La conexión para la alimentación del módulo Neo 6M se la realiza con los pines de 3,3 V en el pin 1 y la tierra (GND) con el pin 6 de los conectores GPIO de la Raspberry Pi. En cuanto a la conexión para la comunicación, es cruzada con los pines UART GPIO; con lo que el pin de transmisión del módulo es conectado al de recepción de la tarjeta y viceversa, como se muestra en la Figura 2.14.

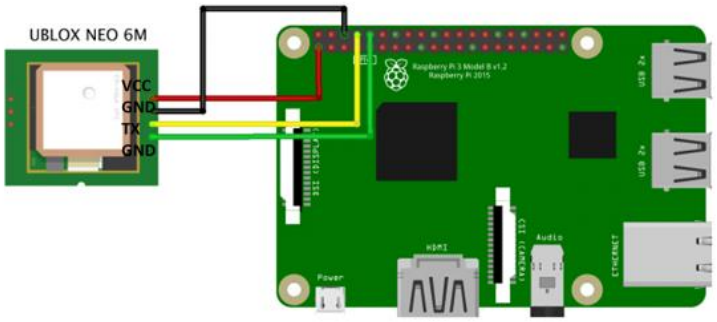


Figura 2.14. Diagrama de conexión módulo GPS y Raspberry Pi 3

- **Instalación y configuración del Módulo GPS**

Por defecto la Raspberry Pi usa la interfaz UART para establecer conexión serial de consola, por lo que esta funcionalidad debe ser desactivada para poder ser utilizada por el

módulo Neo 6M. El archivo cmdline.txt debe ser editado con el comando: *"sudo nano /boot/cmdline.txt"* y dentro de ese archivo eliminar la sección *"console=serial0, 115200"*, realizado esto se guarda el archivo y se reinicia la Raspberry Pi para que surtan efecto los cambios.

La instalación de los paquetes necesarios para la visualización de los datos del módulo directamente por la Raspberry Pi se realiza con el comando: *"sudo apt-get install gpsd gpsd-clients",* para comprobar que la recepción de datos se esté realizando de manera correcta se ejecuta el comando: *"sudo cat/dev/ttyS0"*, siendo ttyS0 la interfaz con la que se identifica al módulo GPS; la Figura 2.15 muestra las conexiones de las que está recibiendo datos GPS.

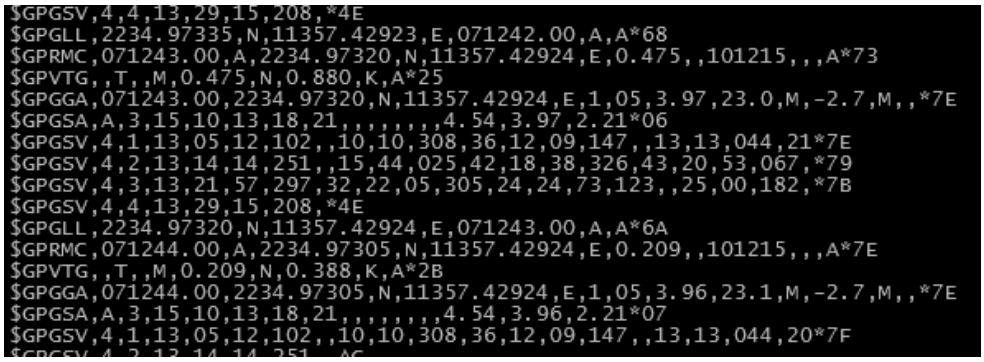


Figura 2.15. Conexiones activas del módulo GPS

- **UPS**

El modelo escogido (Greekworm UPS HAT Board) tiene la capacidad de ser conectado directamente sobre la Raspberry Pi haciendo uso de los pines GPIO, por lo que la energía suministrada a la Raspberry dependerá de este dispositivo. La conexión convencional de energía de 5 VDC y 2,5 A se realiza directamente al UPS HAT Board a través de su puerto micro USB.

- **Cliente FTP**

En la Raspberry Pi 3 se instala un cliente FTP compatible con esta plataforma y que permita conectarse al servidor remoto. Para ello se utiliza el paquete NcFTP, mismo que trabaja mediante líneas de comando, permite subida y descargas de ficheros y carpetas de manera recursiva, automatización de scripts de Shell, etc. Para la instalación del paquete se debe ejecutar la sentencia *"sudo apt-get install ncftp".*

2.5 Implementación de scripts

- **Adquisición de datos meteorológicos**

Tanto el sensor DHT-22 como el sensor BMP-180, son dispositivos digitales que se inicializan en el *void setup* del programa mediante los comandos: "*dht.begin ()* " y "*pressure.begin ()*" respectivamente. A continuación, en la Figura 2.16 se observa un diagrama de flujo con el proceso de adquisición de datos para los valores de temperatura, humedad relativa y presión atmosférica.

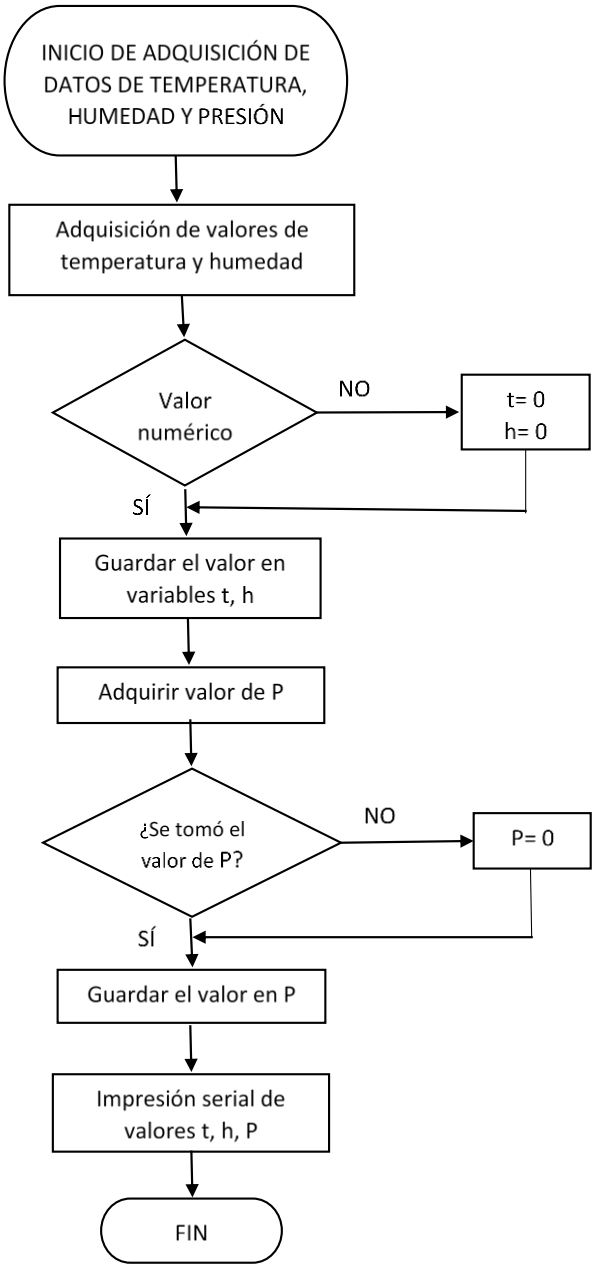


Figura 2.16. Diagrama de flujo de adquisición de datos para los valores de temperatura, humedad y presión

- **Envío de información a la plataforma Raspberry Pi 3**

La conexión entre la plataforma Arduino UNO y la plataforma Raspberry Pi 3 se hace mediante una comunicación serial. La inicialización del puerto serial del Arduino UNO se realiza mediante el comando: "*Serial.begin(9600)*", con una velocidad de transmisión de 9600 baudios[1].

Para el envío de información desde la plataforma Arduino UNO hacia el bloque de lectura, escritura y transmisión de datos; el Arduino Uno realiza una lectura de su buffer serial para verificar si ha recibido un carácter de control por parte de la Raspberry Pi.

Si en el buffer se encuentra un carácter "H", el Arduino UNO responde enviando los datos a la Raspberry Pi, caso contrario se manda un mensaje de "ERROR" y se reinicia el proceso, tal como se puede apreciar en la Figura 2.17.

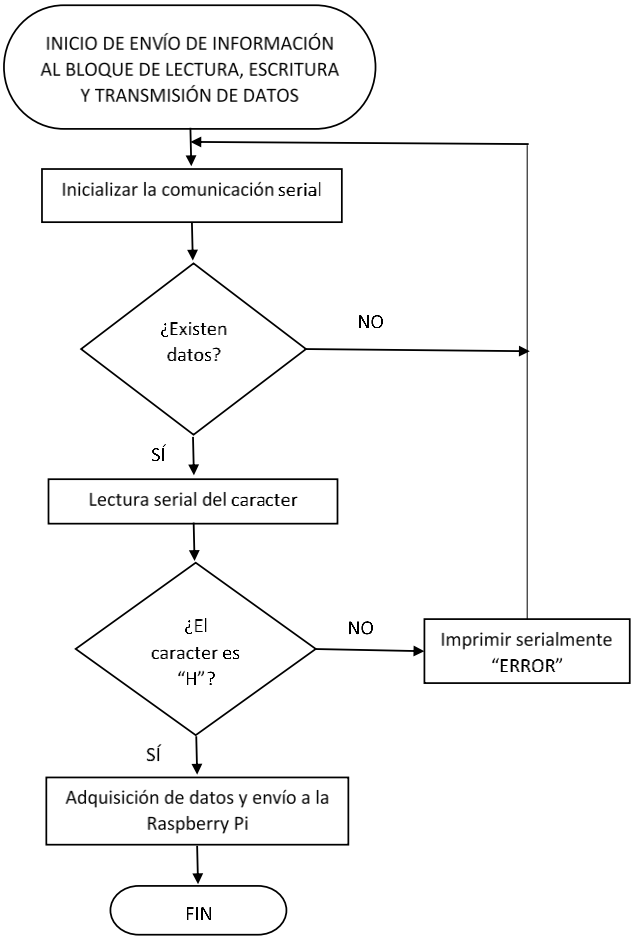


Figura 2.17. Diagrama de flujo para envío de información al bloque de lectura, escritura y transmisión de datos

[1] Baudio: unidad de medida utilizada para representar el número de símbolos por segundo.

- **Bloque de lectura, escritura y transmisión de datos**

➢ **Lectura y escritura de datos en un archivo de texto.**

Este proceso, que es realizado por la Raspberry Pi; comprende el tratamiento de los datos provenientes del Arduino UNO y de los datos GPS. Por lo que se han generado dos scripts separados, debido a que la frecuencia con la que se necesita leer los datos meteorológicos es mayor a la de los datos GPS.

❖ **Lectura y escritura de datos meteorológicos**

El orden para este proceso está presentado en el diagrama de la Figura 2.18, procedimiento que ha sido desarrollado en un script de Python que será ejecutado una vez por minuto por la Raspberry Pi.

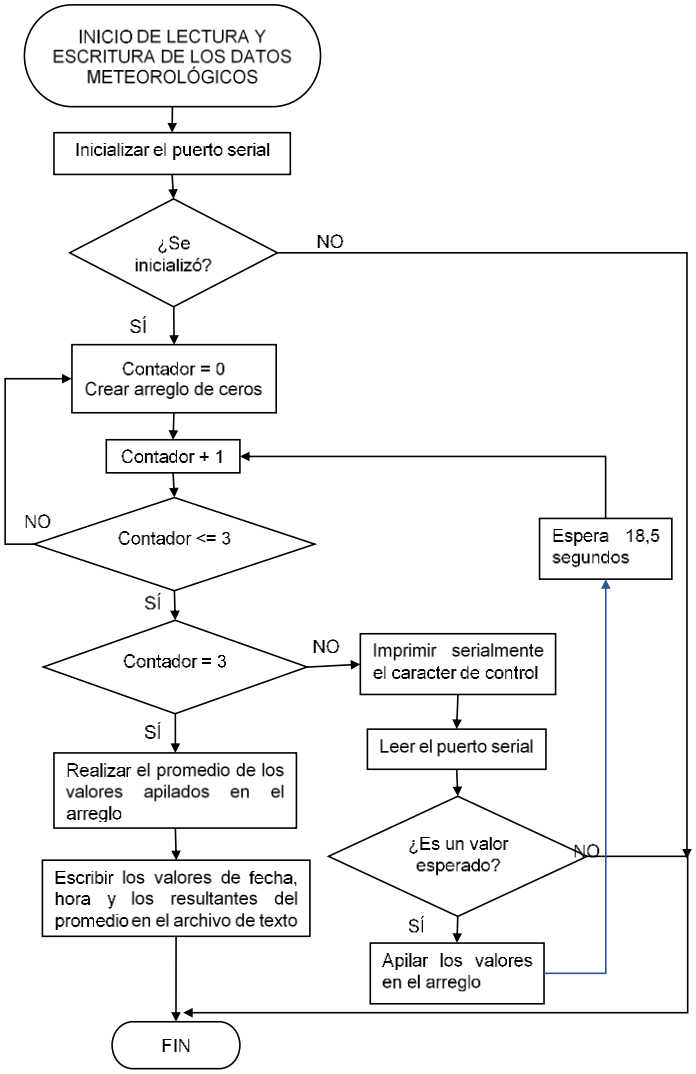


Figura 2.18. Diagrama de flujo para la lectura y escritura de datos meteorológicos en un archivo de texto

Ya que la lectura de los datos proviene de la conexión con el Arduino UNO, se debe inicializar el puerto serial de la Raspberry Pi que mantendrá activa la comunicación con la sentencia: *"arduino = serial.Serial('/dev/ttyACM0', 9600)".* Lo que quiere decir que la conexión llevará el nombre de la variable "arduino" y comunica al puerto /dev/ttyACM0 a 9600 baudios.

La Raspberry Pi envía periódicamente un caracter de control hacia el Arduino (letra "H"), espera que el proceso de adquisición de datos se complete y lee el puerto serial hasta encontrar un salto de línea, guardando el resultado en una variable llamada "mensaje". Las sentencias necesarias para cumplir con este proceso se aprecian en el Código 2.1

```
arduino.write('H') #Caracter de control enviado
time.sleep(1)
mensaje = arduino.readline() #Leer hasta encontrar salto de linea
```

Código 2.1. Comunicación entre Raspberry Pi 3 B y el Arduino Uno mediante Python

Los valores leídos por el Arduino con el formato de punto flotante son guardados temporalmente en una matriz, lo cual permitirá promediarlos posteriormente para determinar el valor final a ser almacenado en el archivo de texto.

```
a= numpy.fromstring(mensaje, dtype=float, sep="\t") #conversion de string a matriz numpy
arreglo = numpy.vstack((arreglo,a)) #creacion matriz de datos
```

Código 2.2. Almacenamiento de los datos leídos en una matriz

El valor promedio de cada minuto de los datos adquiridos es almacenado en un archivo de texto, cuyo nombre corresponde al de su fecha de creación. El archivo de texto es creado o abierto cada vez que el programa requiera escribir datos en él; para lo que se ha definido una función mostrada en el Código 2.3.

```
def archivo_texto():
    now = datetime.datetime.now()
    global name
    name= time.strftime ("%Y-%-m-%-d")
    try:
        text_file = open(name + '.txt', 'r')
    except IOError:
        text_file = open(name + '.txt', 'w')
```

Código 2.3. Creación del archivo de texto asociado a los datos meteorológicos

Finalmente, en el archivo de texto abierto, se escriben fecha, hora y el dato meteorológico correspondiente. Para lo cual se ha utilizado el Código 2.4.

```
text_file = open(name + '.txt', 'a')
text_file.write(time.strftime (" %y  %-m %-d %-H %-M %-S  "))
text_file.write(dato_final)
text_file.write("\n")
text_file.close()
```

Código 2.4. Escritura de datos en el archivo de texto

- **Lectura y escritura de datos GPS en un archivo de texto**

Cada vez que el sistema es reiniciado se necesita inicializar el módulo con el comando: "sudo gpsd /dev/ttyS0 -F /var/run/gpsd.sock", lo que permite que el paquete instalado gpsd interprete los datos del módulo y los pueda mostrar. Para poder inicializar el módulo GPS se ha creado un script en el Bash Shell que contiene la configuración inicial del módulo GPS, el Código 2.5 describe este proceso.

```
sudo systemctl stop gpsd.socket
sudo systemctl disable gpsd.socket
sudo gpsd /dev/ttyS0 -F /var/run/gpsd.sock
```

Código 2.5. Inicialización del módulo GPS

En cuanto al proceso que le proporciona valores de posicionamiento XYZ de la estación se ha desarrollado un script en Python, siguiendo el orden diagramado en la Figura 2.19.

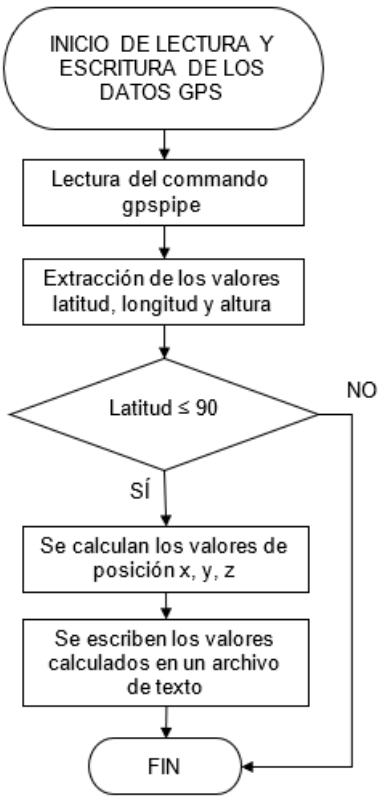


Figura 2.19. Diagrama de Flujo de la lectura y escritura de los datos GPS

El comando de sistema: *"gpspipe"* permite que mediante el paquete gpsd se obtengan lecturas GPS y se las muestre en el terminal; para poder almacenar la respuesta de un comando del sistema en Python es necesario emplear el comando "*subprocess.check_output("args")"* y como argumento el comando del sistema del que se requiere información. Se ha empleado este método para obtener las lecturas de latitud, longitud y altura provenientes del módulo GPS. El Código 2.6 muestra el fragmento del programa en el que se adquieren los valores de posicionamiento que serán transformados acorde al sistema de coordenadas empleado en el formato RINEX.

```
string= subprocess.check_output("gpspipe -w -n 5| grep -m 1 TPV| cut -d "','" -f 6-8", shell=True)
for ch in [':',',','"lat"','"lon"','"alt"']:
    if ch in string:
        string=string.replace(ch, " ")
lla= numpy.fromstring(string, dtype=float, sep="\t")
latitud = float(numpy.take(lla, [0]))
longitud = float(numpy.take(lla, [1]))
altura = float(numpy.take(lla, [2]))
```

Código 2.6. Adquisición de valores de latitud, longitud y altura

La opción -w de gpspipe indica que va a imprimir las sentencias nativas de gpsd delimitadas por (;), la opción -n establece un contador de sentencias antes de cerrar el programa, que para el caso son 5 sentencias dentro de las cuales se encuentran valores de tiempo, latitud y longitud.

Se han empleado tuberías para direccionar la salida de gpspipe al comando grep que con la opción -m permite obtener las primeras líneas que contengan una palabra TPV dadas por un contador. Para este caso busca la primera línea (1) que contenga la palabra TPV. A su vez la salida de este comando se redirecciona al comando *"cut"* que acompañada de -d establece el campo delimitador (',') y la opción -f imprime únicamente los campos especificados y delimitados, que para este caso se utilizan los campos 6 al 8.

En la extracción de valores de latitud, longitud y altura podrían presentarse cifras inválidas, que son valores mucho más grandes que los esperados con el fin de indicar un mal funcionamiento del módulo GPS. Razón por la cual se toma como referencia un valor de latitud válido como pase para realizar los cálculos de posicionamiento XYZ; el Código 2.7 describe el proceso para la transformación de coordenadas.

```
R=a/(math.sqrt(1-((math.exp(2))*((math.sin(latitud))**2))))
x=(R+altura)*math.cos(latitud)*math.cos(longitud)
y=(R+altura)*math.cos(latitud)*math.sin(longitud)
z=(((1-math.exp(2))*R)+altura)*math.sin(latitud)
```

Código 2.7. Transformación de coordenadas latitud, longitud y altura a XYZ

Finalmente, los valores de posicionamiento XYZ son guardados en el archivo “Location.txt”, localizado en el directorio “home/pi/DATA”. Para ser accedidos posteriormente en la creación de los archivos RINEX.

- **Almacenamiento de datos en una base de datos**

La conexión que se realiza será entre la Raspberry Pi y la base de datos alojada en el servidor remoto, mediante un script programado en Python.

En primer lugar, se abre el archivo de texto que contiene los datos meteorológicos en modo de lectura con la sentencia: “*text_file = open(name + '.txt', 'r')”,* del mismo que se leerá la última entrada con el comando: “*last_line = text_file.readlines()[-1]*” y guarda su contenido en la variable “*last_line*”.

Los valores que serán almacenados en la tabla de la base de datos deben ser convertidos al formato específico de cada uno de ellos definido en la creación de la tabla en el servidor remoto. El comando “*numpy.fromstring(args)”* ha sido utilizado para tomar los valores del texto leído y que estén separados por espacios, almacenarlos en una matriz con el formato de punto flotante. En cuanto al formato de la hora se ha necesitado separar los valores leídos con “:” y eliminar los puntos decimales; para la fecha se ha decidido utilizar el comando: “*date = datetime.now().date()”* que proporciona la fecha del sistema. La separación de los valores de la última lectura es realizada como se muestra en el Código 2.8.

```
a= numpy.fromstring(last_line, dtype=float, sep="\t")
datos= numpy.take(a, [6,7,8])
hora = numpy.take(a, [3,4,5])
hour = ':'.join(map(str,hora))
hour = hour.replace(".0", "")
date = datetime.now().date()
```

Código 2.8. Asignación de los datos meteorológicos a variables

Una vez extraídos los datos que serán escritos en la tabla de la base de datos, se realiza la conexión de la base de datos según el diagrama de la Figura 2.20; que representa la función “base_de_datos()”.

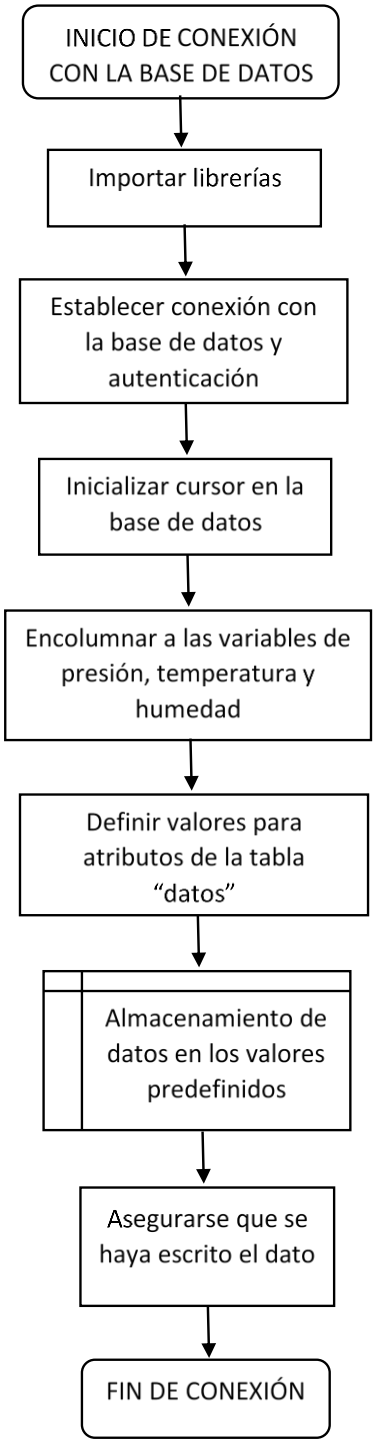


Figura 2.20. Diagrama de bloques para establecer conexión con la base de datos

Una vez llamadas a las librerías necesarias para establecer la conexión con la base de datos MySQL, se procede a establecer la conexión y autenticación mediante el comando: *"db=mysql.connector.connect(user='Usuario',passwd='Contraseña',host='IP',db='Base de datos')".* Con los correspondientes campos de seguridad configurados en la Base de datos del servidor remoto.

