UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

DISEÑO Y DESARROLLO DE UNA RED DE SENSORES DE CONTAMINACIÓN ATMOSFÉRICA PARA SU PROCESAMIENTO EN UN SERVIDOR

T E S I S

QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

P R E S E N T A:
GARCÍA ORTEGA ARTURO SINUHÉ

Director de Tesis:
Dr. Demetrio Fabián García Nocetti
POSGRADO EN CIENCIA E INGENIERIA DE LA COMPUTACIÓN

Ciudad Universitaria, CD. MX. enero, 2018

Agradecimientos

En primer lugar, quiero agradecer a mi madre, quien me ha apoyado incondicionalmente en toda la vida, tanto económica como emocional y académicamente, igualmente a Roberto por su apoyo de experiencia.

A mi hermano Axel por su apoyo incondicional, a su esposa Nadia y a mi sobrina Grettel, que me sirve como inspiración para heredarle a su generación un país mejor.

A Dania, quien es parte importante de mi vida y me ha apoyado en los pasos que he dado en mi vida desde que me conoce y a quien apoyaré en los pasos que dé en su vida. Me ha brindado mucho cariño y eso lo agradezco mucho. También agradezco a sus tíos Marco y Julio por su ayuda en general.

A mis abuelos, en especial a mi finada abuela materna María, por haberme ayudado a crecer de manera integral y a ella le debo la persona que soy hoy en día.

A mis tíos, primos y demás familiares, con quienes he pasado buenos momentos de mi vida y me han inspirado a seguir adelante.

A mi padre que, aunque no siempre ha estado con nosotros, me ha querido ayudar en algunas circunstancias.

A mi asesor, el Dr. Fabián García por su ayuda asesorando este proyecto de maestría; a todos mis profesores de la maestría por compartirme sus conocimientos.

A mis amigos de la maestría por compartirme sus conocimientos y experiencias, tanto académicas como laborales y personales, y por haber pasado momentos amenos con su compañía.

A Conacyt, por haberme brindado el apoyo económico que me ayudó mucho a realizar este trabajo, y que sigan apoyando al talento mexicano que lo requiera.

Al posgrado de Ciencia e Ingeniería en Computación, así como al IIMAS y su personal, el cual fue muy amable conmigo y me ayudó en mis trámites académicos; les deseo lo mejor a todos.

A mis amigos fuera del ambiente de la maestría, quienes desde hace años me han regalado momentos agradables y lo siguen haciendo, por esta razón los estimo mucho.

A mis mascotas, que me sirvieron como desestresante al momento de escribir esta tesis.

Quiero dar mis agradecimientos a las personas que no pude agregar por falta de espacio, que me apoyaron en muchos momentos de mi vida, ya que a ellos les debo mucho de lo que he llegado a ser y a hacer.

Índice

Capítulo 1: Introducción

1.1 Introducción

En estos tiempos, la situación actual de contaminación en las urbes alrededor del planeta ha ido incrementando conforme aumenta la población en las ciudades. Existen diversos tipos de contaminación en las urbes, las cuales pueden ser auditivas, del suelo, del agua y atmosféricas, siendo ésta última el tipo de contaminación de interés para este trabajo.
Se requiere actualmente de diversas soluciones para combatir esta gran problemática de contaminación atmosférica, entre ellas, soluciones tecnológicas que permitan a las personas tomar decisiones con resultados positivos para su salud. Existen diversas formas de obtener información del medio urbano, las cuales pueden ser con ayuda de sensores fijos, sensores móviles, sensores ubicuos (invisibles al usuario) o con intervención del usuario, por medio de aplicaciones conectadas a internet.

Existe hoy en día un concepto llamado el Internet de las Cosas o Internet of Things (IoT) para trabajar con estos sensores. El IoT es un nuevo concepto del siglo XXI, el cual maneja que los objetos del mundo real puedan ser interpretados en el mundo digital[1]. El IoT está creciendo de una manera rápida y es un campo fértil para desarrollar proyectos como éste, ya que promete mucha investigación, implementación y negocio.

El proyecto de tesis manejado en este documento va a basarse en un diseño nuevo de redes Ad-hoc, utilizando un servidor Apache para subir la información recabada y agregando de antemano una aportación propia para la mejora de este tipo de implementaciones de análisis de contaminación. Este trabajo se va a implementar en un ambiente experimental, dejando para un futuro proyecto el poderse implementar en situaciones reales de contaminación urbana.

1.2 Objetivo

Este trabajo va a desarrollar un nuevo protocolo de enrutamiento de redes Ad-hoc, utilizando el protocolo inalámbrico 802.11g (Wi-Fi) que contribuya a almacenar y propagar de manera eficiente, información relacionada con niveles de contaminación del aire; siendo el diseño e implementación física del protocolo de enrutamiento el objetivo principal del proyecto. La información será transmitida de un nodo a otro de la red Ad-hoc y finalmente se enviará hacia un servidor Apache para realizar su almacenamiento y despliegue.

Se desarrollará un nuevo modelo de propagación de redes Ad-hoc, para así comparar sus ventajas frente a los modelos ya existentes.
Este modelo e implementación se encargará de capturar, almacenar y enviar información correspondiente a niveles de contaminación del aire, de acuerdo a las capacidades de hardware y software que poseen los módulos de comunicación a utilizarse. El protocolo a diseñar e implementar se nombrará como Protocolo basado en Saltos y Fuerza de Señal (PSFS).

1.3 Metas

Las metas que se deben completar para poder alcanzar el objetivo general son:

- Investigación de los antecedentes de implementaciones de redes Ad-hoc con sensores y el Internet de las Cosas.
- Investigación de las principales características de las redes Ad-hoc.
- Diseño del protocolo de la red Ad-hoc de sensores de contaminación.
- Desarrollo del protocolo de la red Ad-hoc en dispositivos físicos Arduino.
- Prueba de campo del protocolo ya desarrollado.
- Comparación de resultados del modelo desarrollado con diversos parámetros.
- Comparación de características del modelo desarrollado frente a otros modelos de propagación de redes Ad-hoc.

Capítulo 2: Antecedentes

2.1 Antecedentes en el mundo

2.1.1 Desarrollo de un sensor de mano de dióxido de nitrógeno para smartphones

Investigadores del Centro para electrónica avanzada y sensores, ubicado en la universidad de RMIT, desarrollaron un sensor de bajo costo que puede ser incorporado a teléfonos inteligentes, el cual puede medir las concentraciones de dióxido de nitrógeno en el aire[2].
El sensor opera absorbiendo las moléculas de dióxido de nitrógeno en escamas de bisulfuro de estaño, y los creadores comentan que éste método es un comienzo para crear sensores de mano de NO_2 de bajo costo y personalizable, pudiendo ser incorporado incluso a smartphones.

2.1.2 Sensores personales para ver en el móvil la contaminación del aire

Unos investigadores de la Universidad de California en San Diego diseñaron unos pequeños sensores móviles que miden ozono, óxido nitroso y monóxido de carbono, y envían en tiempo real los datos al teléfono móvil. El proyecto se llama CitiSense y es un sistema de vigilancia de la calidad del aire, capaz de suministrar los datos en tiempo real a teléfonos móviles y ordenadores personales[3].
La información resultante se podrá suministrar al público individualmente o a los organismos públicos, y los científicos involucrados señalan que en el futuro los sensores estarán incluidos directamente en el teléfono.

2.1.3 Sensores personales: Atmotube

Atmotube es un dispositivo de bolsillo con sensores incrustados en su caparazón de titanio. Puede detectar 127 compuestos orgánicos volátiles, así como gases venenosos, tales como el monóxido de carbono[4].
El dispositivo toma lecturas cada 10 segundos. Los resultados alimentan los teléfonos inteligentes de los usuarios, los cuales son visualizados en un mapa que muestra los

niveles de contaminación a través de un área. Atmotube detecta incluso pequeños cambios y el tiempo de respuesta es de menos de un segundo.

2.1.4 World Air Quality Index: sensores, Internet y apps

World Air Quality Index es un proyecto que recopila la información suministrada por las estaciones medidoras, las cuales se encuentran instaladas en los cientos de ciudades que se están monitorizando prácticamente en tiempo real, y continuamente se añaden más núcleos urbanos a este proyecto[5].
Gran cantidad de las apps disponibles en iOS, Android o Windows Phone dedicadas a ofrecer información sobre la calidad del aire localmente o a escala mundial, se alimentan de los datos que recopila el portal Air Quality Index.

2.1.5 Aclima: Una plataforma de sensores móviles

La plataforma de sensores móviles de Aclima en los autos de Google Street View complementan la red de medida regional de aire de EPA, introduciendo un nuevo cuerpo de conocimiento sobre la calidad de aire al nivel de las calles[6].
El monóxido de nitrógeno, el dióxido de nitrógeno y el ozono son los principales contaminantes que se encuentran en el aire de una ciudad en constante interacción. El equipo de Aclima pudo ver cómo los datos correspondientes a estos contaminantes fueron procesados y transmitidos desde los coches a las herramientas y mapas de visualización.

2.2 Antecedentes en México

2.2.1 Sistema de monitoreo para rediseñar control de emisiones

En la ciudad de Guadalajara, se implementó un sensor remoto que mide la concentración de las partículas contaminantes que emiten los automóviles de la urbe[7].
El dispositivo contaba con un rayo láser, el cual tomaba las placas de los automóviles que pasaban frente al dispositivo y con un rayo infrarrojo para detectar las emisiones, con lo cual se creó una base de datos para identificar a los vehículos que contaminaban más en la zona metropolitana de Guadalajara. Posteriormente, la información recabada se entregó a la Secretaría de Medio Ambiente y Desarrollo Territorial (SEMADET) para que con los datos estadísticos que se arrojen de la información de los sensores, se rediseñe el programa de Verificación Vehicular.
El rayo infrarrojo leía en tiempo real la emisión de gases contaminantes como el monóxido de carbono, dióxido de carbono y óxido nítrico. La agencia de protección al

Medio Ambiente recomendó al sensor remoto para conocer la estadística de las emisiones de los vehículos en circulación, y con base en esa estadística plantear y poner en vigor políticas de disminución de contaminación, según sea la autoridad correspondiente al tema.

2.2.2 Radares de contaminación vehicular en la Ciudad de México colocados por la SEMARNAT

En julio de 2016 se implementó el uso de radares en el Valle de México para identificar a los vehículos que contaminan más, y así poder sancionar a sus dueños. Esta implementación forma parte de la Norma Emergente de Verificación Vehicular[8].
El plan del sistema consiste en colocar sensores móviles en puntos estratégicos del Valle de México para medir la contaminación de vehículos en marcha y, en dado caso, poder enviar a los domicilios la fotografía y el talón de pago de la multa.
En los operativos que implementan estos sensores, se les obliga a los automovilistas a bajar la velocidad a poco más de 50 metros en una pequeña caseta rodante, donde está el centro de control. A baja altura, de ambos lados de la calle, se colocan lectores ópticos de rayos infrarrojos y ultravioleta que miden las emisiones contaminantes que salen del escape del vehículo, y envían la información a una computadora. De frente, una cámara toma la fotografía de las placas de circulación, para tener el registro de cada unidad que pasa por el retén. Con este plan, se plantea complementar al nuevo Programa de Verificación Vehicular.

Como se habrá apreciado, en México aún no se han implementado sistemas de monitoreo de contaminantes con dispositivos inalámbricos de un reducido tamaño para su posterior procesamiento, y entrega al usuario final de los datos procesados en una interfaz amigable para él, lo cual se presenta como una oportunidad para explorar este campo en nuestro país. El campo de desarrollo de esta tecnología, así como del desarrollo de protocolos de enrutamiento inalámbrico es prometedor, con lo cual se hará un aporte significativo al rubro de soluciones tecnológicas a la contaminación ambiental y al diseño de protocolos de enrutamiento inalámbricos.

Capítulo 3:
Redes de Sensores

En este capítulo se expondrá la teoría respecto a las redes inalámbricas y las redes de sensores. También se explicarán los diversos algoritmos de enrutamiento de las redes Ad-hoc, para ir analizando las características de cada algoritmo y deducir si se pueden utilizar en el diseño del proyecto.

3.1 Redes Inalámbricas

Son redes de computadoras que no requieren el cableado que se utiliza en las redes tradicionales, ya que se utilizan diversas tecnologías inalámbricas, haciendo la red más dinámica y libre de las limitaciones de las redes cableadas.
Hoy en día las instituciones, empresas y hogares poseen una creciente demanda de estructuras de redes cada vez más inalámbricas, ya que los dispositivos que se utilizan hoy en día se pueden conectar a internet de forma inalámbrica por medio de la tecnología WiFi. Las redes inalámbricas pueden dividirse en dos categorías: modo infraestructura y modo Ad-hoc.

3.1.1 Modo Infraestructura

Son redes inalámbricas basadas en la infraestructura de las redes cableadas. Los nodos se comunican con la estación base estableciendo enlaces inalámbricos, mientras que las estaciones base se comunican entre sí, utilizando redes dorsales preestablecidas de alta velocidad.

Una forma de clasificar a las redes inalámbricas en modo infraestructura es por su topología o uso en: topología de estrella, punto a punto, caso de repetidores y topología de malla[9].

3.1.2 Modo Ad-hoc

En este, los nodos se pueden mover libremente y no dependen de un control central establecido. En estos casos, el nodo puede actuar tanto como un Access Point o como una estación, dependiendo del rango de transmisión de cada uno de los nodos estación presentes en la red[10].

3.2 Redes Inalámbricas Ad-hoc

Las redes Ad-hoc son redes inalámbricas formadas por un conjunto de nodos móviles que operan de forma completamente autónoma y no utilizan ningún tipo de infraestructura fija ni administración centralizada. Características tales como topología dinámica, ancho de banda restringido y consumo energético limitan el funcionamiento de las redes Ad-hoc, y hacen de los algoritmos de enrutamiento una pieza clave para el buen funcionamiento de las mismas. Las redes Ad-hoc, cuando implementan sensores en sus nodos, se les denomina redes de sensores o Wireless Sensor Network (WSN). En la figura 1, se puede observar la apariencia de una red Ad-hoc típica.

Los algoritmos de enrutamiento existentes para redes Ad-hoc se clasifican (ver diagrama 1) en:

1. Algoritmos proactivos
2. Algoritmos reactivos
3. Algoritmos híbridos.

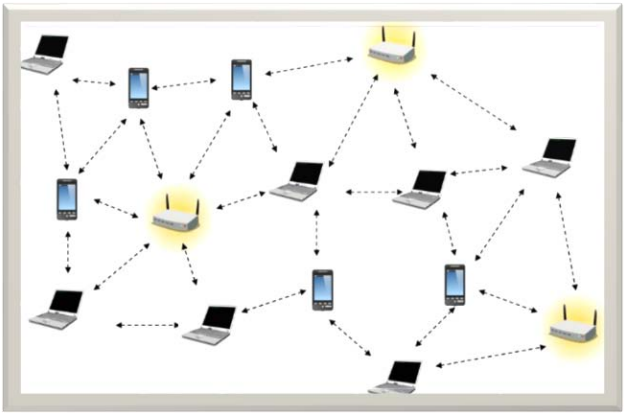


Figura 1. Red Ad-hoc.

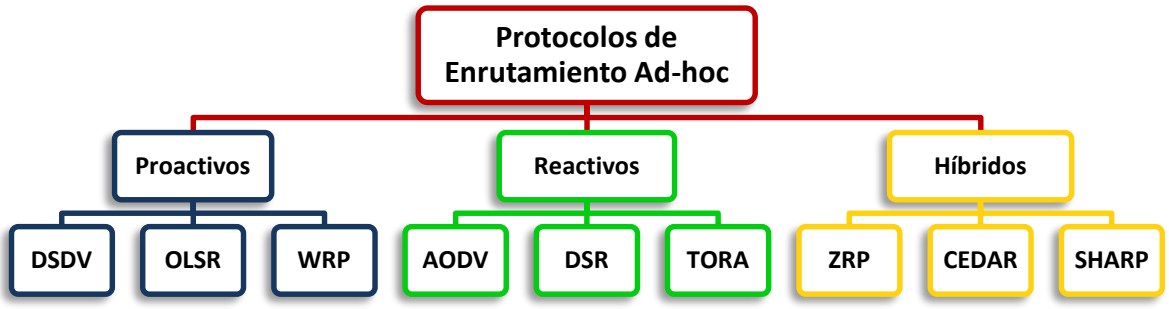


Diagrama 1. Clasificación de algunos de los protocolos de enrutamiento Ad-hoc.

3.2.1 Algoritmos proactivos

Los algoritmos de enrutamiento proactivo se han venido utilizando inicialmente en las redes Ad-hoc. Estos algoritmos mantienen información de enrutamiento de cada uno de los nodos al resto de nodos de la red de forma consistente, e independientemente de que dicha información sea o no utilizada. Los algoritmos proactivos, intentan seleccionar el siguiente salto hacia el destino utilizando índices tales como número de saltos, retardo o utilización de los enlaces, para seleccionar una ruta mínima[11].
Estos algoritmos mantienen constantemente actualizada la información de direccionamiento a través de intercambios de paquetes a intervalos temporales fijos. La desventaja de estos algoritmos es que producen tráfico de señalización, incluso cuando no se transmite ningún paquete de datos; esto puede provocar sobrecarga en la red[12].
En los protocolos proactivos las rutas son buscadas y mantenidas, aunque éstas no estén activas.

Algunos de los algoritmos proactivos existentes en la literatura son:

1. Destination-Sequenced Distance Vector (DSDV)
2. Optimized Link State Routing (OLSR)
3. Wireless Routing Protocol (WRP)

3.2.1.1 Destination-Sequenced Distance Vector (DSDV)

El algoritmo de Vector-Distancia de Destino Secuenciado (DSDV) es un algoritmo de enrutamiento proactivo basado en el algoritmo de distancia de vector Bellman-Ford. Las tablas de enrutamiento son mantenidas y actualizadas, a lo que la emisión periódica de paquetes de actualización de tablas de enrutamiento consume el ancho de banda. Por lo tanto, la principal debilidad de DSDV es que cuando la red crece, estos paquetes también aumentan[13].
Cada nodo perteneciente a la red tiene una tabla de enrutamiento que indica para cada destino cuántos saltos (hop) hacen falta atravesar y cuál es el nodo sucesivo. Derivando del vector de distancia, las actualizaciones de las tablas de enrutamiento se producen mediante el intercambio de información entre nodos cercanos y reaplicando los algoritmos de camino mínimo a menor costo. Cada camino viene etiquetado con un número de secuencia (sequence number), que da una indicación temporal sobre la validez de aquel camino: a números de secuencia más altos corresponden caminos más fiables. En caso de que un nodo notase que un trayecto hacia un destino no funciona, asigna al número de salto un valor alto (que significa infinito) y al número de secuencia un número impar. Un número de secuencia identificado con un número impar señala que aquel camino es inalcanzable mientras que, por el contrario, un número par indica que el destino sí es alcanzable. La figura 2 ejemplifica a este algoritmo.

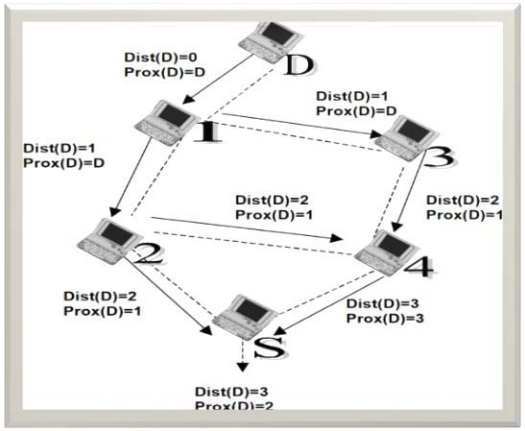


Figura 2. Ejemplo de algoritmo de encaminamiento Vector Distancia.

3.2.1.2 Optimized Link State Routing (OLSR)

El Enrutamiento Optimizado de Estado del Enlace (OLSR) es un protocolo de enrutamiento proactivo que ha sido modelado para hacer frente a las redes Ad-hoc móviles (MANET). Su idea básica es elegir un nodo representativo por cada grupo de nodos vecinos y dividir por lo tanto la red en grupos. Estos nodos representativos entonces seleccionan un conjunto de nodos especializados llamados relés multipuntos (MPRs). La función de los nodos MPR es la de reducir la sobrecarga de mensajes minimizando las transmisiones duplicadas dentro de la misma zona[14].

3.2.1.3 Wireless Routing Protocol (WRP)

El Protocolo de Enrutamiento Inalámbrico (WRP) es un protocolo de enrutamiento de vector-distancia basado en tablas. Cada nodo en la red mantiene una tabla de distancia, una tabla de enrutamiento, una tabla de costo de enlace y una lista de retransmisión de mensaje. La tabla de distancia de un nodo X contiene la distancia de cada nodo de destino Y vía cada vecino Z de X. También contiene el vecino de Z con el que este camino se lleva a cabo. La tabla de enrutamiento del nodo X contiene la distancia de cada nodo destino Y desde el nodo X, el predecesor y el sucesor del nodo X en este camino[15].
Guardando el predecesor y el sucesor en la tabla es benéfico detectando bucles y evitando problemas de conteo al infinito. La tabla de costo de enlace contiene el costo de enlace a cada vecino del nodo, y el número de tiempo agotado desde que un mensaje libre de errores fue recibido de ese vecino. La Lista de Retransmisión de Mensaje (MRL) contiene información para dejar a un nodo saber cuál de sus vecinos no ha admitido su mensaje de actualización para retransmitir dicho mensaje a ese vecino.

3.2.2 Algoritmos reactivos

En los algoritmos reactivos viene invocado un procedimiento para determinar el correcto direccionamiento sólo en el momento en el que el paquete deba efectivamente transmitirse, es decir, descubren la ruta que se necesita solamente cuando se tiene que enviar un paquete a través de la misma. De este modo, se reduce tanto la sobrecarga de la red como el consumo energético, en detrimento de un aumento de los tiempos de entrega.
Los algoritmos reactivos utilizan dos aproximaciones diferentes:

a. Encaminamiento fuente
b. Aprendizaje hacia atrás

En los algoritmos con encaminamiento fuente, la ruta completa se almacena en el propio paquete de descubrimiento de ruta. Estos algoritmos, aunque evitan que los nodos intermedios almacenen información de encaminamiento, por el contrario, consumen ancho de banda adicional debido a la longitud del paquete.
Los algoritmos que utilizan aprendizaje hacia atrás, mantienen tablas de encaminamiento en los nodos intermedios para conseguir que cada paquete solamente almacene la dirección del siguiente nodo, reduciendo el tamaño del paquete.

Algunos algoritmos de tipo reactivo desarrollados son:

1. Ad hoc On-Demand Distance Vector (AODV)
2. Dynamic Source Routing (DSR)
3. Temporally Ordered Routing Algorithm (TORA)

3.2.2.1 Ad hoc On-Demand Distance Vector (AODV)

El algoritmo crea una ruta entre dos nodos, sólo cuando ésta es demandada por el nodo fuente. Esto permite a los nodos tener flexibilidad para entrar y salir de la red. Las rutas permanecerán activas sólo cuando existan paquetes transmitiéndose desde el nodo fuente al nodo destino. Cuando el nodo fuente termine de enviar paquetes y cierto timeout expire, la ruta entre los nodos se cancelará.
Una característica fundamental del protocolo es que los nodos destino de un trayecto, antes de proporcionar información de direccionamiento, crean un número de secuencia de destino, que proporciona a los nodos un instrumento para evaluar cuanto se ha actualizado un determinado recorrido evitando la formación de lazos (loop) en el camino de enrutamiento. Este protocolo usa mensajes particulares llamados RREQ (Route Request), RREP (Route Replies) y RERR (Route Errors) que son enviados y recibidos mediante el protocolo UDP.

3.2.2.2 Dynamic Source Routing (DSR)

La principal característica del Enrutamiento de Origen Dinámico (DSR) es el uso de fuentes de enrutamiento. Esto es, el emisor conoce la ruta completa de saltos hasta llegar al receptor. Estas rutas son almacenadas en memoria caché. Los paquetes contienen la ruta hacia la fuente en su cabecera. Cuando un nodo intenta enviar un paquete a un nodo, del cual no conoce su ruta, éste inicializa un proceso de descubrimiento de ruta, con el propósito de determinar dinámicamente la ruta a tomar[16].
A continuación, se resumen las principales características del protocolo de red DSR:

- Los paquetes de enrutamiento que envían los nodos contienen toda la información de la ruta que éstos deben seguir. Los nodos intermedios pueden adquirir esta información para un posterior uso.
- DSR no requiere paquetes periódicos de mantenimiento de rutas. Lo que significa, que funciona totalmente bajo demanda (OnDemand).
- Cuando todos los nodos se encuentran en un estado estacionario, no existen mensajes de red entre ellos, lo cual evita embotellamientos. Solamente cuando se producen cambios, debidos a movimientos en la red, se producen mensajes de la capa de red.
- Un nodo puede almacenar diferentes rutas para un mismo destino. Esto es una ventaja cuando se rompe el enlace de comunicación en uso. La ventaja se produce por el hecho de que no se tiene que iniciar inmediatamente un nuevo proceso de descubrimiento de rutas.

3.2.2.3 Temporally Ordered Routing Algorithm (TORA)

Este algoritmo está diseñado para reaccionar de manera efectiva ante cambios en la red y particiones de la misma. El nombre de este protocolo proviene de la asunción de tener relojes sincronizados en los nodos, esto se puede obtener vía GPS.

El principal objetivo de TORA es obtener rutas estables, las cuales pueden ser rápida y localmente reparadas, en caso de ruptura de enlaces. El protocolo construye un grafo acíclico directo de rutas hacia el nodo destino, denominado “Direct acyclic graph”. El DAG se obtiene asignando una dirección a cada enlace entre nodos, y un peso, o nivel de referencia a cada nodo. El grafo DAG tiene la siguiente propiedad, los nodos tienen al menos un enlace de salida.

El funcionamiento del protocolo TORA puede ser dividido en tres fases:

- Descubrimiento de ruta.
- Mantenimiento de ruta.
- Cancelación de ruta.

3.2.3 Algoritmos híbridos

Los algoritmos de encaminamiento híbrido intentan mejorar tanto la sobrecarga de los algoritmos proactivos como el retardo inicial de los algoritmos reactivos, combinando de forma adecuada las características de ambos tipos de algoritmos para generar algoritmos adaptativos y parametrizables, y limitando la aplicación de algoritmos proactivos sólo a los nodos adyacentes del nodo que quiere transmitir.

Entre los protocolos híbridos conocidos, se encuentran:

1. Zone Routing Protocol (ZRP)
2. Core-Extraction Distributed Ad-hoc Routing (CEDAR)
3. Sharp Hybrid Adaptive Routing Protocol (SHARP)

3.2.3.1 Zone Routing Protocol (ZRP)[17]

En el Protocolo de Enrutamiento basado en Zonas, la red se divide en zonas de enrutamiento que se superponen, las cuales pueden usar protocolos independientes dentro y entre cada zona. La comunicación dentro de una zona específica se realiza a través del protocolo de enrutamiento IARP (Intrazone Routing Protocol), el cual proporciona descubrimiento de vecinos directamente de forma efectiva (enrutamiento proactivo). La comunicación entre diferentes zonas, es realizada por el protocolo de enrutamiento IERP (Interzone Routing Protocol), el cual proporciona capacidad de enrutamiento entre nodos que deben de comunicarse entre zonas (enrutamiento reactivo).

IERP está basado en un mecanismo de distribución de mensajes conocido como Bordercast Resolution Protocol (BRP). BRP permite que las consultas sean dirigidas fuera de la red local y hacia regiones de la red que no hayan sido cubiertas por la consulta. El proceso de enrutamiento reactivo se divide en dos fases: la fase de solicitud de ruta y la fase de respuesta de ruta. En la solicitud de ruta, el nodo origen envía un mensaje de solicitud de ruta a sus nodos periféricos (bordercasting) usando BRP. Si el receptor de un mensaje de solicitud de ruta conoce el destino, responde enviando un mensaje de respuesta de ruta de regreso al nodo origen. De lo contrario, se continúa el proceso por bordercasting. El proceso bordercasting consiste en que, una vez el nodo periférico ha recibido un mensaje de solicitud de ruta y no encuentra el nodo destino dentro de su red, debe de reenviar el mensaje a sus nodos periféricos. De esta manera, la solicitud de ruta se extiende por toda la red (ver figura 3).