Posteriormente, se inicializa un cursor en la conexión establecida con el comando: *"cursor = db.cursor()*". El cursor permitirá posteriormente ejecutar la sentencia que añade los datos en la base de datos.

Para la inserción de los datos en la tabla "datos" es necesario definir la variable "add_dato" que contiene la sintaxis de SQL para agregar nuevas entradas en la tabla, en donde la sentencia: "*INSERT INTO"*, tiene como argumentos el nombre de la tabla en análisis y los valores de cada atributo de la tabla. Además, las variables generadas en Python deben estar organizadas en una variable que contenga estos valores (variable "dato").

Para ejecutar los comandos de SQL se ha empleado la función: "cursor.execute(), en el Código 2.9 se describe el proceso para añadir las entradas en la tabla.

```
add_dato = ("INSERT INTO datos"
            "(Hora, Fecha, Presion, Temperatura, Humedad)"
            "VALUES (%(hora)s, %(fecha)s, %(presion)s, %(temperatura)s, %(humedad)s)")

dato = {
        'hora' : hour,
        'fecha' : date,
        'presion' : PRESION,
        'temperatura' : TEMPERATURA,
        'humedad' : HUMEDAD,
        }

cursor.execute(add_dato,dato) #agregar dato
```

Código 2.9. Inserción de datos en la tabla "datos"

Finalmente, se asegura que el dato haya escrito en la tabla empleando la sentencia: "*db.commit()*" y procede a desactivar el cursor y finalizar la conexión con la base de datos con los comandos: "*cursor.close()"* y *"db.close()"* respectivamente.

- **Transmisión de datos en formato RINEX**.

El archivo de texto asociado a los datos meteorológicos generado en primera instancia no cuenta con el formato de archivo que maneja el IGM; razón por la cual se generó un programa capaz de crear encabezados y alineación de datos tal cual lo exigen los archivos RINEX.

La programación de este proceso ha sido desarrollada en el Bash Shell de la Raspberry Pi siguiendo el diagrama de flujo de la Figura 2.21 y diseñada para ser ejecutada una vez al día.

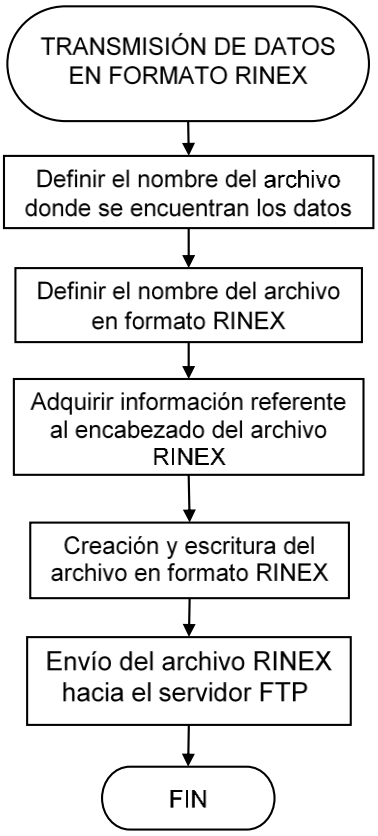


Figura 2.21. Diagrama de flujo de transmisión de datos en formato RINEX

El archivo de datos meteorológicos está nombrado con el formato YYYY-MM-DD, es decir que tendría la siguiente forma: “2018-3-21.txt”. A este nombre se lo volverá a definir con el nuevo formato, el cual consta del nombre de la estación, día del año y los dos últimos dígitos correspondientes al año en curso, es decir: “IGMQ080.18M”.

Para poder acceder a los archivos es necesario localizarse en el directorio que contiene los archivos de datos meteorológicos: /home/pi/DATA; el archivo a enviarse corresponde al del día anterior por lo que a la respuesta del comando “*date*” se le debe restar un día.

El comando: “*sed*” permite editar texto de una entrada como un *stream* de texto o de un archivo; en este caso ha servido para retirar los símbolos que no corresponden con el formato definido. Los comandos que están descritos en el Código 2.10 permiten obtener el nombre del archivo que contiene los datos meteorológicos.

```
###ARCHIVO ABRIR
cd /home/pi/DATA/
hoy=$(date +%Y-%m-%d )
ayer=$(date --date="${hoy} -1 day" +%Y-%m-%d | sed "s/^0*//g; s/\-0*/-/g")
archivo="${ayer}"".txt"
```

Código 2.10. Obtención del nombre del archivo de datos meteorológicos

En el Código 2.11 se describe el proceso para la obtención del nombre del archivo RINEX, mismo que es almacenado en la variable "*nuevo*".

```
###ARCHIVO CREAR
day=$(date --date="${hoy} -1 day" +%j)#dia del año
year=$(date --date="${hoy} -1 day" +%y)#ultimos dos digitos del año
estacion="IGMQ"
nuevo="${estacion}""${day}""0"".${year}""M"
```

Código 2.11. Creación del nombre del archivo RINEX

Ciertos campos que incorporan el encabezado del formato RINEX son tomados por la estación, tales como: valores de posicionamiento XYZ de la estación, versión del sistema operativo que genera el archivo, fecha y hora de su creación. Para la adquisición de estos datos se han empleado las sentencias del Código 2.12. El comando "*tail*" permite listar el contenido de un archivo, y la opción -n 1 listará solamente su última línea.

```
###DATOS ESTACION
version=$(uname -rms)
fecha=$(date +%Y%m%d)
hora=$(date +%H:%M:%S)
xyz=$( tail -n 1 /home/pi/DATA/Location.txt )
```

Código 2.12. Adquisición de campos de la cabecera RINEX

Los datos deben ser ajustados a una alineación propia del archivo RINEX, misma que ha sido desarrollada según el Código 2.13.

```
###ALINEACION
ki="AA BB CC DD EE FF 0.00000 0.000000 0.000000  "
sed -i "1i${ki}" $archivo
tail -n+1 ${archivo}| rev | column -t|rev | sed -e "s/\  /\ /g"|sed "1d"> ${nuevo}
sed -i -e "s/^/ /" ${nuevo}
sed -i -e "s/$/    0.0    0.0    0.0    0.0/" ${nuevo}
```

Código 2.13. Alineación de los datos según el formato RINEX

Para la alineación ha sido necesario la inserción de un patrón "ki" en el archivo de datos meteorológicos. Para alinear los valores listados del archivo se emplea el comando: "*column -t*" que los encolumna separados por espacios en blanco. Ya que el comando empleado para alinear los valores solo se hace hacia la izquierda, por lo que se ha tenido que revertir el contenido de las líneas con el comando: "*rev*" permitiendo alinearlos hacia la derecha como en el formato RINEX.

La salida de este proceso es dirigida al comando "*sed -e "s/\ /\ /g"* que reemplaza los espacios dobles por espacios simples y consecuentemente al comando *"sed "1d""* que se encarga de eliminar el patrón "ki". En la parte correspondiente a los datos es necesario

agregar un espacio al inicio de cada línea, y al final las columnas de los sensores no conectados que son valores de “0.0”.

El encabezado es añadido en el archivo RINEX con el comando *“sed -i "1i header ${nuevo}””,* donde “*header*” contiene todo el texto del encabezado RINEX.

- **Envío del archivo RINEX al servidor FTP**

Ya obtenido el archivo RINEX, debe ser enviado al servidor FTP remoto. A continuación, se describe la lógica utilizada para el envío del archivo RINEX al servidor remoto FTP conforme el diagrama de flujo de la Figura 2.22.

El archivo RINEX generado con anterioridad y almacenado en la carpeta /home/pi/DATA debe ser enviado al servidor remoto FTP, esto se consigue ejecutando el comando *“ncftpput -u “Usuario” -p “Contraseña” IP /directorio_destino /home/pi/DATA/${nuevo}”*. En caso de que el archivo no pueda ser enviado, se reintentará el envío en un intervalo de 10 minutos hasta obtener éxito. Si el proceso se completó, el archivo almacenado en la carpeta /home/pi/DATA será movido a la carpeta /home/pi/SENT.

Ya enviado el archivo RINEX, es necesario limpiar la carpeta /home/pi/DATA para almacenar nuevos archivos. Para este proceso se envían los archivos remanentes de días anteriores al servidor mediante el comando *“ncftpput -R -u “Usuario” -p “Contraseña” IP /directorio_destino /home/pi/DATA”*, en donde la bandera *-R* permite enviar todos los archivos de esa carpeta de manera recursiva.

Una vez enviados todos los archivos de la carpeta /home/pi/DATA se procede a eliminarlos a excepción del archivo de datos meteorológicos que se encuentra generándose en ese momento, conforme al Código 2.14.

```
cd /home/pi/DATA/
shopt -s extglob
rm -v !("$actual")
```

Código 2.14. Conservación del archivo actual

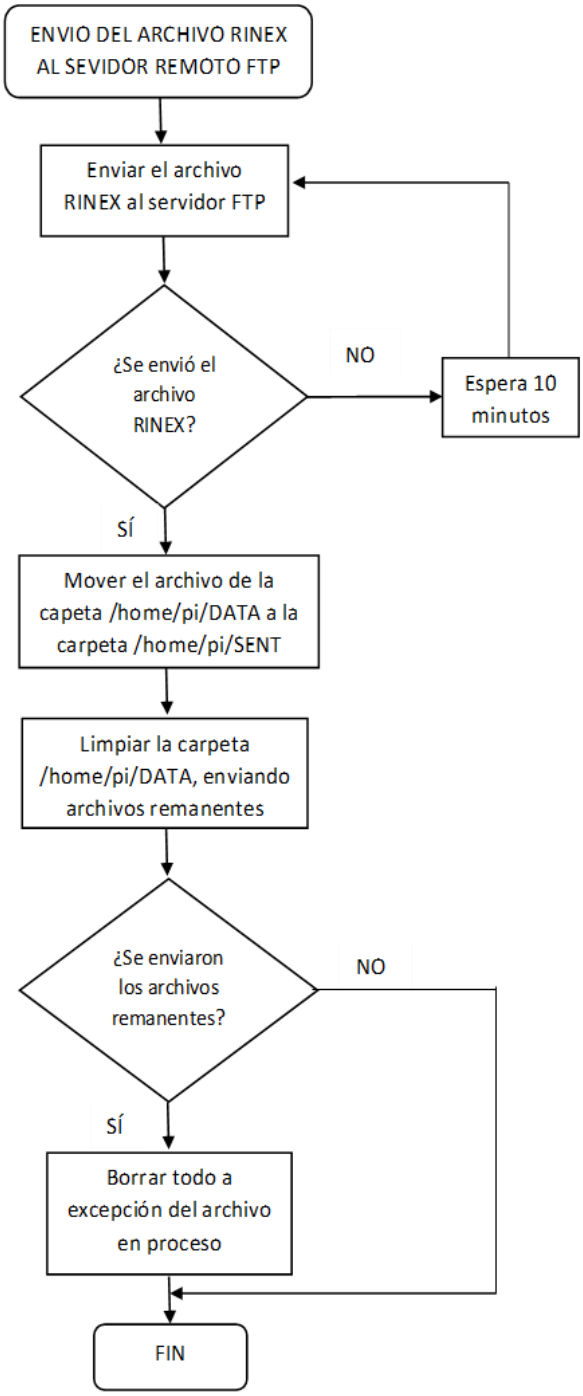


Figura 2.22. Diagrama de flujo para envío de archivos RINEX al servidor FTP

2.5.1 Automatización de scripts

En el presente proyecto se manejan dos tipos de scripts, los desarrollados en el Bash Shell de Raspbian y los desarrollados en Python. Cada uno de estos scripts corresponde a una tarea específica del prototipo que debe ser ejecutada automáticamente cada cierto tiempo según lo requiera basándose en el diseño de la estación.

- **Crontab**

Cron es un “demonio” que permite ejecutar automáticamente comandos del sistema operativo a una hora o fecha específica en segundo plano. En la distribución de Raspbian utilizada para el proyecto ya viene instalado y configurado para ejecutarse durante el arranque.

Crontab (CRON TABle) es un archivo que contiene un cronograma de entradas tipo *cron* a ser ejecutadas en tiempos específicos. Para ingresar a este archivo es necesario en la consola de Raspbian ingresar el comando: “*sudo crontab -e”*, que permite editar o crear el archivo de no existir uno.

El formato al que se deben acoger las entradas del archivo *crontab* para comandos que requieran ser ejecutados periódicamente se encuentra especificado en la Figura 2.23. Los campos que no se requieran utilizar para ejecutar un comando son sustituidos con un asterisco (“*”) [60].

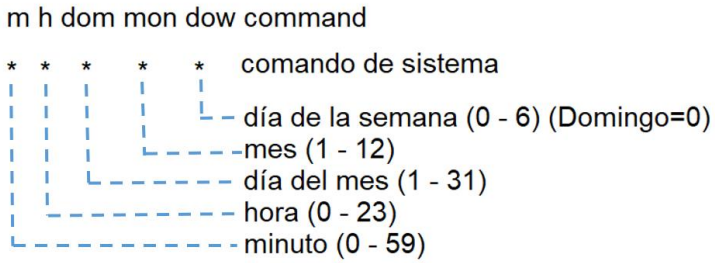


Figura 2.23. Formato de entradas *cron*

Además, para un comando que se requieran ejecutarse una sola vez al inicializar el sistema de la Raspberry Pi, la entrada *cron* tendrá la siguiente sintaxis: “*@reboot comando”*.

- **Configuración de crontab**

Los scripts que son parte de la Raspberry Pi deben ser ejecutados según las especificaciones del proyecto, en la Tabla 2.1 se presenta la información para la automatización de todos los procesos del prototipo.

Tabla 2.1. Características de los scripts desarrollados

Descripción del script	Comando para la ejecución	Frecuencia de ejecución
Configuración inicial de módulo GPS	sudo bash /home/pi/gps.sh	Al reiniciar el sistema
Lectura y escritura de datos meteorológicos en un archivo de texto	Sudo python /home/pi/LECTURA.py	Cada minuto

Descripción del script	Comando para la ejecución	Frecuencia de ejecución
Transmisión de datos en formato RINEX	sudo bash /home/pi/justify.sh	A las 19:00 horas
Almacenamiento de datos en una base de datos	sudo python /home/pi/BASE.py	Cada 10 minutos
Lectura y escritura de datos GPS en un archivo de texto	sudo python /home/pi/XYZ.py	Cada 3 horas

Cada uno de los scripts antes mencionados se encuentra asociado a una entrada tipo cron. La configuración realizada en el archivo crontab se la detalla en la Figura 2.24.

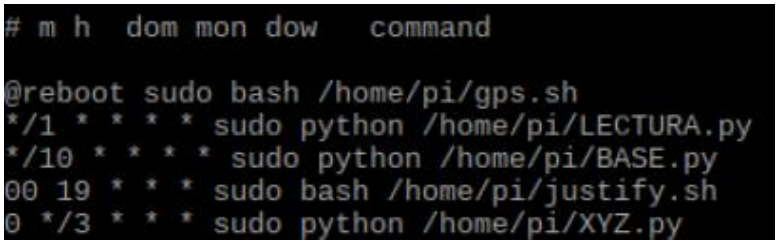


Figura 2.24. Entradas cron del archivo crontab

2.6 Instalación y configuración de los servicios que conforman al subsistema receptor

2.6.1 Bloque de almacenamiento de datos

La instalación del gestor de datos MySQL, perteneciente al paquete LAMP, requiere de la ejecución del comando: *"sudo apt-get install mysql-server".* La gestión y administración de la base de datos se realiza mediante una herramienta gráfica llamada PhpMyAdmin, misma que evita el uso de sentencias SQL.

PhpMyAdmin es instalada al ejecutar el comando: *"sudo apt-get install phpmyadmin"*, durante la instalación solicita el ingreso de la clave del usuario *"root"* de MySQL. Para acceder a la base de datos se ingresa la ruta *"http://localhost/phpmyadmin/"* en el navegador de preferencia.

Una vez ingresado en PhpMyAdmin, se procede a la creación de la base datos llamada "Mediciones", que a su vez está formada por una tabla llamada "datos". Los campos que conforman la tabla son:

Clave Primaria (id_mediciones): identificador único del número de medición. Corresponde a un número entero autoincremental.

Fecha: representa la fecha en la que se toma la muestra, corresponde al tipo *"date"* y se muestra en el formato año-mes-día.

Hora: representa la hora en la que se toma la muestra, corresponde al tipo *"time"* y se muestra en el formato HH:MM:SS

Presión, Temperatura, Humedad: correspondiente al tipo *"float"*, muestra un número decimal, para presión atmosférica en mbar (hPa), temperatura en ^{0}C y humedad relativa expresada en porcentaje.

La base de datos "Mediciones" y los campos que conforman su tabla "datos", se pueden visualizar en la Figura 2.25.

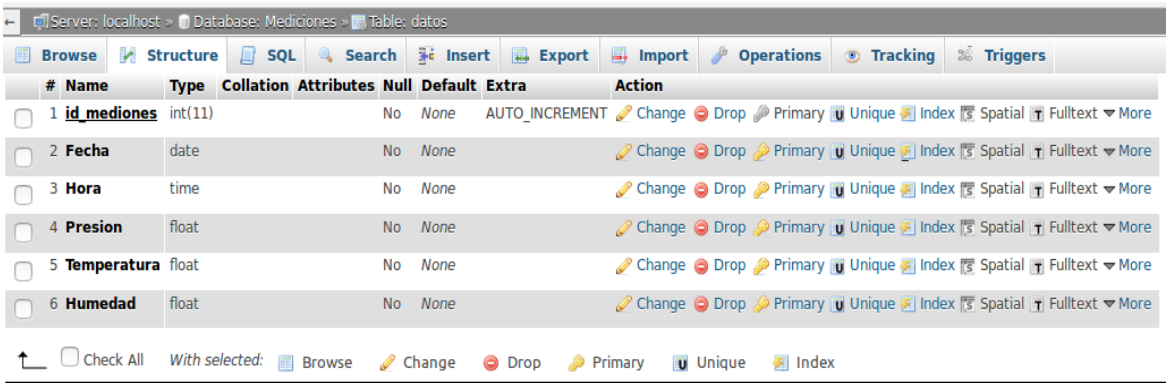


Figura 2.25. Campos que forman la tabla "Datos"

2.6.2 Bloque de visualización y administración de datos

- **Servidor Web**

El servidor Web correrá bajo la implementación del servidor Apache, perteneciente al paquete LAMP, que permite la creación de páginas y servicios web gracias a que integra los lenguajes HTML[2](*Hyper Text Markup Language*, Lenguaje de Marcado de Hyper Texto) y CSS[3](*Cascading Style Sheets*, Hojas de Estilo en Cascada). La instalación es sencilla, pues basta con ejecutar el comando: *"sudo apt-get install apache2".*

Adicional al servidor web se instala el paquete PHP, que en conjunto con otros complementos facilitan la conexión con la base de datos. La instalación se realiza a través del comando: *"sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql".*

[2] HTML: lenguaje que permite la definición de contenido de una página web como texto, imágenes, etc.

[3] CSS: lenguaje utilizado para definir y crear el diseño visual de documentos web e interfaces de usuarios escritas en HTML.

En base a lo antes mencionado, se procede a la creación de la página web basada en PHP, HTML y CSS presentando una interfaz web dinámica y amigable con el usuario.

La página web estará formada por: pestaña de inicio, pestaña proyecto, pestaña datos IGMQ y pestaña descarga de archivos, tal como se aprecia en la Figura 2.26.

Figura 2.26. Interfaz web disponible para el usuario

Pestaña Inicio: indica las entidades participantes del proyecto y objetivo de la página web.

Pestaña Proyecto: desarrolla una breve descripción acerca del proyecto de titulación.

Pestaña Datos IGMQ: muestra los valores almacenados en la base de datos en grupos de 360, 180 o 60.

Pestaña Descarga de archivos: permite conectarse al servidor FTP para realizar la descarga de los archivos RINEX generados por la estación IGMQ.

Para establecer la conexión de la base datos con el servidor web, se crea un script PHP utilizando las funciones: *"mysqli_connect"* y *"mysqli_query".* La primera función permite la conexión con la base de datos "Mediciones", mientras que la segunda función permite extraer los datos almacenados en la tabla "datos" para visualizarlos en la página web de manera ordenada y encolumnada.

- **Servidor FTP (*File Transfer Protocol*, Protocolo de Transferencia de Archivos)**

Para la instalación del servidor FTP se requiere de un servidor disponible para Ubuntu y de fácil administración para el usuario.

En base a lo antes mencionado se utiliza el servidor VSFTPD, servidor que permite la transferencia de archivos de manera segura y eficiente. Para instalar este servicio se debe ejecutar el comando "*sudo apt-get install vsftpd".* Una vez instalado el paquete se procede a editar el archivo de configuración /etc/vsftpd.conf acorde a las necesidades del proyecto. Es importante deshabilitar la opción de conexión anónima que viene activada por defecto, dado que esta permite la conexión de cualquier usuario hacia el servidor sin antes habérsele asignado un usuario y contraseña.

Posteriormente se crean usuarios FTP con contraseñas robustas, mismos que están restringidos a acceder a un solo directorio específico por motivos de seguridad.

2.6.3 Bloque de tratamiento de datos

El modelo predictivo se llevará a cabo utilizando el software libre WEKA creado por el grupo de "*Machine Learning"* de la Universidad de Waikato en Nueva Zelanda. Este software ha sido desarrollado como un entorno de análisis de datos para aplicar, analizar y evaluar técnicas del aprendizaje automático; también posee una colección de algoritmos de "*Machine Learning*" permitiendo utilizarlos en tareas de procesamiento, clasificación, agrupamiento y asociación de datos [61]. Para ejecutar la última versión de WEKA se requiere de Java 8 o posteriores, para su instalación en Ubuntu es necesario ejecutar el comando "*sudo apt-get install weka".*

El formato de archivo que contiene los datos a analizarse es el conocido como ARFF (Attribute-Relation File Format).

WEKA ha desarrollado una interfaz gráfica, misma que permite acceder y configurar las distintas herramientas integradas tal como se muestra en la Figura 2.27.

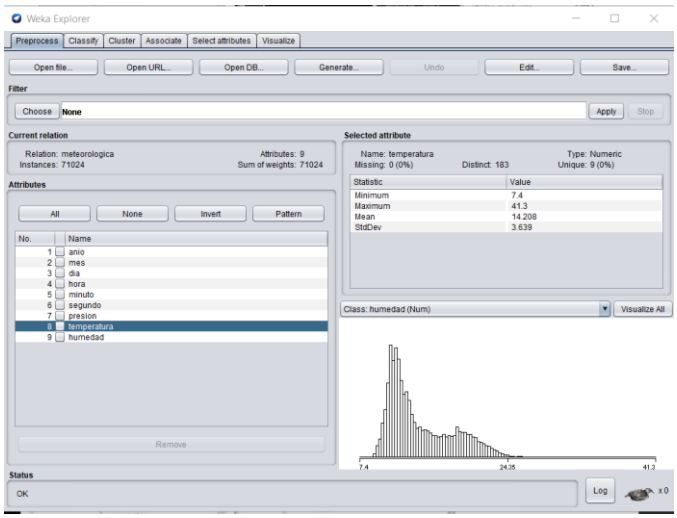


Figura 2.27. Interfaz gráfica de WEKA

Los datos de entrada al software WEKA almacenados en el archivo “arff” deben ser cargados en la opción “*Open File*” para su ejecución. Este tipo de archivo consta de dos secciones: cabecera y datos.

La cabecera del archivo “arff” incluye el nombre de la relación, declaración de los atributos y su tipo. Es importante identificar el tipo de dato contenido en el archivo es decir si son de tipo numérico o nominal.

La sección de datos comienza con la declaración “@data”, seguido de los datos. Cada línea de datos representa una instancia, en la cual los atributos deben separarse por comas o espacios.

Para este caso se ha utilizado un archivo “arff” conformado por datos a partir del 18 de abril del 2018 hasta el 24 de junio del 2018, obedeciendo al formato antes mencionado, tal como se muestra en la Figura 2.28.

```
@relation meteorologica

@attribute dia real
@attribute hora real
@attribute temperatura real

@data
108 4 11.425
108 9 11.4
108 14 11.4
```

Figura 2.28. Estructura de archivo "arff" utilizado

En la Figura 2.28 el atributo “dia” corresponde al valor del día del año en curso, mientras que el atributo “hora” corresponde al valor del minuto del día en cuestión. En tanto que el atributo “temperatura” corresponde al valor de temperatura promediada cada cinco minutos.

Una vez realizado lo antes mencionado, en la opción “*Classify*” se muestran todos los algoritmos disponibles en WEKA tanto para valores continuos como para valores discretos, conforme a la Figura 2.29.

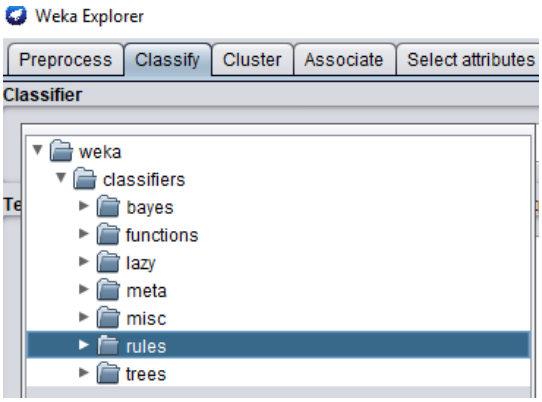


Figura 2.29. Algoritmos disponibles en WEKA

Como se observa en la Figura 2.29 existen varios tipos de algoritmos, para este proyecto se han probado y escogido los más óptimos para trabajar con regresiones.

Dentro del clasificador “functions” se encuentran los algoritmos de regresión lineal, regresiones lineales simples, etc. Considerados los más simples para los casos de predicción numérica cuyo enfoque se basa en el establecimiento de una ecuación matemática como modelo para representar las interacciones entre las diferentes variables [62].

Los clasificadores “lazy” admiten tanto la clasificación como la regresión. Entre los algoritmos más idóneos para el proyecto está el algoritmo “IBK”, cuyo principio es la comparativa de cada instancia con los K vecinos más cercanos en el conjunto de datos de entrenamiento [63].

Los clasificadores “rules” cuentan con algoritmos capaces de predecir valores numéricos y discretos. Es así como el algoritmo “M5RULES” se basa en la generación de reglas de regresión basadas en modelos “tree” [64].

Los clasificadores “trees” cuentan con varios algoritmos, el más representativo es el algoritmo “RandomForest” mismo que produce árboles de clasificación y regresión dependiendo de la variable si es categórica o numérica [62]. Uno de los principales problemas que presentaron los clasificadores *trees* para los datos ingresados fue el *overfitting*, es decir, que el modelo aprende el detalle y ruido de los datos de entrenamiento en la medida en que tiene un impacto negativo en el rendimiento del modelo en nuevos datos.

En base a lo antes mencionado, se decidió generar modelos con los algoritmos “IBK” y “M5RULES”, realizando las configuraciones correspondientes para cada algoritmo.

- **Configuración del algoritmo "M5RULES"**

La configuración necesaria para el algoritmo se realiza en las propiedades que presenta el mismo, en la Figura 2.30 muestra los distintos campos configurables.

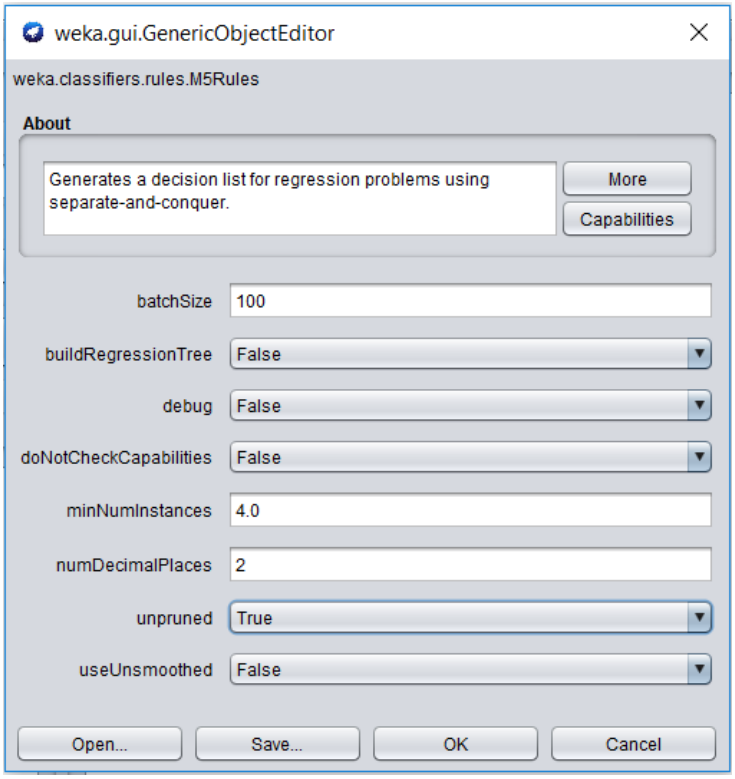


Figura 2.30. Configuración del algoritmo M5RULES

Cada uno de los campos presentes en la Figura 2.30 se detallan a continuación:

BatchSize: permite especificar el número de instancias analizadas presentes en una regla, se ha trabajado con un "batch size" =100.

BuildRegressionTree: en el caso de desear obtener árboles de decisión de regresión se activa esta opción. Para este proyecto la configuración es en "False".

Debug: añade información adicional al resultado de la clasificación, para este caso la configuramos en "False".

DoNotCheckCapabilities: permite verificar las capacidades del algoritmo antes de construir el clasificador, sin embargo, extiende el tiempo de ejecución del algoritmo. Para este caso se lo configura en "False".

MinNumInstances: indica el número de instancias que permitirán la creación de la regla, cuya notación para el proyecto será N. En este caso se configuró para los valores 4 y 6 con la finalidad de observar las diferencias obtenidas.

numDecimalPlaces: indica la cantidad de decimales que se obtendrá en los valores de salida del modelo. Para este caso se configuró en el valor de 2.

Unpruned: indica si se deben preservar todas las reglas generadas por el algoritmo u omitir las consideradas obvias. Para este caso se la configura en "True", consiguiéndose un modelo de predicción robusto para el valor de temperatura, ya que cuenta con un mayor número de reglas.

UseUnsmoothed: permite realizar un filtrado de predicciones. Para este caso se lo configura en "False".

- **Configuración del algoritmo "IBK"**

Para configurar este algoritmo se debe modificar las configuraciones preestablecidas por el software WEKA, conforme a la Figura 2.31.

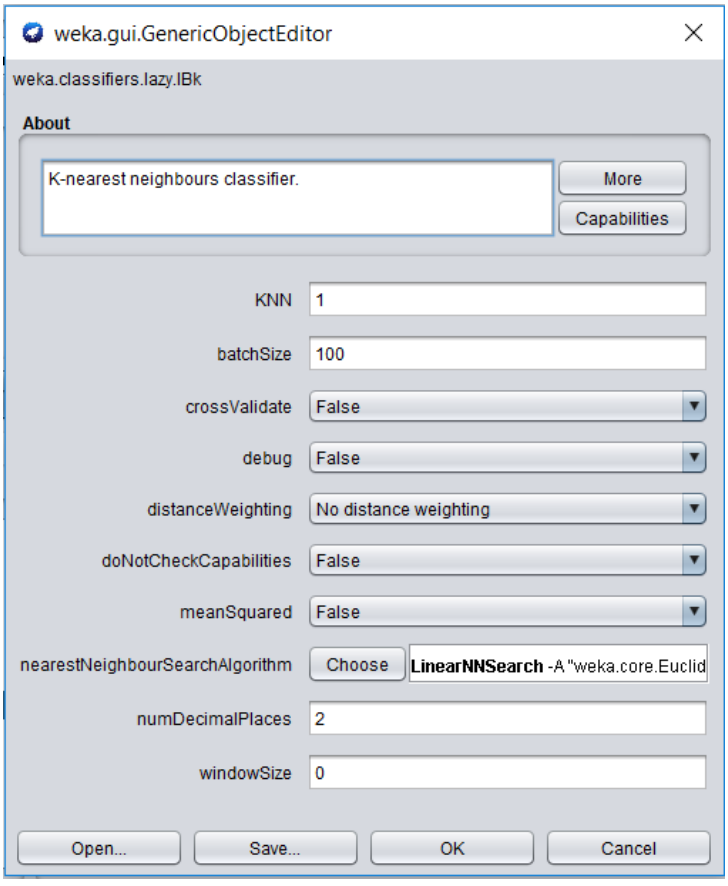


Figura 2.31. Configuración del algoritmo IBK

A continuación, se describe los principales parámetros a configurarse.

KNN: este parámetro indica el número de instancias vecinas similares se van a utilizar para la predicción. Para este caso se tomaron valores de 100 y 250.

CrossValidate: permite descubrir automáticamente el mejor valor de K para la predicción. Para este caso se lo configura en “False”.

DistanceWeighting: permite configurar la ponderación de la distancia del método utilizado. Para este caso se conserva el valor predeterminado.

NearestNeighbourSearchAlgorithm: controla la forma en que se almacenan y buscan los datos de entrenamiento. Se recomienda dejar con el valor predeterminado “LinearNNSearch” para valores numéricos.

Para las demás opciones se ha decidido mantener la configuración predeterminada por el software, dado que de esta manera se han conseguido resultados aceptables.

Para las pruebas de predicción el propio software permite realizar una división de los datos ingresados, de tal manera que un porcentaje de estos será utilizado para entrenar el modelo y el resto de los datos servirán para comparar los resultados obtenidos por el modelo. La opción pertinente para realizar la división de los datos es llamada “*Percentage split*” que se encuentra en la ventana de “*Test options*”, donde es necesario especificar el porcentaje requerido para el grupo de entrenamiento como se muestra en la.

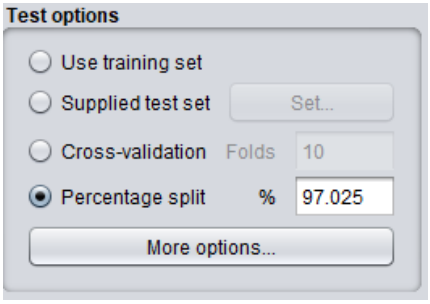


Figura 2.32. Ventana "Test Options"

Además, en la opción “*More options…”* se ha configurado que se mantenga el orden para el porcentaje de división de tal manera que las predicciones correspondan a los últimos valores del grupo de datos ingresado, dentro de esta opción se configura que los resultados del modelo sean mostrados como texto plano.

Bajo la ventana “*Test options*” se encuentra un selector que permite escoger la variable a ser predicha por el modelo, y el botón “*Start”* que genera el modelo.

Una vez que el modelo ha sido generado, en la ventana “*Classifier output”* aparecerán los datos importantes referentes al modelo, al igual que los resultados de la predicción y la comparación con los valores reales como se muestra en la Figura 2.33.

```
=== Predictions on test split ===

    inst#     actual  predicted      error
        1     10.55      12.359      1.809
        2     10.35      12.373      2.023
        3     10.4       12.388      1.988
        4     10.475     12.402      1.927
        5     10.425     12.416      1.991
        6     10.45      12.226      1.776
        7     10.5       12.212      1.712
        8     10.425     12.198      1.773
        9     10.05      12.183      2.133
       10     10.025     12.169      2.144
```

Figura 2.33. Comparación de valores reales y predichos mostrado por WEKA

Al final de la salida del clasificador se muestran resultados adicionales como los valores de coeficiente de correlación de la muestra, al igual que valores de error absoluto promedio, error medio cuadrático, error absoluto relativo y error cuadrático relativo.

3. RESULTADOS Y DISCUSIÓN

En esta sección se presentan los resultados del proyecto de titulación. Para ello se realizan las pruebas de funcionamiento de la estación prototipo, comparación con la estación científica, predicción del valor de temperatura y finalmente el costo referencial para de la estación prototipo.

3.1 Instalación del sistema prototipo

La estación meteorológica prototipo ha sido instalada en la terraza del edificio principal del Instituto Geográfico Militar, ubicado entre la Avenida Seniergues y la calle Gral. Telmo Paz y Miño, en la ciudad de Quito. Y con el servidor remoto ubicado en la Escuela Politécnica Nacional en el edificio de Química-Eléctrica.

La ubicación geográfica en la que han sido instalados los componentes se muestra en la Figura 3.1 mediante una captura satelital del programa Google Earth.

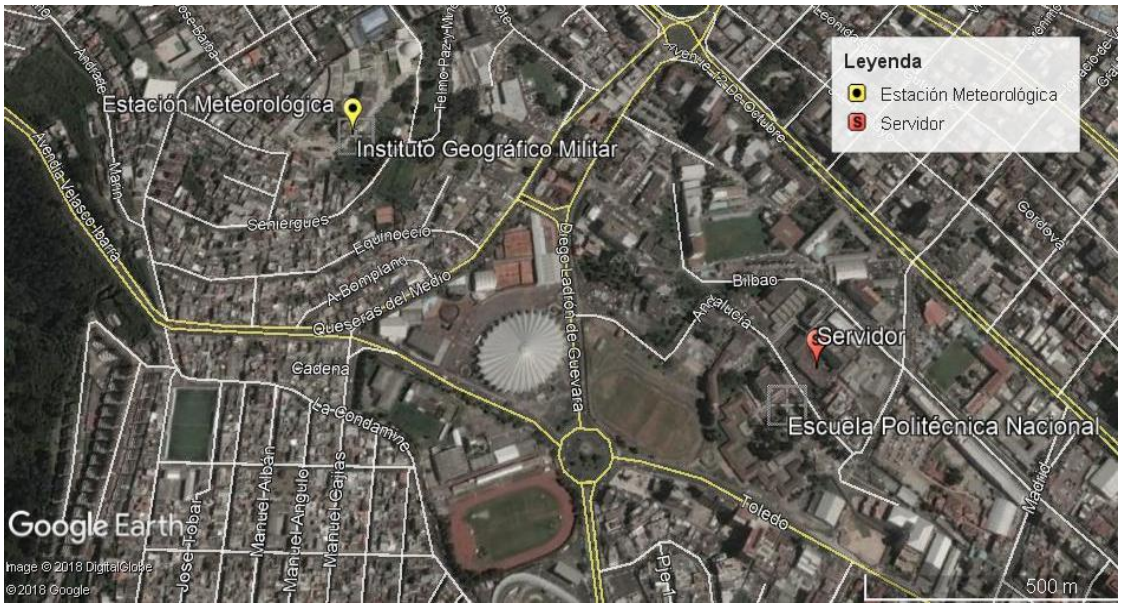


Figura 3.1. Ubicación geográfica de los componentes del proyecto

Con la finalidad de proteger los sensores, y a su vez proporcionar las condiciones necesarias para la medición de los parámetros meteorológicos conforme a lo mencionado en el capítulo 1, se ha utilizado una pantalla de radiación de ventilación natural que contiene una placa con los sensores de temperatura/humedad relativa y de presión atmosférica. Es así como en la Figura 3.2 se aprecia la placa diseñada e implementada para los sensores. En la parte derecha de la Figura 3.2 se observa la vista lateral de la placa, mientras que en la parte izquierda se observa a la placa en su vista frontal.

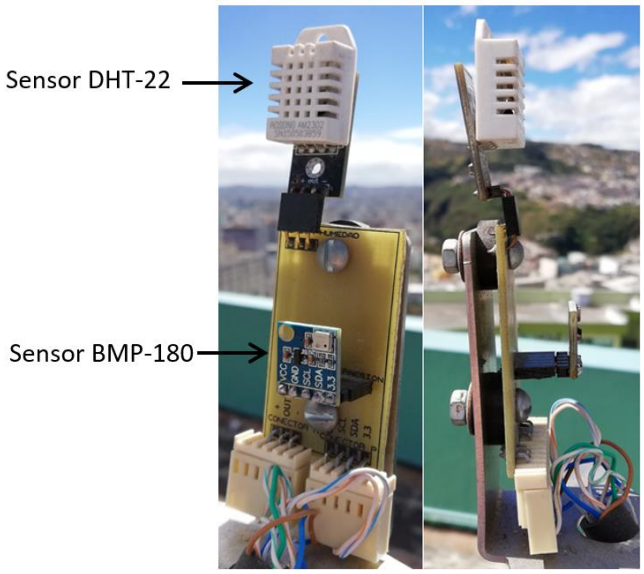


Figura 3.2. Placa de interconexión de los sensores

De igual manera, los demás elementos electrónicos que conforman el subsistema transmisor han sido ordenados dentro de una caja térmica apartándolos del entorno tal como se observa en la Figura 3.3.

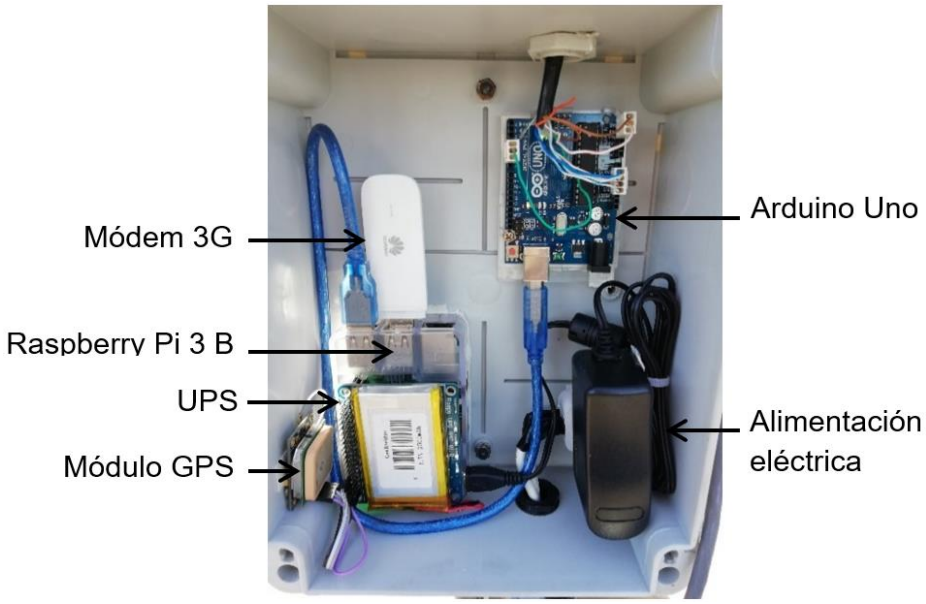


Figura 3.3. Caja térmica con dispositivos electrónicos del subsistema transmisor

Tanto la pantalla de radiación como la caja han sido montados sobre un poste de longitud variable, lo cual permite variar la altura a la que se encuentran los sensores conforme a lo expresado por la WMO para mediciones de temperatura del aire. En la Figura 3.4 se muestra la estructura montada que ha sido ajustada a una altura de 1,75 metros sobre el suelo para las pruebas realizadas.

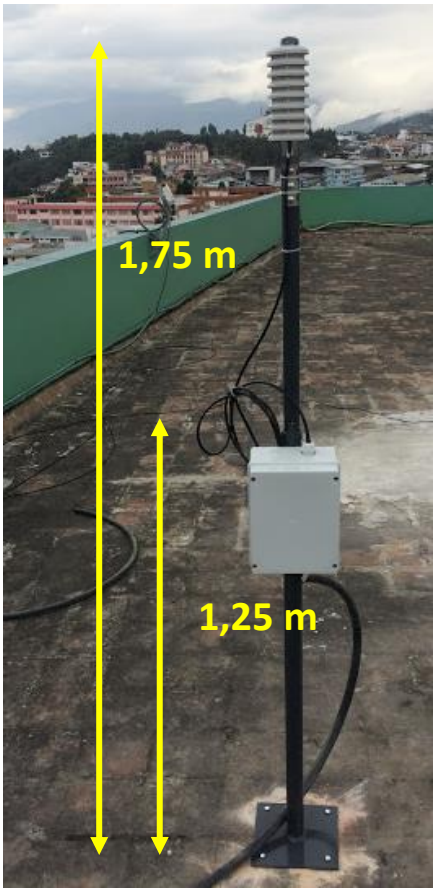


Figura 3.4. Estructura del sistema prototipo

El prototipo de estación meteorológica ha sido instalado el día 22 de diciembre del 2017, y expuesto a las condiciones climáticas de la zona. Todas las pruebas de funcionamiento se las han realizado con el prototipo instalado.

3.2 Pruebas de funcionamiento

- **Pruebas de funcionamiento de los scripts**

Primeramente, se han realizado pruebas de cada uno de los scripts desarrollados para el subsistema transmisor, mismos que han sido ejecutados de manera independiente desde el terminal de comandos de Raspbian.

- **Lectura y escritura de datos meteorológicos en un archivo de texto**

El objetivo de esta prueba es verificar la adquisición de datos de los sensores DHT-22 y BMP-180, y su almacenamiento en un archivo de texto diario. Para esto basta con ejecutar el script asociado a este proceso, y constatar que los valores promediados de los datos adquiridos desde los sensores se encuentran en el archivo de texto diario creado por el mismo script dentro de la carpeta /home/pi/DATA.

En la Figura 3.5 se muestran los tres datos provenientes de los sensores y el dato final que será escrito en el archivo de texto.

Figura 3.5. Datos meteorológicos adquiridos

El archivo de texto es identificado por su fecha de creación, los datos almacenados están asociados a su fecha de adquisición dentro del archivo de texto creado. En la Figura 3.6 se puede apreciar el contenido del archivo generado para la prueba realizada el día 20 de abril de 2018, y su correspondencia con la lectura presentada en la Figura 3.5.