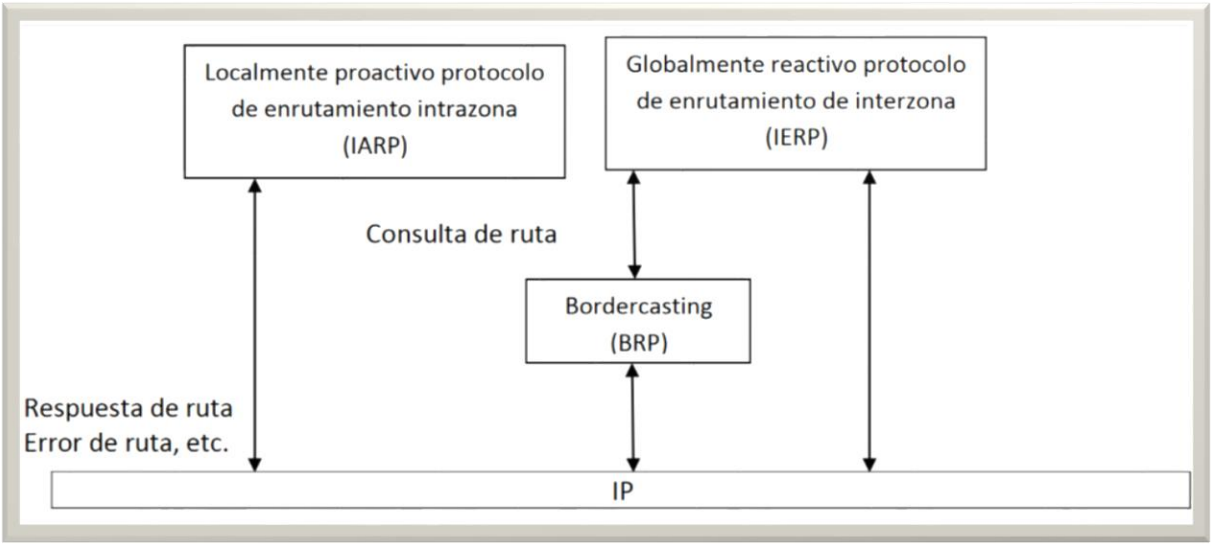


Figura 3. Arquitectura del protocolo ZRP.

3.2.3.2 Core-Extraction Distributed Ad-hoc Routing (CEDAR)[18]

El algoritmo híbrido de enrutamiento Ad-hoc distribuido de Extracción de Núcleo dinámicamente establece un núcleo de la red, y entonces de forma incremental propaga el estado de enlace de enlaces estables de alto ancho de banda a los nodos del núcleo. El procesamiento de la ruta es bajo demanda, y es llevado a cabo por nodos del núcleo usando sólo estado local. Este algoritmo consiste de tres componentes clave:

1. Extracción de núcleo: Un conjunto de nodos es elegido de forma distribuida y dinámica para formar el núcleo de la red, aproximando un conjunto mínimo dominante de la red Ad-hoc y utilizando sólo procesamiento y estado local.
2. Propagación de estado de enlace: El enrutamiento de QoS en CEDAR es conseguido propagando la información de disponibilidad de ancho de banda de enlaces estables en el núcleo conocido a los nodos más lejanos de la red, mientras la información sobre enlaces dinámicos o enlaces de ancho de banda bajo se mantiene en forma local.
3. Cálculo de la ruta: El cálculo de la ruta primero establece un camino-núcleo del dominador del origen al dominador del destino. El camino del núcleo provee la direccionalidad de la ruta del origen al destino.

3.2.3.3 Sharp Hybrid Adaptive Routing Protocol (SHARP)[19]

En el Protocolo de Enrutamiento Adaptativo Híbrido Nítido (SHARP), cada nodo determina la vecindad de la red, llamada zona proactiva, en la cual la información de encaminamiento perteneciente a sí misma está diseminada proactivamente. El protocolo

SHARP se basa en un algoritmo de enrutamiento proactivo original. Sin embargo, el algoritmo puede utilizar cualquier algoritmo de enrutamiento reactivo cuyos costos pueden ser caracterizados analíticamente. SHARP encuentra el "punto dulce" entre los dos regímenes de enrutamiento, ajustando dinámicamente el alcance del enrutamiento proactivo y reactivo.
Los requerimientos para el rendimiento de la red varían entre aplicaciones. Sin embargo, las aplicaciones no tienen control sobre el rendimiento de los protocolos de enrutamiento tradicional. En contraste, el protocolo SHARP habilita cada aplicación para perseguir diferentes métricas cuantitativas, para guiar a la compensación inherente entre la sobrecarga incrementada para información proactiva, contra latencia reducida y tasa de pérdida. Cada nodo SHARP puede, de forma separada, perseguir diferentes garantías de rendimiento específicas a cada aplicación.

3.3 Red de sensores o WSN

Una red de sensores o Wireless Sensor Network (WSN) es una infraestructura comprendida de elementos de detección, procesamiento y comunicación inalámbrica que brindan a un administrador la habilidad de instrumentar, observar y reaccionar a eventos y fenómenos en un entorno específico. El entorno puede ser el mundo físico, un sistema biológico, o una estructura de tecnologías de la información. Las típicas aplicaciones incluyen: recolección de datos, monitoreo (ver figura 4), vigilancia y telemetría médica. Además de sistemas de detección, frecuentemente se presenta interés en sistemas de control y activación.
Existen cuatro componentes básicos en una red de sensores:

1. Un conjunto de sensores distribuidos o localizados.
2. Una red de interconexión (usualmente, pero no siempre inalámbrica).
3. Un punto central de agrupación de la información.
4. Un conjunto de recursos computacionales en el punto central (o más allá) para manejar la correlación de datos, tendencias de eventos, consulta de estado y minería de datos.

Dada la gran cantidad de información capturada, los métodos algorítmicos para la administración de los datos juegan un papel importante en las redes de sensores. La infraestructura de procesamiento y comunicación asociada con las redes de sensores es, con frecuencia, específica a su entorno y arraigada en la naturaleza basada en aplicación-dispositivo de estas redes. Por ejemplo, a diferencia de la mayoría de las otras configuraciones, el procesamiento dentro de la red es deseable en las redes de sensores; además, la energía del nodo (y/o vida de la batería) es una consideración clave del diseño. La información recolectada es típicamente paramétrica por naturaleza, aunque con la aparición del video de tasa de bits baja (como el MPEG-4) y los algoritmos de imagen, algunos sistemas también soportan este tipo de datos[20].

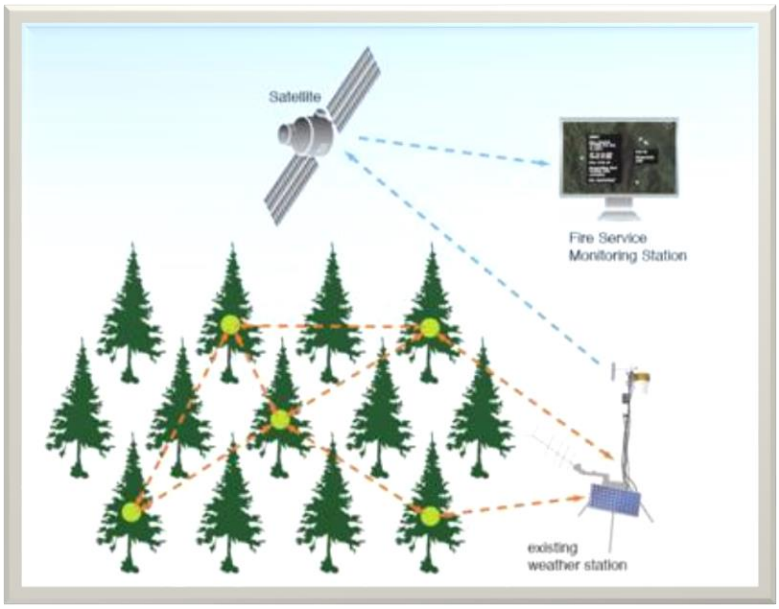


Figura 4. Red de sensores inalámbrica (WSN).

3.3.1 Topologías de red WSN

Los nodos de una red de sensores inalámbrica están típicamente organizados en tres tipos de topologías[21] de red (véase Figura 5):

1. Topología de estrella, donde cada nodo se conecta directamente al gateway.
2. Topología de árbol, donde cada nodo se conecta a un nodo de mayor jerarquía en el árbol y después al gateway. Los datos son ruteados desde el nodo de menor jerarquía en el árbol hasta el gateway.
3. Topología de malla; la característica de esta topología es que los nodos se pueden conectar a múltiples nodos en el sistema y pasar los datos por el camino disponible de mayor confiabilidad.

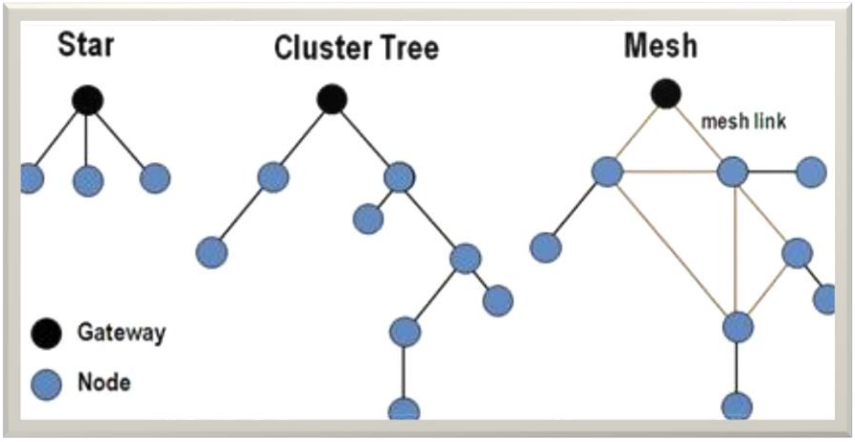


Figura 5. Topologías de redes WSN.

3.4 Consideraciones de los protocolos ya existentes respecto al proyecto

Dado a que en todos los protocolos se requiere que el nodo realice transmisiones multicast a sus vecinos, se debe diseñar e implementar un nuevo protocolo que se adecúe a las características que posee el módulo ESP8266, ya que éste no puede hacer transmisiones multicast. Uno de los objetivos del proyecto es el de diseñar e implementar un nuevo protocolo de red para topologías de árbol fuente. Dado que los nodos en modo SoftAP no pueden hacer transmisiones multicast, se optó por hacer una red de sensores de árbol, ya que por el contrario, sí puede tener varios nodos estación conectados a la vez (aunque solamente pueda tener hasta 4 nodos en modo cliente conectados cuando el nodo softAP esté en modo servidor). La mayoría de los protocolos existentes no pueden ser utilizados en la red Ad-hoc de árbol que se va a implementar, ya que éstos protocolos están diseñados para funcionar con redes de malla, en la cual existen múltiples destinos (todos los nodos pueden ser destinos), mientras que en la red de árbol sólo existe un destino, el cual es en este caso el router con acceso al servidor de alojamiento de datos recolectados.

Capítulo 4:
Diseño e implementación del sistema de redes de sensores

4.1 Componentes

El sistema de redes de sensores a desarrollar requiere de diversos componentes, de los cuales el principal será de la plataforma Arduino o semejante. Todo funcionará como un solo sistema de monitoreo de gases nocivos para el ser humano, pudiendo no solamente mostrar la información capturada de los sensores en una pantalla LCD, sino que también se podrá enviar dicha información entre módulos por vía WiFi, y si hay conexión con el router y éste tiene acceso al servidor, entonces enviar los datos capturados al servidor.

La tabla 1 nos muestra los componentes de cada nodo:

No. Componente	Nombre o modelo	Tipo	Cantidad
1	Arduino MEGA 2560[22] (figura 6)	Microcontrolador	3
2	MQ-135[23] (figura 7)	Sensor	3
3	MQ-7[24] (figura 8)	Sensor	3
4	ESP8266[25] (figura 9)	Módulo WiFi	3
5	LCD 1602[26] (figura 10)	Pantalla LCD	3
6	DS1307 (figura 11)	Módulo de fecha y hora RTC	3
7	Solar Bank (figura 12)	Cargador Solar	3
8	LM1117 (figura 13)	Regulador de voltaje	3
9	Módulo micro SD (figura 14)	Lector de tarjetas micro SD	3

Tabla 1. Componentes de un nodo.

En total se ensamblaron 3 nodos de monitoreo de contaminación con cada uno de los componentes antes mencionados.

A continuación, se mencionarán las características útiles de cada uno de los componentes de los nodos.

1. Microcontrolador Arduino MEGA

El Arduino Mega 2560 es uno de los pilares del proyecto; se encarga de procesar toda la información de entrada y salida. Es el cerebro principal del sistema, ya que sabe cómo interpretar cada dato que recibe y lo puede enviar fuera de él. Puede programarse para lograr diversas funciones de manera simultánea.

Sus características son:

- Microcontrolador de 8 bits
- 54 pines digitales de entrada/salida
- 16 entradas analógicas
- Microcontrolador de 16 MHz
- Voltaje de operación: 5V
- Voltaje de salida: 3.3V y 5V
- Voltaje de alimentación ideal entre 7 y 12 V con corriente máxima de 800 mA.
- Corriente de operación: 93 mA
- Límite de voltaje de alimentación de 20 V
- Alimentación por USB (500 mA), Jack DC y entrada Vin.
- Memoria flash de 256 KB
- SRAM de 8 KB
- EEPROM de 4 KB
- 4 puertos seriales
- Valores A/D de 0 a 1023
- Pines I2C para conectar reloj externo

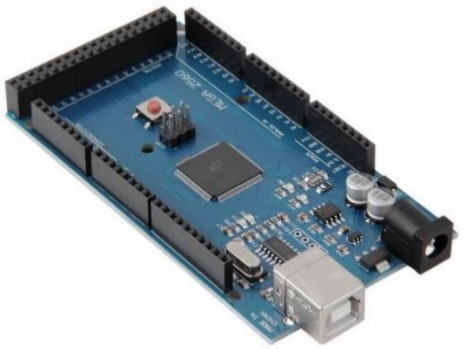


Figura 6. Arduino Mega.

2. Sensor de gases nocivos MQ-135

El sensor de gas MQ-135 se encarga de devolver un voltaje analógico al Arduino de acuerdo a las concentraciones de CO_2 que mida.

Sus características son:

- Voltaje de operación: 5V
- Corriente de operación: 150 mA
- Potencia usada máxima: 800 mW
- Detecta compuestos como Alcohol, amoniaco(NH_3), CO_2, NO y NO_3.
- Tiempo de precalentamiento: 20 segundos
- Detección de partes por millón: 10ppm~1000ppm
- Humedad de operación: <95%RH
- Concentración de oxígeno para mayor precisión: 21%
- Temperatura de operación: -20°C a 70°C
- Salida Analógica/Digital
- Calibración digital por resistencia

Figura 7. MQ-135.

3. Sensor de C0 MQ-7

El sensor de gas MQ-7 se encarga de devolver un voltaje analógico al Arduino de acuerdo a las concentraciones de CO que mida.

Sus características son:

- Voltaje de operación: 5V
- Corriente de operación: 150 mA
- Potencia usada promedio: 350mW
- Detecta monóxido de carbono (CO).
- Tiempo de precalentamiento: 60 segundos
- Detección de partes por millón: 20ppm~2000ppm
- Humedad de operación: <95%RH
- Concentración de oxígeno para mayor precisión: 21%
- Temperatura de operación: -20°C a 50°C
- Salida Analógica/Digital
- Calibración digital por resistencia

Figura 8. MQ-7.

4. Módulo WiFi ESP8266

El módulo ESP8266 es el otro pilar del proyecto, ya que se encarga de conectar al nodo con otros nodos o con un router por medio de WiFi 802.11g y, una vez conectado, envía los datos capturados de los sensores a la red. Además, propaga otros datos como el SSID, la contraseña del router, la fecha y hora de la red. Es el componente al que le corresponden las cuestiones de red, y como un plus, se puede programar ya que posee un microcontrolador integrado, aumentando su versatilidad. De acuerdo a experimentaciones, éste módulo puede alcanzar en condiciones óptimas de ambiente (en la intemperie sin obstáculos), distancias de comunicación de hasta 366 metros[27].

Sus características son:

- Protocolos 802.11 b/g
- Modos de SoftAP y station
- Stack integrado de TCP/IP
- Uso de corriente en standby: <10uA
- 1 MB de memoria flash (Programable)
- CPU de 32-bit de bajo consumo
- Voltaje de operación: 3.3V
- Corriente de operación entre 0.9 uA y 215 mA
- Frecuencia de operación: 2.4 GHz
- 11 canales de frecuencia espaciados en 5MHz
- Potencia máxima de 20.5 dBm (0.11 W)

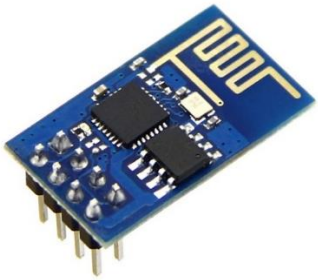


Figura 9. ESP8266.

5. Pantalla LCD 1602

La pantalla LCD se encarga de visualizar el menú y los datos capturados en el momento de CO y CO_2, además de mostrar la fecha y hora. El menú puede mostrar información como el SSID y contraseña del router, además de poderse capturar dicha información. También puede mostrar las direcciones IP del nodo, ya sea como estación o como SoftAP. Además, se puede ver y modificar el brillo y contraste de la pantalla LCD por medio de PWMs (modulación de ancho de pulso).

Sus características son:

- Formato del display: 16 Caracteres x 2 Líneas
- Modos de datos de entrada: 4-Bits u 8-Bits
- Tamaño de caracteres del display: 5 x 8 puntos
- Voltaje de alimentación: 5V±10%
- Luz de fondo
- Consumo de corriente con luz de fondo: 25 mA
- Control de contraste

Figura 10. LCD 1602.

6. Integrado DS1307 para fecha y hora

El reloj de tiempo real DS1307 se encarga de almacenar la hora capturada de internet o de otros nodos, y gracias al cristal de cuarzo que posee, puede ir corriendo el tiempo que capturó, además de seguir corriendo aún sin energía gracias a su batería CR2032.

Sus características son:

- Varios formatos de fecha/hora: formato de 12-24 hrs, ajuste automático de 28,30 o 31 días según el mes y ajuste de año bisiesto.
- Voltaje de operación: 5V
- Corriente máxima de operación: 1.5 mA
- Uso de la interfaz I^2C (solo dos canales: datos y reloj)
- Socket para batería CR2032

Figura 11. DS1307.

7. Cargador Solar

El cargador solar se utilizará en condiciones finales de experimentación para probar la autonomía de la red de nodos, ya que las redes de sensores deben ser lo suficientemente autónomas para que no requieran constantemente intervención humana y no dependan tanto del cableado eléctrico para ser alimentadas, lo cual limita su portabilidad.

Sus características son:

- Voltaje de operación: 5V
- Capacidad: 5000 mAh
- Carga solar: de 200 mA/h
- Corriente de entrada: 1A (corriente DC)
- Corriente de salida: 2 puertos x 1A

Figura 12. Cargador solar.

8. Regulador de voltaje LM1117

El integrado LM1117 se encarga de convertir el voltaje de 5V a 3.3V, que es el voltaje de operación del módulo ESP8266, ya que, si se alimenta con 5 voltios éste se puede quemar.

Sus características son:

- Voltaje de entrada: 5V
- Voltaje de salida 3.3V
- Corriente máxima de salida: 800mA

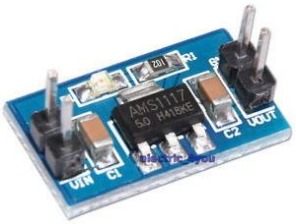


Figura 13. LM1117.

9. Módulo micro SD

El lector de memorias micro SD se encarga de almacenar la información capturada de los sensores, así en caso de no estar conectado el nodo a la red, se evitará la perdida de información.

Sus características son:

- Voltaje de operación: 5V
- Corriente de operación: 80 mA
- Interfaz: SPI
- Límite de capacidad: 2 TB

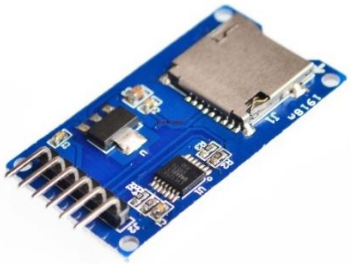


Figura 14. Módulo micro SD.

10. Interfaz USB-TTL CP2102

La Interfaz USB-TTL CP2102 sirve para probar la comunicación del ESP8266 con una computadora por medio de un puerto serie habilitado en un puerto USB de la computadora. También se utiliza para programar el ESP8266 a una velocidad máxima de 921600 baudios sin problemas (ver figura 15).

Sus características son:

- Voltaje de trabajo: 5V
- Voltajes de salida: 3.3 V/5 V
- Comunicación y programación de computadora a microcontrolador TTL.
- Tasa de bits soportada: 300 bps a 1 Mbps

Figura 15. CP2102.

4.2 Problemática en el diseño

A continuación, se enlistarán los problemas que se encontraron al momento de diseñar el prototipo del nodo inalámbrico, los cuales constaron de cuestiones de voltajes, corrientes, gasto de energía, uso de energía móvil, entre otros aspectos importantes.

1. En el aspecto del monitoreo de gases nocivos para la inhalación humana, se consideró en usar el sensor de gases MQ-135, el cual puede medir alcohol, dióxido de carbono, amoniaco, óxidos de nitrógeno y se puede calibrar de acuerdo a la ecuación potencial del gas que se quiera leer. Este sensor, al no medir solamente un gas en particular, sino que mide la calidad del aire, es decir, todos los gases en su conjunto, no es muy preciso al momento de medir un solo gas. Si en el ambiente existe un solo tipo de gas, entonces la precisión de dicho gas aumenta, pero si existen además del gas a medir en cuestión otros gases en menores proporciones, entonces la precisión disminuye. Como elemento adicional, se tomó en cuenta al componente MQ-7, el cual mide solamente concentraciones de monóxido de carbono, con lo cual se puede mejorar el sistema al poder medir dos lecturas de calidad del aire. En ambos casos, el problema es al momento de calibrar cada componente, ya que cada uno tiene diferentes curvas de calibración según el gas a calibrar, y las ecuaciones resultantes de la calibración no son lineales. La solución a este problema se detalla en el apartado 4.4.2.

2. En el tema de la alimentación, por tratarse de un sistema de sensores inalámbricos, la alimentación directa con eliminadores de pared no es siempre una opción a tomarse en cuenta, por lo que se analizaron diversas formas de alimentar al módulo de forma portátil.
   La primera fue alimentar el módulo con una batería de 9V, pero el inconveniente de alimentar al módulo de esta forma es la densidad energética que provee la batería, la cual oscila entre 500 a 600mAh, insuficiente para alimentar al nodo por mucho tiempo, y la corriente proporcionada no pasa de los 300mA, la cual es muy baja y no nos sirve para la demanda de corriente que pudiese tener el módulo, la cual pudiese superar los 478mA.
   Como segunda opción, se analizó utilizar baterías recargables 18650, ya que proporcionan una densidad de corriente de 2300mAh y un voltaje de 3.7V, pudiendo satisfacer de forma exitosa la demanda de voltaje y corriente del

módulo. El detalle de esta batería es su costo, el cual es caro para implementar varios módulos.
Finalmente, se optó por usar cargadores portátiles USB, los cuales son más baratos que las baterías 18650 y vienen en diversas capacidades, desde 2200mAh hasta 10000mAh y brindan de 1 a 2 amperios de intensidad de corriente. Las baterías solares fueron la opción a utilizarse.

3. En la cuestión de almacenamiento de datos, mientras no se envían, se llegó a la conclusión de almacenarlos en la memoria SD, así no se corre el riesgo de sobrescribir los datos en otros ya existentes y se tiene una mayor cantidad de memoria para respaldar información en caso de que el nodo no disponga de un camino para enviar su información a internet.

4. En el asunto del envío de datos con fecha y hora de la toma del dato, primero se contempló en usar el reloj interno del sistema Arduino, pero éste reloj se reinicia cada vez que la placa Arduino pierde corriente, a lo que se eligió como mejor opción al integrado DS1307, ya que posee una batería CR2032 para mantener almacenada la fecha y hora, además de permitir que el reloj siga avanzando aunque se corte la energía del módulo.

5. En el caso de los módulos de WiFi, se utilizó el módulo de WiFi ESP8266 para enviar información de control y de datos a través de la red Ad-hoc.

6. En el tema de los protocolos de enrutamiento, se requirió diseñar e implementar un nuevo protocolo de enrutamiento proactivo que se adecuara a las capacidades de hardware del módulo WiFi ESP8266, ya que por diseño de hardware no puede llevar a cabo múltiples conexiones de nodo estación a nodos SoftAP, es decir, no puede realizar multicast a otros nodos, lo cual es necesario para implementar muchos de los protocolos de enrutamiento existentes para las redes Ad-hoc. Lo que sí se pudo llevar a cabo es la conexión de varios nodos estación a un sólo nodo SoftAP, lo cual facilita diseñar una red Ad-hoc de sensores con topología de árbol.

4.3 Implementación en el tema de redes

En el área de redes se tomó en cuenta las capacidades de hardware y software de los módulos ESP8266, ya que no funcionan de la misma forma que los componentes de una red cableada. Éstos módulos son muy sensibles a las variaciones de voltaje, lo cual fue un problema más a enfrentar al momento de implementar el módulo ya que, si no se alimenta adecuadamente, las variaciones de voltaje o una corriente insuficiente provoca que el módulo se reinicie de forma frecuente o no opere de manera normal, afectando el rendimiento de la red.

Se aprovecharon las características del módulo tales como:

- La capacidad de ser **programable** por medio de una interfaz USB-TTL, es decir, posee memoria flash para introducirle información de programación, con lo que el módulo es un microcontrolador independiente, o un componente 'inteligente'.
- Modificar las direcciones IP de softAP e IP de estación.
- Habilitar un servidor en cada nodo para permitir conectar hasta 4 nodos clientes.
- Habilitar el modo de softAP, con el cual se puedan conectar cuantas estaciones permita la tabla DHCP.
- Habilitar una tabla DHCP para los nodos estación que se conecten al softAP.
- Poder conectarse a una página web o a la dirección IP de un servidor por medio de un nodo conectado a un router con acceso a internet.

Nota: Es de importancia destacar que no es lo mismo la relación softAP-estación que la relación Servidor-Cliente. En la primera relación un módulo se configura como softAP y el otro módulo como estación. En el softAP se pueden conectar cuantos módulos se permitan conectar de acuerdo a la configuración de la tabla DHCP. En cambio, en la relación Servidor-Cliente se pueden conectar al servidor hasta 4 clientes (límite de hardware), y se debe de habilitar el modo servidor en el módulo ESP.

Los módulos ESP tendrán como SSID (nombre del nodo) un formato especifico, el cual será de la siguiente forma:

chipID	**Número de saltos**

donde los campos van separados por comas.
El primer campo se refiere al identificador único de cada módulo, el cual se basa en los 24 últimos bits de la dirección MAC.
El segundo campo es el número de saltos que debe realizar el nodo para llegar al router con acceso a internet. Un ejemplo de un SSID de nodo con este formato sería de la siguiente forma: 3a916f,3

4.3.1 Diseño del protocolo PSFS

El protocolo de comunicación de la red de sensores de gases contaminantes se basa en la comunicación WiFi para elegir qué nodo se conectará a internet. En un principio, se pensó en que todos los nodos estuviesen conectados a la vez a internet (topología de estrella), pero esto trae como consecuencia que el punto de acceso (AP) se sature, a lo que futuros nodos de sensores u otros dispositivos finales como tablets, smartphones o laptops no se podrán conectar a la red del punto de acceso, dado que, si los nodos de sensores son muchos, éstos agotarán la tabla de asignación de direcciones IP del punto de acceso. Esto afectaría mucho la experiencia del usuario (QoE[28]) con el uso de internet y del sistema de monitoreo de gases. Además, no todos los nodos pueden estar lo suficientemente cerca del router como para permanecer conectados a él.

El protocolo diseñado se nombró como **Protocolo basado en Saltos y Fuerza de Señal (PSFS)**, dado a que, para hallar el mejor camino, se basa en crear una lista de los vecinos detectados con una combinación del número de saltos hacia el router y la intensidad de la señal, eligiendo al vecino con el número más bajo de esta lista.

La red diseñada e implementada se clasifica como una **red inalámbrica** en modo **Ad-hoc**, siendo una **red de sensores (WSN)** con topología de **árbol**.

Los algoritmos de enrutamiento se clasificaron como algoritmos **proactivos**, ya que los nodos constantemente están revisando si la conexión con su nodo superior sigue vigente, para así enviar los datos recolectados al nodo destino (el cual es el router), o para solicitar una actualización de fecha y hora con el conocimiento de que hay una conexión constante hacia el nodo destino, así como la actualización de datos de router. El protocolo se define, según la topología de la red, como un **protocolo basado en árbol fuente**.