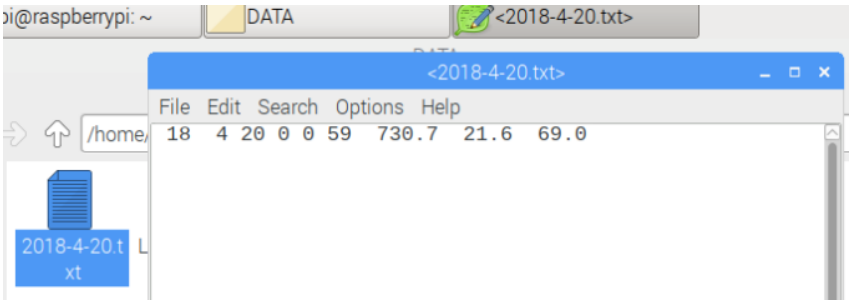


Figura 3.6. Archivo de datos meteorológicos

- **Lectura y escritura de datos GPS en un archivo de texto**

Esta prueba tiene como finalidad constatar el correcto funcionamiento del script que realiza la lectura de los datos GPS, convierte las lecturas al sistema de coordenadas XYZ y los almacena en el archivo de texto “Location.txt” alojado en la carpeta /home/pi/DATA/.

En la Figura 3.7 se muestran los valores de latitud, longitud y altitud adquiridos por el prototipo instalado en el Instituto Geográfico Militar, además de los valores convertidos al sistema de coordenadas XYZ.

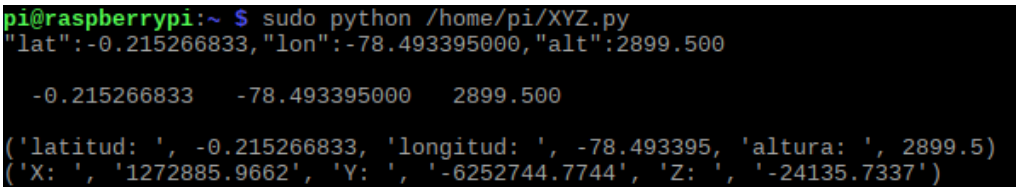


Figura 3.7. Lectura de datos GPS

El archivo “Location.txt” es creado de no existir uno, de lo contrario los datos siguen siendo añadidos dentro del mismo. Los datos almacenados dentro del archivo corresponden a los valores transformados al sistema XYZ, como se observa en la Figura 3.8.

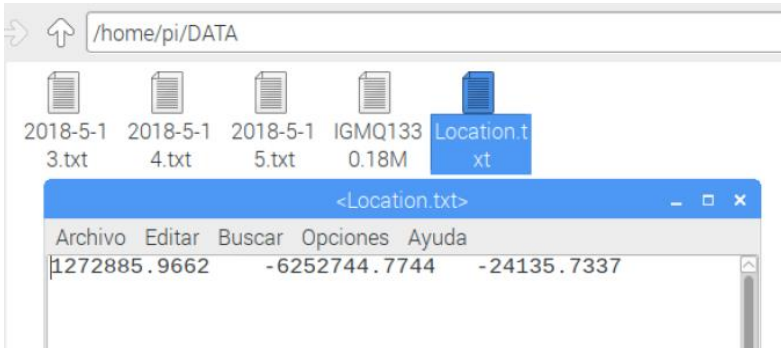


Figura 3.8. Archivo de datos GPS

- **Almacenamiento de datos en la base de datos**

Esta prueba se ha realizado para evidenciar que el script desarrollado permite acceder remotamente a la base de datos y que el prototipo añade la información contenida en el archivo de texto diario.

Para la prueba se ha empleado el mismo archivo de la Figura 3.6, que es leído por el script y almacenado en la base de datos alojada en el servidor presente en la Escuela Politécnica Nacional. En la Figura 3.9 se muestra la entrada presente en la base de datos tras a ejecución del script.

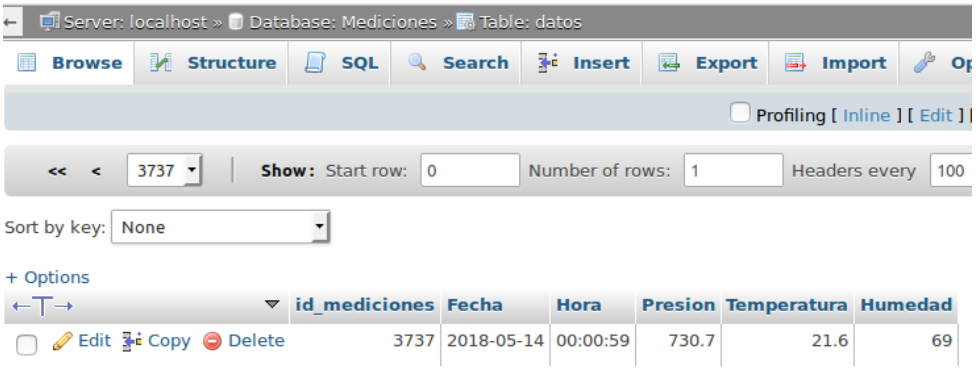


Figura 3.9. Dato almacenado en la base de datos

- **Transmisión de datos en formato RINEX**

La prueba del script de este proceso ha sido llevada a cabo para verificar la creación del archivo RINEX, su envío al servidor FTP y almacenamiento en la carpeta /home/pi/SENT dentro del prototipo una vez haya sido enviado; además de constatar que el proceso de limpieza de la carpeta /home/pi/DATA/ se realice correctamente.

La prueba se realizó el día 15 de mayo del 2018, contando con el archivo diario correspondiente a ese día y de dos días anteriores. El proceso realizado por el script se muestra en la Figura 3.10.

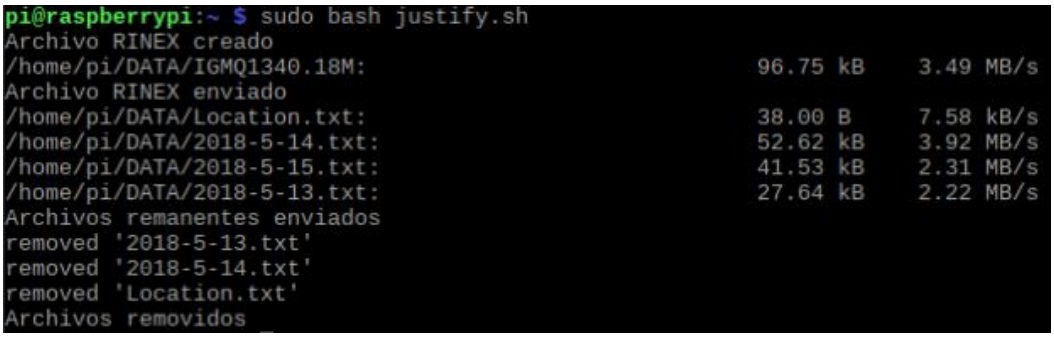


Figura 3.10. Transmisión de archivos al servidor FTP

En la Figura 3.11 se aprecia el archivo RINEX creado a partir de esta prueba correspondiente al día 14 de mayo de 2018, y alojado en la carpeta /home/pi/SENT.

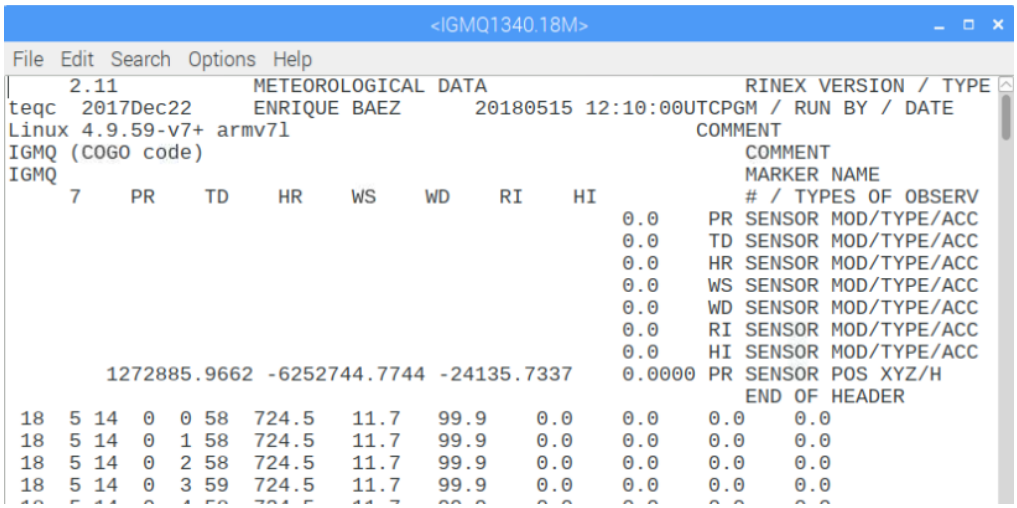


Figura 3.11. Archivo RINEX generado por el prototipo

El proceso de limpieza consiste en el envío de todos los archivos contenidos en la carpeta /home/pi/DATA hacia el servidor FTP, y la eliminación de dichos archivos a excepción del archivo de texto diario que se encuentra editando actualmente el prototipo. En la Figura 3.10 se logra observar que dentro del proceso llevado a cabo por el script no existe la eliminación del archivo “2018-05-15.txt”, correspondiente al archivo de texto diario del día de la prueba.

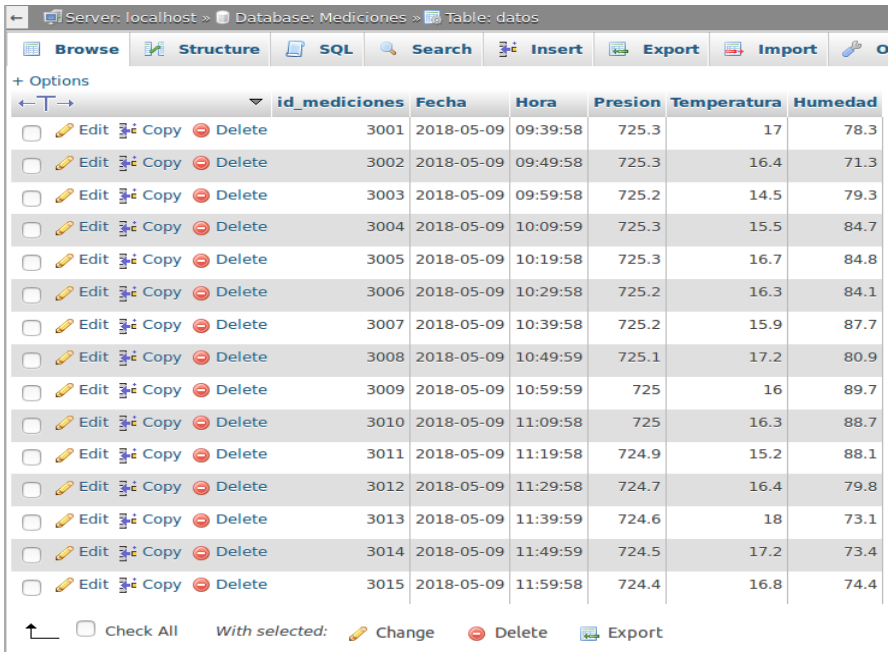
- **Pruebas de funcionamiento automatizado del sistema prototipo**

En esta sección se indican las pruebas del funcionamiento automatizado de todo el sistema prototipo, tanto del subsistema transmisor como del subsistema receptor.

Para ello se comprobará la transmisión de los datos meteorológicos hacia el subsistema receptor a través de la red celular. Para esta exposición se toman valores correspondientes al día 9 de mayo del 2018, tanto en la base de datos instalada en el servidor remoto como en la página web. Además, se verificará la existencia de dichos valores en el archivo diario generado por la estación prototipo y el resultado de consumo de datos celulares para el funcionamiento de todos los servicios.

- **Prueba de visualización de datos en la base de datos**

Ya adquiridos los datos meteorológicos, estos son almacenados en la base de datos MySQL del servidor remoto. En la Figura 3.12, se muestran los valores correspondientes al día 9/5/2018 representados en el gestor de la base de datos PhpMyAdmin, mismos que son actualizados cada 10 minutos.



	id_mediciones	Fecha	Hora	Presion	Temperatura	Humedad
Edit Copy Delete	3001	2018-05-09	09:39:58	725.3	17	78.3
Edit Copy Delete	3002	2018-05-09	09:49:58	725.3	16.4	71.3
Edit Copy Delete	3003	2018-05-09	09:59:58	725.2	14.5	79.3
Edit Copy Delete	3004	2018-05-09	10:09:59	725.3	15.5	84.7
Edit Copy Delete	3005	2018-05-09	10:19:58	725.3	16.7	84.8
Edit Copy Delete	3006	2018-05-09	10:29:58	725.2	16.3	84.1
Edit Copy Delete	3007	2018-05-09	10:39:58	725.2	15.9	87.7
Edit Copy Delete	3008	2018-05-09	10:49:59	725.1	17.2	80.9
Edit Copy Delete	3009	2018-05-09	10:59:59	725	16	89.7
Edit Copy Delete	3010	2018-05-09	11:09:58	725	16.3	88.7
Edit Copy Delete	3011	2018-05-09	11:19:58	724.9	15.2	88.1
Edit Copy Delete	3012	2018-05-09	11:29:58	724.7	16.4	79.8
Edit Copy Delete	3013	2018-05-09	11:39:59	724.6	18	73.1
Edit Copy Delete	3014	2018-05-09	11:49:59	724.5	17.2	73.4
Edit Copy Delete	3015	2018-05-09	11:59:58	724.4	16.8	74.4

Figura 3.12. Datos meteorológicos almacenados en la base de datos

- **Prueba de visualización en la página web**

La prueba de visualización de los datos meteorológicos en la página web se ha realizado accediendo a la pestaña “Datos IGMQ”, en donde los datos son observados tan pronto como son almacenados en la base de datos. Todo esto gracias a la integración del servicio web con el servicio de base de datos alojados en el servidor remoto. En la Figura 3.13, se observan los valores correspondientes al día 9/5/2018, ordenados de forma descendente tal cual se muestran en la base de datos MySQL, comprobándose la integración de los servicios antes mencionados.

NO	FECHA	HORA	PRESION	TEMPERATURA	HUMEDAD
1	2018-05-09	11:59:58	724.4	16.8	74.4
2	2018-05-09	11:49:59	724.5	17.2	73.4
3	2018-05-09	11:39:59	724.6	18	73.1
4	2018-05-09	11:29:58	724.7	16.4	79.8
5	2018-05-09	11:19:58	724.9	15.2	88.1
6	2018-05-09	11:09:58	725	16.3	88.7
7	2018-05-09	10:59:59	725	16	89.7
8	2018-05-09	10:49:59	725.1	17.2	80.9
9	2018-05-09	10:39:58	725.2	15.9	87.7
10	2018-05-09	10:29:58	725.2	16.3	84.1
11	2018-05-09	10:19:58	725.3	16.7	84.8
12	2018-05-09	10:09:59	725.3	15.5	84.7
13	2018-05-09	09:59:58	725.2	14.5	79.3
14	2018-05-09	09:49:58	725.3	16.4	71.3
15	2018-05-09	09:39:58	725.3	17	78.3

Figura 3.13. Datos meteorológicos observables en la página web

- **Prueba de visualización de datos en archivo diario generado**

El archivo diario generado, descargable en el servidor FTP, correspondiente al día 9/5/2018 estará conformado por los datos meteorológicos adquiridos durante ese día. En la Figura 3.14 se observan los valores meteorológicos correspondientes al archivo diario generado por el subsistema transmisor.

```
18  5  9  9 38 58  725.3  16.7  74.6  0.0  0.0  0.0  0.0
18  5  9  9 39 58  725.3  17.0  78.3  0.0  0.0  0.0  0.0
18  5  9  9 40 58  725.3  17.2  78.0  0.0  0.0  0.0  0.0
18  5  9  9 41 59  725.3  17.3  75.9  0.0  0.0  0.0  0.0
18  5  9  9 42 58  725.3  17.1  73.6  0.0  0.0  0.0  0.0
18  5  9  9 43 59  725.3  16.6  75.2  0.0  0.0  0.0  0.0
18  5  9  9 44 58  725.3  16.4  76.8  0.0  0.0  0.0  0.0
18  5  9  9 45 59  725.3  16.6  75.1  0.0  0.0  0.0  0.0
18  5  9  9 46 58  725.3  16.6  74.5  0.0  0.0  0.0  0.0
18  5  9  9 47 58  725.3  16.4  75.5  0.0  0.0  0.0  0.0
18  5  9  9 48 59  725.3  16.4  74.1  0.0  0.0  0.0  0.0
18  5  9  9 49 58  725.3  16.4  71.3  0.0  0.0  0.0  0.0
18  5  9  9 50 59  725.2  16.1  71.3  0.0  0.0  0.0  0.0
18  5  9  9 51 58  725.2  15.9  76.3  0.0  0.0  0.0  0.0
18  5  9  9 52 59  725.3  16.1  79.2  0.0  0.0  0.0  0.0
18  5  9  9 53 58  725.3  16.4  76.0  0.0  0.0  0.0  0.0
18  5  9  9 54 58  725.2  16.1  73.3  0.0  0.0  0.0  0.0
18  5  9  9 55 59  725.2  15.2  73.2  0.0  0.0  0.0  0.0
18  5  9  9 56 58  725.3  14.6  76.2  0.0  0.0  0.0  0.0
18  5  9  9 57 59  725.3  14.5  77.6  0.0  0.0  0.0  0.0
18  5  9  9 58 58  725.3  14.5  79.1  0.0  0.0  0.0  0.0
18  5  9  9 59 58  725.2  14.5  79.3  0.0  0.0  0.0  0.0
18  5  9 10  0 58  725.3  14.3  78.2  0.0  0.0  0.0  0.0
```

Figura 3.14. Archivo diario generado

Como se observa en la Figura 3.14 resaltan algunos valores meteorológicos, esto con la finalidad de comprobar el funcionamiento total del sistema prototipo. Como se nota los valores resaltados en el archivo diario corresponden a los datos visualizados en la base de datos (tres primeros valores de la Figura 3.12) y a los datos presentados en la página web (tres últimos valores de la Figura 3.13).

- **Pruebas de acceso al servidor FTP**

Se han realizado las pruebas del servicio de descarga de archivos generados por el sistema prototipo ingresando a través de la página web desarrollada y mediante un cliente de servicio FTP.

- **Acceso a través de la página web**

La descarga de archivos alojados en el servidor FTP ha sido probada accediendo a la pestaña de “Descarga de Archivos” en la página web desarrollada; que muestra una ventana emergente donde se piden ingresar las credenciales para inicio de sesión de usuario FTP, tal como se aprecia en la Figura 3.15.

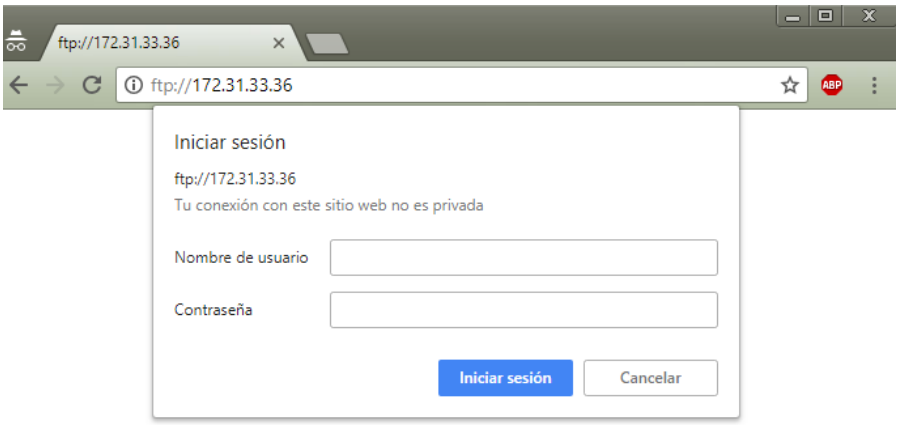


Figura 3.15. Autenticación en el servidor FTP a través de la página web

Al ingresar al servicio FTP se puede constatar la existencia de los archivos RINEX recibidos en el servidor remoto, al igual que la carpeta “DATA” que contiene los archivos de datos sin formato RINEX y remanentes que no pudieron ser enviados al momento de su creación debido a no contar con conexión a la red celular. Los datos se aprecian como en la Figura 3.16.

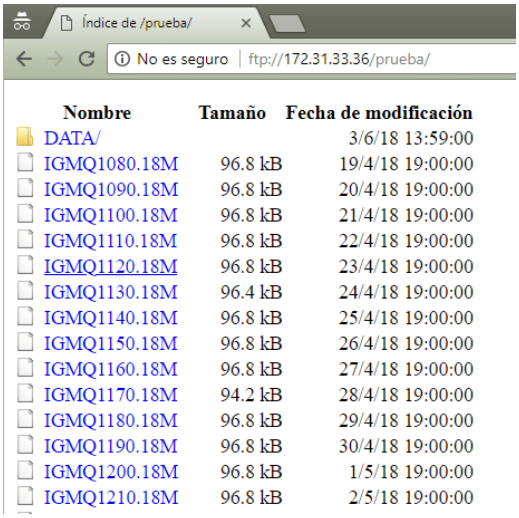


Figura 3.16. Acceso al servidor FTP a través de la página web desarrollada

❖ Acceso a través de un cliente FTP

Es posible acceder al servicio con un cliente FTP, a través de la dirección IP pública asociada al servidor. Las pruebas de conexión al servicio a través de un cliente FTP se han realizado en el programa Filezilla configurando la dirección del servidor, nombre de usuario y contraseña; en este caso no ha sido necesario especificar el puerto ya que se ha utilizado el puerto predeterminado para el servicio FTP. Los archivos disponibles en el servidor FTP se muestran como en la Figura 3.17.

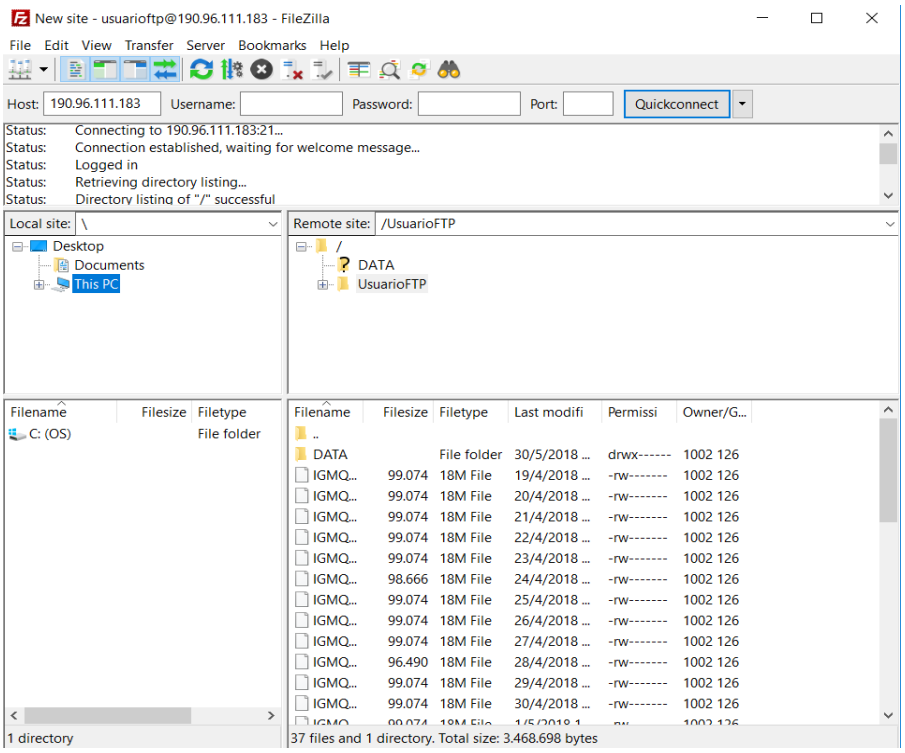
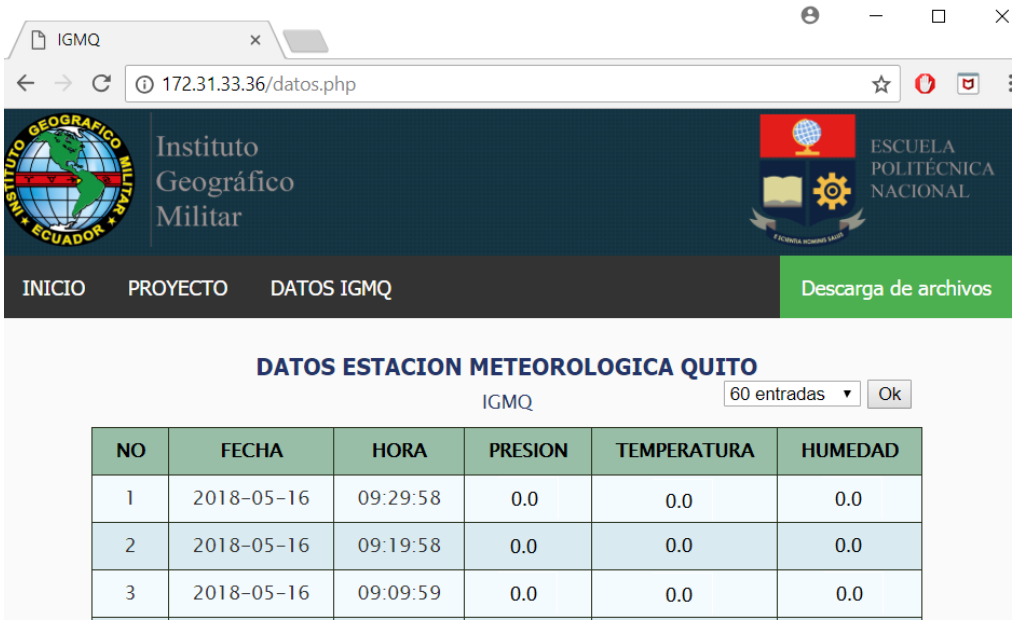


Figura 3.17. Acceso al servidor FTP a través de una IP pública

- **Otros escenarios**

➢ **Desconexión o fallo de los sensores**

Si la estación presenta fallo en los sensores o si los mismos han sido desconectados por alguna situación, sus lecturas serán representados por ceros tal como se muestra en la Figura 3.18 y en la Figura 3.19. De esta manera resulta fácil darse cuenta de este incidente y tomar las respectivas correcciones.



NO	FECHA	HORA	PRESION	TEMPERATURA	HUMEDAD
1	2018-05-16	09:29:58	0.0	0.0	0.0
2	2018-05-16	09:19:58	0.0	0.0	0.0
3	2018-05-16	09:09:59	0.0	0.0	0.0

Figura 3.18. Representación de desconexión en la página web

```
18  5  6  9  8 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9  9 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 10 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 11 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 12 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 13 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 14 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 15 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 16 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 17 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 18 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 19 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 20 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 21 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 22 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 23 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 24 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 25 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 26 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 27 59  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 28 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 29 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
18  5  6  9 30 58  0.0  0.0  0.0  0.0  0.0  0.0  0.0
```

Figura 3.19. Representación de desconexión en el archivo diario

- **Alimentación del sistema con el UPS**

Como se explicó en el capítulo 1 y capítulo 2, la estación prototipo cuenta con un sistema de alimentación "*backup",* con el fin de extender el tiempo de funcionamiento en caso de que el sistema eléctrico se encuentre suspendido por alguna razón. La prueba de desconexión del sistema prototipo de la alimentación eléctrica inició el día 21 de mayo del 2018 a las 11:23 horas y finalizó el mismo día a las 15:28 horas, tras haberse agotado la batería; registrándose como resultado un período de operabilidad con todos los servicios activos cercano a las cuatro horas. Durante este tiempo de "*backup"* la estación no presentó pérdida de datos ni desconexiones con la red celular, comprobándose el funcionamiento del UPS utilizado.

- **Interrupción en la comunicación inalámbrica**

El envío de los archivos diarios y la transmisión de los datos meteorológicos hacia el servidor remoto se realiza a través de la red móvil 3G de Movistar. La prueba se ejecutó cuando el plan de datos celulares activo para el sistema caducó, con la finalidad de observar el comportamiento esperado del prototipo ante esta situación. Durante tres días consecutivos, sin plan de datos celulares, se observó que los archivos diarios eran almacenados en la carpeta /home/pi/DATA de la Raspberry Pi, a la espera de ser enviados.

Al momento de contar con conexión a la red celular, todos los archivos almacenados en la carpeta /home/pi/DATA fueron recibidos y almacenados en la carpeta /DATA del servidor FTP. En ninguno de esos días se presentaron pérdidas de los archivos generados por el sistema prototipo.

- **Resultado de consumo de datos celulares para el sistema prototipo**

El cálculo de consumo de datos del sistema prototipo se realizó considerando el tamaño de los datos enviados a la base de datos, tamaño típico de los archivos: diario generado en formato RINEX, diario generado en formato .txt y archivo en proceso, tal como se observa en la Tabla 3.1.

Tabla 3.1. Cálculo de KBytes generados en un día

Consideraciones	Tamaño (KB)
Archivo diario RINEX	99,07
Archivo diario .txt	53,87
Archivo en proceso .txt	45,30
Datos enviados a la base de datos	16,56
Total	214,8

Para el cálculo del tamaño de dato enviado a la base de datos se utilizó la Ecuación 3.1.

$$C = \frac{115\ Bytes}{dato} * \frac{6\ datos}{1\ hora} * \frac{24\ horas}{1\ día} = 16{,}56 KB/día$$

Ecuación 3.1. Cálculo para la conexión con la base de datos

De acuerdo con la Tabla 3.1, se consumen alrededor de 214,8 KB por día, si esto lo estimamos para un mes, dará un valor alrededor de 6 MB por mes. Cabe mencionar que no se considera el consumo de establecimiento de conexión con la base de datos, conexión con el servidor FTP y actualización del reloj.

Para este proyecto se contrató el servicio de la operadora Tuenti, ya que la misma utiliza la misma infraestructura de la operadora Movistar para su funcionamiento.

Para este proyecto se contrató un plan mensual de 200 MB, por ser el más apropiado existente en el mercado.

El histórico de uso proporcionado por la operadora Tuenti, indica un consumo diario de alrededor de 450 KB y un total mensual de 13,5 MB. La Figura 3.20 muestra el consumo diario proporcionado por la operadora entre los días 17 de abril del 2018 al 16 de mayo del 2018.

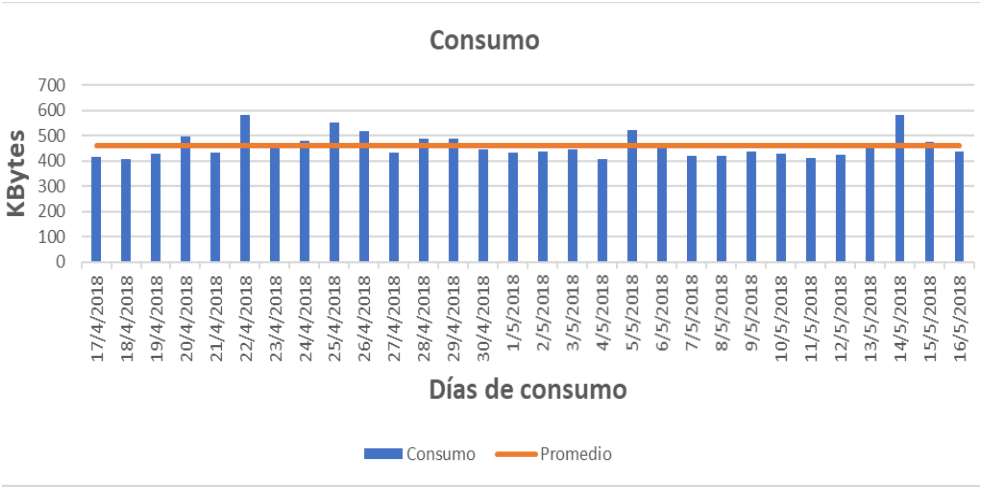


Figura 3.20. Histórico de consumo de datos

3.3 Comparaciones

- **Comparación estación prototipo y estación MET-4A**

Para las pruebas de comparación de datos se ha instalado una estación PAROSCIENTIFIC MET-4A compartiendo espacio con la estación prototipo como se muestra en la Figura 3.21.

Figura 3.21. Instalación de la estación prototipo y estación MET-4A

La comparación entre las estaciones meteorológicas se llevó a cabo con datos meteorológicos obtenidos cada minuto, desde el día 6 de junio del 2018 hasta el día 11 de junio del 2018. A continuación, se presentan los resultados obtenidos de las comparaciones entre las dos estaciones conforme a las Figuras 3.21, 3.22 y 3.23.

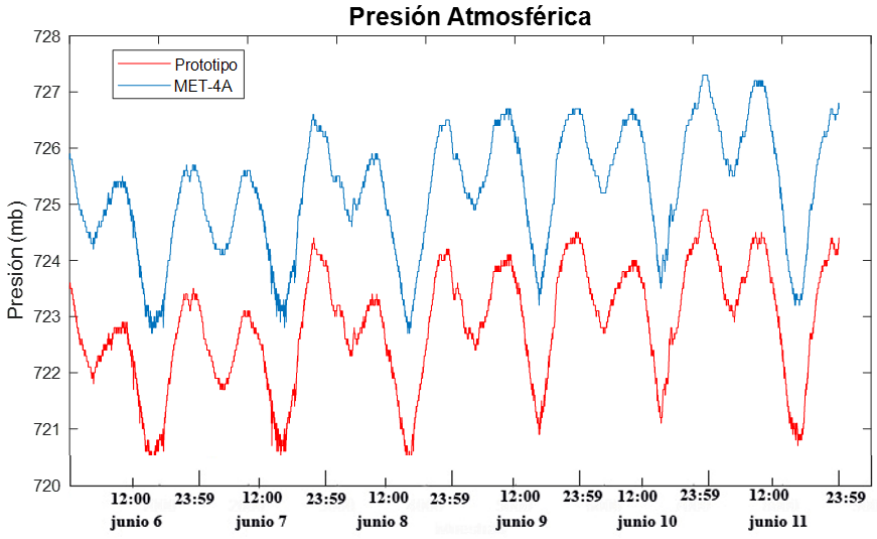


Figura 3.22. Comparación del valor de presión atmosférica Prototipo vs. MET-4A

Como se observa en la Figura 3.22, la tendencia de la gráfica prototipo es idéntica a la gráfica MET-4A, salvo pequeñas variaciones dependientes de la temperatura. Además, se observa que el sensor de presión atmosférica responde de manera inmediata a cambios pequeños de presión atmosférica, reduciendo así el posible error a generarse.

Asimismo, en la Tabla 3.2 se observan los errores obtenidos para la presión atmosférica los cuales resultan ser mínimos que, en términos de diferencia de valores, esta se encuentra en promedio a 2,41 mbar por debajo para la estación científica.

Tabla 3.2. Errores obtenidos para el valor de presión atmosférica

Variable meteorológica	Error absoluto promedio	Error relativo promedio (%)
Presión Atmosférica	2,41	0,33

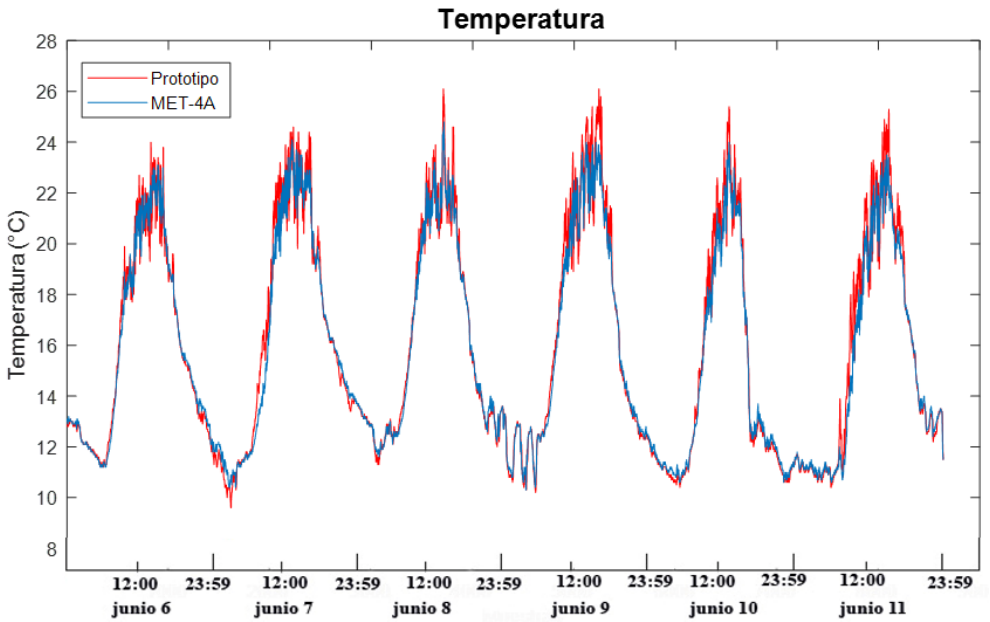


Figura 3.23. Comparación del valor de temperatura Prototipo vs. MET-4A

En la Figura 3.23, la gráfica correspondiente a la temperatura del aire medida por la estación prototipo presenta gran similitud con la tendencia de la gráfica de la temperatura medida por la estación MET-4A, salvo algunos picos detectados durante intervalos de tiempos pequeños; obteniéndose los mayores valores de error para temperaturas superiores a los 22˚C.

De la misma forma, en la Tabla 3.3 se muestran los errores obtenidos para la temperatura, mismos que resultan ser relativamente pequeños que, en promedio son de ± 0,45˚C. Para valores de temperatura medida por la estación MET-4A mayores a los 22˚C, se reportaron errores cercanos a los 2˚C.

Tabla 3.3. Errores obtenidos para el valor de temperatura

Variable meteorológica	Error absoluto promedio	Error relativo promedio (%)
Temperatura	0,45	2,63

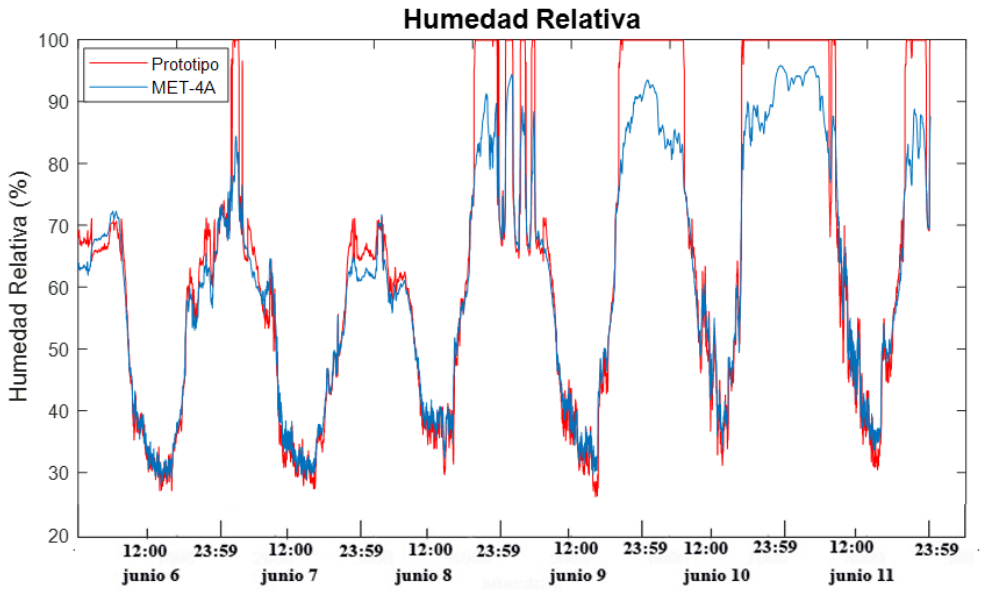


Figura 3.24. Comparación del valor de humedad relativa Prototipo vs. MET-4A

Como se observa en la Figura 3.24, la gráfica humedad relativa prototipo muestra tener una gran similitud con la tendencia de la gráfica MET-4A excepto para valores superiores al 87%, donde al sensor de humedad utilizado se le dificulta detectar cambios y presenta una lectura constante de 99.9%, es decir el sensor entra en saturación. Esta dificultad se refleja en los altos valores de error presentados en la Tabla 3.4.

Tabla 3.4. Errores obtenidos para el valor de humedad relativa

Variable meteorológica	Error absoluto promedio	Error relativo promedio (%)
Humedad relativa	4,97	7,17

- **Comparativa de resultados entre los algoritmos "M5RULES" e "IBK"**

Para las pruebas de predicción, en el software WEKA se han separado los últimos cinco, cuatro, tres, dos y un día de tal manera que se encuentren fuera del grupo de entrenamiento; el propio software realiza la comparación entre los valores predichos y los ingresados como reales, en base a esto presenta los resultados que han sido organizados en las Tablas 3.2, 3.3, 3.4, 3.5 y 3.6. Adicionalmente, se muestran algunas gráficas que representan a las predicciones realizadas por estos algoritmos.

❖ **Predicción para cinco días**

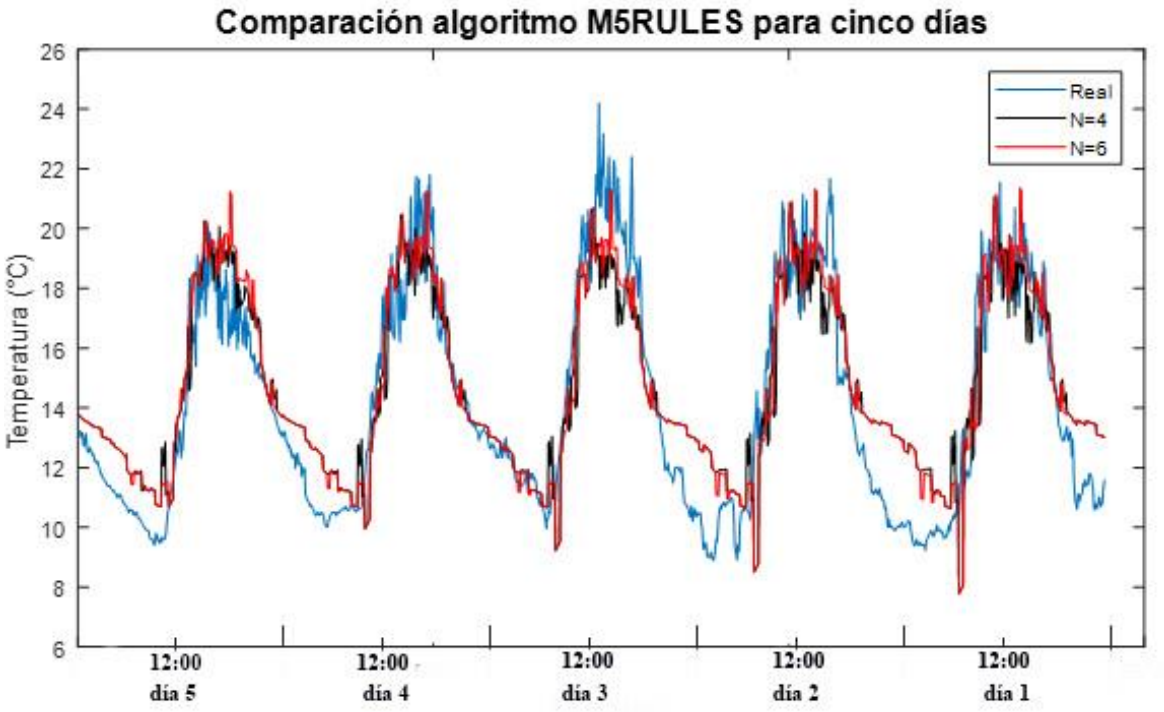


Figura 3.25. Comparación del valor de temperatura predicho para cinco días (algoritmo M5RULES) vs el valor medido

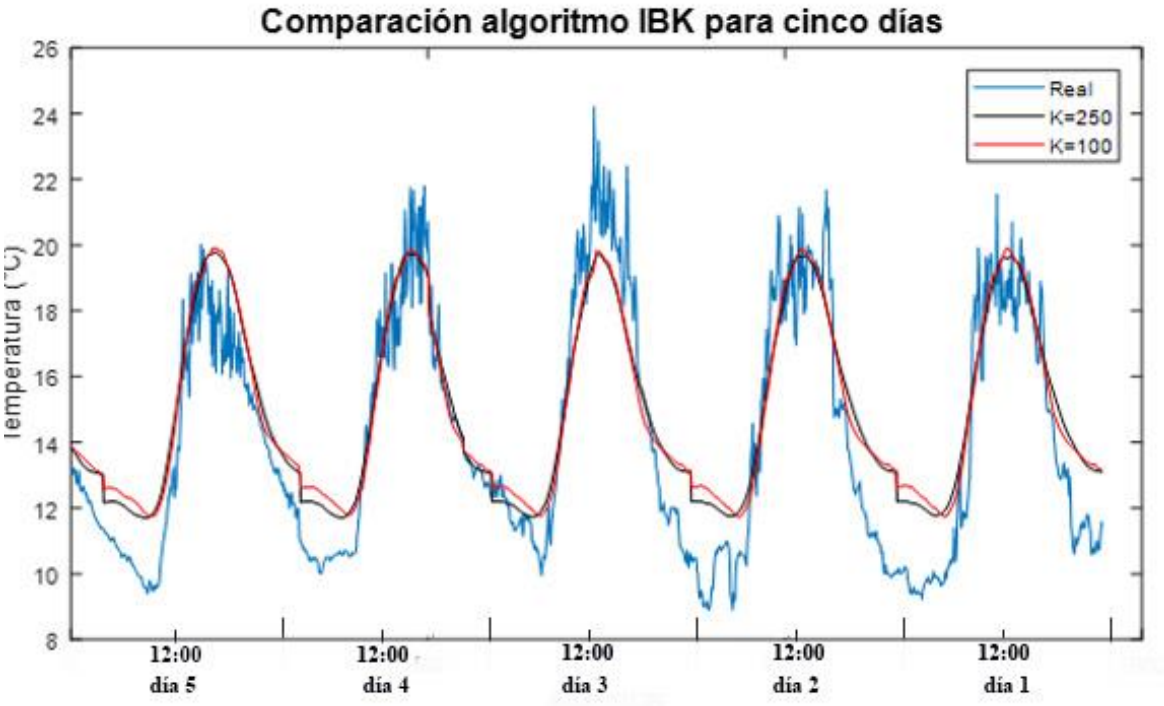


Figura 3.26. Comparación del valor de temperatura predicho para cinco días (algoritmo IBK) vs el valor medido