4.3.2 Algoritmos del protocolo PSFS

Los algoritmos diseñados para el Protocolo basado en Saltos y Fuerza de Señal fueron cuatro en esencia, los cuales, trabajando en conjunto hacen que la red WSN funcione para llevar a cabo su propósito: el de recolectar y actualizar información. Dichos algoritmos son:

1. Algoritmo de Saltos.
2. Algoritmo de Propagación de Datos de Router.
3. Algoritmo de Sincronización.
4. Algoritmo de Transferencia de Datos.

4.3.2.1 *Algoritmo de saltos*

Este algoritmo consiste en explorar todos los nodos cercanos para comparar sus saltos hacia el router y potencia de señales, de modo que se elija el camino más corto y efectivo al router. En el diagrama 2 se puede observar el diagrama de flujo de este algoritmo.

1. Todos los nodos comienzan con un número de saltos igual a 0, y permanecerá así siempre y cuando no estén conectados a una red, como lo muestra la figura 16.

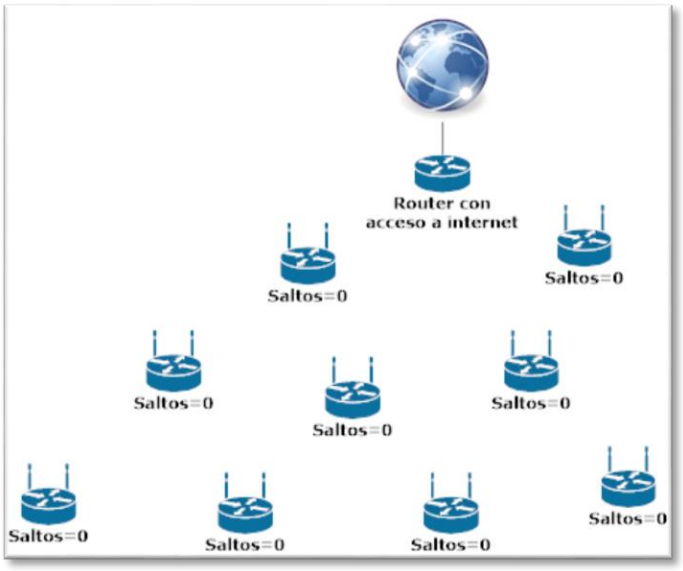


Figura 16

2. Si el nodo es designado para conectarse al router y lo consigue, entonces el número de saltos se fija en 1 (lo cual señala que se requiere de un salto para llegar al router) (ver figura 17).

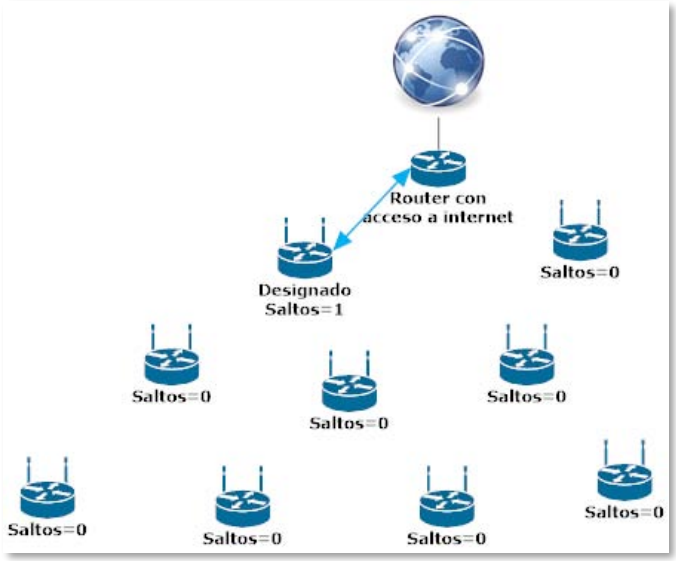
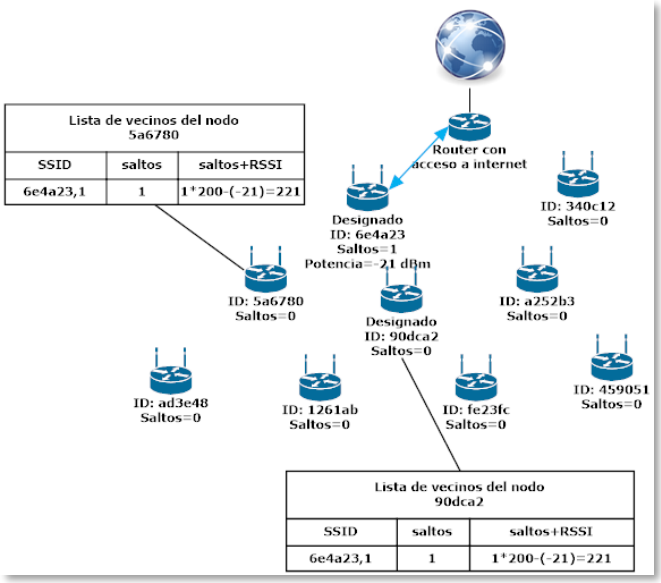
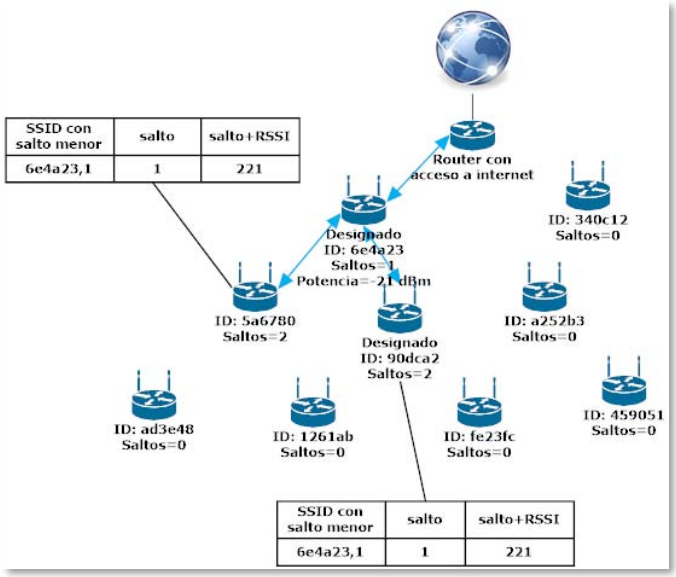


Figura 17

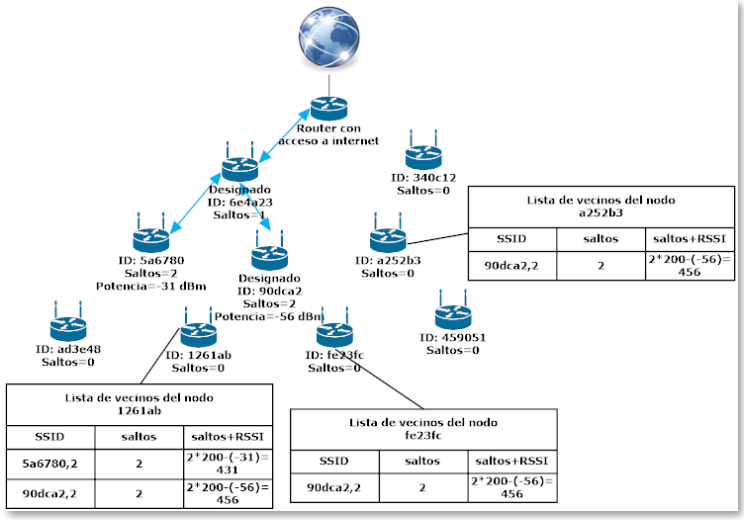
3. Si el nodo no está designado para conectarse al router o si está designado, pero no posee como vecino al router, creará una lista con los saltos multiplicados por 200 y con las potencias de señal(RSSI) restadas de cada nodo vecino (véase la nota al final de esta sección para conocer el porqué de estas operaciones). Si el salto del nodo vecino es cero, el nodo vecino se ignora para incluirse en la lista. La lista se ordenará de menor a mayor número de saltos con potencia de señales con un algoritmo bubble sort (dado el poco número de nodos vecinos probables), y se elegirá al SSID con el menor número de saltos y mayor potencia de señal para conectarse a él. El nodo colocará su salto igual al salto del vecino al que se haya conectado más 1. En las figuras 18 a-f se aprecia paso a paso cómo se lleva a cabo este procedimiento.



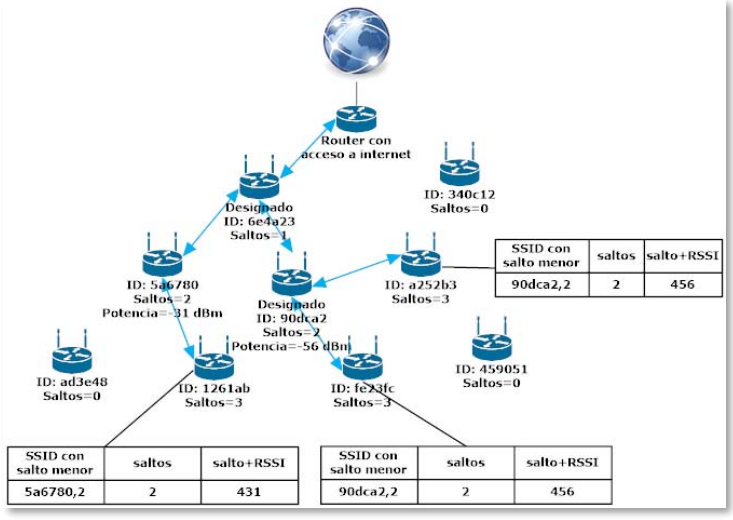
a)



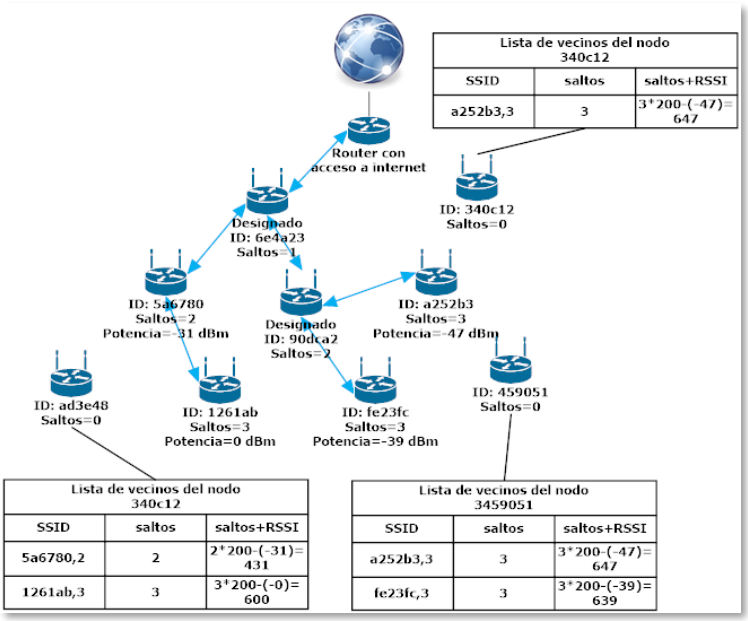
b)



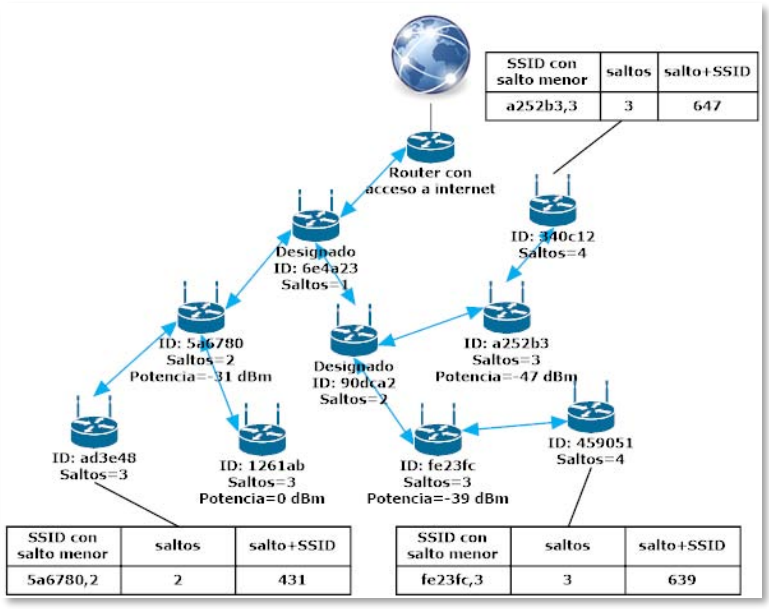
c)



d)



e)



f)

Figura 18

4. En la figura 19 se ve que, si un nodo se desconecta de la red o se deja de designar para conectarse al router, entonces su número de saltos se colocará en 0.

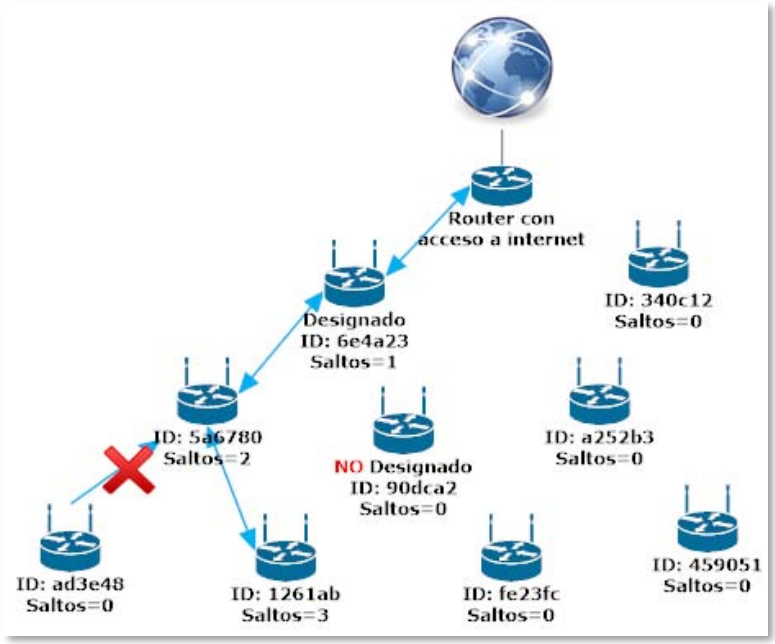


Figura 19

Nota: Cabe mencionar que, en caso de existir dos vecinos con el mismo número de saltos, se elegirá el que tiene mejor potencia de señal. El cálculo del mejor candidato se realiza de la siguiente manera:
1. Se multiplica por 200 el número de saltos. Esto se hace debido a que la potencia de señal se mide en dBm, con un rango promedio de -100 dBm a + 5dBm. Se requiere de un rango para que las potencias de cada vecino con saltos diferentes no se traslapen, entonces el rango de cada salto se definió de -50 dBm a +149 dBm.
2. A la cantidad resultante se le resta la potencia en dBm. Generalmente la potencia aparece con números negativos, siendo el mayor numero la mejor señal y el menor número la peor (teniendo en cuenta el signo). Esta operación resulta en una suma que causa que los vecinos con peor señal se queden arriba del ordenamiento, y el mejor candidato quede hasta abajo del ordenamiento.

Fórmula general: saltoyRSSI= num_saltos*200 - RSSI

Como ejemplo tomaremos dos nodos con el mismo número de saltos (2 saltos), pero diferente potencia de señal:

Operación con nodo vecino 1: saltoyRSSI= 2*200 – (-31 dbm) = 431

Operación con nodo vecino 2: saltoyRSSI= 2*200 – (3 dbm) = 397

En el ordenamiento, el nodo 2 quedará más abajo que el nodo 1, por lo tanto, es mejor candidato para conectarse a él.

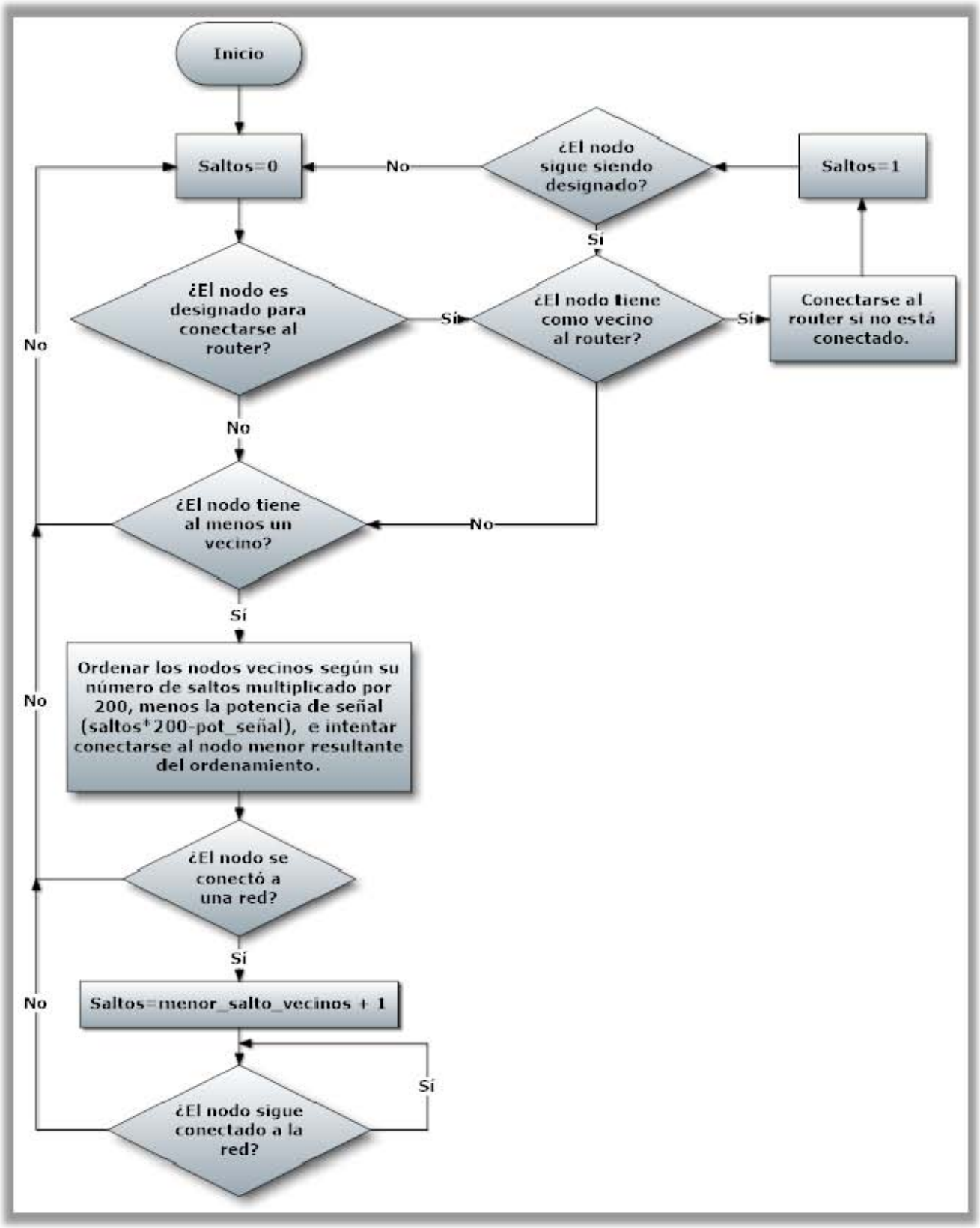


Diagrama 2. Algoritmo de saltos

4.3.2.2 *Algoritmo de Propagación de Datos de Router*

Este algoritmo consta de propagar los datos referentes al router, tales como el nombre del router, su contraseña y la página web o dirección IP donde se enviarán los datos de los sensores almacenados en la micro SD. En el diagrama 3 se puede observar el diagrama de flujo de este algoritmo.

1. Se designan él o los nodos que vayan a conectarse al router de forma manual. Estos nodos deberán de estar fijos para garantizar que el resto de los nodos se puedan conectar a éstos sin la preocupación de que se muevan y pierdan la conexión a internet. Si el nodo designado se conecta al router, entonces su Id de propagación se colocará en 1. La figura 20 ilustra este paso.

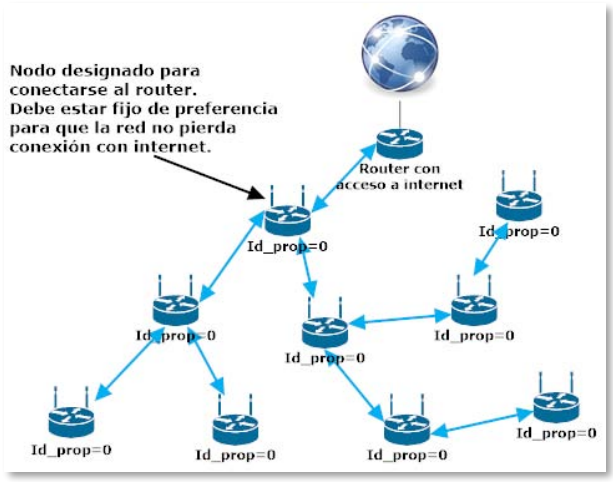
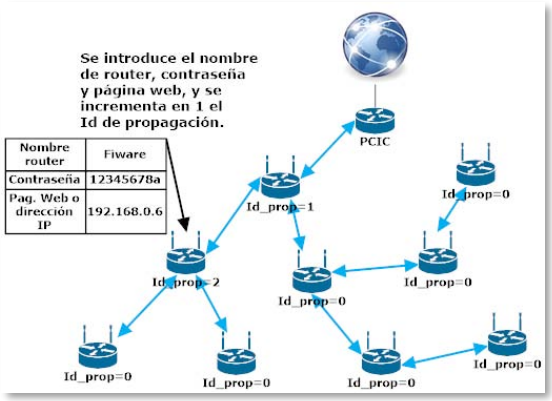
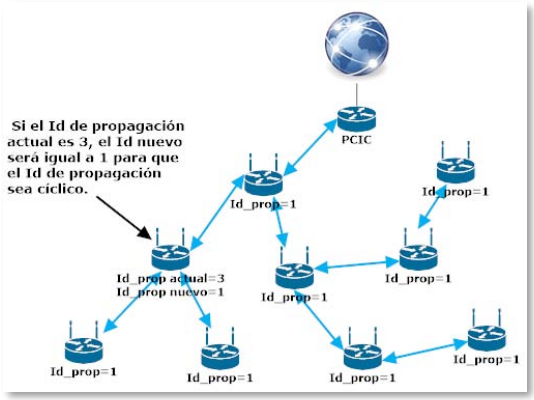


Figura 20

2. Se introducen en un nodo conectado a una red con acceso al router los datos del router (nombre, contraseña y página web) para que éstos se propaguen a través de la red aunados a un Id de propagación, el cual aumenta en 1 hasta llegar a 3, de ahí se reinicia el conteo a 1 para hacer el conteo del Id de propagación cíclico (véase figuras 21 a-b).



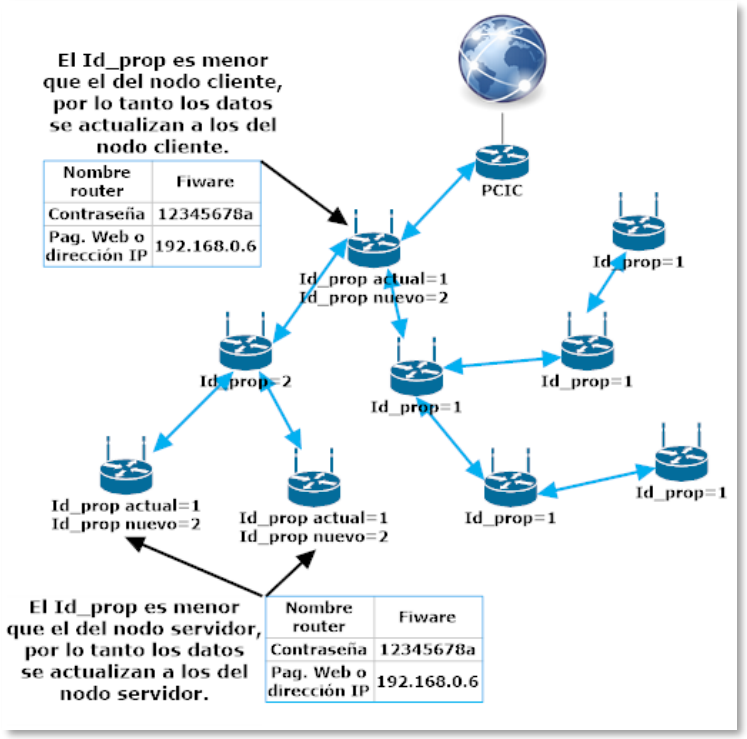
a)



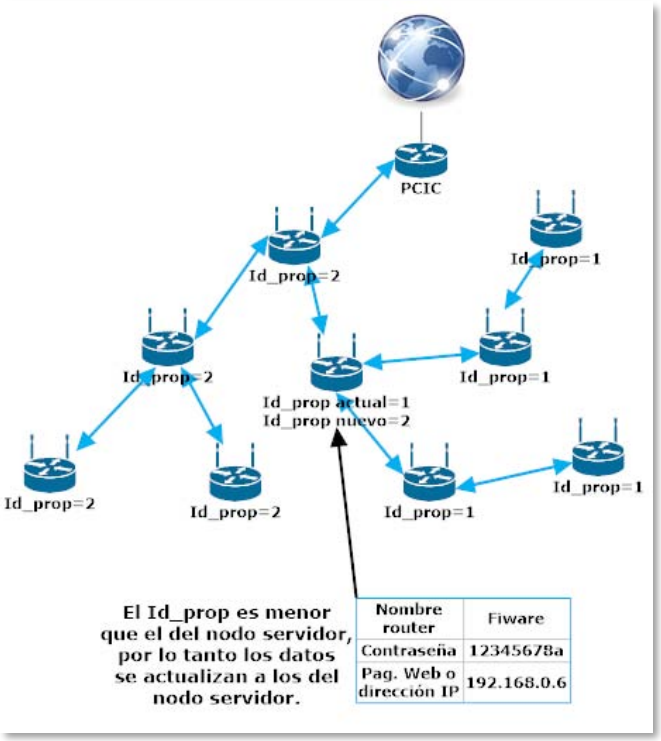
b)

Figura 21

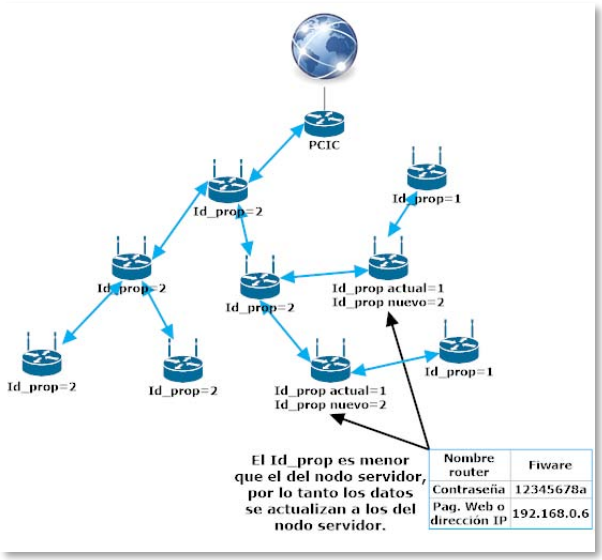
3. Los nodos cliente solicitan cada cierto tiempo (red proactiva) una petición de actualización de datos de router. Si el Id de propagación del cliente es menor, entonces los datos del cliente se actualizarán a los del nodo servidor, incluido el Id de propagación. Si dicho Id es igual, entonces el nodo cliente ignorará los datos provenientes del nodo servidor.
   Si el id del cliente es mayor que el Id de propagación del nodo servidor (el nodo cliente ha modificado sus datos y se ha elegido propagarlos), entonces el nodo servidor modificará sus datos del router a los del cliente, incluido el Id de propagación. Las figuras 22 a-d ejemplifican de manera más gráfica este paso.

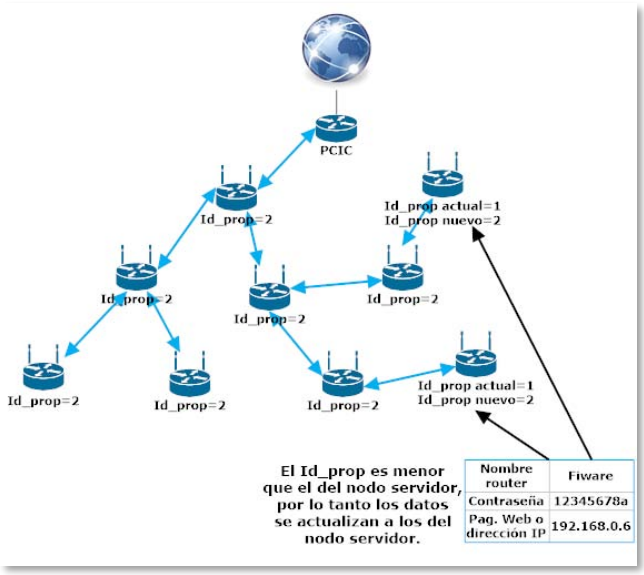


a)



b)





c) d)

Figura 22

4. En la figura 23 se puede ver que, si un nodo se desconecta de la red, su Id de propagación se ajusta a 0 para que, en caso de volver a conectarse a la red, automáticamente consiga los datos de router del nodo al que se haya conectado.