Tabla 3.5. Resultado de la predicción para cinco días

Classifiers	Rules-M5Rules	Rules-M5Rules	Lazy-IBK	Lazy-IBK
Instances	4	6	-	-
KNN	-	-	250	100
Tiempo tomado en construir el modelo (s)	938,51	553,36	0,05	0,05
Coeficiente de correlación	0,9226	0,9052	0,9433	0,9378
Error absoluto promedio	1,2858	1,3216	1,3412	1,384
Error cuadrático promedio	1,5824	1,6462	1,5891	1,6488
Error absoluto relativo promedio (%)	39,9787	41,1082	41,7195	43,0491
Error cuadrático relativo promedio (%)	43,7674	45,5459	43,966	45,6162

❖ **Predicción para cuatro días**

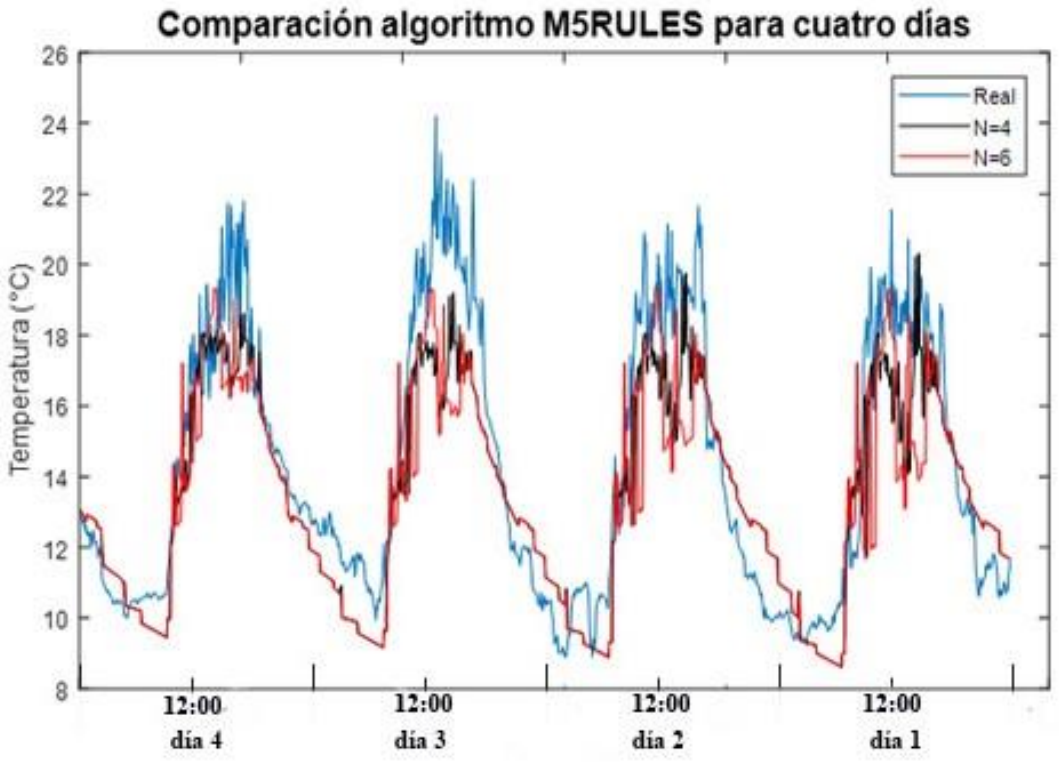


Figura 3.27. Comparación del valor de temperatura predicho para cuatro días (algoritmo M5RULES) vs el valor medido

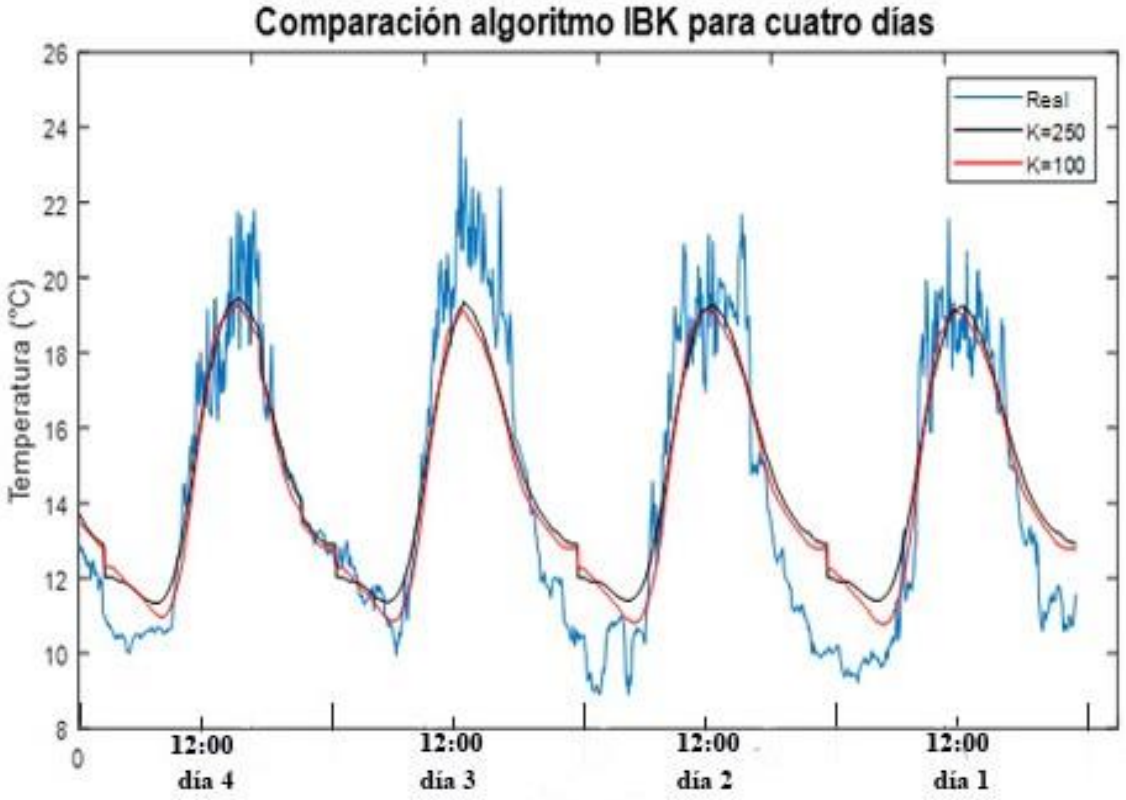


Figura 3.28. Comparación del valor de temperatura predicho para cuatro días (algoritmo IBK) vs el valor medido

Tabla 3.6. Resultado de la predicción para cuatro días

Classifiers	Rules-M5Rules	Rules-M5Rules	Lazy-IBK	Lazy-IBK
Instances	4	6	-	-
KNN	-	-	250	100
Tiempo tomado en construir el modelo (s)	925,75	581,02	0,01	0,01
Coeficiente de correlación	0,9064	0,8605	0,9457	0,9395
Error absoluto promedio	1,4043	1,5837	1,2655	1,2082
Error cuadrático promedio	1,8015	2,1194	1,5361	1,4953
Error absoluto relativo promedio (%)	42,0926	47,4694	37,9338	36,2158
Error cuadrático relativo promedio (%)	48,1696	56,6714	41,0726	39,9837

❖ **Predicción para tres días**

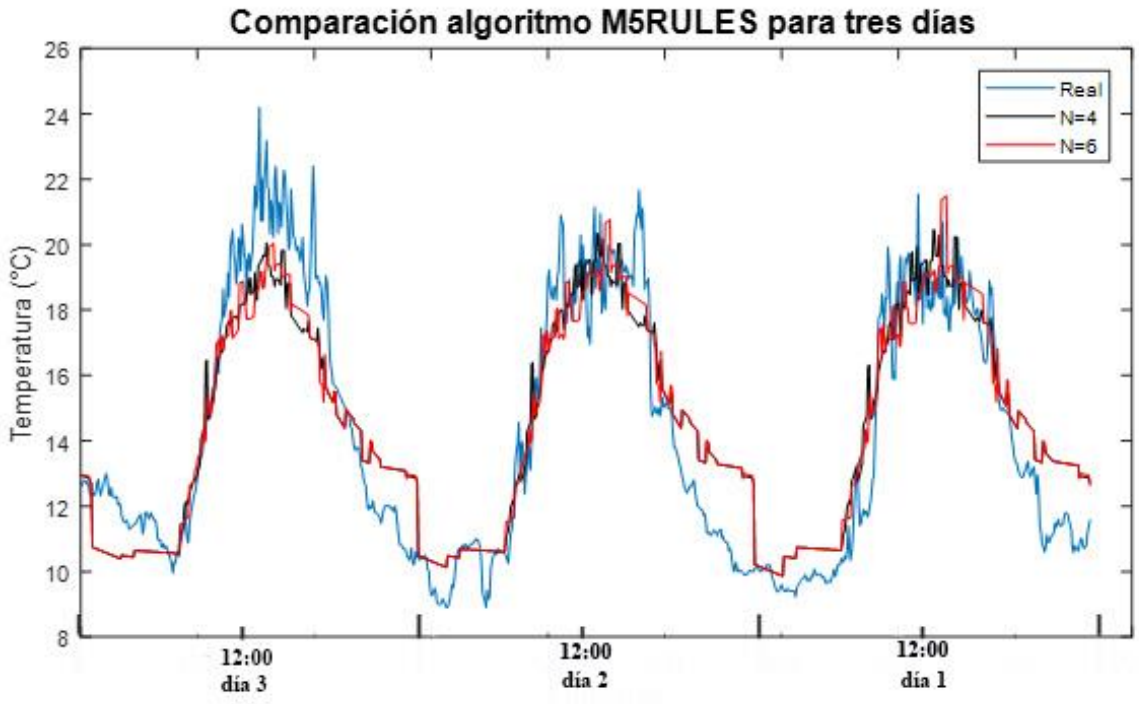


Figura 3.29. Comparación del valor de temperatura predicho para tres días (algoritmo M5RULES) vs el valor medido

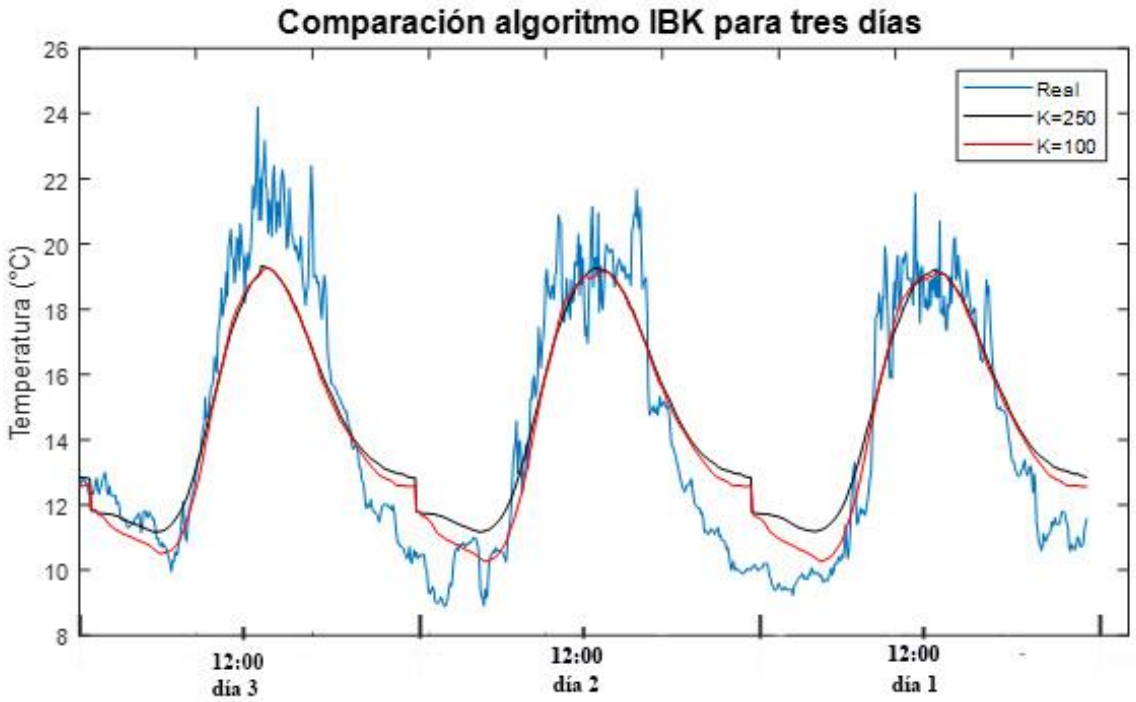


Figura 3.30. Comparación del valor de temperatura predicho para tres días (algoritmo IBK) vs el valor medido

Tabla 3.7. Resultado de la predicción para tres días

Classifiers	Rules-M5Rules	Rules-M5Rules	Lazy-IBK	Lazy-IBK
Instances	4	6	-	-
KNN	-	-	250	100
Tiempo tomado en construir el modelo (s)	924,53	582,79	0,05	0,05
Coeficiente de correlación	0,9255	0,9245	0,9434	0,9422
Error absoluto promedio	1,2424	1,2414	1,3477	1,1884
Error cuadrático promedio	1,5413	1,5459	1,6072	1,455
Error absoluto relativo promedio (%)	35,5365	35,5079	38,5497	33,9928
Error cuadrático relativo promedio (%)	39,8108	39,9296	41,512	37,5818

- **Predicción para dos días**

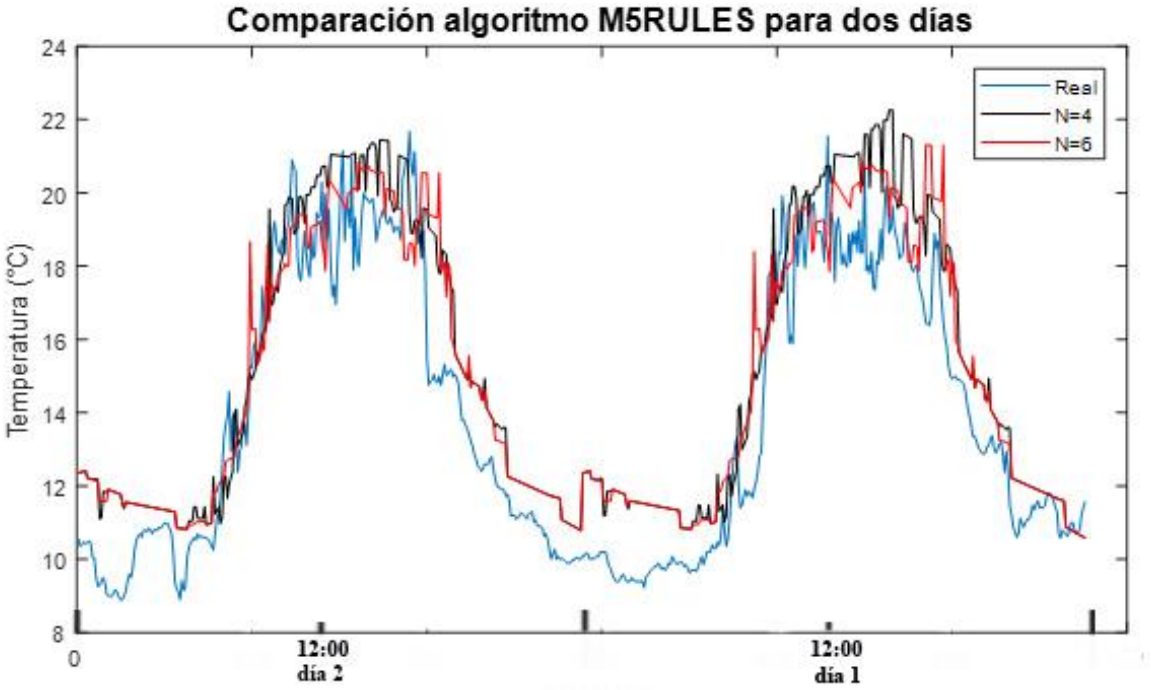


Figura 3.31. Comparación del valor de temperatura predicho para dos días (algoritmo M5RULES) vs el valor medido

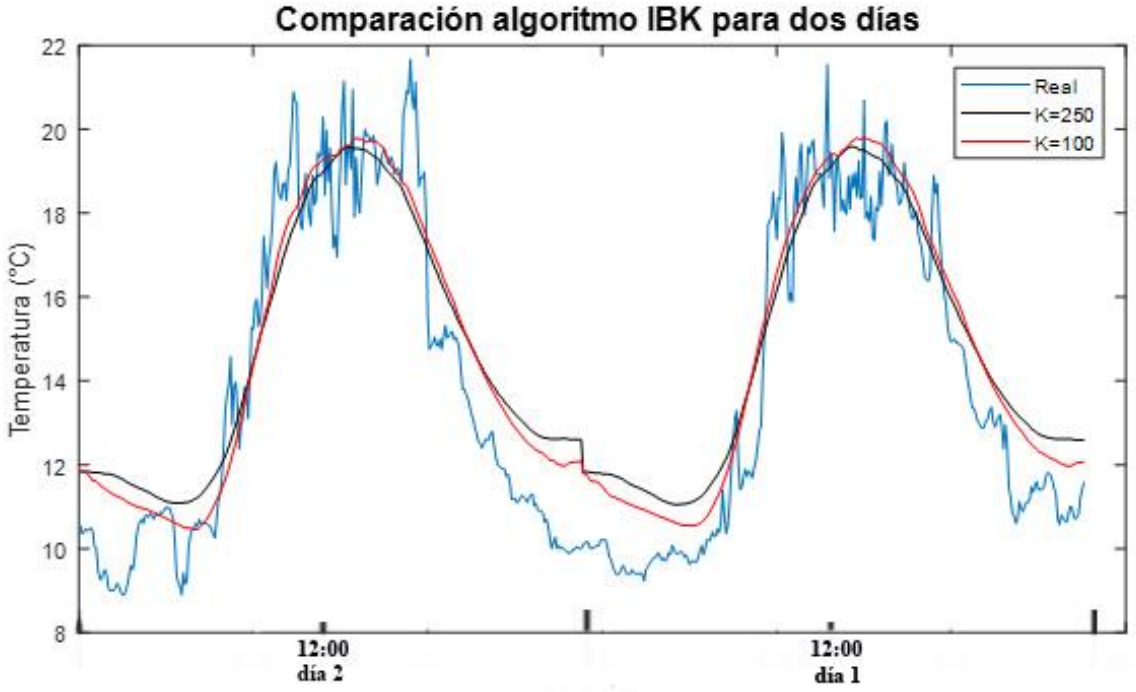


Figura 3.32. Comparación del valor de temperatura predicho para dos días (algoritmo IBK) vs el valor medido

Tabla 3.8. Resultado de la predicción para dos días

Classifiers	Rules-M5Rules	Rules-M5Rules	Lazy-IBK	Lazy-IBK
Instances	4	6	-	-
KNN	-	-	250	100
Tiempo tomado en construir el modelo (s)	865,88	595,01	0,05	0,05
Coeficiente de correlación	0,9416	0,9436	0,957	0,96
Error absoluto promedio	1,4868	1,377	1,3248	1,1336
Error cuadrático promedio	1,7923	1,6782	1,5217	1,3203
Error absoluto relativo promedio (%)	41,9509	38,8464	37,3733	31,9801
Error cuadrático relativo promedio (%)	46,6376	43,6676	39,5963	34,3548

❖ **Predicción para un día**

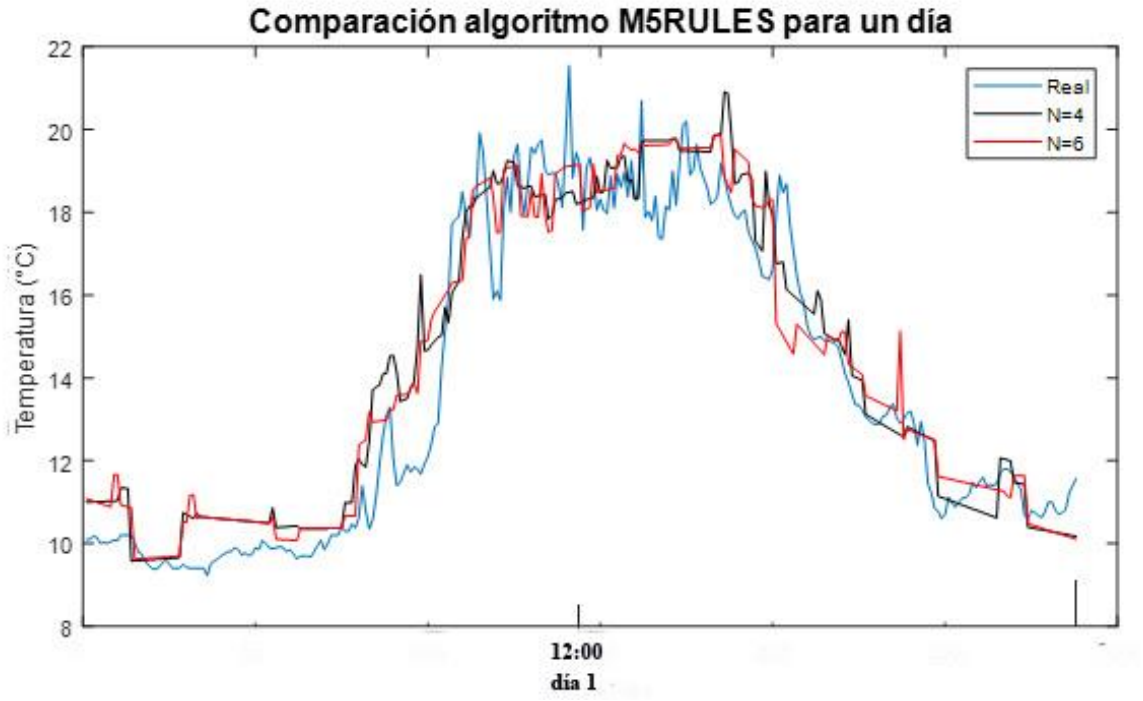


Figura 3.33. Comparación del valor de temperatura predicho para un día (algoritmo M5RULES) vs el valor medido

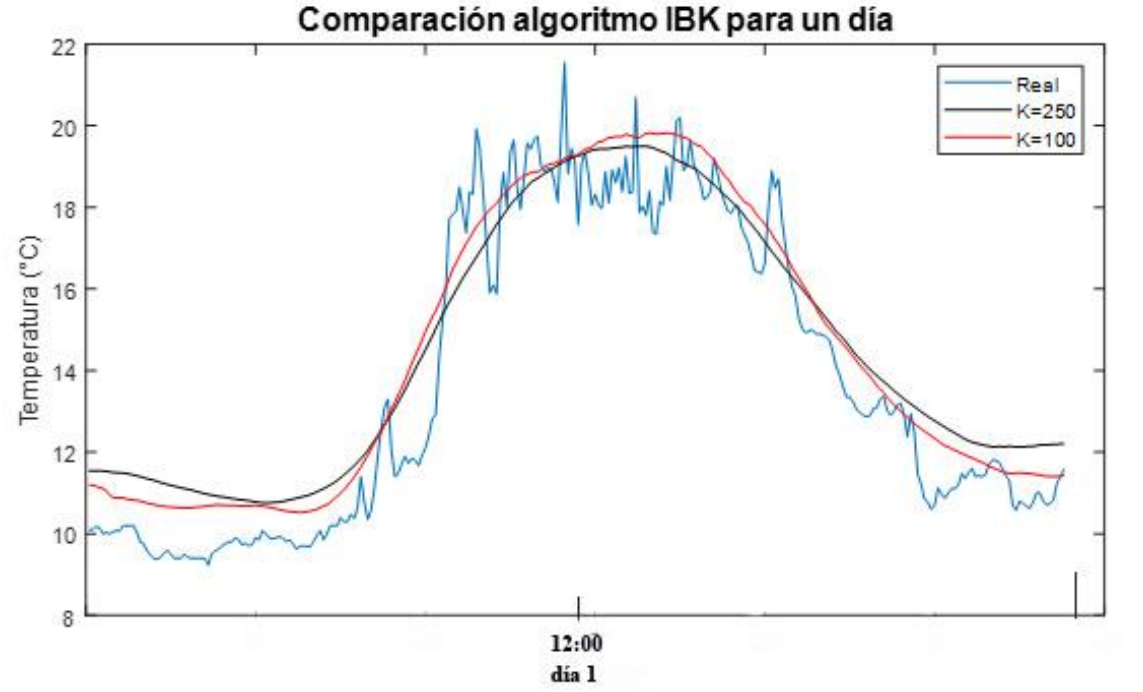


Figura 3.34. Comparación del valor de temperatura predicho para un día (algoritmo IBK) vs el valor medido

Tabla 3.9. Resultado de la predicción para un día

Classifiers	Rules-M5Rules	Rules-M5Rules	Lazy-IBK	Lazy-IBK
Instances	4	6	-	-
KNN	-	-	250	100
Tiempo tomado en construir el modelo (s)	834,06	574,93	0,05	0,05
Coeficiente de correlación	0,9596	0,9573	0,9721	0,9747
Error absoluto promedio	0,8695	0,8739	1,088	0,9232
Error cuadrático promedio	1,1301	1,1463	1,2384	1,0902
Error absoluto relativo promedio (%)	24,7547	24,902	31,0016	26,3064
Error cuadrático relativo promedio (%)	30,06	30,513	32,9623	29,0197

Como se observa en las gráficas, los algoritmos M5Rules detectan con más facilidad los cambios de valores de temperatura, mismos que son reflejados en sus predicciones. Esto debido a que su forma de aprendizaje se basa en dar más peso a los últimos datos ingresados. En tanto que en los algoritmos IBK se observan gráficas más generalizadas, es decir la gráfica obtenida para la predicción no muestra cambios de valores de temperatura y para predicciones futuras se obtiene un patrón de predicción muy similar que a los primeros generados.

Para las tablas con los resultados obtenidos se observa que todos los algoritmos poseen un coeficiente de correlación cercano al valor de uno, mismo que resulta óptimo para las predicciones resultantes.

También se observa que existe una diferencia en las gráficas para los diferentes valores de N del algoritmo M5Rules. Para un N=6 se observa que, al ingresar valores de temperatura de notable diferencia con respecto a los de días anteriores, se van a obtener predicciones que cambian exageradamente entre días. Efecto que se ve reducido con un N=4 donde las predicciones cambian gradualmente según transcurren los días.

Para los tiempos de construcción del modelo, se nota que los algoritmos M5Rules presentan altos tiempos de construcción en comparación a los algoritmos IBK. Además, para el algoritmo M5Rules con N=4 le lleva más tiempo construir el algoritmo que para un algoritmo M5Rules N=6; debido a que con un N=4 se van a crear más reglas en el modelo. En tanto, los algoritmos IBK poseen los tiempos de creación del modelo más pequeños debido a que el procesamiento realizado por el algoritmo se limita a encontrar un valor de entre los valores KNN más cercanos.

Otra de las diferencias son los errores obtenidos; para un algoritmo M5Rules con N=4 el error será generalmente menor que para un algoritmo M5Rules con N=6, esto debido a que se asemejan significativamente a los valores tomados como reales. Por otra parte, los errores para los algoritmos IBK son los más bajos, no obstante, las predicciones resultantes no muestran cambios significativos en los valores de temperatura entre días.

Por lo tanto, en base a lo antes mencionado se adopta al algoritmo M5Rules con N=4, como el algoritmo adecuado para la creación del modelo de predicción para el valor de temperatura.

- **Comparación del retraso troposférico con valores de temperatura medidos y valores de temperatura predichos**

Como se ha mencionado en el primer capítulo, los datos provenientes de las estaciones meteorológicas son empleados para realizar el cálculo del retraso troposférico; para la estimación se han empleado tanto los datos de temperatura medidos por la estación prototipo, como la predicción realizada para cinco días.

Además de todos los datos de temperatura, se han establecido valores constantes de presión atmosférica y humedad relativa, con la finalidad de que los resultados reflejen las diferencias entre los valores de temperatura medidos y los predichos.

Los valores de distancia cenital y coeficiente Ω corresponden a valores típicos de una estación GNSS situada en la ciudad de Quito, para la presión atmosférica se utiliza un valor de 723,5 mbar, valor promedio de todos los datos; en cuanto a la humedad relativa, se establecen valores de 20, 50, 80 y 100%.

A continuación, se calcula el error absoluto y el error relativo para cada una de las muestras, la Tabla 3.10 muestra los valores de error absoluto máximo y promedio, además del error absoluto promedio para cada uno de los casos de humedad relativa analizada.

Tabla 3.10. Resultado del cálculo del retraso troposférico utilizando la predicción para cinco días

Humedad relativa (%)	**20**	**50**	**80**	**100**
Presión Atmosférica (mbar)	**723,5**	**723,5**	**723,5**	**723,5**
Error absoluto promedio	$6{,}5374 \times 10^{4}$	0,0016	0,0026	0,0033
Error absoluto máximo	0,0027	0,0068	0,0109	0,0136
Error absoluto relativo promedio (%)	0,0392	0,0964	0,1518	0,1877

En la Figura 3.35 se comparan los valores de retraso troposférico obtenidos empleando las Ecuaciones 1.3 y 1.4.

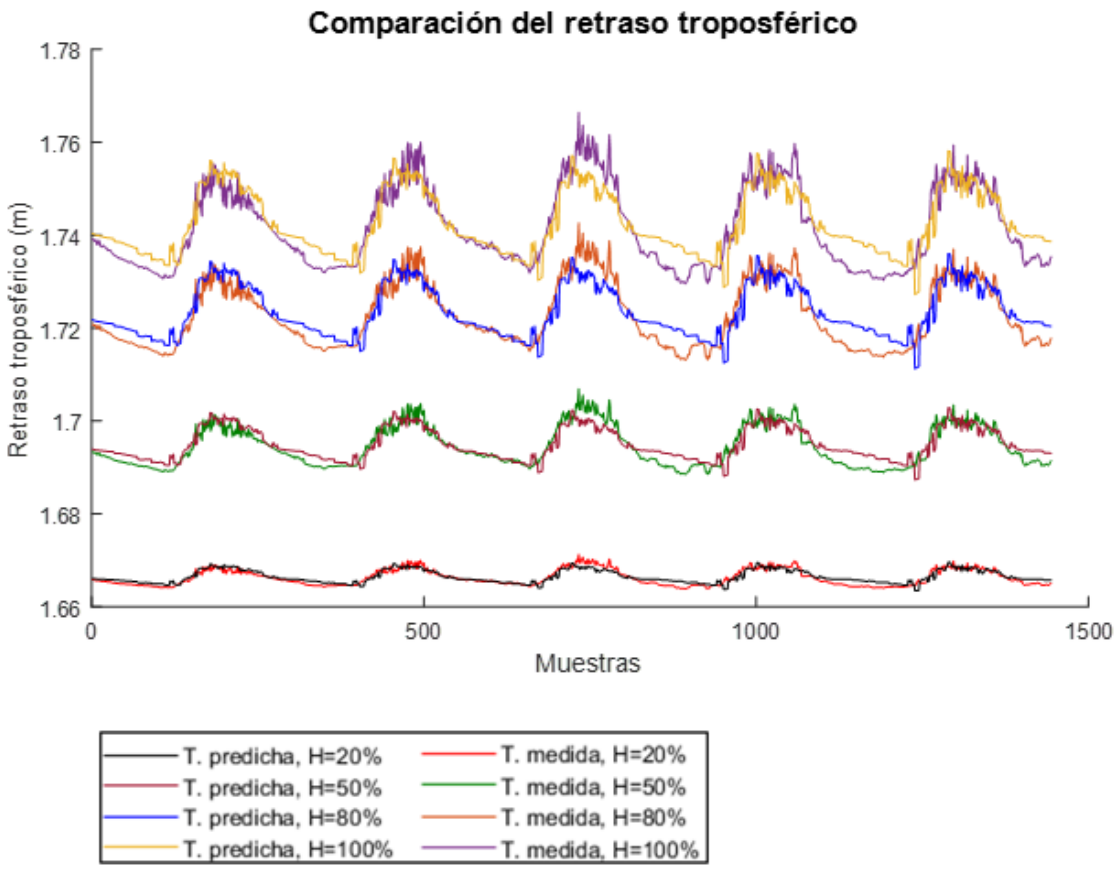


Figura 3.35. Comparación del valor de retraso troposférico

Tanto la Tabla 3.10, como la Figura 3.35 muestran que los errores calculados aumentan en los valores más altos de humedad; aun así, la diferencia de los valores de retraso troposférico obtenidos con la humedad relativa más alta posible (100 %) se mantienen por debajo de los 1,36 cm. En cuanto a los valores menores de humedad relativa, se observa que los valores calculados difieren en un máximo de entre 0,27 cm y 1,09 cm.

Los resultados permiten realizar una buena caracterización del comportamiento del retraso troposférico y establecer un rango en base a los valores de temperatura predichos y los posibles valores de humedad relativa en la zona, debido a los bajos valores de error porcentual obtenidos.

Conociendo el comportamiento de los datos, el rango del retraso troposférico puede ser reducido en su valor más alto, al asociar las menores temperaturas con los valores de humedad relativa más alta. Con lo cual para la muestra se podría establecer un rango de 1,66 a 1,74 metros para el valor de retraso troposférico.

3.4 Costo referencial del sistema prototipo

A continuación, en las Tablas 3.11, 3.12 y 3.13 se presentan los costos referenciales que se utilizaron para la construcción del sistema prototipo. Cabe recalcar que en el presupuesto no se consideraron los costos de ingeniería invertidos para el desarrollo e implementación del sistema prototipo.

Se puede omitir el costo del computador, dado que no es necesario comprar uno nuevo y a su vez es posible utilizar uno disponible que se encuentre en perfectas condiciones.

Cabe destacar que el costo de adquisición para la estación científica MET-4A utilizada actualmente por el IGM está alrededor de los $15,000.

Tabla 3.11. Costo referencial Subsistema Transmisor

Subsistema Transmisor			
Dispositivo	**Cantidad**	**Costo Unitario ($)**	**Costo Total ($)**
Sensor T-H DHT-22	1	10,00	10,00
Sensor BMP-180	1	4,00	4,00
SBC Raspberry Pi	1	60,00	60,00
SBM Arduino UNO	1	17,00	17,00
Módulo GPS u-blox Neo 6M	1	33,00	33,00
Solar Radiation Shield	1	40,00	40,00
UPS para Raspberry Pi3	1	24,59	24,59
Pedestal con graduación	1	80,00	80,00
Caja hermética de plástico	1	10,00	10,00
Sim Card Tuenti 4G	1	5,00	5,00
Modem 3G E173s-6	1	50,00	50,00
Prensas para cable y otros	4	0,35	4,40
Cable UTP Cat.5 exteriores	3	0,85	2,55
Cables conexión sensores	8	0,25	2,00
Cable para extensión eléctrica	5	0,60	3,00
Placa y circuito electrónico	1	20,00	20,00
	Subtotal		365,54

Tabla 3.12. Costo referencial Subsistema Receptor

Subsistema Receptor			
Dispositivo	**Cantidad**	**Costo Unitario ($)**	**Costo Total ($)**
Computador	1	500,00	500,00
	Subtotal		500,00

Tabla 3.13. Costo referencial Sistema Prototipo

Sistema Prototipo	
Subsistemas	**Costo Total ($)**
Subsistema Transmisor	365,54
Subsistema Receptor	500,00
TOTAL	865,54

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- Se diseñó e implementó un prototipo de estación meteorológica inalámbrica capaz de adquirir valores de temperatura, humedad relativa y presión atmosférica, para asociarlos a la predicción del valor de temperatura; cumpliendo con todos los objetivos establecidos para este proyecto de titulación.

- Se ha diseñado y construido un prototipo que logra emular el funcionamiento de una estación meteorológica de la REGME al haber sido expuesto a las condiciones climáticas.

- El sensor de humedad contenido en el DHT22 presentó limitaciones al encontrarse en condiciones de alta humedad, presentando en sus lecturas el valor más alto posible; por lo cual se determina que no es adecuado para sensar en condiciones de humedad relativa mayores al 90%.

- Tras la comparativa realizada para la variable meteorológica de temperatura, se observó que el sensor DHT22 utilizado en el prototipo es más sensible a la detección de cambios, que el sensor contenido en la estación meteorológica científica MET-4A.

- El sensor de presión atmosférica BMP180 resultó ser el más acertado para la construcción de la estación meteorológica ya que presentó asemejarse notablemente a los valores presentados por la estación científica; obteniéndose un error porcentual promedio de 0,33%.

- El consumo por el envío de información a través del módem 3G fue de alrededor de 13,5 MB en un mes de análisis. Por lo que con un bajo costo se ha logrado monitorear los datos adquiridos en el servidor remoto ubicado en la Escuela Politécnica Nacional.

- Conforme al presupuesto realizado en el capítulo 3, se observa que el costo de construcción del sistema prototipo es alrededor de la treintava parte del costo de adquisición de una estación científica MET-4A utilizada por el IGM; lo cual supone un ahorro significativo para dicha institución en caso de que se decida adquirir la estación prototipo.

- Mediante la programación PHP, HTML y CSS se consiguió la integración con servicios de base de datos (MySQL) y transferencia de archivos (FTP),

desarrollando una interfaz web totalmente transparente y amigable con el usuario; misma en la que se puede acceder a los datos MySQL y a la descarga los archivos RINEX.

- De acuerdo con el análisis realizado para la predicción del valor de temperatura, empleando algoritmos de Machine Learning, se determinó que el algoritmo más óptimo fue el algoritmo M5Rules N=4. Ya que este algoritmo generaliza, de todos sus datos, un patrón de predicción y se enfoca en dar más peso a los últimos datos para incluir los cambios en las predicciones de los siguientes días; dando como resultado una predicción acertada para los valores de temperatura deseados.

- En base a los resultados conseguidos de la comparación del retraso troposférico con valores de temperatura medidos y valores de temperatura predichos, se obtuvo un rango de valores para el retraso troposférico de entre 1,66 m y 1,74 m, cuyo margen de tolerancia fue de $\pm$ 1 cm. Resultando ser valores admisibles de retraso troposférico para la situación geográfica en la que se encuentra la estación.

- La inclusión del dispositivo GPS en el diseño de la estación prototipo permitió incluir los datos de posicionamiento en el encabezado de los archivos RINEX, resultando un archivo más completo.

- WEKA, al ser una herramienta computacional dirigida al aprendizaje automático posee la capacidad de trabajar con gran cantidad de datos, gestionándolos acorde a las especificaciones de cada algoritmo para modelarlos. Por lo tanto, el resultado de una buena predicción está en realizar un análisis previo de los datos, entrenar el modelo con un algoritmo que se apegue a las características de los mismos y constatar las predicciones con valores reales para verificar la fiabilidad de los resultados.

4.2 Recomendaciones

- Se recomienda utilizar una pantalla de protección de radiación adecuada para la implementación del sistema prototipo, considerando su forma de ventilación y diseño para evitar el ingreso de agentes dañinos para los sensores.

- Para el uso de Raspberry Pi 3 B se recomienda utilizar el sistema operativo Raspbian Jessie with Pixel, ya que cuenta con una interfaz gráfica diseñada especialmente para los dispositivos de la marca Raspberry.

- Para uso de WEKA al contar con un número extenso de datos, es recomendable crear un grupo que resulte representativo de toda la muestra; esto para reducir el tiempo de procesamiento en la creación de los modelos.
- Es recomendable restringir el acceso de los usuarios FTP a un solo directorio donde se encuentren los archivos generados por la estación.
- Se recomienda emplear dispositivos diseñados para ser usados con Raspberry Pi 3 B con la finalidad de evitar problemas de compatibilidad.
- Es importante dimensionar de manera correcta la base de datos, evitando de esta manera pérdida de datos o colapso del servidor de almacenamiento. Obteniéndose un servidor de base de datos robusto y escalable.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] Campbell Scientific Spain S.L., “Estaciones Meteorológicas,”[En línea]. Available: https://s.campbellsci.com/documents/es/product-brochures/b_weatherstation.pdf. [Último acceso: 4 Noviembre 2017].

[2] R. Chadwick, “RADIATION SHIELDS TO REDUCE ERROR,”10 Octubre 2009. [En línea]. Available: http://wxqa.com/shields.html. [Último acceso: 14 Noviembre 2017].

[3] Campbell Scientific, Inc., “Campbell Scientific,”2017. [En línea]. Available: https://www.campbellsci.com/. [Último acceso: 4 Noviembre 2017].

[4] World Meteorological Organization, “Instruments and Methods of Observation,”2016. [En línea]. Available: http://www.wmo.int/pages/prog/www/IMOP/CIMO-Guide.html. [Último acceso: 16 Noviembre 2017].

[5] World Meteorological Organization, “Recent changes in thermometer screen design and their impact,”1998. [En línea]. Available: https://library.wmo.int/pmb_ged/wmo-td_871.pdf. [Último acceso: 17 Noviembre 2017].

[6] Russell Scientific Instruments Ltd, “1920 Stevenson´s screen,”EKM, 2017. [En línea]. Available: http://www.russell-scientific.co.uk/1290-stevensons-screen---medium-155-p.asp. [Último acceso: 17 Noviembre 2017].

[7] Fondriest Enviromental, Inc, “Young Radiation Shields,”2017. [En línea]. Available: http://www.fondriest.com/rm-young-radiation-shields.htm. [Último acceso: 17 Noviembre 2017].

[8] Comet System, s.r.o., “COMETEO Fan Aspirated Radiation Shield for Weather Sensors,”2015. [En línea]. Available: http://www.cometsystem.com/products/reg-F8200. [Último acceso: 17 Noviembre 2017].

[9] G. Werner, “RINEX The Receiver Independent Exchange Format Version 3.01,”22 Junio 2009. [En línea]. Available: ftp://igs.org/pub/data/format/rinex301.pdf. [Último acceso: 30 Enero 2018].

[10] R. W. G. a. R. T. C. f. M. S. S. C. 1. (.-S. International GNSS Service (IGS), “International GNSS Service,”14 Julio 2015. [En línea]. Available: ftp://igs.org/pub/data/format/rinex303.pdf. [Último acceso: 28 Enero 2018].

[11] F. Medina, “¿Qué son los archivos RINEX?,”1 Enero 2010. [En línea]. Available: https://www.ecomexico.net/proyectos/soporte/Varios/Que%20son%20los%20archivos%20RINEX.pdf. [Último acceso: 30 Enero 2018].

[12] A. Francois, “Précision,incertitude et altération linéaire des données géographiques,”24 Abril 2017. [En línea]. Available:

https://www.sigterritoires.fr/index.php/precisionincertitude-et-alteration-lineaire-des-donnees-geographiques-1/. [Último acceso: 7 Mayo 2018].

[13] P. G. Caviedes, “Geodesia Teoría y Práctica,”*GEO Ciencias & Datos,* vol. VIII, pp. 59-60, 2017.

[14] International Federation of Air Traffic Controller's Associations, “A Begginer's Guide to GNSS in Europe,”EVP Europe, 1999.

[15] W. Suparta y M. Kemal, *Modeling of Tropospheric Delays Using ANFIS*, Bangi: Springer, 2016.

[16] N. G. Villén, “Errores atmosféricos en GNSS (GPS),”11 Abril 2016. [En línea]. Available: https://nagarvil.webs.upv.es/errores-atmosfericos-gnss-gps/. [Último acceso: 28 Noviembre 2017].

[17] L. M. Espinal Zapata, “Estudio de los efectos troposféricos en la precisión de las mediciones GPS en el suroccidente colombiano,”Universidad Del Valle, Santiago de Cali, 2012.

[18] A. A. Custodio, “Sensores Inteligentes: La revolución tecnológica de la instrumentación,”Mayo 1999. [En línea]. Available: http://www.raco.cat/index.php/Buran/article/viewFile/178815/240291. [Último acceso: 14 Noviembre 2017].

[19] Aosong Electronics Co.,Ltd, “Digital-output relative humidity & temperature sensor/module.,”12 Diciembre 2015. [En línea]. Available: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf. [Último acceso: 1 Noviembre 2017].

[20] D-Robotics UK, “DHT11 Humidity & Temperature Sensor Datasheet,”30 Julio 2010. [En línea]. Available: http://www.micropik.com/PDF/dht11.pdf. [Último acceso: 1 Noviembre 2017].

[21] Bosch Sensortec, “BMP085 Digital Pressure Sensor Datasheet,”15 Octubre 2009. [En línea]. Available: https://www.sparkfun.com/datasheets/Components/General/BST-BMP085-DS000-05.pdf. [Último acceso: 1 Noviembre 2017].

[22] Bosch Sensortec, “BMP180 Digital Pressure Sensor Datasheet,”Abril 2013. [En línea]. Available: https://cdn-shop.adafruit.com/datasheets/BST-BMP180-DS000-09.pdf. [Último acceso: 1 Noviembre 2017].

[23] Raspberry pi, “Raspberry Pi 1 model A+,”2017. [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-1-model/. [Último acceso: 17 Noviembre 2017].

[24] S. Compare, “RaspberryPI models comparison,”12 Enero 2018. [En línea]. Available: http://socialcompare.com/es/comparison/raspberrypi-models-comparison. [Último acceso: 24 Enero 2018].

[25] Raspberry Pi , “Raspberry Pi 2 Model B,”2017. [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/. [Último acceso: 24 Enero 2018].

[26] Raspberry Pi, “Raspberry Pi 3 Model B,”2017. [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/. [Último acceso: 17 Noviembre 2017].

[27] Raspberry Pi, “Audio Configuration,”2015. [En línea]. Available: https://www.raspberrypi.org/documentation/configuration/audio-config.md. [Último acceso: 17 Noviembre 2017].

[28] P. Vis, “Raspberry Pi CSI Camera Interface,”[En línea]. Available: https://www.petervis.com/Raspberry_PI/Raspberry_Pi_CSI/Raspberry_Pi_CSI_Camera_Interface.html. [Último acceso: 17 Noviembre 2017].