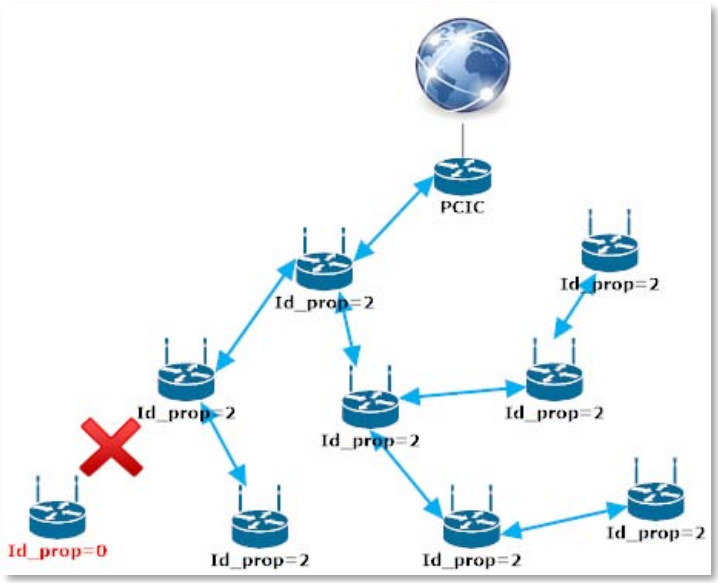


Figura 23

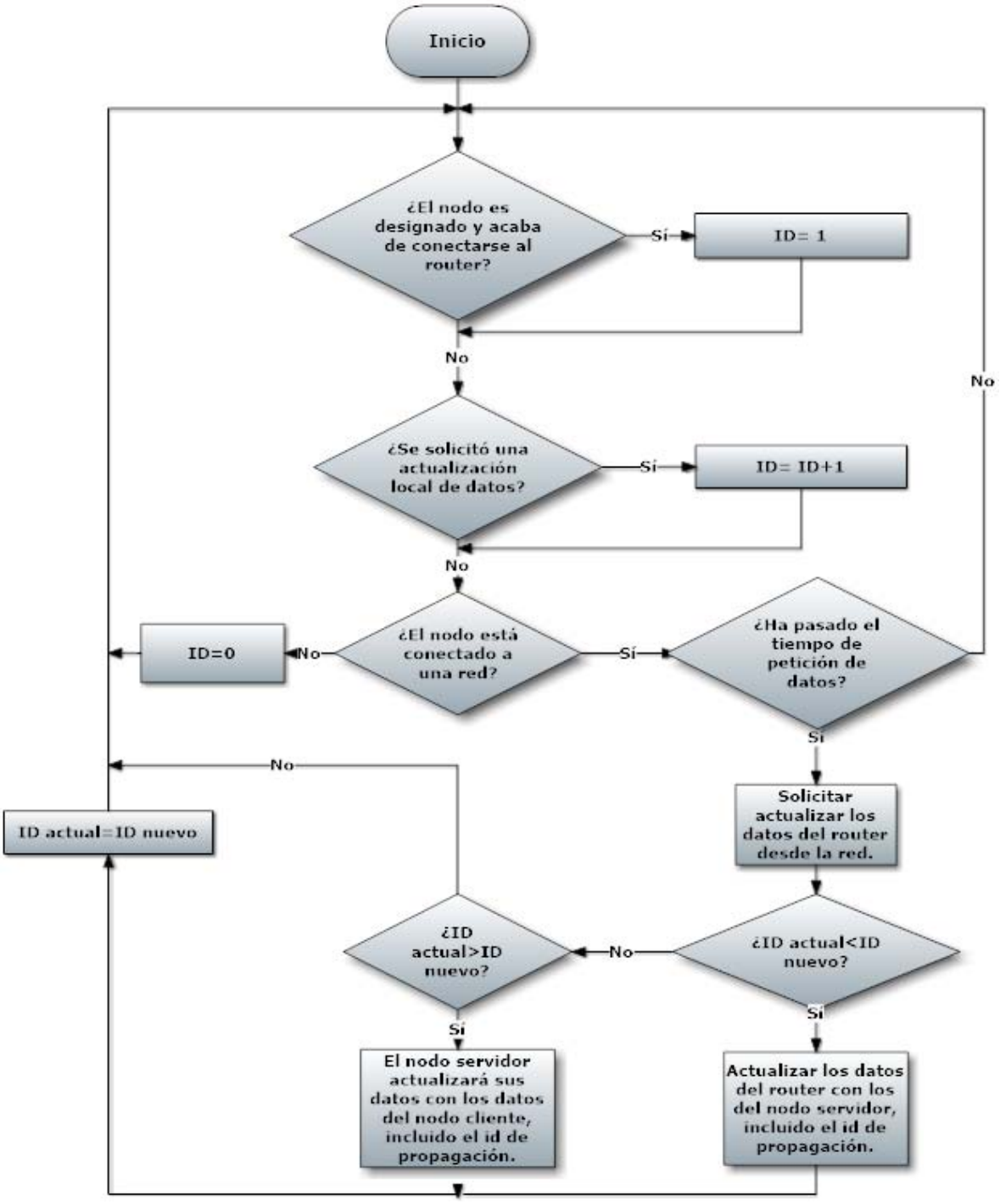
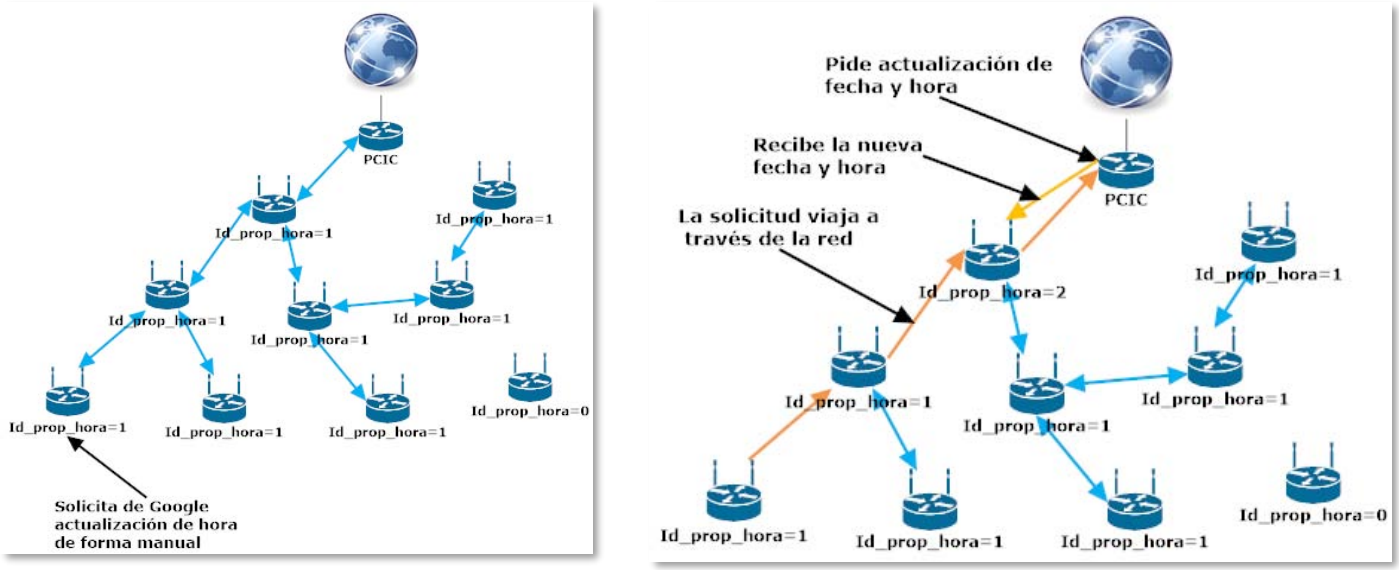


Diagrama 3. Algoritmo de Propagación de Datos de Router

4.3.2.3 *Algoritmo de Sincronización*

Se requiere que cada nodo esté sincronizado en la fecha y hora con el resto de los nodos, a lo que el nodo o nodos que fueron elegidos para acceder a internet obtendrán la hora del servidor de Google (Ver diagrama 4 para el diagrama de flujo). Los pasos a realizar son los siguientes:

1. Se obtiene la fecha y hora del servidor Google ya sea solicitándola cada cierto tiempo, de forma manual, cada vez que se encienda el nodo o cuando el integrado RTC no está corriendo. Esto lo pueden hacer tanto el nodo conectado al router como los nodos conectados a éste. La petición viaja por la red hasta llegar al nodo conectado a internet, el cual pide una actualización de fecha y hora e incrementa su Id de actualización de hora en 1 (ver figuras 24 a-b).



a) b)

Figura 24

2. Los nodos cliente piden la fecha y hora cada cierto tiempo de este u otros nodos servidores que ya hayan sido actualizados. Si el Id de actualización de hora del nodo cliente es menor que el del nodo servidor, entonces la fecha y hora se actualizará. La figura 25 explica de manera gráfica este paso.

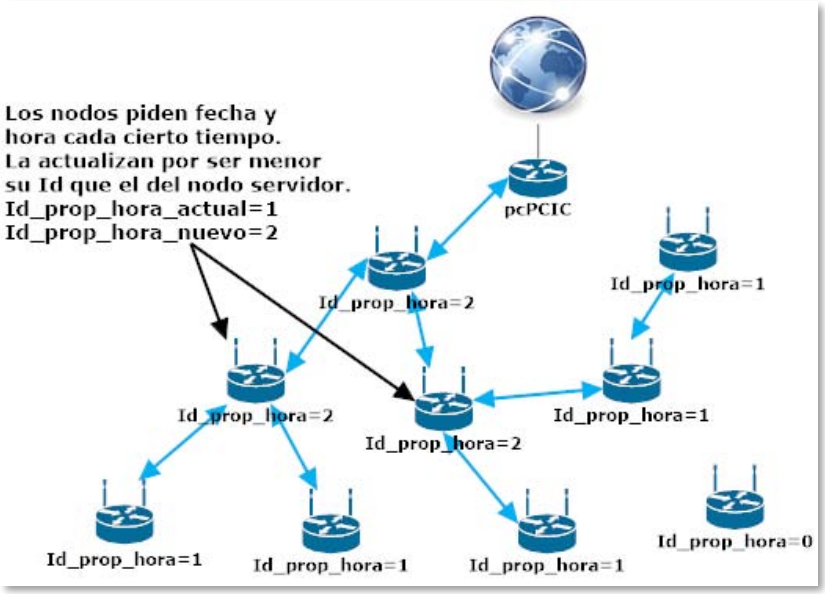


Fig. 25

3. En la figura 26 se observa que cuando un nodo se conecta a una red, pide la fecha y hora de otro nodo sin pedir petición de actualización al servidor de Google, siempre y cuando el RTC del otro nodo esté corriendo. Esto se sabe por medio de un identificador *running* que indica si el módulo RTC del otro nodo está corriendo.

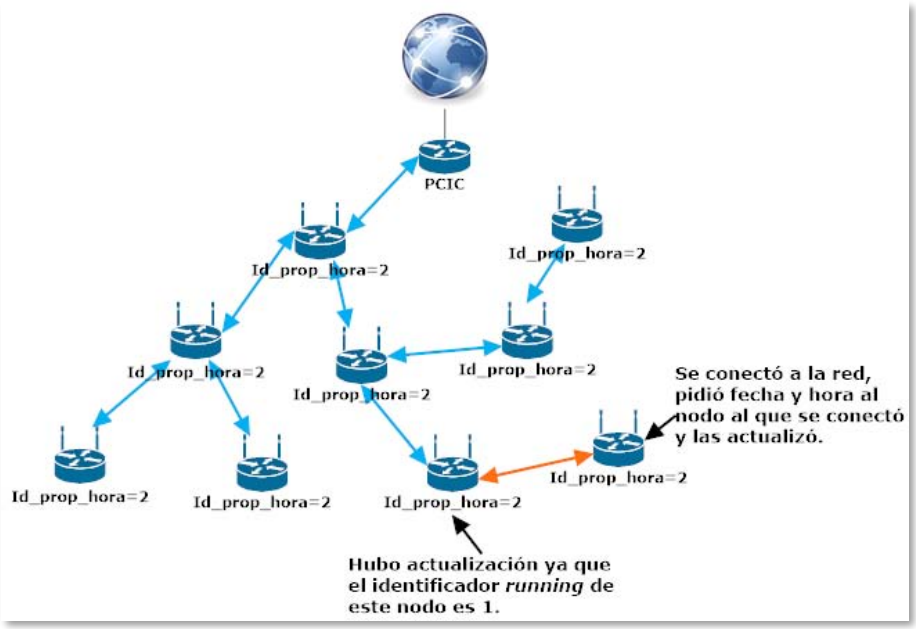


Figura 26

4. Si un nodo recibe un mensaje de sincronización y ya recibió con anterioridad otro mensaje con el mismo Id de actualización de hora, entonces el mensaje es descartado (ver figura 27).

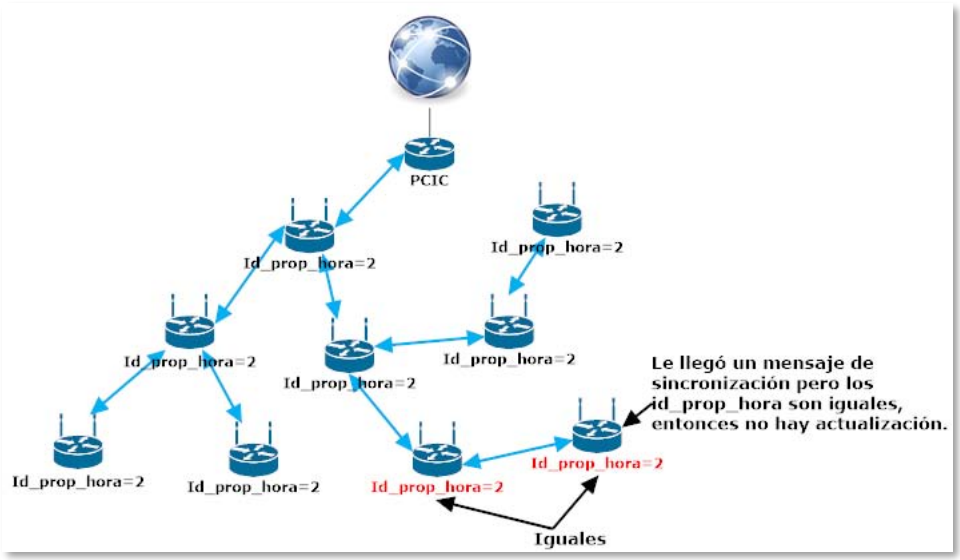


Figura 27

5. En la figura 28 se ve que, si el nodo se desconecta de la red, entonces el Id de actualización de hora se colocará en cero para actualizarse automáticamente si se vuelve a conectar a la red.

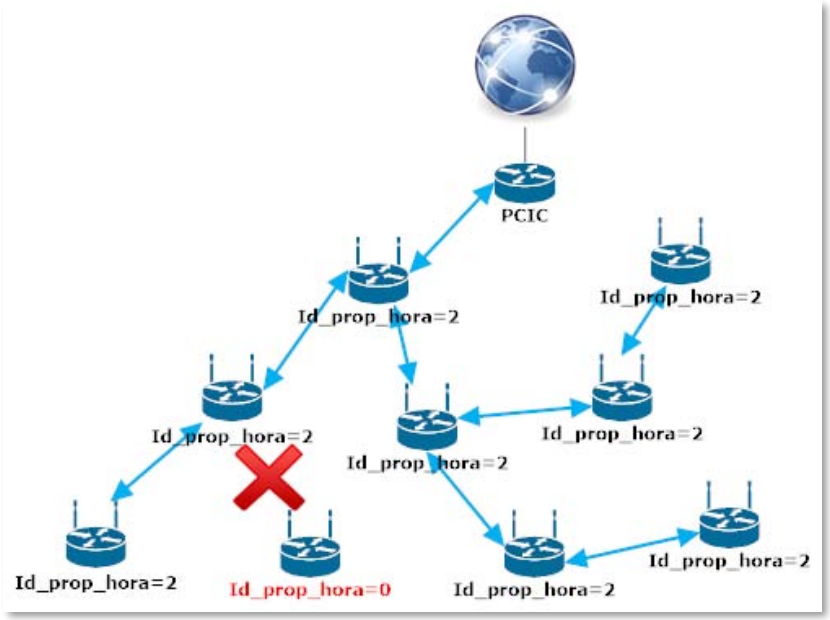


Figura 28

6. Si el Id de actualización de hora de los nodos llega a ser 3 y existe una nueva actualización, entonces el nuevo Id será 1 para hacer los Id de actualización de hora cíclicos (ver figura 29).

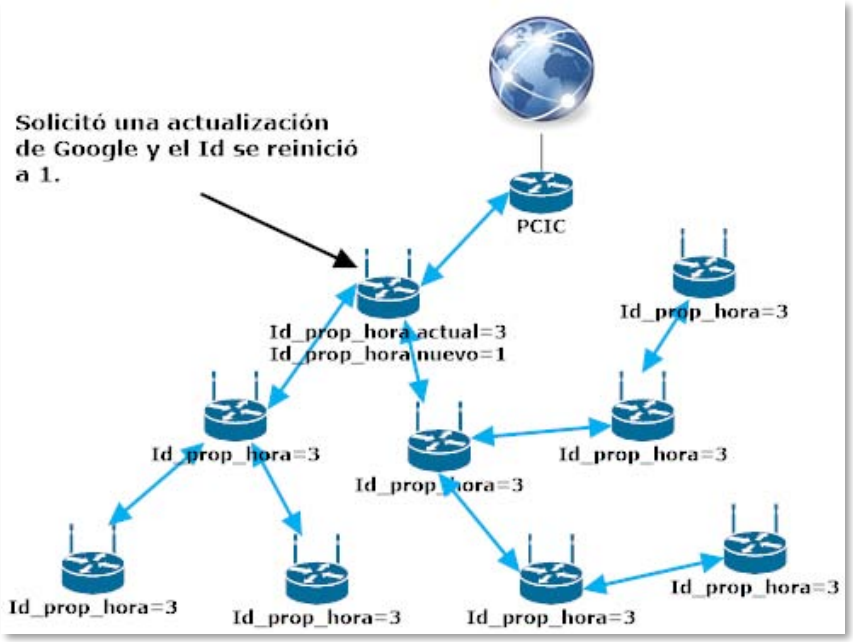


Figura 29

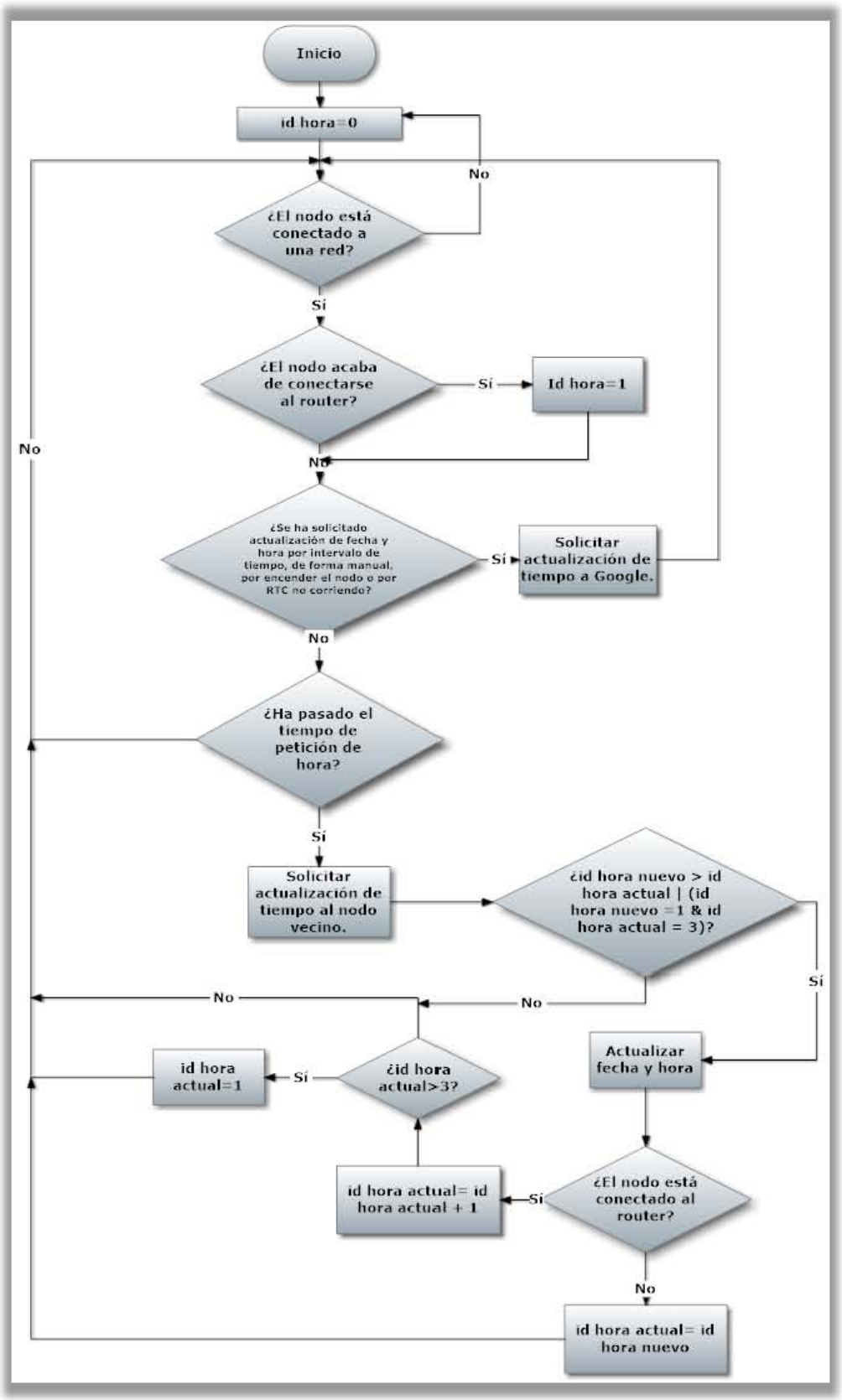


Diagrama 4. Algoritmo de sincronización

4.3.2.4 *Algoritmo de Transferencia de Datos*

Este algoritmo consiste en los pasos que describen cómo transmitir la información recabada de los sensores al servidor Apache a través de los nodos de la red Ad-hoc. En el diagrama 5 se puede observar el diagrama de flujo de este algoritmo.

1. El nodo enviará cada 11 segundos al nodo servidor que esté conectado lo que lleve recolectado en la memoria micro SD, ya sea de sus propios sensores o lo que haya recibido de su nodo cliente (si es que tiene). Se leen los índices de inicio y fin de paquetes como referencia del rango de los paquetes a enviar. Si al concluir el envío recibe con éxito todos los mensajes de confirmación de cada paquete, el índice de inicio de la memoria se actualiza para ser igual al índice de fin de paquetes. Si en el transcurso del envío no se recibe un mensaje de confirmación de algún paquete enviado, entonces se aplaza su envío hasta el siguiente envío de datos, es decir, se vuelve a intentar enviar después de que transcurran 11 segundos, y así hasta que los índices de inicio y fin sean iguales (ver figura 30 a-b).

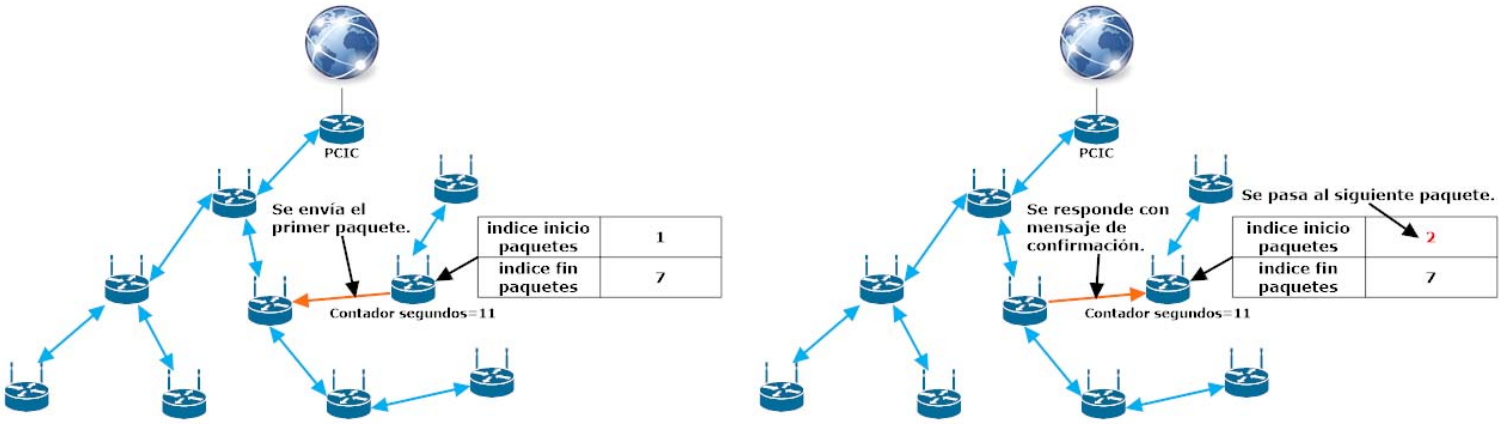


Figura 30

2. En la figura 31 se observa que, si el nodo está designado y está conectado al router, entonces los datos de los sensores deberá enviarlos al servidor Apache para su posterior graficación.

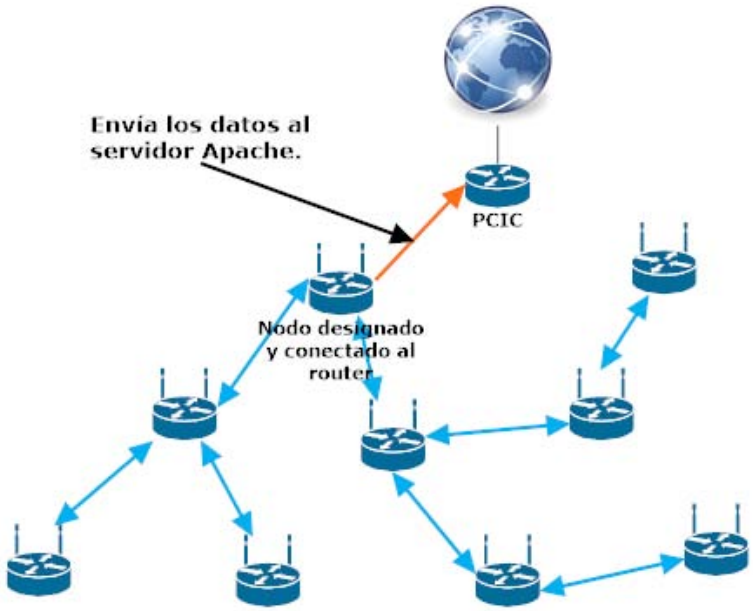


Figura 31

3. Si el nodo no está conectado a la red, los datos recolectados seguirán almacenándose en la memoria micro SD hasta que se conecte a una red de sensores (véase figura 32).

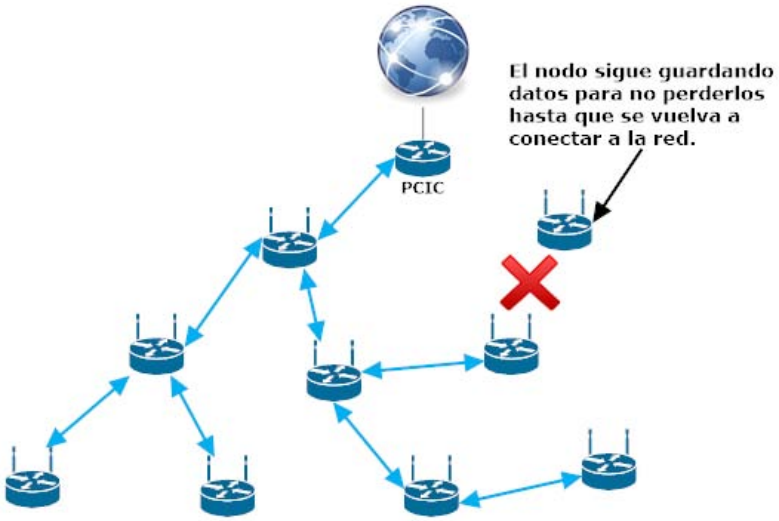


Figura 32

4. En la figura 33 se ve que, si el o los nodos designados están conectados al router, pero no poseen una memoria micro SD, los datos recolectados serán enviados cada 11 segundos al nodo servidor en un intento de rescatar dichos datos en ausencia de memoria. Si un nodo no posee memoria micro SD o no está conectado al router, entonces los datos recolectados se perderán hasta resolver uno o ambos problemas.

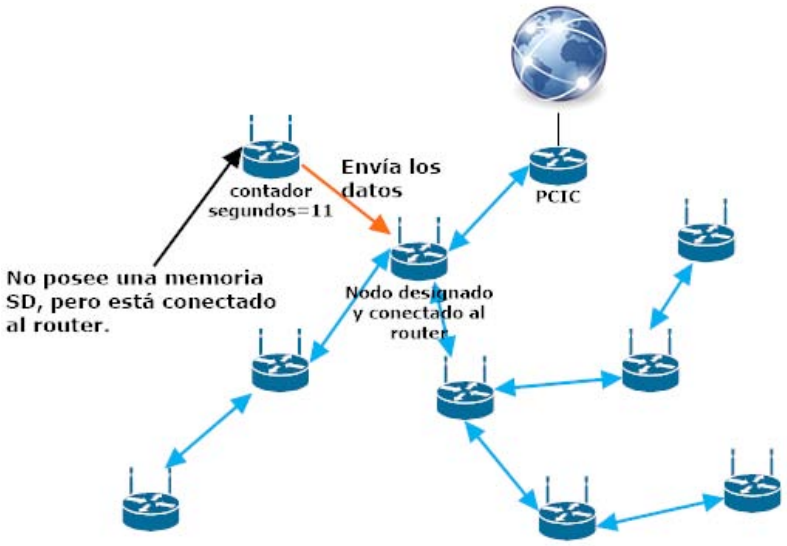


Figura 33

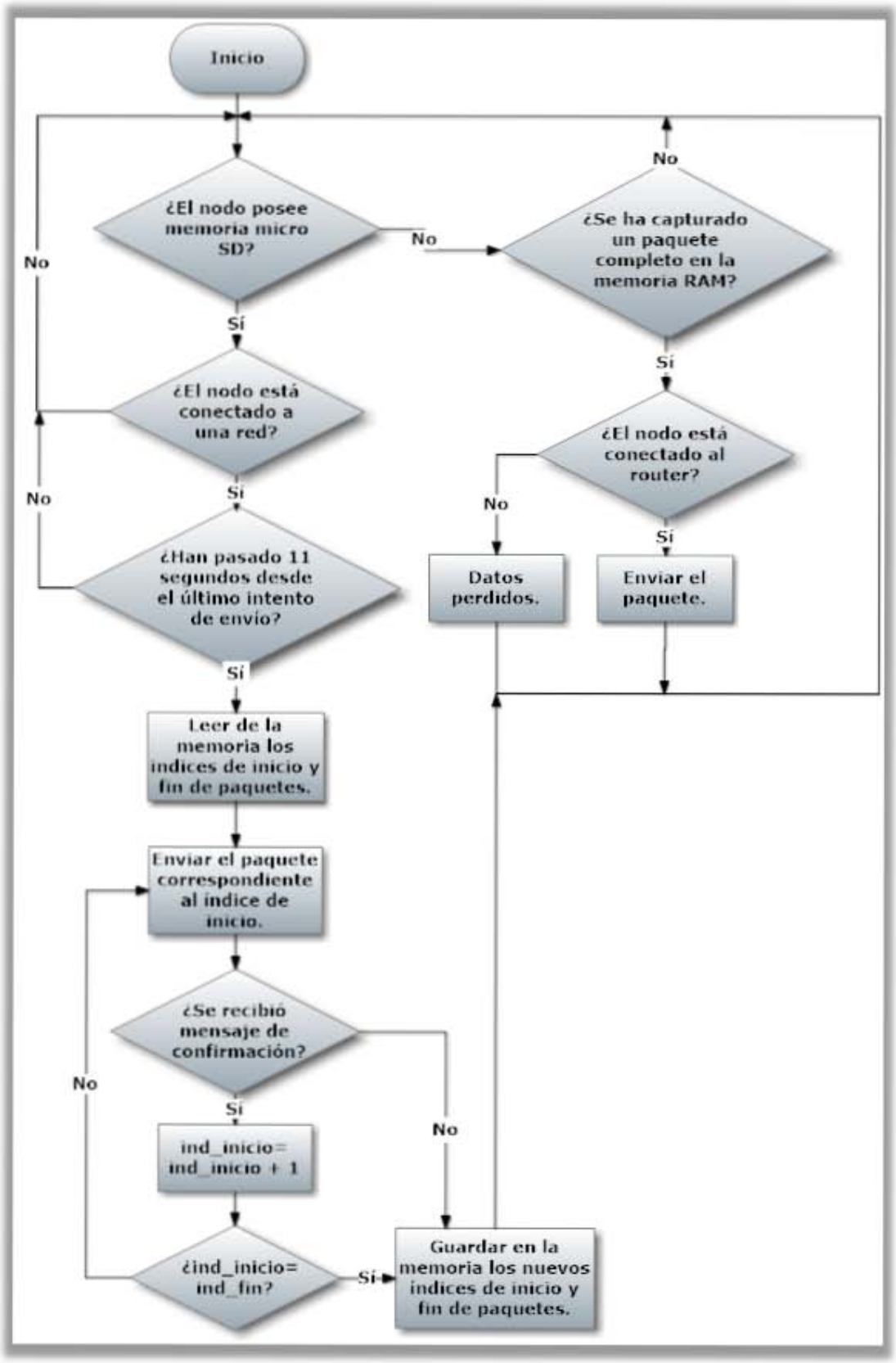


Diagrama 5. Algoritmo de transferencia de datos

4.3.3 Lenguaje de comunicación del protocolo

El protocolo PSFS posee un lenguaje propio para enviar y recibir mensajes de datos y de control. Este lenguaje se divide en mensajes de confirmación o router, mensajes de tiempo y mensajes de envío de datos capturados.

Nota: Los parámetros en *cursivas* indican valores variables. Las palabras en letra no cursiva son para identificar el tipo de mensaje, y son palabras fijas.

4.3.3.1 Mensajes de router

Estos mensajes sirven para propagar los datos del router como el SSID, la contraseña y la página web donde se alojarán los datos capturados de los sensores.

ACT_ROUTER,*Id_propagacion,SSID,página_web,contraseña,*: Este mensaje se utiliza para propagar cada dos segundos los datos del router de un nodo a su superior. Si el nodo superior tiene un Id de propagación menor al de este mensaje, entonces sus datos serán sustituidos por los de este mensaje. Si, tanto el Id del nodo emisor como el Id del nodo superior son iguales, o si el Id del nodo superior es mayor al de este mensaje, entonces el mensaje será ignorado. Este mensaje recibe como respuesta un mensaje ACK, el cual se describe a continuación.
ACK*,Id_propagacion,SSID,página_web,contraseña*. Es el mensaje de respuesta a ACT_ROUTER. Si el Id de propagación en este mensaje es mayor que el Id del nodo receptor, entonces se actualizarán los datos del nodo con los datos de este mensaje. Si, en cambio, el Id de propagación de este mensaje es menor o igual que el del nodo receptor, el mensaje será ignorado por los vecinos.

4.3.3.2 Mensajes de tiempo

Estos mensajes se utilizan para solicitar actualizaciones de fecha y hora, ya sea a los otros nodos o a un router conectado a internet.

TIME_REQ: Este mensaje solicita una petición de actualización de fecha y hora a otro nodo de acuerdo a su Id de hora actual. Este mensaje recibirá como respuesta **ACK,*Id_actualización_hora,fecha_y_hora*** o una cadena vacía. Si el Id de actualización de hora en el nodo es menor al recibido del otro nodo, entonces se actualizará la fecha y hora a los datos provenientes del otro nodo. Si el Id de hora actual es igual o mayor, los nuevos datos se ignorarán. Si la respuesta fue una cadena vacía, significa que hubo un problema de comunicación con el Arduino, y por lo tanto no habrá evaluación del Id de actualización de hora ni actualización de fecha y hora si fuese el caso.

TIME_INT: Este mensaje solicita una actualización de fecha y hora a Google. El mensaje se propaga desde el nodo solicitante hasta el router con internet, el cual devuelva la fecha y hora actualizados. Recibe como respuesta el mensaje **ACK.**

HEAD / HTTP/1.1 \r\n\r\n: Este mensaje lo envía el nodo conectado al router al servidor de Google para obtener la fecha y hora. Este mensaje es el resultado de la petición de actualización de la fecha y hora con el mensaje TIME_INT. La repuesta que obtiene del servidor Google contiene el prefijo **Date:**, seguido del día de la semana, la fecha y hora. Este mensaje se modifica de forma que los nodos puedan entenderlo, cambiando el prefijo **Date:** y el día de la semana por **TIME:date**, seguido de la fecha y hora.
Una vez hecho esto, el nodo conectado al router, al haber recibido la nueva fecha y hora, se encargará de propagar por toda la red esta nueva información, con la ayuda del mensaje **TIME_REQ**.

4.3.3.3 Mensajes de envío de datos capturados

Estos mensajes se utilizan para notificar a otro nodo o al router sobre el envío de datos capturados por los sensores.

ENVDATOS: ***Datos en formato de red Ad-hoc***: Este mensaje es para transmitir los datos capturados de los sensores de un nodo a otro, recibiendo como mensaje de respuesta a **ACK**, con lo cual se confirma que todos los datos se han enviado exitosamente al otro nodo. El formato de los datos de red Ad-hoc es de la forma: **Id de nodo,Fecha de captura,Hora de captura,ppm de CO_2,ppm de CO**, (por ejemplo, 6e3a27,01/06/2017,13:24:08,00000670,00000003). Si, por el contrario, el nodo receptor no recibió alguno de los datos, enviará al nodo emisor un mensaje **ERROR**.

ERROR: Este mensaje se envía en caso de no haber recibido correctamente todo o parte de un mensaje de datos capturados. Avisa de un error de envío de dichos datos, con lo cual el nodo sabe desde qué paquete de datos debe de iniciar la siguiente ocasión que deba enviar datos capturados.

4.3.4 Método de asignación de direcciones IP a los nodos

Los módulos ESP8266 poseen dos direcciones IP por diseño de hardware: una IP de softAP y una IP de estación.
La IP de softAP consiste en la dirección IP del modo que funciona como Soft Access Point, es decir, como un punto de acceso portátil al cual se pueden conectar otros dispositivos con WiFi, incluidos otros módulos ESP8266. Se manejan dos tipos de conexiones en este modo: la conexión al softAP y la conexión al servidor. Para conectarse al servidor del módulo ESP, primero se debe conectar al softAP. Si el módulo se configura como servidor,

puede tener hasta 4 clientes conectados a dicho servidor, pero al softAP se pueden conectar una cantidad de nodos equivalentes al número de direcciones IP que posee la tabla del protocolo DHCP habilitado en el softAP.
La IP de la estación es la dirección IP del modo del ESP que funciona como estación, es decir, como un nodo esclavo de un nodo softAP. Esta IP, así como su dirección Gateway y máscara de subred, se pueden asignar de forma manual (estática) o de manera automática, por medio del protocolo DHCP de un AP o softAP con dicho protocolo habilitado.

Cuando se habilitan ambos modos en el módulo ESP, ambas direcciones IP están activadas y no pueden pertenecer a la misma subred, a lo que se deben de ajustar para que no se traslapen entre sí de acuerdo a un método implementado para tal propósito, descrito a continuación.

4.3.4.1 Ajuste de la dirección IP softAP

La dirección IP softAP se obtiene con base en la dirección MAC del modo softAP, utilizando el ChipId, el cual es un identificador de unicidad para los módulos ESP8266. El ChipId sólo utiliza los últimos tres bytes de la dirección MAC del softAP.

Al ChipId se le hace un acarreo de 8 bits a la izquierda para que dé lugar a una subred de 8 bits o 254 direcciones utilizables. La fórmula usada para calcular el número de direcciones utilizables es:

$$\text{No. direcciones utilizables} = 2^{\text{bits subred}}-2 = 2^{8}-2 = 254$$

Lo siguiente es convertir este valor en octetos para ser asignados como dirección IP. Esto se realiza de esta forma:

- Primer octeto= acarreo de 24 bits a ChipId_acarreado y operación AND con 255.
- Segundo octeto= acarreo de 16 bits a ChipId _acarreado y operación AND con 255.
- Tercer octeto= acarreo de 8 bits a ChipId _acarreado y operación AND con 255.
- Cuarto octeto= 1. El 1 es para asignarle al softAP la primera dirección utilizable de la subred.

Estos octetos dan como resultado a la dirección IP softAP basada en los últimos 3 bytes de la dirección MAC.
Esta dirección IP ya está ajustada para permitir asignar hasta 253 direcciones IP a los nodos estación de acuerdo al acarreo a la izquierda de 8 bits. Con este acarreo también se evita el traslape de direcciones IP entre el modo estación y softAP. Es importante recalcar que en el ESP8266 no se pueden traslapar las dos subredes de softAP y de estación, de lo contrario, el nodo no se podrá comunicar con otros nodos o con el router.

La máscara de subred debe coincidir con el número de bits acarreados de la dirección IP de softAP, que en este caso fueron 8, quedando así: 255.255.255. 0

4.3.4.2 Ajuste de evasión de traslape de IPs

Existe una pequeña probabilidad de que la dirección softAP del módulo ESP conectado al router se traslape con la dirección IP de estación asignada por el router, es decir, que ambas subredes se traslapen. Tomando en cuenta este riesgo, se implementó una forma de evitar este traslape.
Tomemos como ejemplo la dirección softAP 192.168.0.65 con máscara de subred 255.255.255.0 y la dirección de estación 192.168.0.15 con máscara de subred 255.255.255.0 de un módulo conectado a un router.

El primer paso es realizar una operación AND a ambas direcciones con la máscara de subred con menor valor, la cual en este caso es cualquiera de las dos máscaras por ser iguales, es decir, 255.255.255.0. Se comparan ambos resultados, y si son iguales, se procede con el siguiente paso.

Dirección softAP: 192.168.0.65 & 255.255.255.0 =**192.168.0.0**

Dirección estación: 192.168.0.15 & 255.255.255.0 =**192.168.0.0**

Ambos resultados son iguales

El segundo paso consiste en hacer una operación AND a la dirección softAP con la máscara de subred con menor valor, después se le suma la máscara de subred con menor valor negada más 2, quedando la operación así: IP_softAP = IP_softAP & máscara_menor_val + ~ máscara_menor_val + 2.

La nueva dirección de softAP 192.168.1.1 esquiva a la subred del router, con lo cual se evita que se traslape con la dirección de estación 192.168.0.15 y provoque un error de comunicación hacia el router o hacia otros nodos.

4.3.5 Implementación de seguridad

Existe cierto riesgo de que un intruso pueda acceder a uno de los nodos de la red y, por lo tanto, sea capaz de desviar los datos recolectados para que nunca lleguen a su destino. Para evitar este tipo de problema, se optó por habilitar el cifrado de seguridad que viene implementado en los módulos ESP8266, eligiendo el protocolo WPA2-PSK.

4.3.5.1 Mecanismo de seguridad para evitar que intrusos aparenten ser nodos

Se implementó otro mecanismo de seguridad para evitar que los nodos se conecten a dispositivos que no sean ESP8266, el cual consiste en revisar las direcciones MAC de los dispositivos cercanos al nodo. Si las direcciones MAC comienzan con 5E:CF:7F, (que es la parte de la dirección MAC refiriéndose al Id de fábrica de los ESP8266) entonces el nodo las tomará en cuenta para el cálculo del nodo más cercano, de lo contrario, cualquier otra dirección MAC será ignorada. Con este mecanismo se evita que un intruso confunda a los nodos, causando que éstos intenten conectarse al intruso, inhabilitando la red.

4.4 Implementación de temas no relacionados con redes

4.4.1 Calibración de los sensores de CO y CO_2

Los sensores utilizados en el proyecto miden concentraciones de dióxido de carbono (CO_2) y monóxido de carbono (CO), siendo componentes activos (requieren alimentación propia para funcionar) y devolviendo un voltaje de salida analógico, el cual mientras sea menor, menor será la concentración del gas medido en partes por millón (ppm), y mientras el voltaje sea mayor, más será la concentración de dicho gas.
Hay un punto a tomar en cuenta en cuanto a la interpretación de los sensores de gases, y consiste en que la concentración de gas en ppm no corresponde de forma lineal con el voltaje de salida del sensor, sino que la función que los enlaza es potencial, a lo que se requiere calibrar los sensores de acuerdo a una función exponencial. Sin embargo, en los datasheet de los sensores sólo se brinda una gráfica logarítmica de **ppm** versus **Rs/Ro**; estos parámetros se explicarán más adelante. Lo que se debe hacer es, por lo tanto, construir una gráfica lineal con los puntos que entregue cada gráfica del datasheet, y luego hacer una regresión logarítmica para así obtener la función que modele a la relación potencial entre el voltaje y la concentración.

4.4.1.1 Sensor de CO_2

Se utilizó la regresión logarítmica, ya que es la regresión que mejor se adecúa a los puntos de muestreo de la gráfica que nos brinda el datasheet (se adecúa mejor que la regresión lineal, exponencial, potencial o polinómica). La figura 34 nos proporciona la gráfica logarítmica de relación R_S/R_0 vs ppm del CO_2, basada en el datasheet del MQ-135[29].

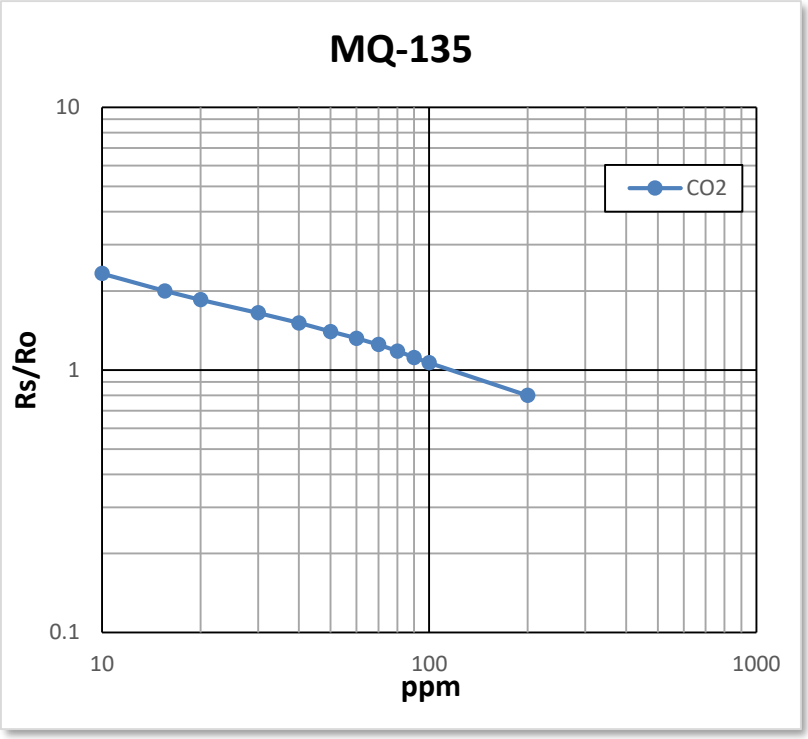


Figura 34

Estos datos los obtuvo el fabricante con una temperatura de 20 °C, a una humedad de 65% y con una concentración de oxigeno de 21%, la cual es la concentración promedio en la atmósfera terrestre. Los datos de interés para calibrar el sensor son los del dióxido de carbono, a lo que se capturaron en una gráfica lineal; también se agregó una línea de regresión logarítmica para obtener así un patrón de referencia de los datos capturados del datasheet y, por lo tanto, la función potencial de la relación R_S/R_0 vs ppm. La figura 35 nos muestra la gráfica resultante.

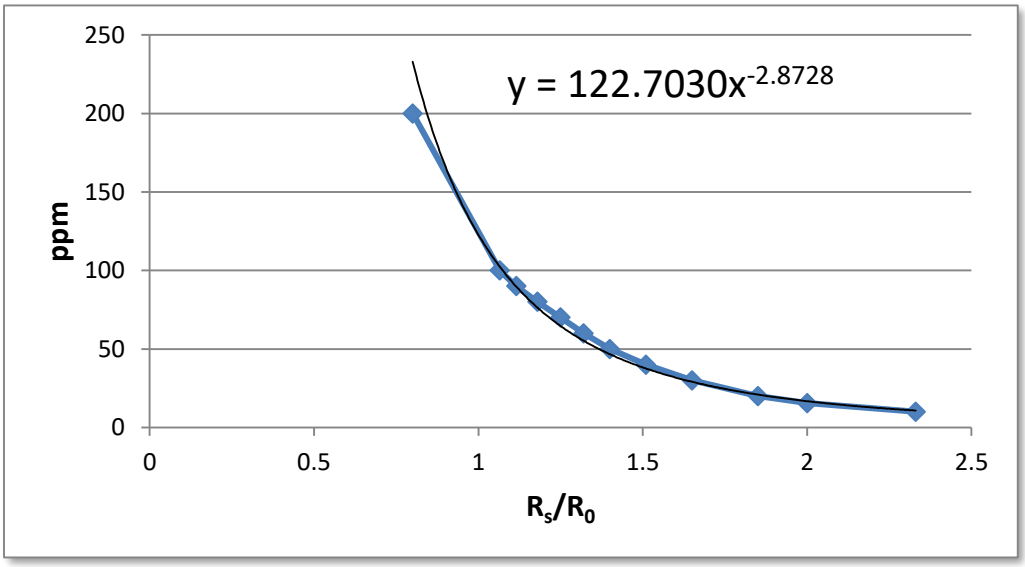


Figura 35. Gráfica lineal de relación R_S/R_0 vs ppm del CO_2.

La función resultante de *R_S/R_0 vs ppm* es entonces:

$$\text{ppm} = 122.703\left(\frac{R_s}{R_0}\right)^{-2.8728} \qquad (1)$$

Sin embargo, todavía no se puede ver la relación directa del voltaje leído con la concentración en ppm. Para esto hay que conocer el significado de R_s y R_0.
R_s es la resistencia resultante del sensor de medir diversas concentraciones de gases, tomando en cuenta el voltaje analógico leído y la resistencia de carga R_L, que en general en los sensores de medición de gases es de 1KΩ. La ecuación del voltaje leído es de la siguiente forma:

$$v_{leído} = \frac{5 * R_L}{R_S + R_L}$$

La cual, al despejar R_S, queda de la siguiente manera:

$$R_S = R_L\left(\frac{5}{v_{leído}} - 1\right) \qquad (2)$$

donde 5 es el voltaje de alimentación del sensor, el voltaje que brinda el Arduino.
R_0 es la resistencia del sensor medida en laboratorio, usando una muestra de concentración de CO_2 conocida. Para fines de simplicidad y dada la falta de instrumentos de medición de concentración de gases como referencias, se utilizó la concentración promedio de CO_2 existente en la atmósfera terrestre, la cual es de 407.05 ppm[30] a junio de 2017. Despejando de (1) a R_0 se obtiene:

$$R_0 = \frac{R_s}{\sqrt[-2.8728]{\frac{\text{ppm}}{122.703}}} = R_s * \sqrt[2.8728]{\frac{\text{ppm}}{122.703}} \qquad (3)$$

Si sustituimos en (2) las variables con R_L= 1KΩ y $v_{leído}$=0.36 V (es el voltaje leído del sensor con una concentración de CO2 de 407.05 ppm), entonces se obtiene lo siguiente:

$$R_S = 1000\left(\frac{5}{0.36} - 1\right) = 12888.8889\ \Omega$$

Ahora, si en (3) sustituimos las variables con R_s=12888.8889 Ω y ppm=407.05, el resultado será:

$$R_0 = 12888.8889 * \sqrt[2.8728]{\frac{407.05}{122.703}} = 19565.88272\ \Omega$$

Ya se conoce ahora el valor de R_0 de acuerdo a la calibración efectuada con la concentración de CO_2 en el ambiente. Lo siguiente es sustituir a (2) en (1), obteniendo la siguiente función:

$$\text{ppm} = 122.703 * \left(\frac{R_L}{R_0}\left(\frac{5}{v_{\text{leído}}} - 1\right)\right)^{-2.8728} \qquad (4)$$

Se sustituyen los valores R_L y R_0 con 1000 Ω y 19565.88272 Ω en esta función, obteniendo:

$$\text{ppm} = 122.703 * \left(\frac{1000}{19565.88272}\left(\frac{5}{v_{\text{leído}}} - 1\right)\right)^{-2.8728}$$

Estos cálculos con números reales se pueden realizar afortunadamente con el Arduino Mega, ya que su hardware permite realizar operaciones complejas en punto flotante como potencias (ya sean positivas, negativas o fraccionarias). Con microcontroladores más sencillos no se hubiese podido realizar, ya que solamente trabajan con punto fijo, imposibilitando este tipo de operaciones.

Otro punto a tomar en cuenta es que un microcontrolador tiene un límite de resolución para interpretar y digitalizar los valores de voltaje que devuelve el sensor. En el caso del Arduino Mega, la resolución está definida por 10 bits, es decir, 1024 valores. Así, el voltaje digitalizado está definido por las siguientes funciones, donde el 5 es el voltaje de referencia del microcontrolador:

$$\text{valor\_leido\_en\_10bits} = \text{redondear}\left(\frac{V_{\text{analógico}} * 1023}{5}\right)$$

$$V_{\text{digital}} = \frac{\text{valor\_leido\_en\_10bits} * 5}{1023} \qquad (5)$$

Así, como ejemplo, si el sensor devuelve un voltaje de 1.02 V, el valor digital se obtendrá así:

$$V_{\text{digital}} = \frac{\text{redondear}\left(\frac{1.02 * 1023}{5}\right) * 5}{1023} = \frac{209 * 5}{1023} = 1.0215 \text{ V}$$

Y este valor se tomará en cuenta para el cálculo de las ppm, en lugar del valor de 1.02 V, sustituyéndose en (4):

$$\text{ppm} = 122.703 * \left(\frac{1000}{19565.88272}\left(\frac{5}{1.0215} - 1\right)\right)^{-2.8728} = 12668.96094 \text{ ppm}$$

Para agregar las funciones al sketch de Arduino tomando en cuenta el valor leído en 10 bits, se tuvo que escribir la función sustituyendo en (2) utilizando a (5) el $V_{digital}$ en el $V_{leído}$:

$$R_S = R_L\left(\frac{5}{\frac{valor\_leido\_en\_10bits * 5}{1023}} - 1\right)$$

$$R_S = 1000 * \left(\frac{1023}{valor\_leido\_en\_10bits} - 1\right) \qquad (6)$$

Una vez calculado R_s, se sustituye en (1) junto con el valor obtenido de R_0:

$$ppm = 122.703\left(\frac{R_s}{19565.88272}\right)^{-2.8728} \qquad (7)$$

Quedando las operaciones en código de Arduino de esta manera:

```
float Rs = 1000 * (1023.0 /analogRead(A0) - 1);
ppm[0] = 122.703*pow(Rs / 19565.88272, -2.8728);
```

Tanto Rs como ppm[0] son números declarados como flotantes.

4.4.1.2 Sensor de CO

Lo siguiente es calibrar es el sensor de CO, lo cual será más sencillo de explicar, ya que es el mismo método que el que se utilizó con el sensor de CO_2. Para esto se utilizó la gráfica de sensibilidad a diversas concentraciones del sensor MQ-7 obtenida de su datasheet. La figura 36 nos proporciona la gráfica logarítmica de relación R_S/R_0 vs ppm del CO_2, basada en el datasheet del MQ-7[31].