[29] Raspberry Pi, “Raspberry Pi display,”[En línea]. Available: https://www.raspberrypi.org/documentation/hardware/display/. [Último acceso: 17 Noviembre 2017].

[30] Raspberry Pi, “GPIO,”2016. [En línea]. Available: https://www.raspberrypi.org/documentation/hardware/raspberrypi/gpio/README.md. [Último acceso: 17 Noviembre 2017].

[31] Raspberry Pi, “Raspberry Pi Zero,”2016. [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-zero/. [Último acceso: 17 Noviembre 2017].

[32] Raspberry Pi, “Power Supply,”[En línea]. Available: https://www.raspberrypi.org/documentation/hardware/raspberrypi/power/README.md . [Último acceso: 17 Noviembre 2017].

[33] Raspberry Pi, “The Raspberry Pi UARTS,”[En línea]. Available: https://www.raspberrypi.org/documentation/configuration/uart.md. [Último acceso: 17 Noviembre 2017].

[34] Arduino, “Arduino Introduction,”2017. [En línea]. Available: https://www.arduino.cc/en/Guide/Introduction. [Último acceso: 21 Noviembre 2017].

[35] Arduino.cc, “Arduino Leonardo with headers,”2017. [En línea]. Available: https://store.arduino.cc/usa/arduino-leonardo-with-headers. [Último acceso: 21 Noviembre 2017].

[36] Arduino, “Building an Arduino on a Breadboard,”2017. [En línea]. Available: Building an Arduino on a Breadboard. [Último acceso: 21 Noviembre 2017].

[37] Ingeniería Electrónica, “Tipos de Arduino, detalles y diferencias entre las placas,”2018. [En línea]. Available: https://ingenieriaelectronica.org/tipos-de-arduino-detalles-y-diferencias-entre-las-placas/. [Último acceso: 26 Enero 2018].

[38] Arduino.cl, “Arduino Uno R3,”2017. [En línea]. Available: http://arduino.cl/arduino-uno/. [Último acceso: 21 Noviembre 2017].

[39] Arduino, “Arduini Uno Rev 3,”2017. [En línea]. Available: https://store.arduino.cc/usa/arduino-uno-rev3. [Último acceso: 21 Noviembre 2017].

[40] Arduino.cc, “SPI Library,”2017. [En línea]. Available: https://www.arduino.cc/en/Reference/SPI. [Último acceso: 21 Noviembre 2017].

[41] T. Igoe, “Microcontroller Pin Functions,”27 Agosto 2016. [En línea]. Available: https://itp.nyu.edu/physcomp/lessons/microcontrollers/microcontroller-pin-functions/. [Último acceso: 21 Noviembre 2017].

[42] J. Maugard, “KILL MY BILL,”13 Marzo 2017. [En línea]. Available: https://www.killmybill.es/modem-usb-3g-4g/. [Último acceso: 17 Noviembre 2017].

[43] Telefónica, “Telefónica - Universidad Politécnica de Madrid,”[En línea]. Available: http://www.catedra-telefonicamovistar.etsit.upm.es/sites/default/files/desarrolloEvolucionLTE.pdf. [Último acceso: 23 Noviembre 2017].

[44] OpenSignal, “OpenSignal,”Copyright, 2017. [En línea]. Available: https://opensignal.com/. [Último acceso: 23 Noviembre 2017].

[45] HUAWEI Technologies Co, “E3531 Specs,”Huawei Technologies Co, 2018. [En línea]. Available: http://consumer.huawei.com/en/mobile-broadband/e3531/specs/. [Último acceso: 15 Enero 2018].

[46] BotScience, “Soluciones en Open Hardware,”[En línea]. Available: http://botscience.net/store/index.php?route=product/product&product_id=73. [Último acceso: 26 Enero 2018].

[47] Adafruit, “Adafruit Ulimate GPS Breakout,”[En línea]. Available: https://www.adafruit.com/product/746. [Último acceso: 26 Enero 2018].

[48] I. D. Bernal Espita, “Administacion Informática,”WordPress.com., 2018. [En línea]. Available: https://administacioninformatica.wordpress.com/2012/08/31/definicion-de-ups-y-su-funcion/. [Último acceso: 22 Marzo 2018].

[49] Amazon.com, “Makerfocus-Raspberry-2500mAh-Lithium-Battery,”Amazom.com, 1996-2018. [En línea]. Available: https://www.amazon.com/Makerfocus-Raspberry-2500mAh-Lithium-Battery/dp/B01MQYX4UX. [Último acceso: 22 Marzo 2018].

[50] R. Margaret, “WhatIs.com,”Tech Target, 2018. [En línea]. Available: http://whatis.techtarget.com/definition/machine-learning. [Último acceso: 13 Marzo 2018].

[51] M. Varone, D. Mayer y M. Staff, “EXPERT SYSTEM,”Expert System S.p.A., 18 Febrero 2018. [En línea]. Available: http://www.expertsystem.com/machine-learning-definition/. [Último acceso: 13 Marzo 2018].

[52] Raspberry Pi Foundation, “Introducing Pixel,”[En línea]. Available: https://www.raspberrypi.org/blog/introducing-pixel/. [Último acceso: 24 Enero 2018].

[53] Python Softwae Foundation, “Python Softwae Foundation,”2001-20018. [En línea]. Available: https://www.python.org/. [Último acceso: 13 Marzo 2018].

[54] GNU Operating System, “Linux and the GNU System,”[En línea]. Available: https://www.gnu.org/gnu/linux-and-gnu.es.html. [Último acceso: 30 Enero 2018].

[55] The Apache Software Foundation, “Apache HTTP Server Project,”[En línea]. Available: https://httpd.apache.org/. [Último acceso: 30 Enero 2018].

[56] Oracle Corporation, “MysSQL 5.7 Reference Manual,”2018. [En línea]. Available: https://dev.mysql.com/doc/refman/5.7/en/introduction.html. [Último acceso: 30 Enero 2018].

[57] The PHP Group, “Whay is PHP?,”[En línea]. Available: http://php.net/manual/es/intro-whatis.php. [Último acceso: 30 Enero 2018].

[58] Raspberry Pi Foundation, “Installing Operating System Images,”[En línea]. Available: https://www.raspberrypi.org/documentation/installation/installing-images/README.md. [Último acceso: 24 Enero 2018].

[59] Resion.io, “Etcher,”2018. [En línea]. Available: https://etcher.io/. [Último acceso: 21 Marzo 2018].

[60] A. Choice, “Crontab - Quick Reference,”2018. [En línea]. Available: http://www.adminschoice.com/crontab-quick-reference. [Último acceso: 2 Abril 2018].

[61] Machine Learning Group at the University of Waikato, “Weka 3: Data Mining Software in Java,”The Univerdity of Waikato, 2009. [En línea]. Available: https://www.cs.waikato.ac.nz/ml/weka/. [Último acceso: 1 Julio 2018].

[62] C. Espino Tmón, “Análisis predictivo: técnicas y modelos utilizados y aplicaciones del mismo - herramientas Open Source que permiten su uso,”16 Enero 2017. [En línea]. Available: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/59565/6/caresptimTFG0117mem%C3%B2ria.pdf. [Último acceso: 1 Julio 2018].

[63] S. Vijayarani y M. Muthulakshmi, “Comparative Analysis of Bayes and Lazy Classification Algorithms,”*International Journal of Advenced Research in Computer and Communication Engineering,* vol. II, nº 8, pp. 3118-3124, 2013.

[64] Knowledge Engineering Group, “Separate-and-conquer Regression,”Enero 2010. [En línea]. Available: http://www.ke.tu-darmstadt.de/publications/reports/tud-ke-2010-01.pdf. [Último acceso: 1 Julio 2018].

[65] J. R. Clynch, “Geodetic Coordinate Conversions,”Febrero 2006. [En línea]. Available: http://clynchg3c.com/Technote/geodesy/coordcvt.pdf. [Último acceso: 3 Abril 2018].

[66] D. T. Sandwell, “Reference Earth Model-WGS84,”2002. [En línea]. Available: https://www.eoas.ubc.ca/~mjelline/Planetary%20class/14gravity1_2.pdf. [Último acceso: 3 Abril 2018].

[67] C. Pilapanta, A. Viteri y A. Tierra, “Análisis, evaluación y comparación de los nuevos modelos de propagación atmosférica, con datos obtenidos por parte del sistema de medición meteorológico MET 4 ubicado en la estación GNSS ESPE-ECUADOR,”Instituto Geográfico Militar, Quito, 2013.

6. ANEXOS

ANEXO I. Código de adquisición de datos meteorológicos

ANEXO II. Código de lectura y escritura de datos meteorológicos

ANEXO III. Almacenamiento de datos en una base de datos

ANEXO IV. Transmisión de datos en formato RINEX

ANEXO V. Envío del archivo RINEX al servidor FTP

ANEXO VI. Código de lectura y escritura de datos GPS en un archivo de texto

ANEXO VII. Placa de interconexión de los sensores

ANEXO I

Código de adquisición de datos meteorológicos

```
#include "DHT.h"        // Libreria Sensor TEMPERATURA y HUMEDAD
#include <SFE_BMP180.h> // Libreria Sensor PRESION
#include <Wire.h>       // Libreria COMUNICACION SERIAL
#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321

const int DHTPin = 5;     // Pin digital al que se conecta el sensor DHT
SFE_BMP180 pressure;      // Funciones dentro de la libreria
DHT dht(DHTPin, DHTTYPE); // Funciones dentro de la libreria
void setup()
{
  Serial.begin(9600);     // Inicializa la comunicacion serial
}
void loop()
{     Serial.flush();
  if (Serial.available()) //Si esta disponible la comunicacion serial
  {
    char c = Serial.read(); //Guardamos la lectura del caracter de control en una variable char
    /* Serial.flush();         //Vaciar el buffer */
    if (c == 'H') //Si el caracter recibido es "H"
    {

        pressure.begin();       // Inicializa sensor BMP180
  dht.begin();          // Inicializa sensor DHT
  //if (pressure.begin())
     //Serial.println("BMP180 init success");
    //else
    //{
     //Serial.println("BMP180 init fail\n\n");
     //while(1); // Pause forever.
    //}


      float pres[5];
      float temp[5];
      float hume[5];

      float sumapres;
      float sumatemp;
      float sumahume;

      int i;
      for (i = 0; i < 5; i = i + 1) {
      // Lectura de Temperatura y Humedad  (250 ms)
      double T = 1;
      double P = 0;

      float h = dht.readHumidity();
      float t = dht.readTemperature();

    if (isnan(h) || isnan(t)) // isnan: is not a number
```

```
{
    t = 0;
    h = 0;
  }

   //Lectura de Temperatura y Presion
  char status;

  status = pressure.startTemperature();
  if (status != 0) //se inicializa la temperatura
  {
  delay(status);
  status = pressure.getTemperature(T);
    if (status = 0)//si no se ha tomado el valor de temperatura
    {
      double T = 1;
    }
  }
  status = pressure.startPressure(3);
  if (status != 0) //se inicializa la presión
  {
    delay(status);
    status = pressure.getPressure(P, T);
    if (status = 0)//si no se ha tomado el valor de presión
    {
      double P = 1;
    }
  }

  //Impresión Datos
  pres[i]=P;
  temp[i]=t;
  hume[i]=h;
  sumapres+=pres[i];
  sumatemp+=temp[i];
  sumahume+=hume[i];
  }
  sumapres=sumapres/i;
  sumatemp=sumatemp/i;
  sumahume=sumahume/i;
  if (sumahume <= 80)
  {
    sumahume = sumahume*0.89;
    }
  else if ((sumahume > 80) && (sumahume <= 95))
  {
   sumahume = sumahume * 0.8;
  }
  else
  {
  sumahume = sumahume;
  }

  Serial.print(sumapres);
```

```
      Serial.print ("\t");
      Serial.print(sumatemp);
      Serial.print ("\t");
      Serial.print(sumahume);
      Serial.print("\n");
    }

      else  //Si el caracter de control es diferente de "H"
      {
        Serial.print("ERROR");
        Serial.print("\n");
      }
    }
}
```

ANEXO II

Código de lectura y escritura de datos meteorológicos en un archivo de texto

```
import time
import serial
import datetime
import numpy
import threading
import os
os.chdir("/home/pi/DATA")

#NOMBRE Y ARCHIVO
def archivo_texto():
   now = datetime.datetime.now()
   global name
   name= time.strftime ("%Y-%-m-%-d")
   try:
      text_file = open(name + '.txt', 'r')
   except IOError:
      text_file = open(name + '.txt', 'w')
def datos():
      var=0
      arreglo = numpy.zeros((0,3))
      while True:
         var= var+1
         if var<=3:
               arduino.write('H') #Caracter de control enviado
               print('Caracter de control enviado')
               time.sleep(1)##############################19.98
               mensaje = arduino.readline() #Leer hasta encontrar salto de linea
               a= numpy.fromstring(mensaje, dtype=float, sep="\t") #conversion de string a
matriz numpy
               arreglo = numpy.vstack((arreglo,a)) #creacion matriz de datos
               time.sleep(18.5)

               if (var==3):
                  archivo_texto()
                  text_file = open(name + '.txt', 'a')
                  #now = datetime.datetime.now()
                  text_file.write(time.strftime (" %y  %-m %-d %-H %-M %-S  "))
                  promedio=arreglo.mean(axis=0) #promedio de la matriz de 3 datos
                  numpy.around(promedio,  decimals=1,  out=promedio)  #Redondeo  a  un
decimal
                  dato_final = '  '.join(map(str,promedio)) #conversion numpy a string
                  text_file.write(dato_final)
                  text_file.write("\n")
                  print("DATO")
                  #base_de_datos()

                  #new=threading.Thread(target=base_de_datos)
                  #new.start()
         else:
               var=0
```

```
                arreglo = numpy.zeros((0,3))
                return
                        #print arreglo
                   #arduino.close()

#COMUNICACION SERIAL
arduino = serial.Serial('/dev/ttyACM0', 9600)
arduino.flushInput()
arduino.flushOutput()
datos()
```

ANEXO III

Código de almacenamiento de datos en una base de datos

```
import os
import time
import datetime
import numpy

def base_de_datos():
    import mysql.connector
    import pymysql
    import MySQLdb
    from datetime import date, datetime

    #ABRIR CONEXION A LA BASE DE DATOS
    db = mysql.connector.connect(user='root', passwd='epnigm', host='190.96.111.183',
db='Mediciones')  #Host,usuario,password,baseDeDatos

    cursor = db.cursor()
    date = datetime.now().date()
    #hour = datetime.now().time()

    PRESION = str(datos[0])
    TEMPERATURA = str(datos[2])
    HUMEDAD = str(datos[1])
    add_dato = ("INSERT INTO datos"
                "(Hora, Fecha, Presion, Temperatura, Humedad)"
                "VALUES (%(hora)s, %(fecha)s, %(presion)s, %(temperatura)s, %(humedad)s)")

    dato = {
            'hora' : hour,
            'fecha' : date,
            'presion' : PRESION,
            'temperatura' : TEMPERATURA,
            'humedad' : HUMEDAD,
            }
      print(date)
    print(hour)
    print(PRESION)
    print(TEMPERATURA)
    print(HUMEDAD)

    cursor.execute(add_dato,dato) #agregar dato
    db.commit() #asegura que haya sido escrito el dato
    cursor.close()
    db.close() #termina la conexion

#NOMBRE Y ARCHIVO
def archivo_texto():
    now = datetime.datetime.now()
    global name
    name= time.strftime ("%Y_%-m_%-d")
os.chdir("/home/pi/DATA")
```

```
archivo_texto()
text_file = open(name + '.txt', 'r')
last_line = text_file.readlines()[-1]
print (last_line)
a= numpy.fromstring(last_line, dtype=float, sep="\t") #conversion de string a matriz numpy
#fecha=numpy.take(a, [0,1,2])
hora = numpy.take(a, [3,4,5])
datos= numpy.take(a, [6,7,8])
print (datos)
hour = ':'.join(map(str,hora))
hour = hour.replace(".0", "")
print(hour)
text_file.close()
base_de_datos()
```

ANEXO IV

Código de transmisión de datos en formato RINEX

```
###ARCHIVO ABRIR
cd /home/pi/DATA/
hoy=$(date +%Y-%m-%d | sed "s/^0*//g; s/\-0*/-/g")
actual="${hoy}"".txt"
ayer=$(date --date="${hoy} -1 day" +%Y-%m-%d | sed "s/^0*//g; s/\-0*/-/g")
archivo="${ayer}"".txt"

###ARCHIVO CREAR
day=$(date --date="${hoy} -1 day" +%j)
year=$(date --date="${hoy} -1 day" +%y)
estacion="IGMQ"
nuevo="${estacion}""${day}""0"".${year}""M"

###DATOS ESTACION
version=$(uname -rms)
fecha=$(date +%Y%m%d)
hora=$(date +%H:%M:%S)

###ALINEACION
ki="AA BB CC DD EE FF 0.00000 0.000000 0.000000  "
sed -i "1i${ki}" $archivo
tail -n+1 ${archivo}| rev | column -t|rev | sed -e "s/\  /\ /g"|sed "1d"> ${nuevo}
sed -i -e "s/^/ /" ${nuevo}
sed -i -e "s/$/    0.0    0.0    0.0    0.0/" ${nuevo}

###HEADER
sed -i "1i\ \ \ \ \ 2.11           METEOROLOGICAL DATA                 RINEX VERSION / TYPE\nteqc  2017Dec22     ENRIQUE BAEZ        ${fecha} ${hora}UTCPGM / RUN BY / DATE\n${version}                                        COMMENT\n${estacion} (COGO code)                                COMMENT\n${estacion}                                            MARKER NAME\n     7    PR  TD  HR  WS  WD  RI  HI        # / TYPES OF OBSERV\n                                                    0.0  PR SENSOR MOD/TYPE/ACC\n                                                    0.0    TD SENSOR MOD/TYPE/ACC\n                                                    0.0    HR SENSOR MOD/TYPE/ACC\n                                                    0.0    WS SENSOR MOD/TYPE/ACC\n                                                    0.0    WD SENSOR MOD/TYPE/ACC\n                                                    0.0    RI SENSOR MOD/TYPE/ACC\n                                                    0.0    HI SENSOR MOD/TYPE/ACC\n       0.0000        0.0000        0.0000        0.0000 PR SENSOR POS XYZ/H\n                                                    END OF HEADER" ${nuevo}
sed -i "/AA BB/d" ${archivo}
until ncftpput -u "administrador" -p "EpnIgm_18%&" 190.96.111.183 /home/ftp/administrador/prueba/ /home/pi/DATA/${nuevo}
do
sleep 600
done
mv /home/pi/DATA/${nuevo} /home/pi/SENT/
ncftpput -R -u "administrador" -p "EpnIgm_18%&" 190.96.111.183 /home/ftp/administrador/prueba/ /home/pi/DATA
cd /home/pi/DATA/
shopt -s extglob
rm -v !("$actual")
exit 1
```

ANEXO V

Código de configuración inicial de módulo GPS

```
sudo systemctl stop gpsd.socket
sudo systemctl disable gpsd.socket
sudo gpsd /dev/ttyS0 -F /var/run/gpsd.sock
```

ANEXO VI

Código de lectura y escritura de datos GPS en un archivo de texto

```
import subprocess
from subprocess import check_output
import math
import numpy
import os
###LECTURA GPS
string= subprocess.check_output("gpspipe -w -n 5| grep -m 1 TPV| cut -d '"',"' -f 6-8", shell=True)
for ch in [':',',','"lat"','"lon"','"alt"']:
    if ch in string:
        string=string.replace(ch, " ")
lla= numpy.fromstring(string, dtype=float, sep="\t")
latitud = float(numpy.take(lla, [0]))
longitud = float(numpy.take(lla, [1]))
altura = float(numpy.take(lla, [2]))
longitud= float(subprocess.check_output("gpspipe -w -n 5| grep -m 1 GST| cut -d '":"' -f 9 | cut -b 1-11", shell=True))
altura= float(subprocess.check_output("gpspipe -w -n 5| grep -m 1 GST| cut -d '":"' -f 10 | cut -b 1-11", shell=True))
###CONVERSION
a= float(6378137)
b= float(6356752.3)
if (latitud <=90):
    latitud=latitud*(2*math.pi)/360
    longitud=longitud*(2*math.pi)/360
    R=((a)**2)/(math.sqrt((((a)**2)*(math.cos(latitud))**2)+(((b)**2)*(math.sin(latitud))**2)))
    x=(R+altura)*math.cos(latitud)*math.cos(longitud)
    y=(R+altura)*math.cos(latitud)*math.sin(longitud)
    z=(((((a)**2)/((b)**2))*R)+altura)*math.sin(latitud)
    x=str(format(x,'.4f'))
    y=str(format(y,'.4f'))
    z=str(format(z,'.4f'))
###ESCRITURA
    os.chdir("/home/pi/DATA")
    try:
        text_file = open('Location.txt', 'r')
    except IOError:
        text_file = open('Location.txt', 'w')
    text_file = open('Location.txt', 'a')
    text_file.write(x)
    text_file.write("\t")
```

```
    text_file.write(y)
    text_file.write("\t")
    text_file.write(z)
    text_file.write("\n")
    exit
else:
    exit
```

ANEXO VII

Placa de interconexión de los sensores

La placa diseñada para la interconexión de los sensores con la plataforma Arduino UNO se observa en la figura, misma que fue desarrollada en el software Proteus 8.1 con las herramientas ISIS y ARES.

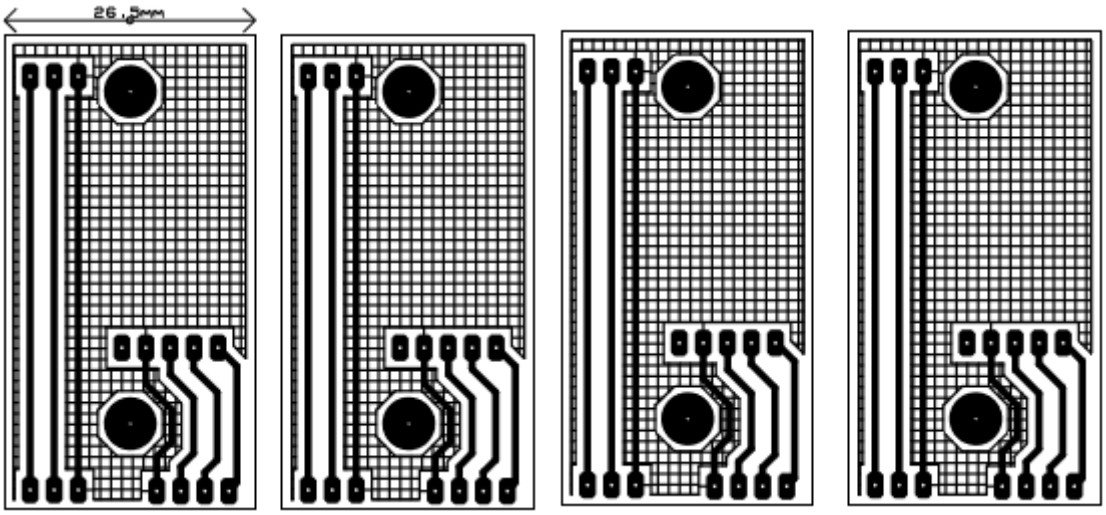


Figura I. 1 Placa de interconexión de sensores usando Proteus

ORDEN DE EMPASTADO