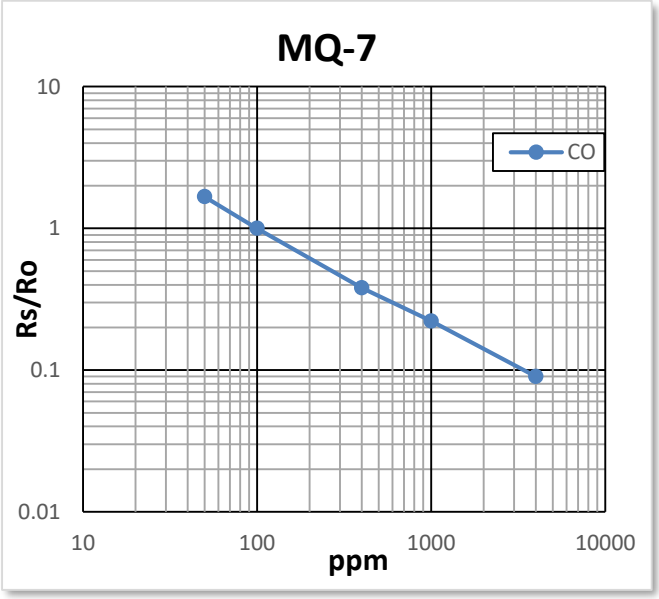


Figura 36. Gráfica logarítmica de relación R_S/R_0 vs ppm del CO, basada en el datasheet del MQ-7.

Estos datos los obtuvo el fabricante con una temperatura de 20 °C, a una humedad de 65% y con una concentración de oxigeno de 21%, la cual es la concentración promedio en la atmósfera terrestre. Como en el caso anterior, se hizo una gráfica lineal extrayendo los puntos de concentración de CO, y se agregó una línea de regresión logarítmica para observar su coincidencia con los puntos de muestreo. Finalmente se obtuvo la función potencial de la relación R_S/R_0 vs ppm del CO (véase figura 37).

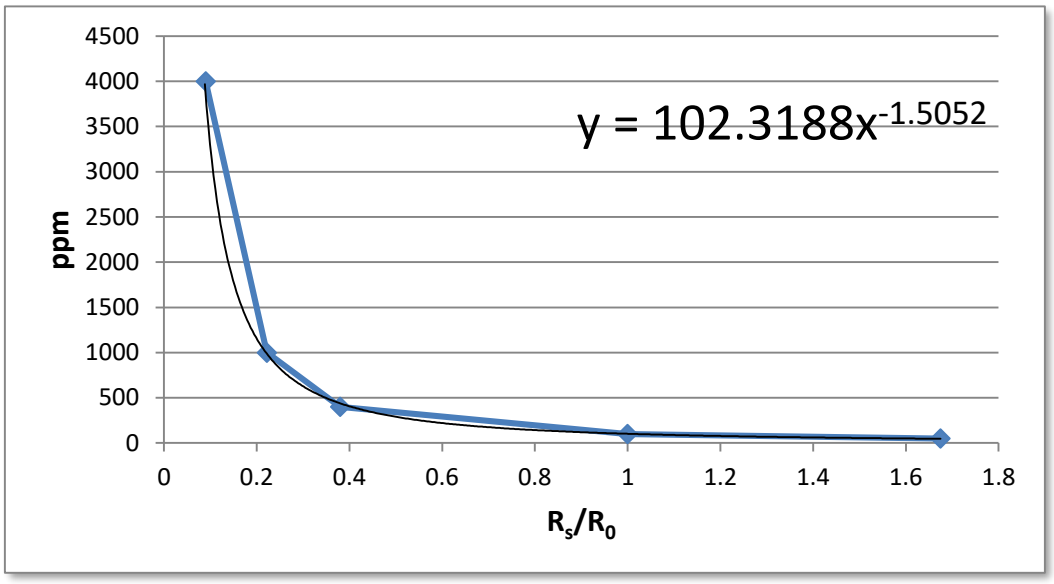


Figura 37. Gráfica lineal de relación R_S/R_0 vs ppm del CO.

La función resultante de *R_S/R_0 vs ppm* es entonces:

$$\text{ppm} = 102.3188\left(\frac{R_S}{R_0}\right)^{-1.5052} \qquad (8)$$

Se utilizan las ecuaciones (2) y (3) para hallar a R_s y R_0 con una concentración ambiente promedio de 0.1 ppm del CO según lo estimado en estudios[32], lo cual devuelve un voltaje de 0.44 V:

$$R_S = 1000\left(\frac{5}{0.44} - 1\right) = 10363.63636\ \Omega$$

$$R_0 = 10363.63636 * \sqrt[1.5052]{\frac{0.1}{102.3188}} = 103.7068676\ \Omega$$

Se sustituyen los valores R_L y R_0 con 1000 Ω y 103.7068676 Ω en (8), obteniendo:

$$\text{ppm} = 102.3188*\left(\frac{1000}{103.7068676}\left(\frac{5}{V_{\text{leído}}} - 1\right)\right)^{-1.5052}$$

Para agregar las funciones al sketch de Arduino, se tuvieron que utilizar las ecuaciones (6) y (7) para obtener el R_s y las ppm del sensor de CO:

$$R_S = 1000 * (\frac{1023}{\text{valor\_leido\_en\_10bits}} - 1)$$

$$\text{ppm} = 102.3188(\frac{R_s}{103.7068676})^{-1.5052}$$

Quedando las operaciones en código de Arduino de esta manera:

```
Rs = 1000 * (1023.0 / analogRead(A1) - 1);
ppm[1] = 102.3188*pow(Rs / 103.7068676, -1.5052);
```

Algo importante a tomar en cuenta sobre ambos sensores es su confiabilidad de precisión, en la cual los fabricantes sugieren encender por lo menos por media hora los sensores para que se calienten lo suficiente y devuelvan resultados más precisos y confiables, ya que experimentando se observó que los sensores, cuando no se calientan lo suficiente, pueden devolver mediciones imprecisas.

Tanto la pantalla LCD como los datos recolectados mostrarán a las ppm como números enteros, ya que para el fin del proyecto no se requiere utilizar números decimales, y facilita su visualización tanto en la pantalla LCD como en el servidor.

Capítulo 5: Resultados

En este capítulo se comentarán los resultados de la experimentación con la implementación del protocolo PSFS. Se midieron parámetros como el rendimiento en el tiempo de envío de datos de un nodo a otro, la cantidad de envío de datos por segundo, el tiempo de convergencia y la tasa de error.

En la experimentación se probó el protocolo PSFS con 3 nodos solamente (ver figura 38), dejando para trabajo futuro el poder implementar el protocolo con más nodos y en un ambiente real más amplio. Los parámetros se midieron en un entorno donde los nodos estuviesen fijos, a lo que el parámetro como la velocidad de los nodos se descarta.

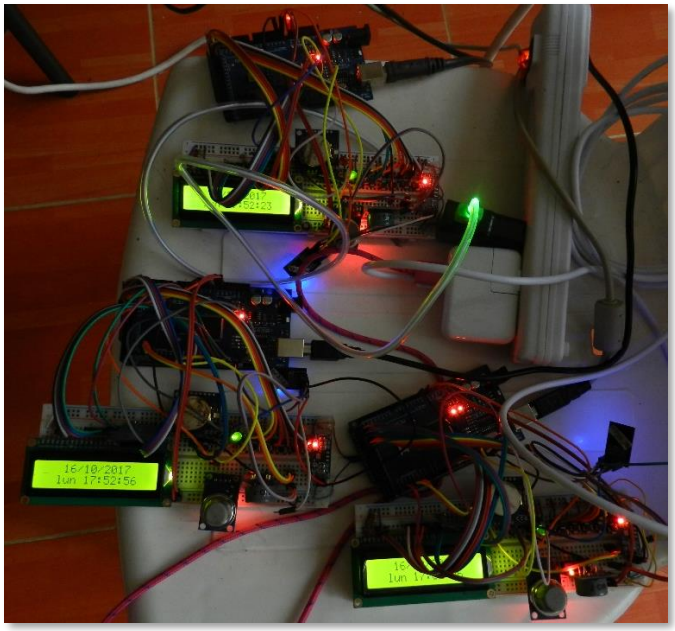


Figura 38. Tres nodos implementados y funcionando.

El primer punto a tomar en cuenta será el del rendimiento del tiempo en el envío de datos de un nodo a otro, el cual se define y limita por la velocidad de lectura/escritura de la memoria micro SD, la velocidad de procesamiento del Arduino Mega, la tasa de transferencia de los puertos seriales y la tasa de transferencia de datos del módulo ESP8266. La velocidad mínima de escritura de una memoria micro SD clase 4 es de 4 MB/s en un procesador con una frecuencia de reloj igual o mayor a 1GHz, alcanzando velocidades de escritura promedio de 5 MB/s (ver figura 39), pero dada la frecuencia de operación del procesador del Arduino Mega, la cual es de 16MHz, la velocidad de escritura máxima de la memoria micro SD es de aproximadamente 248.82 kB/s de acuerdo a una

prueba de escritura (ver figura 40). Se dividió al número de kilobytes escritos entre el tiempo de escritura.

Figura 39. Velocidad promedio de una memoria micro SD clase 4 en una computadora, de acuerdo a una prueba de 4 lecturas y 4 escrituras, cuyos resultados aparecen en verde. Los promedios de lectura y escritura aparecen en blanco.

```
Bytes escritos: 254464
Kilobytes escritos: 249
Tiempo transcurrido: 1000704 microsegundos
```

Figura 40. Resultados obtenidos de escribir datos en una memoria micro SD clase 4 en un Arduino Mega.

La tasa de transferencia más alta entre el Arduino Mega y el ESP8266 por puerto serial obtenida por experimentación y que sea compatible con ambos circuitos es de 950000 bits por segundo, o 118.75 kB/s.
La velocidad de transferencia del protocolo WiFi u 802.11g implementado en el módulo ESP8266 es de 45 Mbps o 5.6 MB/s en promedio.
Todas estas características se deben tomar en cuenta para poder comprender el porqué de los resultados de velocidad de transmisión de datos de un nodo a otro, ya que, de las velocidades mencionadas, la menor de todas es la que limitará la velocidad de transmisión de datos. En este caso, el limitante de dicha velocidad es la velocidad de transferencia por el puerto serial entre el Arduino Mega y el ESP8266, a lo que la velocidad de transferencia de un nodo a otro será de **118.75 kB/s**. Esto es el equivalente a enviar de un nodo a otro o al servidor un aproximado de 239 paquetes de datos o 2629 datos capturados por segundo, incluidos en los paquetes el Id del nodo, la fecha, la hora y las concentraciones de CO_2 y CO en ppm.

Otro parámetro medido fue el tiempo de convergencia de los nodos, el cual depende del número de nodos existentes en la red; mientras más nodos se conecten, más tiempo tardará la red en converger. Un nodo tarda aproximadamente 7 segundos en conectarse a otro nodo o a un router, por lo que el tiempo de convergencia se mide en los niveles de saltos existentes en la red. Cada nivel de salto posee un número de salto diferente,

colocándose del nivel 1 en un salto, el nivel 2 en dos saltos y así sucesivamente hasta alcanzar el *n* nivel. Entonces, el tiempo de convergencia (en segundos) se puede calcular como:

$$t_c = t_1 + t_2 * (n-1)$$

donde:

- t_c es el tiempo de convergencia
- t_1 es el tiempo de conexión entre el nodo designado y el router, generalmente 5 segundos
- t_2 es el tiempo de conexión entre nodos, generalmente 7 segundos
- *n* es el número de niveles en la red

En la figura 41 se puede apreciar un ejemplo para calcular el tiempo de convergencia para una red con 5 niveles.

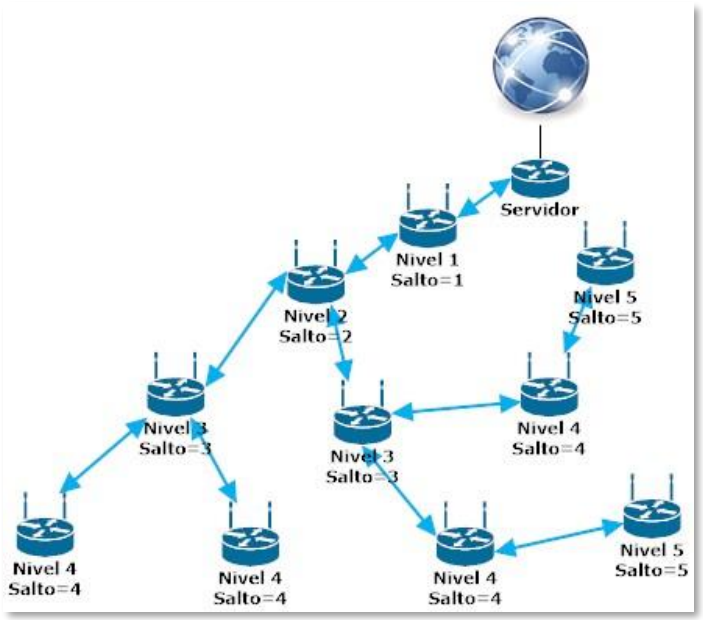


Figura 41. Red con 5 niveles. Tiempo de convergencia: $t_c = 5+7*(4) = 33$ segundos

Un punto a tomar en cuenta es cuando un nodo se desconecta de otro, lo cual causa que sus nodos subordinados se desconecten también como una reacción en cadena, y provoca que exista un tiempo de desconexión de estos nodos, el cual es de aproximadamente 2 segundos de acuerdo a experimentaciones con el módulo Wi-Fi.

En la sección de la red en la que haya acontecido una desconexión, va a existir un mayor tiempo de convergencia al momento de volver a conectar esa sección a la red ya conectada, dependiendo tanto del tiempo de conexión como del tiempo de desconexión, quedando el cálculo como:

$$t_c=(t_2+t_3) *n$$

donde:

- t_c es el tiempo de convergencia
- t_2 es el tiempo de conexión entre nodos, generalmente 7 segundos
- t_3 es el tiempo de desconexión entre nodos, generalmente 2 segundos
- n es el número de niveles de la sección desconectada

En la figura 42 se puede observar un ejemplo para calcular el tiempo de convergencia de una sección de red desconectada de 3 niveles.

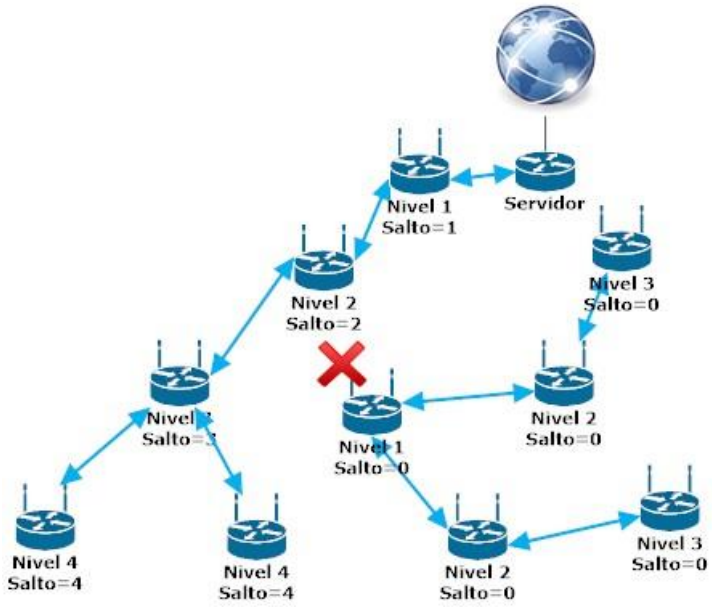


Figura 42. Sección con 3 niveles. Tiempo de convergencia: t_c=9*3=27 segundos

La tasa de error se midió de acuerdo a la cantidad de datos que llegan a su destino de manera correcta y completa; se obtuvo dividiendo el número de datos enviados correctamente entre el número total de líneas enviadas multiplicadas por 11. Es decir:

$$\%E = (1 - \frac{datos\ correctos}{\text{num\_tot\_lin} * 11}) * 100$$

En una muestra de 36 líneas se obtuvieron 385 datos enviados correctamente, a lo que la tasa de error calculada es:

$$\%E = \left(1 - \frac{385}{36 * 11}\right) * 100 = 2.77\%$$

Un nodo, una vez encontrado el nodo con el número de saltos más bajo, no vuelve a buscar otro nodo con un número de saltos más bajo por cuestiones de eficiencia en el tiempo de ejecución del módulo ESP8266, ya que al explorar los SSID de los nodos cercanos, debe invertir un tiempo continuamente en realizar dicha exploración. Si el nodo se desconectase de su nodo superior, volvería a explorar los nodos cercanos para calcular a cuál de todos conectarse de acuerdo al menor número de saltos y la mayor fuerza de señal.

La cuestión en cuanto a autonomía de energía, al no estar conectados los nodos a una fuente fija de energía, se midió con las baterías solares recargables, las cuales manejan una capacidad de almacenamiento de energía de 5000mAh. Estimando que todos los componentes del nodo gastan 750mA por hora, esto nos da una independencia aproximada de 6 horas con 40 minutos. Las baterías solares ayudan a mejorar la movilidad y la autonomía energética de los nodos, al no estar restringidos con una conexión energética por cable, aunque se puede optar por conectarlos a la corriente eléctrica, ya sea con o sin batería solar, para que ésta se cargue mientras continúa suministrando energía al nodo.

Los datos capturados de los tres nodos se graficaron en una interfaz gráfica de Matlab, eligiendo el modo de Graficar en tiempo real (véase Anexo A.2: Interfaz de Matlab). En las gráficas de las figuras 43 y 44 se puede apreciar cómo se van introduciendo los datos de CO_2 y CO capturados en el servidor y se van mostrando a la vez en la interfaz de Matlab, pudiendo observar los datos de los tres nodos a la vez en un intervalo de 10 minutos.

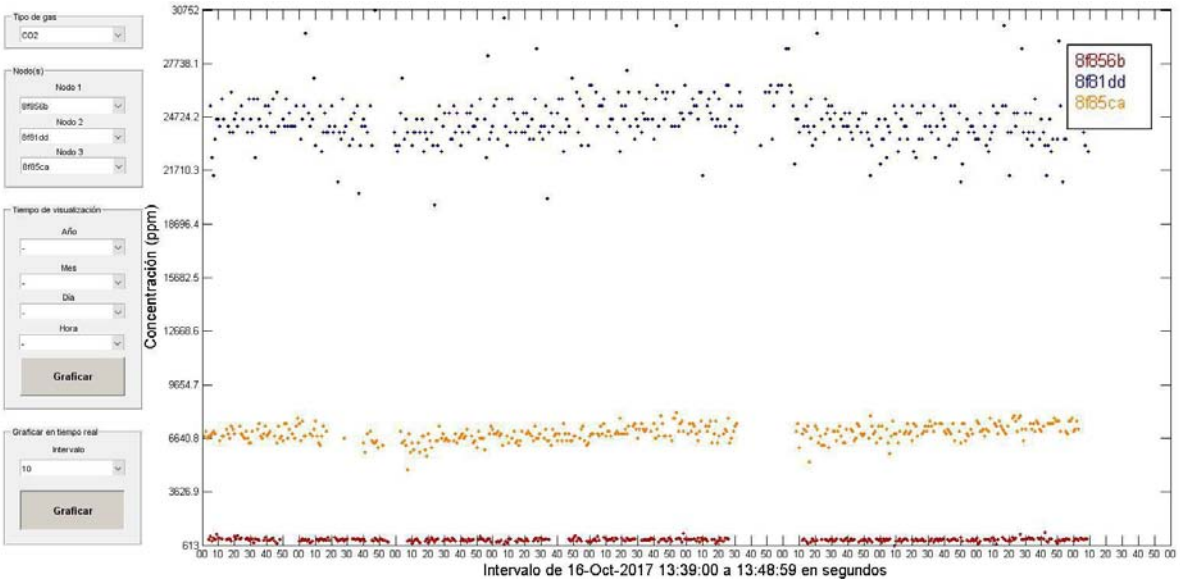


Figura 43. Gráfica de tiempo vs concentración de CO_2 de tres nodos en un intervalo de 10 minutos.

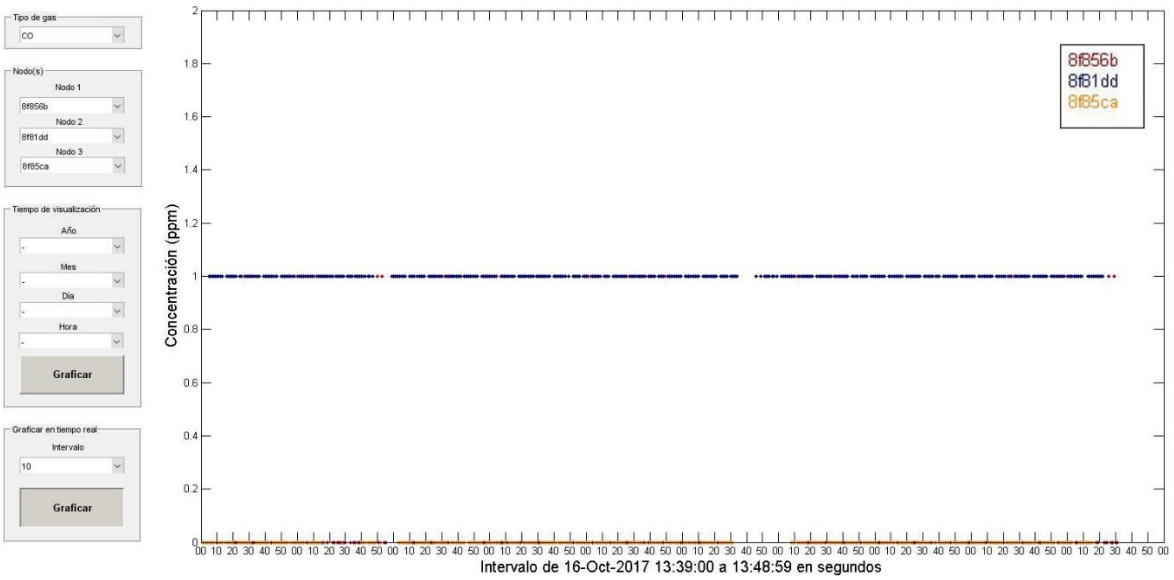


Figura 44. Gráfica de tiempo vs concentración de CO de tres nodos en un intervalo de 10 minutos.

5.1 Experimentación con exploración continua de vecinos

Se experimentó con una implementación que lleva a cabo una exploración de nodos cercanos cada cierto tiempo aleatorio que va en intervalos de 2 segundos, desde los 4 hasta los 12 segundos. Esta implementación tenía como propósito el de explorar nuevos nodos candidatos a ser softAP del nodo explorador, para que de esta manera encuentre un mejor camino para enviar sus datos capturados. Desafortunadamente, se observó un decremento en el desempeño de la red, ya que, al explorar nuevos nodos, el módulo WiFi no puede recibir nuevas conexiones de nodos estación, ni recibir datos provenientes de los nodos conectados, lo que hace más lento el envío de datos de los nodos al servidor. Por esta razón, se optó por descartar la característica de explorar nuevos nodos una vez conectado el módulo WiFi a la red. A continuación, se mostrarán dos gráficas de captura de datos de tres nodos. En la figura 45 se observa un flujo de datos de forma continua, ya que no hay exploración continua de nuevos nodos. En cambio, en la figura 46 al implementar la exploración continua de nuevos nodos, el desempeño cae, viéndose reflejado con la ausencia de datos en algunos intervalos de tiempo.

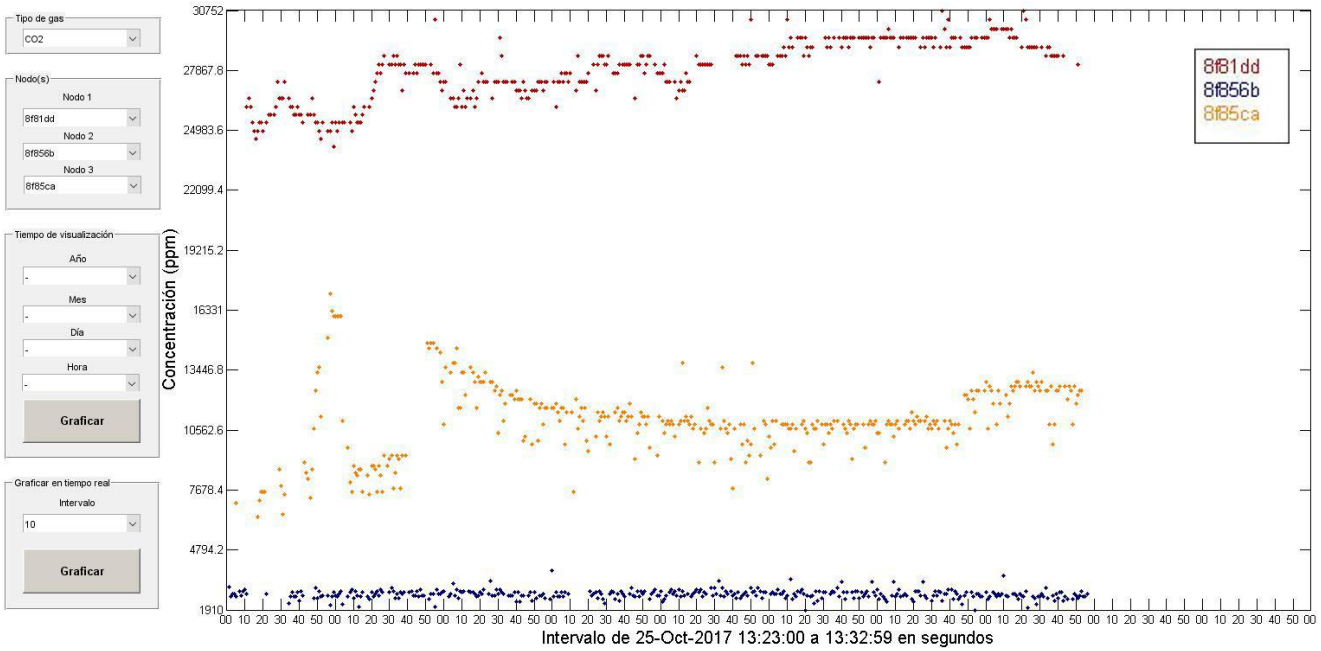


Figura 45. Implementación sin exploración continua.

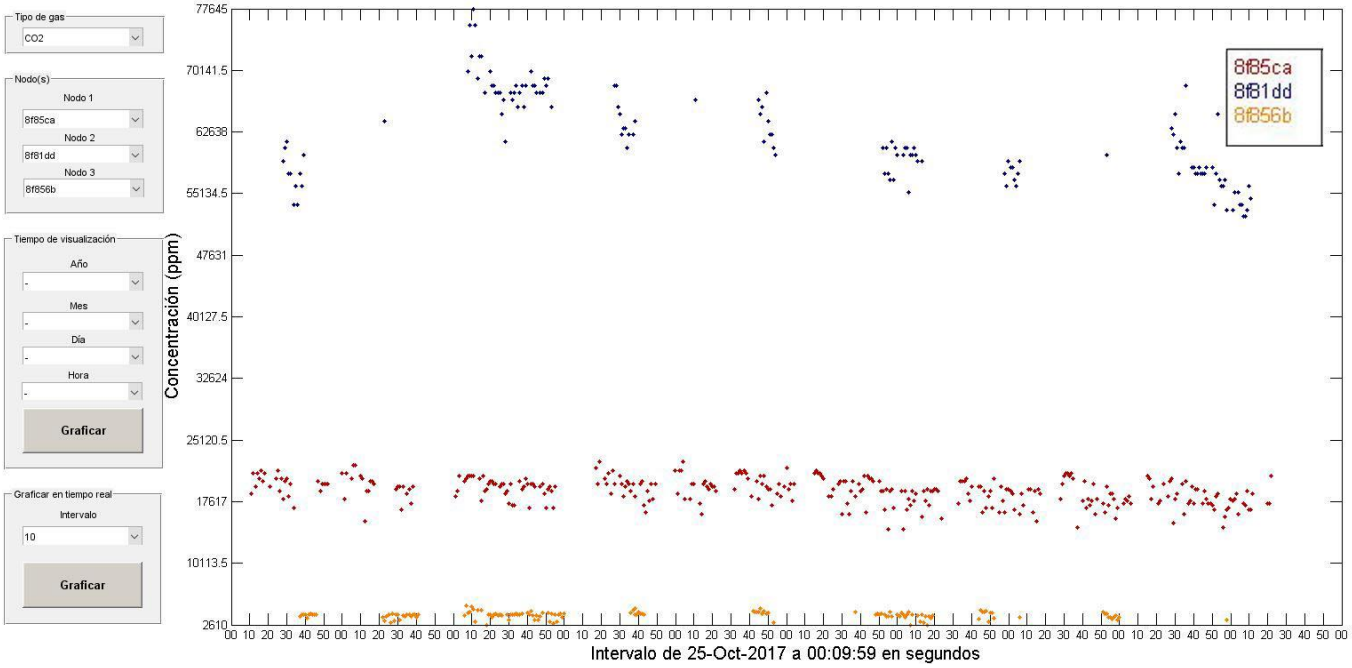


Figura 46. Implementación de exploración continua de nodos.

5.2 Experimentación con configuraciones

Se conectaron los tres nodos con dos configuraciones posibles de conexiones entre nodos para probar la eficiencia del protocolo PSFS.

5.2.1 Configuración 1

Figura 47

En esta configuración, mostrada en la figura 47, se configuró a un nodo como el designado para conectarse al router y los dos nodos restantes enviarán su información recabada cada 11 segundos. El nodo conectado al router enviará también cada 11 segundos su información recabada más la información que almacene de los otros dos nodos.

Los resultados con esta configuración fueron los siguientes:

Tiempo de convergencia: 12 segundos.
Velocidad de datos: 118.75 Kb/s
Tasa de error: 5.17%

En esta configuración existe más carga de recepción de datos en el nodo conectado al router, lo cual causa que dicho nodo se trabe cuando, además de enviar los paquetes de datos almacenados cada 11 segundos al servidor, debe recibir los paquetes de datos de los otros dos nodos. Esto se resolvió definiendo un tiempo de respuesta del módulo WiFi hacia el Arduino Mega el cual, si sobrepasa los 20 segundos, envía una señal de reinicio al módulo WiFi. Cuando el nodo vuelva a conectarse al router con acceso al servidor, los datos que haya almacenado hasta ese momento serán enviados cuando se venza el intervalo de envío de paquetes de datos, y si existe un error en un paquete de datos, el índice de datos en la memoria SD se ajustará en ese paquete, para volver a enviarlo en el próximo envío de paquetes, asegurando que la pérdida de envío de datos al servidor sea mínima. En conclusión, la configuración fue un éxito, a pesar de que de manera esporádica el nodo conectado al router se atore y se deba reiniciar, desintegrando la red.

5.2.2 Configuración 2

Figura 48

En esta configuración, mostrada en la figura 48, se dispusieron los nodos de tal manera que un nodo fuese asignado para conectarse al router con acceso al servidor, el segundo nodo se conectase al primer nodo como estación y el tercer nodo se conectase al segundo nodo como estación, haciendo así una red lineal.

Los resultados con esta configuración fueron los siguientes:

Tiempo de convergencia: 25 segundos.
Velocidad de datos: 118.75 Kb/s
Tasa de error: 4.13%

En esta configuración existe menos carga de recepción de datos en cada nodo, pero el tiempo de convergencia en caso de desconexión es mayor al existir más niveles que en la configuración 1. Sin embargo, la probabilidad de que se atore un nodo por la cantidad de tráfico es menor que en la configuración anterior. Pese a que existe la posibilidad de que se desconecten los nodos, los datos se almacenan en las memorias micro SD hasta que los nodos se conecten a la red, minimizando la pérdida de datos. En conclusión, la configuración funcionó con éxito, a pesar de a veces sufrir desconexiones, dado el atoramiento de los nodos en ciertas ocasiones por el hardware del módulo WiFi.

En conclusión, la configuración 1 es más conveniente de implementar si se dispone únicamente de 3 nodos, ya que:

- converge más rápido,
- el envío de datos hacia el servidor es más rápido dados los pocos saltos entre nodos y,
- los nodos adyacentes al nodo conectado al router no requieren de memoria micro SD.

5.3 Comparación con otros protocolos

En esta sección se va a comparar el protocolo PSFS con protocolos ya existentes para revisar su eficiencia contra la eficiencia de estos protocolos.

En el tema de paquetes retransmitidos, en el caso que el nodo no reciba un mensaje de confirmación de paquete recibido, se retransmite el paquete si hay una memoria micro SD instalada en el nodo, aunque esta situación sucede en un paquete de datos de cada cincuenta que son enviados de un nodo a otro o un 2% de los casos, y cuando se trata del envío de paquetes de un nodo hacia el servidor, esto sucede en 1% de los casos. Si el nodo no posee una memoria micro SD y no está conectado al router, no enviará paquetes de retransmisión, ahorrando recursos como tiempo y tráfico en la red. Esto es ventaja frente a otros protocolos como el reactivo AODV[33].

En cuanto a paquetes descartados, el protocolo del proyecto va empatado con protocolos como AODV, DSDV y DSR, ya que con pocos nodos la cantidad de paquetes descartados es escasa, de un aproximado del 2.5%, aunque con más nodos no se pudo experimentar por falta de dispositivos físicos.

El protocolo, al no tener que actualizar frecuentemente tablas de vectores de distancia, no ocupa procesamiento, tiempo, ni tráfico en la red para dicho fin, lo cual lo pone en ventaja frente a los protocolos basados en vector de distancias (VD)[34].

Este protocolo implementó dos características únicas, las cuales no se hallan en otros protocolos. La primera característica es la capacidad de transmitir la fecha y hora a todos los nodos conectados a la red (característica sólo disponible con el protocolo TORA), además de la posibilidad de solicitar una actualización de fecha y hora desde el nodo conectado al router o sus subordinados directos. La segunda característica es la capacidad de transmitir los datos del router, como el SSID, la contraseña y la dirección IP o página web del servidor a otros nodos de la red, además de tener la capacidad de cambiar dichos datos desde cualquier nodo, siempre y cuando esté conectado a la red.

En el tema de convergencia, este protocolo podría converger de manera mucho más rápida que el resto de los demás protocolos de enrutamiento para redes Ad-hoc, si no tomase tanto tiempo en conectarse a otro nodo. Es decir, de parte del diseño del algoritmo para conectarse, podría converger a una red de manera muy rápida y eficaz, pero esto en la práctica no es posible debido al tiempo de conexión de un nodo con el otro, aún proporcionando la dirección MAC y el canal de comunicación inalámbrica del nodo. Otro factor que ralentiza la conexión con otro nodo es la negociación con el servidor DHCP de los nodos, el cual se debe implementar, ya que no es posible asignar direcciones IP estáticas a los nodos estación, por la posibilidad de asignar direcciones repetidas.

Capítulo 6:
Conclusiones y trabajo futuro

6.1 Conclusiones

En este trabajo se desarrolló un nuevo protocolo de enrutamiento de redes Ad-hoc llamado Protocolo basado en Saltos y Fuerza de Señal (PSFS), con el objetivo de almacenar y propagar de manera eficiente, información relacionada con niveles de contaminación del aire. La información capturada se envió y almacenó en un servidor Apache para su visualización en una interfaz amigable para el usuario final.
El protocolo desarrollado e implementado arrojó resultados positivos en parámetros como la velocidad de transmisión de datos, la velocidad de convergencia, la cantidad de nodos estación conectados a nodos servidor, la flexibilidad de datos, la tasa de éxito en la transferencia de datos y la escalabilidad pequeña. A continuación, se detallarán las conclusiones obtenidas del trabajo.

- La velocidad de transmisión de datos y su persistencia, a pesar de estar limitada por la velocidad de los puertos seriales del módulo WiFi y del Arduino, fue lo suficientemente rápida como para poder enviar la información recabada de manera oportuna. Además, el protocolo fue diseñado para almacenar la información recabada cuando el nodo no esté conectado a otro nodo o al router con acceso al servidor, para enviarla posteriormente cuando logre conectarse a la red, minimizando así la pérdida de datos recolectados.

- La velocidad de convergencia es rápida, tomando en cuenta que los nodos demoran un poco de tiempo en conectarse a otro nodo o al router. Si se quiere un tiempo de convergencia menor, se sugiere montar una red Ad-hoc con una cantidad de nodos no mayor a 50, ya que con cantidades mayores se calcula un tiempo de convergencia estimado de 40 segundos, dependiendo qué tan alejados estén los nodos unos de otros (depende de los niveles).

- La cantidad de nodos estación conectados a un mismo nodo softAP no debe de superar a los 254 nodos, ya que la tabla de asignación de direcciones IP por DHCP solamente puede asignar hasta 254 direcciones, aunque por motivos de rendimiento, se sugiere que no se encuentren conectados más de 5 nodos a un solo nodo softAP. Con esto se desea limitar el tráfico sobre un mismo nodo, ya que a mayor número de nodos conectados al nodo softAP, mayor será la probabilidad de que se atore el nodo dada la sobrecarga de tráfico.

- Con el protocolo se pueden propagar datos esenciales para la red de sensores como la fecha, la hora, el SSID del router, su contraseña y la dirección IP o página web del servidor de manera automática y casi instantánea. Se puede solicitar una actualización de fecha y hora desde cualquier nodo conectado al router o a los nodos conectados a éste. Los datos del router pueden ser cambiados y propagados desde cualquier nodo conectado a la red, dándole un plus a la flexibilidad de propagación de datos de la red. Además, si un nodo se desconecta de la red y se vuelve a conectar, éste copiará los datos del nodo al que se conectó.

- La tasa de éxito tuvo como problema principal la compatibilidad de los voltajes de comunicación entre el módulo de WiFi ESP8266 y el Arduino Mega, lo cual causaba que algunos datos se perdiesen en las transmisiones. Dos soluciones posibles para solucionar este problema son: utilizar una velocidad menor de transmisión de los puertos seriales o, utilizar un ajustador de nivel de voltaje (level shifter) entre el Arduino Mega y el ESP8266. De haber solucionado este problema, la tasa de éxito probablemente hubiese sido aproximadamente del 100 por ciento; en pocas palabras, la tasa de éxito dependió del canal de comunicación de los puertos seriales y no del rendimiento del protocolo PSFS.

- La escalabilidad a pequeña o mediana escala del protocolo es posible, dado el poco tiempo de convergencia que requiere el protocolo para conectar todos los nodos, lo cual lo hace ideal para zonas medianamente amplias y con nodos ya sean fijos o con poco movimiento.

6.2 Trabajo futuro

En el proyecto, viendo la posibilidad de realizar trabajo a futuro tanto en el diseño como en la implementación, hay puntos donde se pueden realizar mejorías:

- **Buscar rutas con menores saltos una vez conectado el nodo.** En el protocolo, una vez que el nodo se ha conectado a la red, ya no buscará nuevos enlaces con menor número de saltos por cuestiones de eficiencia en la transmisión y recepción de datos. Lo que se pretende agregar es que el nodo siga buscando de forma proactiva nuevos enlaces con menos saltos una vez conectado a la red, para así buscar nuevas rutas más cortas para el envío de datos. Esto se debe hacer sin que se comprometa el rendimiento de envío continuo de datos.

- **Minimizar la tasa de error en la transmisión de datos.** Este punto, al ser cuestión de hardware, se puede mejorar, ya sea disminuyendo la velocidad de comunicación entre el módulo ESP8266 y el Arduino Mega, o utilizando un level shifter entre ambos dispositivos para que la tasa de éxito se aproxime a un 100 por ciento.

- **Diseñar un mecanismo limitante del número de estaciones conectadas a un softAP:** Si bien se implementó que el servidor DHCP de los nodos pueda brindar hasta 254 direcciones IP para los nodos estación, lo conveniente es conectar hasta 5 nodos estación a un solo nodo softAP para evitar congestionamiento de tráfico en este nodo. Se puede diseñar un mecanismo para limitar el número de nodos estación conectados a la vez y así evitar que los nodos softAP se atoren por el alto tráfico de paquetes de datos.

- **Utilizar más nodos y agregarles mayor movilidad:** El protocolo fue diseñado para funcionar con más de tres nodos, así que, para fines prácticos, se debe de probar con más nodos para comprobar su robustez frente a un mayor número de nodos. Este protocolo fue diseñado para funcionar en ambientes donde los nodos fuesen fijos o se moviesen a velocidades bajas. Un punto a considerar en el futuro sería el de mejorar el protocolo para funcionar en entornos donde los nodos se desplacen a altas velocidades (redes MANET).

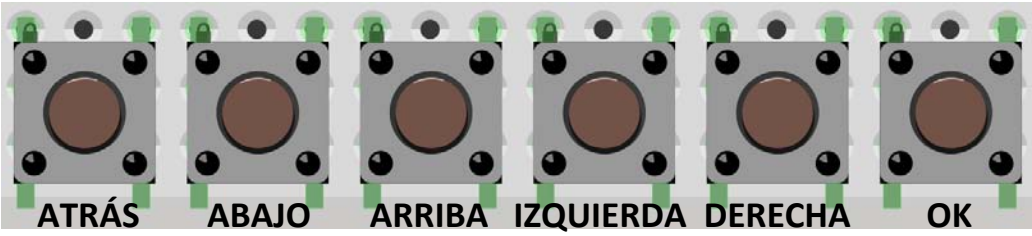
- **Enviar datos sin necesidad de memoria:** El protocolo requiere que, para almacenar y enviar los datos al servidor, cada nodo posea una memoria micro SD. Una característica a agregar sería la de poder enviar los datos capturados en ausencia de una memoria micro SD a través de la red hasta el servidor, aunque esto signifique agregar más carga de datos a la red. Esto se podría implementar para disminuir la dependencia a la memoria externa.

Anexo A:
Manuales de Usuario

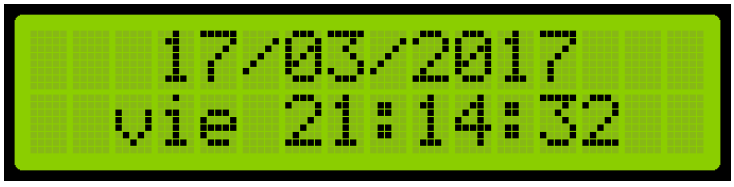
En este apartado se brindará una guía de usuario sobre cómo utilizar tanto el menú de los nodos en la pantalla LCD, como la interfaz en Matlab para visualizar los datos capturados.

A.1 Menú de la pantalla LCD

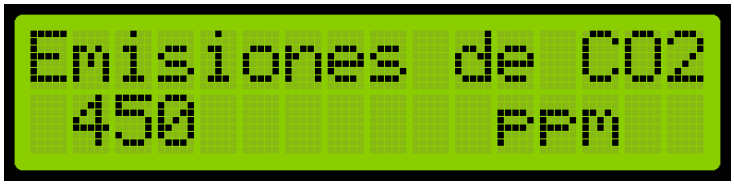
En la pantalla LCD se puede navegar en múltiples pantallas con ayuda de los botones de acción. A continuación de mostrarán los botones de acción implementados en la protoboard:



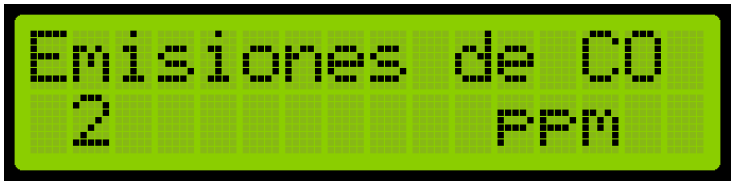
La visualización de los parámetros principales se hace a lo largo de tres pantallas sin tener que ingresar a un menú. La primera pantalla muestra la fecha con el día de la semana en formato de fecha dd/mm/aaaa, y también muestra la hora en formato de 24 hrs. HH:mm:ss, de acuerdo a los algoritmos discutidos en la sección 4.4.3. La segunda pantalla muestra las emisiones de CO_2 en ppm con formato entero hasta 8 dígitos, y la tercera pantalla muestra las emisiones de CO en ppm con formato entero hasta 8 dígitos.



Pantalla 1: Fecha y hora.

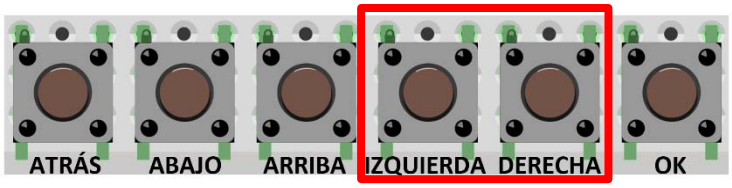


Pantalla 2: Emisiones de CO_2.



Pantalla 3: Emisiones de CO.

Se puede navegar a través de las tres pantallas por medio de los botones de **IZQUIERDA** o **DERECHA**.



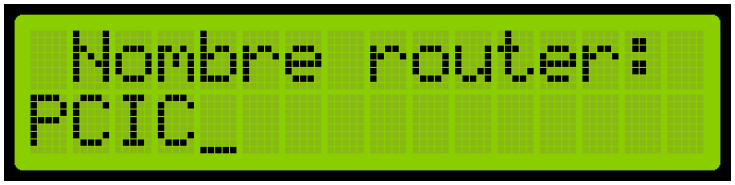
Al presionar el botón de **OK** se puede acceder al **menú de Ajustes e Información**, el cual consta de 7 opciones a elegir con los botones de **ABAJO** y **ARRIBA**, y la opción a elegir se resalta con una flecha y se elige con el botón **OK**. Estas opciones son:

1. Propagar datos del router
2. Cambiar datos del router
3. Ver datos del router
4. Ver direcciones IP
5. Designar para conectarse a internet
6. Solicitar actualización de hora
7. Ajustes de pantalla

Cabe mencionar que, para regresar de cualquier opción al menú de ajustes e información, basta con presionar el botón **ATRÁS**.

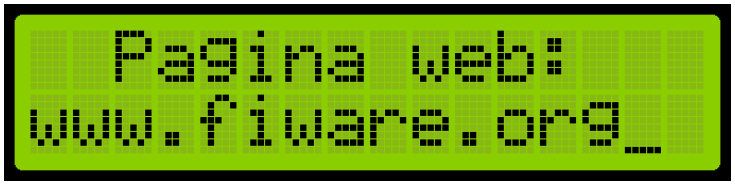
A.1.1 Propagar datos del router

En la opción **Propagar datos del router** se pueden tanto cambiar como pedir que se propaguen por la red Ad-hoc los datos del router, al cual el o los nodos designados se deberán conectar para enviar la información capturada a internet. Dichos datos son el SSID del router, su contraseña y la página web a la que se enviarán los datos.
En la primera pantalla se puede introducir el nombre o SSID del router, el cual puede ser de hasta 32 caracteres y se puede desplazar a lo largo de él, pudiendo retornar al primer carácter desde el último carácter. Si el nombre ya ha sido introducido, entonces se muestra el SSID ya existente, pudiéndose modificar. Al terminar se debe presionar el botón **OK**.



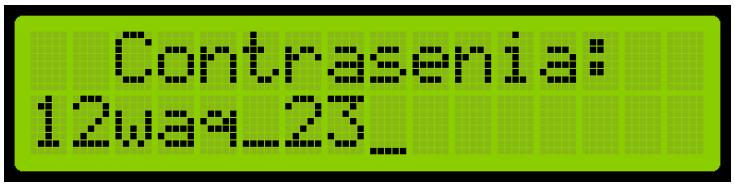
Pantalla de inserción del SSID del router.

En la siguiente pantalla se debe introducir el nombre o IP del sitio web a donde irán los datos capturados, el cual puede medir hasta 100 caracteres e igual se puede desplazar a lo largo de la cadena, pudiendo retornar al primer carácter desde el último carácter. Si la página o dirección IP ya ha sido introducida, entonces se muestra la información ya existente, pudiéndose modificar. Al terminar se debe presionar el botón **OK**.



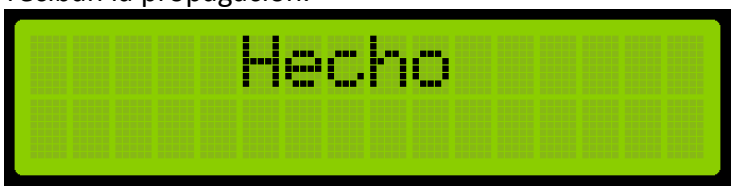
Pantalla de inserción de la página web.

En la tercer y última pantalla se introduce la contraseña del router, con longitud de 8 a 32 caracteres y se puede desplazar a lo largo de ella, pudiendo retornar al primer carácter desde el último carácter como en el caso del SSID. Si ya se ha introducido la contraseña, ésta no se mostrará en la pantalla por motivos de seguridad. Al terminar se debe presionar el botón **OK**.



Pantalla de inserción de la contraseña del router.

Una vez hecho todo esto, si todo salió bien se mostrará una pantalla con la leyenda “Hecho”, y se regresará al menú de ajustes e información. La información introducida se guardará en la memoria EEPROM del módulo ESP8266, tanto en el nodo propagador como en los nodos que reciban la propagación.

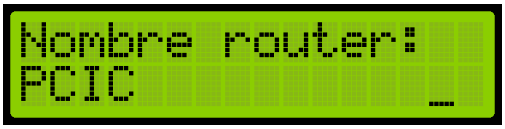


A.1.2 Cambiar datos del router

La opción **Cambiar datos del router** funciona solamente para cambiar la información respectiva al router sin propagar esta información a través de la red. La introducción de datos del router funciona de la misma manera que la opción anterior. La información introducida se guardará en la memoria EEPROM del módulo ESP8266.

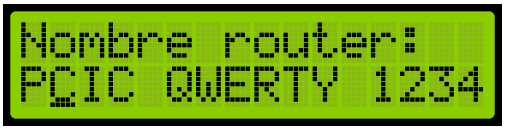
Es de importancia mencionar cómo se deben manejar los botones de acción para escribir los datos de entrada del Arduino respecto a estas dos opciones del menú.
Los botones de **IZQUIERDA** y **DERECHA** se utilizan para desplazarse a lo largo de toda la cadena, ya sea del SSID, de la contraseña o del nombre o IP de la página web. Si se mantiene presionado cualquiera de estos botones, se puede desplazar más rápido por la cadena.

Desplazar a la derecha.



Mantener presionado **DERECHA**.

Desplazar a la izquierda.



Mantener presionado **IZQUIERDA**.

Los botones de **ABAJO** y **ARRIBA** funcionan para seleccionar entre los distintos caracteres de acuerdo a la semántica elegida. Hay cuatro semánticas: caracteres especiales, números, letras mayúsculas y letras minúsculas. Si, por ejemplo, se tiene seleccionada la semántica de mayúsculas, al presionar rápidamente el botón de **ARRIBA** se puede ir eligiendo de las letras mayúsculas de la A a la Z, y si se presiona rápidamente el botón de **ABAJO** se puede ir desplazando de la Z a la A.

Orden ascendente de las letras mayúsculas.

Orden descendente de las letras mayúsculas.

Para cambiar entre las cuatro semánticas se debe mantener presionado alguno de los botones de **ABAJO** o **ARRIBA**. En la semántica de caracteres especiales se pueden elegir los siguientes caracteres: **espacio_blanco** ! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [\ ] ^ _ ` { | }

Cambio de semántica: mantener presionado **ABAJO** o **ARRIBA**.

Nota: En la introducción de datos desde los botones de acción se tiene un conjunto limitado de caracteres especiales en ASCII y no hay vocales con acentos o diéresis.

A.1.3 Ver datos del router

La opción de **Ver datos del router** permite ver lo que es el SSID y la página web introducidas con anterioridad u obtenidas de propagaciones por la red. En caso de no haber introducido nada o no haber obtenido ninguna información de la red, no se mostrará nada en la pantalla. Con los botones de **IZQUIERDA** y **DERECHA** se puede desplazar entre la cadena del dato, saltándose de 16 en 16 caracteres.

Como ejemplo se tomará el nombre de la página web *www.fiware.org/index/iot/ad-cocnetwork/datacapture*, el cual posee 50 caracteres.

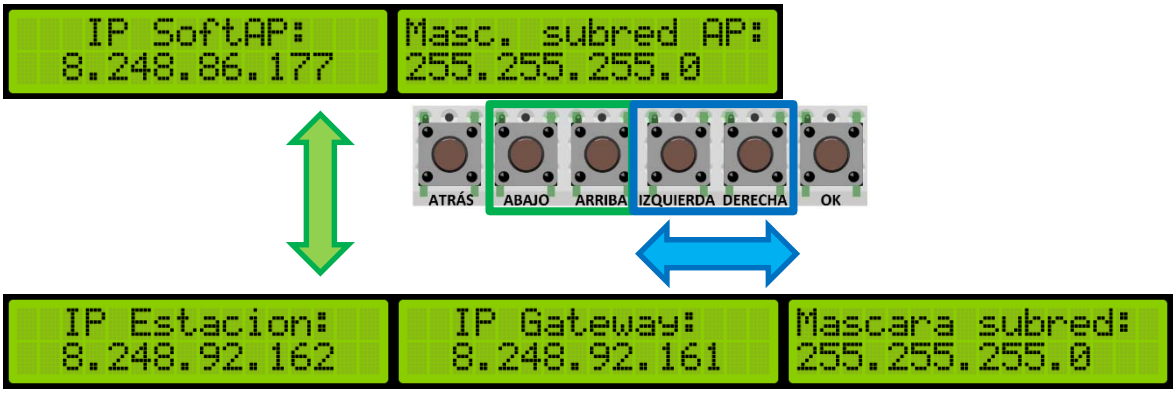
Vista del nombre de la página web dividido en 4 partes de 16 caracteres cada una.

Con los botones de **ABAJO** y **ARRIBA** se puede elegir ver entre el SSID y la página web.

Cambio de vista entre la página web y el nombre del router.

A.1.4 Ver direcciones IP

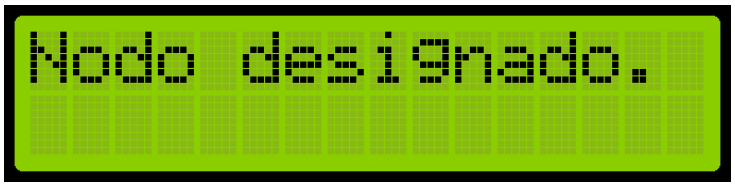
Con la opción **Ver direcciones IP** se pueden visualizar las direcciones IP del nodo, tanto en modo SoftAP como en modo estación. Con los botones de **ABAJO** o **ARRIBA** se puede alternar entre ver los parámetros del SoftAP o los parámetros de la estación. Con los botones de **IZQUIERDA** o **DERECHA** se puede alternar entre ver la dirección IP, la dirección IP de la puerta de enlace predeterminada y la máscara de subred.



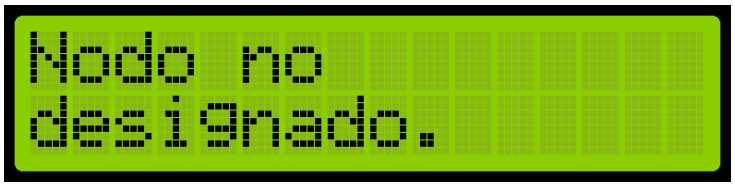
Navegación entre las diversas IP del nodo.

A.1.5 Designar para conectarse a internet

En la opción **Designar para conectarse a internet** se puede elegir al nodo para que se le permita conectarse al router que tiene acceso a internet presionando el botón OK.

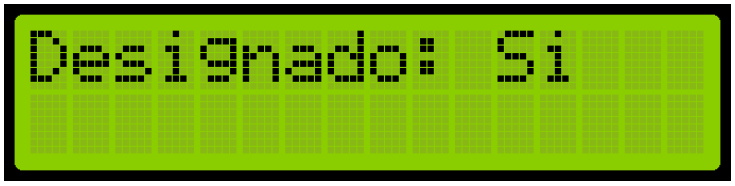


Volviendo a presionar el botón de OK se puede dejar de designar al nodo para conectarse al router con internet.



El ajuste de designación o no designación se guarda en la memoria del ESP8266.

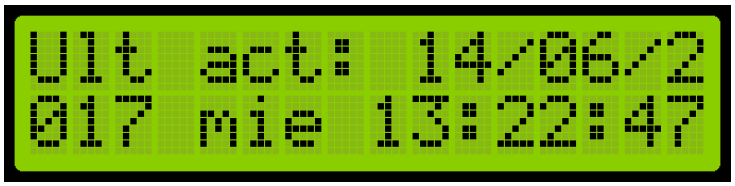
Para consultar si el nodo ha sido o no designado, se debe resaltar con la flecha del **menú de Ajustes e Información** esta opción y presionar el botón **DERECHA**, con lo cual se mostrará si el nodo ha sido designado, mostrando uno de dos mensajes: 'Designado: Si' o 'Designado: No'. Para regresar al menú anterior, se debe presionar el botón **ATRÁS**.



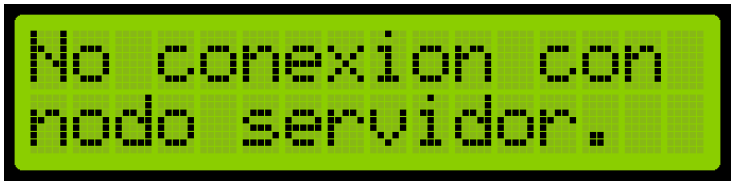
A.1.6 Solicitar actualización de hora

Con la opción **Solicitar actualización de hora** se puede pedir una actualización de hora manual, la cual se puede solicitar desde el nodo conectado al router o desde sus subordinados, y éste solicita al servidor de Google la hora actual. La respuesta de fecha y hora se propaga de vuelta a toda la red Ad-hoc, almacenándose en el RTC de cada módulo la fecha y hora actual. Si la petición fue enviada con éxito hasta el router, la pantalla devolverá un mensaje diciendo “Hecho”.

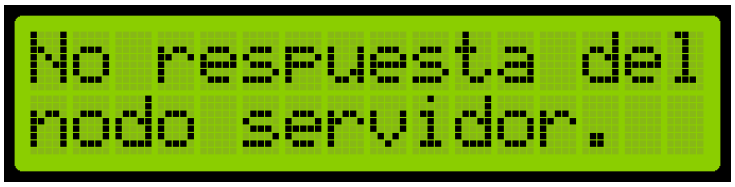
Si hubo con anterioridad una actualización de fecha y hora, se puede consultar la fecha y hora de la última actualización presionando el botón **DERECHA** con la opción **Solicitar actualización de hora** resaltada en la flecha del **menú de ajustes e información**. Para regresar al menú anterior, se debe presionar el botón **ATRÁS**. Si no ha habido una actualización de fecha y hora, entonces el mensaje mostrado será **N/A**.



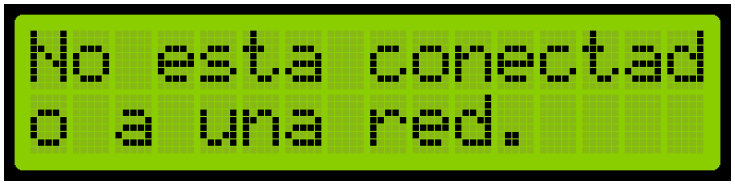
Si ocurrió un error en la solicitud de actualización de hora, la pantalla puede devolver tres mensajes de error:
El primer mensaje se muestra si no hubo conexión con el nodo servidor.



El segundo mensaje aparece si hubo conexión con el nodo servidor, pero no hubo mensaje de respuesta al mensaje de petición.

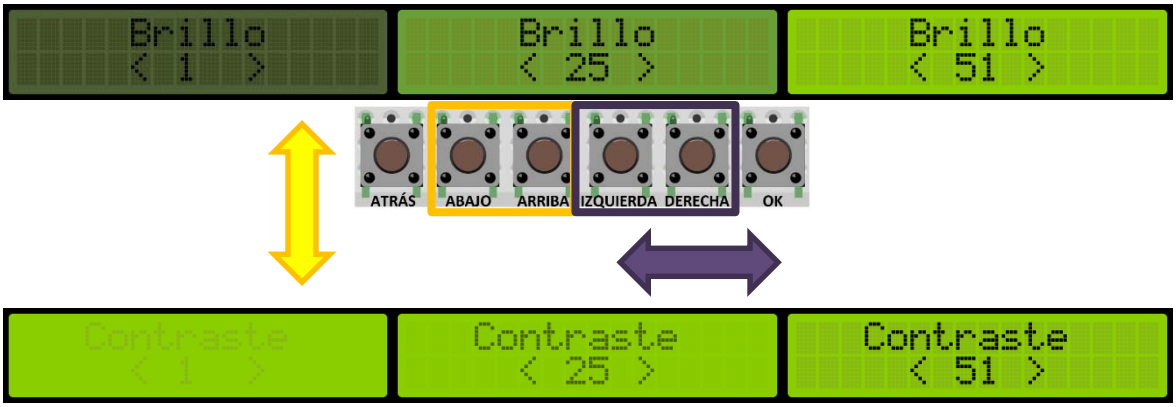


El tercer mensaje aparece cuando el nodo no está conectado a una red Ad-hoc.



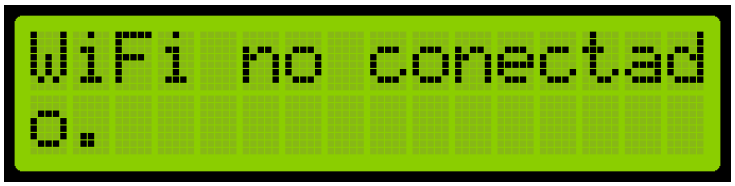
A.1.7 Ajustes de pantalla

Con la opción **Ajustes de pantalla** se puede configurar parámetros de la pantalla LCD como el brillo y contraste, lo cual se logra mediante dos PWM insertados en los pines de la pantalla LCD. Con los botones de **ABAJO** o **ARRIBA** se puede elegir entre ajustar el brillo o el contraste. Con los botones de **IZQUIERDA** o **DERECHA** se puede ajustar el nivel de cada parámetro del 1 al 51. Si se mantiene presionado cualquiera de estos dos botones, el número cambiará más rápido.
Los parámetros ajustados se guardan en la memoria EEPROM del Arduino Mega.



Navegación y ajuste del brillo y contraste.

Cabe mencionar que, si el módulo ESP8266 no se encuentra conectado al Arduino Mega, no está encendido o tiene alguna otra falla, el Arduino Mega devolverá un mensaje de falla, el cual indicará que el modulo WiFi no está conectado. Este mensaje lo muestra el Arduino cuando quiere recuperar del módulo ESP8266 información como las direcciones IP, los datos del router, quiere designarse para conectarse a internet o quiere solicitar actualización de fecha/hora y no tiene éxito.



Mensaje de error de comunicación con el ESP8266.

A.2 Interfaz de Matlab

En la interfaz de Matlab se pueden elegir diversos parámetros para representar los datos capturados de los sensores. Se puede tanto visualizar datos capturados con anterioridad de acuerdo a la emisión que se requiera observar, el Id del nodo y la fecha, como visualizar los datos capturados en tiempo real de acuerdo a un intervalo de tiempo (ver figura 49).

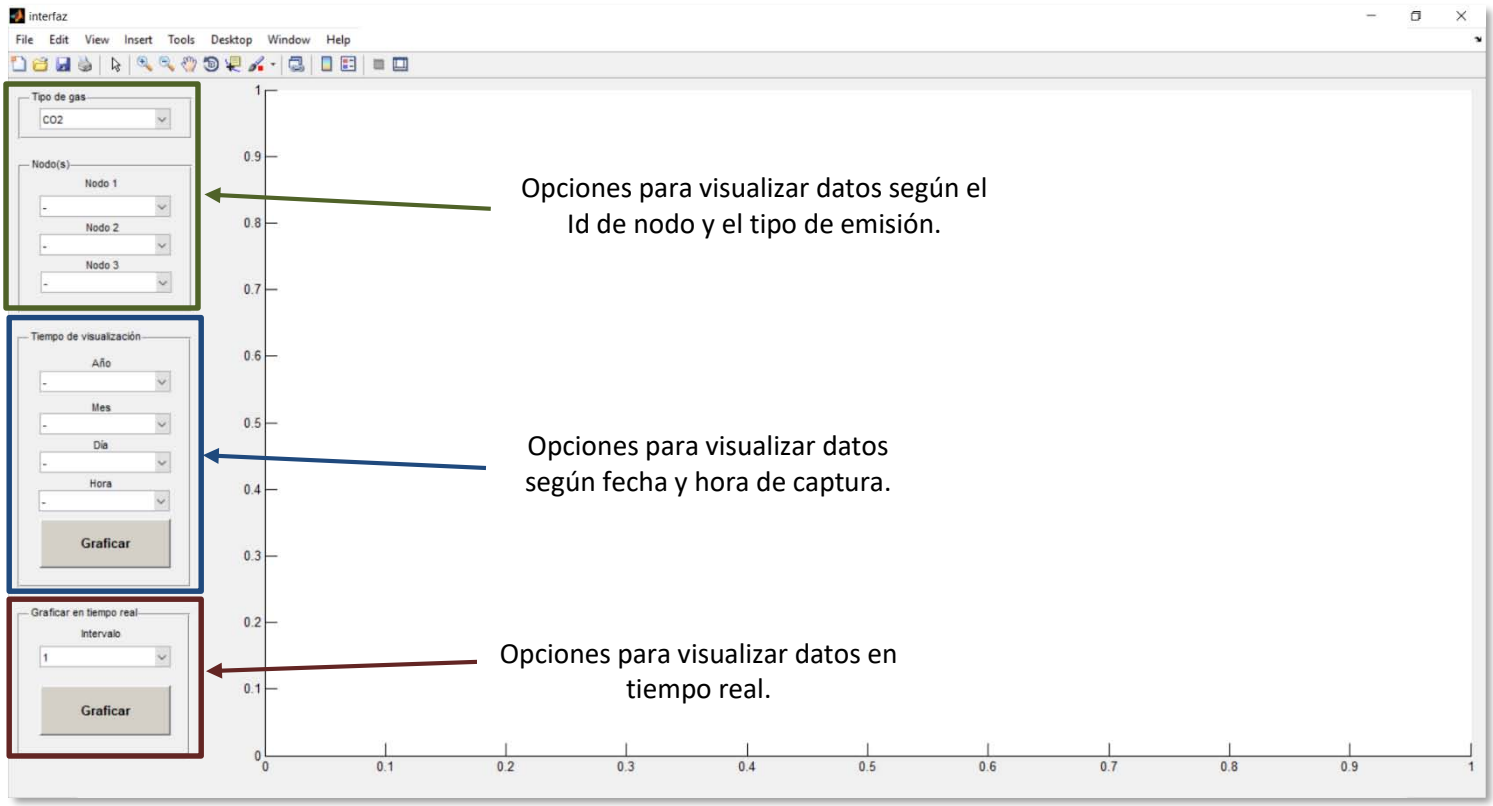


Figura 49. Interfaz de Matlab

En el recuadro **Tipo de gas** se puede seleccionar el tipo de emisión de gas que se quiere visualizar, con las opciones de poder ver las emisiones de $C0^2$, CO o ambos (ver figura 50).

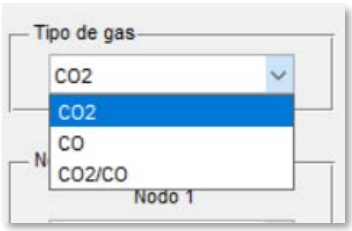


Figura 50

En el recuadro **Nodo(s)** (figura 51), se pueden seleccionar hasta tres de los nodos detectados en las capturas del servidor para visualizar sus datos correspondientes, o se puede seleccionar ver el promedio de todos los nodos, el cual se logra sumando todos los valores capturados en un segundo de los nodos y dividiendo el resultado entre el número de valores sumados.

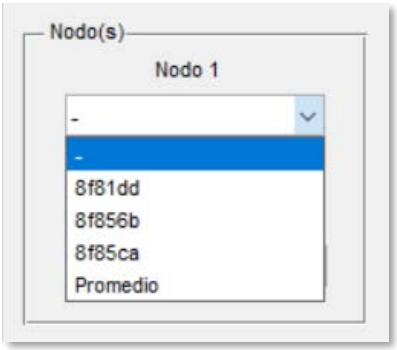


Figura 51

A.2.1 Graficar con datos antiguos

En la interfaz de Matlab se puede graficar con datos ya almacenados con anterioridad eligiendo tres parámetros, los cuales son Tipo de gas, Nodos y Tiempo de visualización. El tiempo de graficación será mayor mientras más datos capturados se procesen, a lo que se recomienda utilizar esta opción con pocos datos capturados. Como ya se explicó cómo introducir los parámetros de Tipo de gas y Nodos, se procederá a explicar cómo introducir el parámetro de Tiempo de visualización.

Esta opción posee un parámetro único para ella, el cual es el **Tiempo de visualización.** En este parámetro se puede elegir el intervalo de tiempo en el que se requiera visualizar los datos capturados anteriormente, pudiendo elegir el rango de un año, de un mes, de un día o de una hora. Los años disponibles se muestran de acuerdo a los datos que se encuentren almacenados en el servidor; los meses se pueden elegir de entre los 12 meses existentes; los días se pueden elegir de acuerdo al mes elegido, siendo de entre 28, 29, 30 o 31 días los disponibles; las horas se pueden elegir desde las 00:00 horas hasta las 23:00 horas. En el eje de las abscisas, en el caso de elegir el rango de un año, se secciona en intervalos de meses. En el caso de elegir el rango de un mes, el eje x se parte en intervalos de días. En el caso de elegir el rango de un día, el eje x se secciona en intervalos de horas, y en el caso de elegir el rango de una hora, el eje x se secciona en intervalos de minutos. El eje de las ordenadas se mide en partes por millón o ppm.
Finalmente, con el botón **Graficar** (figura 53) se lleva a cabo la graficación de los datos de acuerdo a los parámetros introducidos de Tipo de gas, Nodos y Tiempo de visualización (ver figura 50).

Figura 52

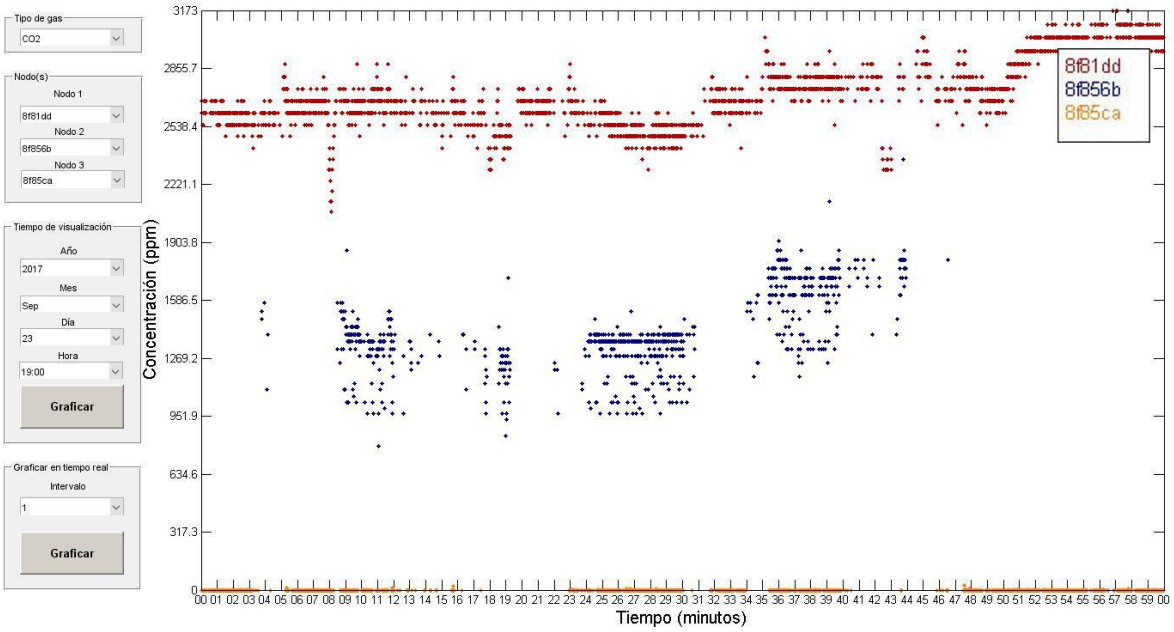


Figura 53. Ejemplo de gráfica de datos almacenados.

A.2.2 Graficar en tiempo real

Esta opción permite graficar los últimos datos entrantes del servidor y visualizarlos según vayan siendo almacenados al servidor. Como parámetros posee, además de Tipo de gas y Nodos, un parámetro propio llamado el **Intervalo** de tiempo en minutos, el cual posee las opciones de visualizar los últimos 1,2,3,4,5,6,10,12,15,20,30 y 60 minutos de los nodos. El intervalo del eje de abscisas es en segundos, independientemente del rango elegido. El eje de las ordenadas se mide en partes por millón o ppm. Esta opción es más rápida que la opción de graficar datos antiguos, ya que no tiene que procesar toda la información capturada en el servidor, sino los últimos minutos que se elijan en el intervalo de tiempo.
El botón **Graficar** (figura 54) inicia la graficación en tiempo real de acuerdo a los parámetros introducidos, y el botón permanecerá sumido indicando que se están graficando continuamente los datos entrantes. Para parar la graficación, se debe de volver a presionar el botón **Graficar**, levantándose el botón para así volver a graficar en caso de volver a presionarlo. La figuras 55 a-e ejemplifican una graficación en tiempo real.

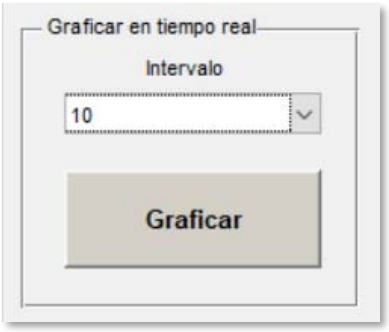
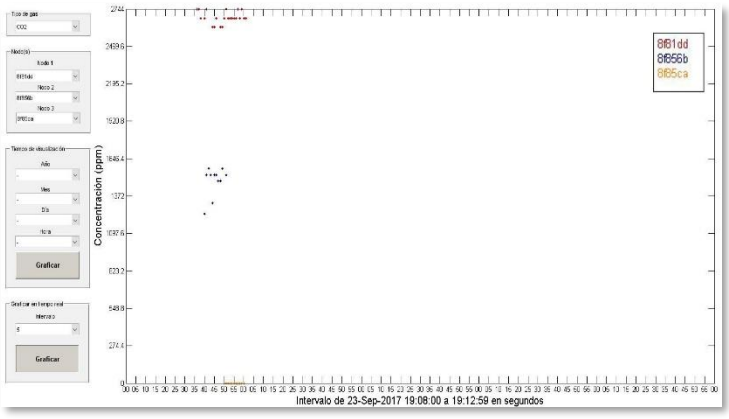
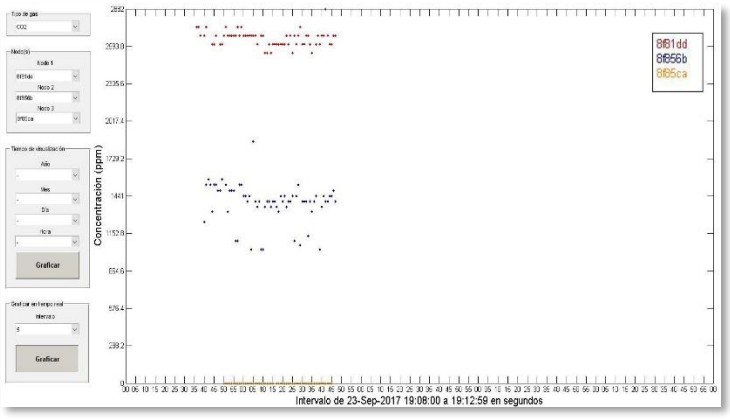


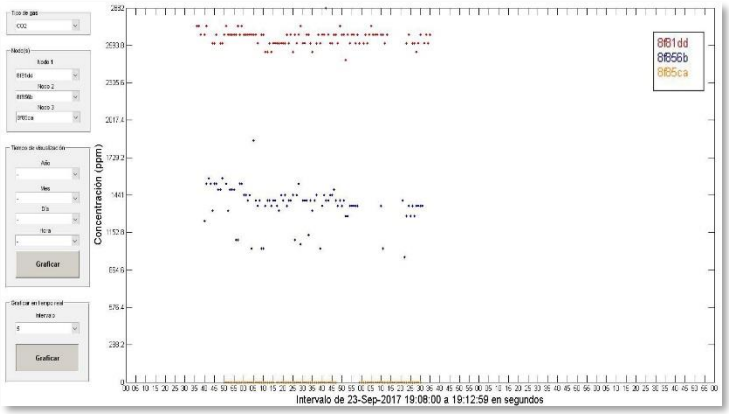
Figura 54



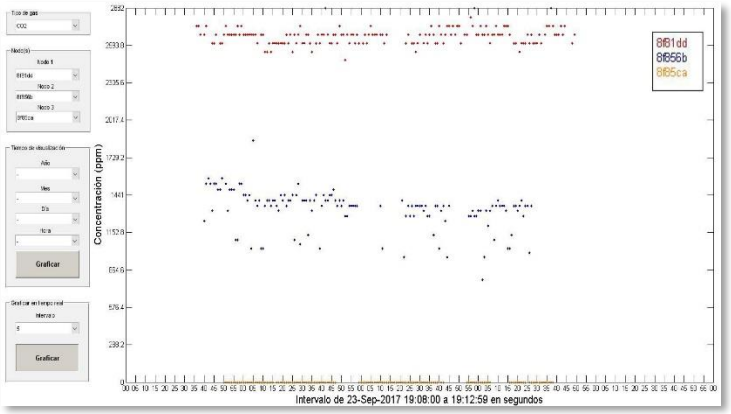
a)



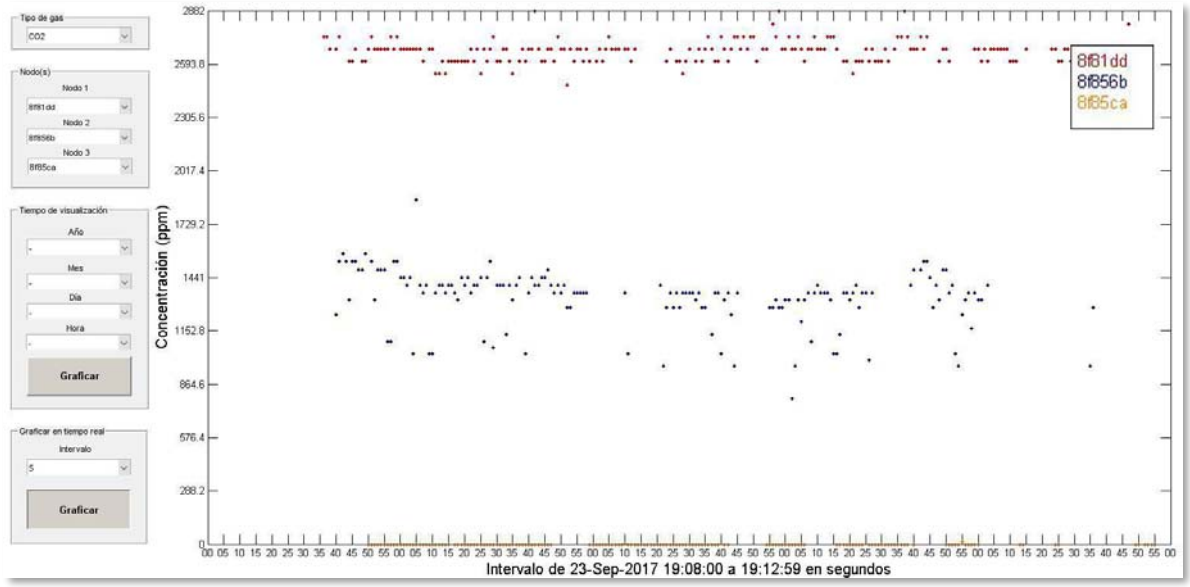
b)



c)



d)



e)

Figura 55. Ejemplos de graficación en tiempo real.

Nota: Para poder observar los Id de nodo identificados y el o los años de captura, primero se debe presionar cualquiera de los botones de **Graficar**, para realizar la exploración de estos datos y poderlos mostrar en sus respectivos menús popup.

Anexo B:
Diagramas de conexiones

El cableado que se tuvo que implementar para unir a todos los componentes en un sólo sistema se ilustra en la figura 56.

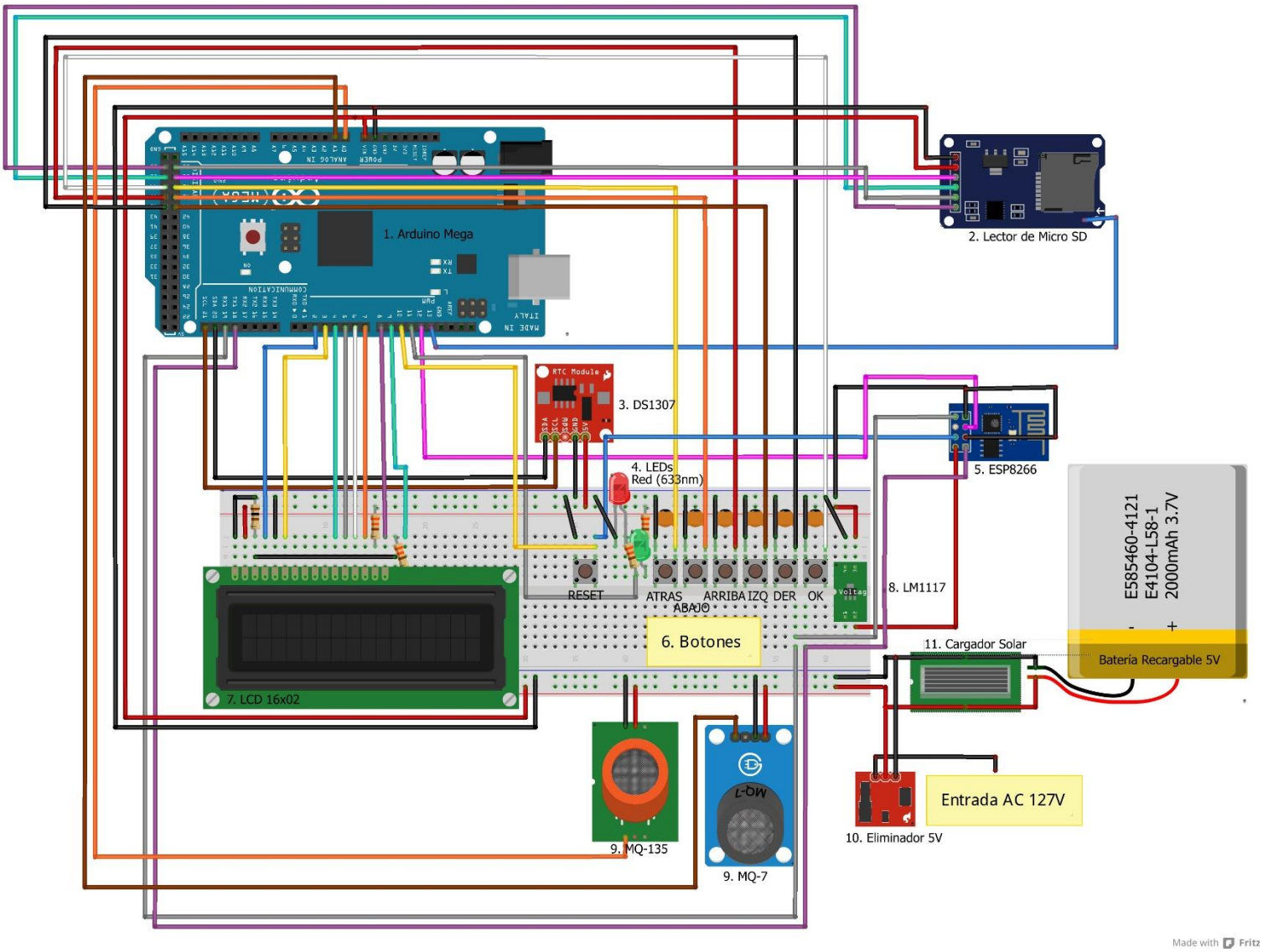


Figura 56. Diagrama físico del nodo.

1. Los **LED verde y rojo** indican el estado de conexión del nodo, con lo cual rojo indica que no se ha conectado a otro nodo o a un router, y verde indica una conexión con cualquiera de los dos de forma exitosa.
2. Los **6 botones de acción** sirven para desplazarse por el menú del nodo mostrado en el LCD. Básicamente ejecutan las acciones de Atrás, Abajo, Arriba, Izquierda, Derecha y OK (se encuentran acomodados en este orden de izquierda a derecha). Hay un séptimo botón, el cual es un Reset manual para el módulo WiFi, y está separado del resto de los botones.
3. El **eliminador de 5V** es un componente opcional, el cual se utiliza en condiciones experimentales para no tener que utilizar baterías. Es básicamente un cargador USB con 5 voltios y 1 amperio en general, el cual se conecta a la corriente alterna habitual de las construcciones de 127V.
4. Los capacitores en los botones de acción sirven para evitar el efecto debouncing, que consiste en que al pulsar un pusbutton, éste no se cierra de forma inmediata, sino que acontecen varias seudopulsaciones en fracciones de segundo antes de estabilizarse la pulsación. El Arduino interpreta esto como varias pulsaciones y, por lo tanto, se desplaza más rápido por los menús del proyecto, lo cual resulta molesto para el usuario. Las dos posibles soluciones son por hardware y por software; en este caso se optó por la solución por hardware instalando un capacitor por cada botón.

Nota: El pin IO0 del ESP8266 va conectado a tierra sólo si se va a programar. De otro modo, puede ir conectado tanto a 5V como no puede estar conectado (alta impedancia).
Cuando se programa el ESP8266, primero se debe poner en tierra el pin IO0, luego se resetea el módulo poniendo en tierra el pin RST, y después se debe proceder a verter el programa compilado. En el Visual Micro (o el IDE que se utilice) se puede ir viendo el progreso de la carga del programa dentro del ESP8266. Cuando diga 100%, la carga ha finalizado.

En la figura 57 se muestran las respectivas conexiones del Arduino Mega 2560 con el resto de los componentes del sistema. Las conexiones se encuentran encerradas en cuadros, los cuales tienen el nombre del componente al que deben ir las conexiones. La figura 58 es el diagrama esquemático. Las figuras 59 y 60 son fotografías de los nodos físicos con sus componentes y en funcionamiento.

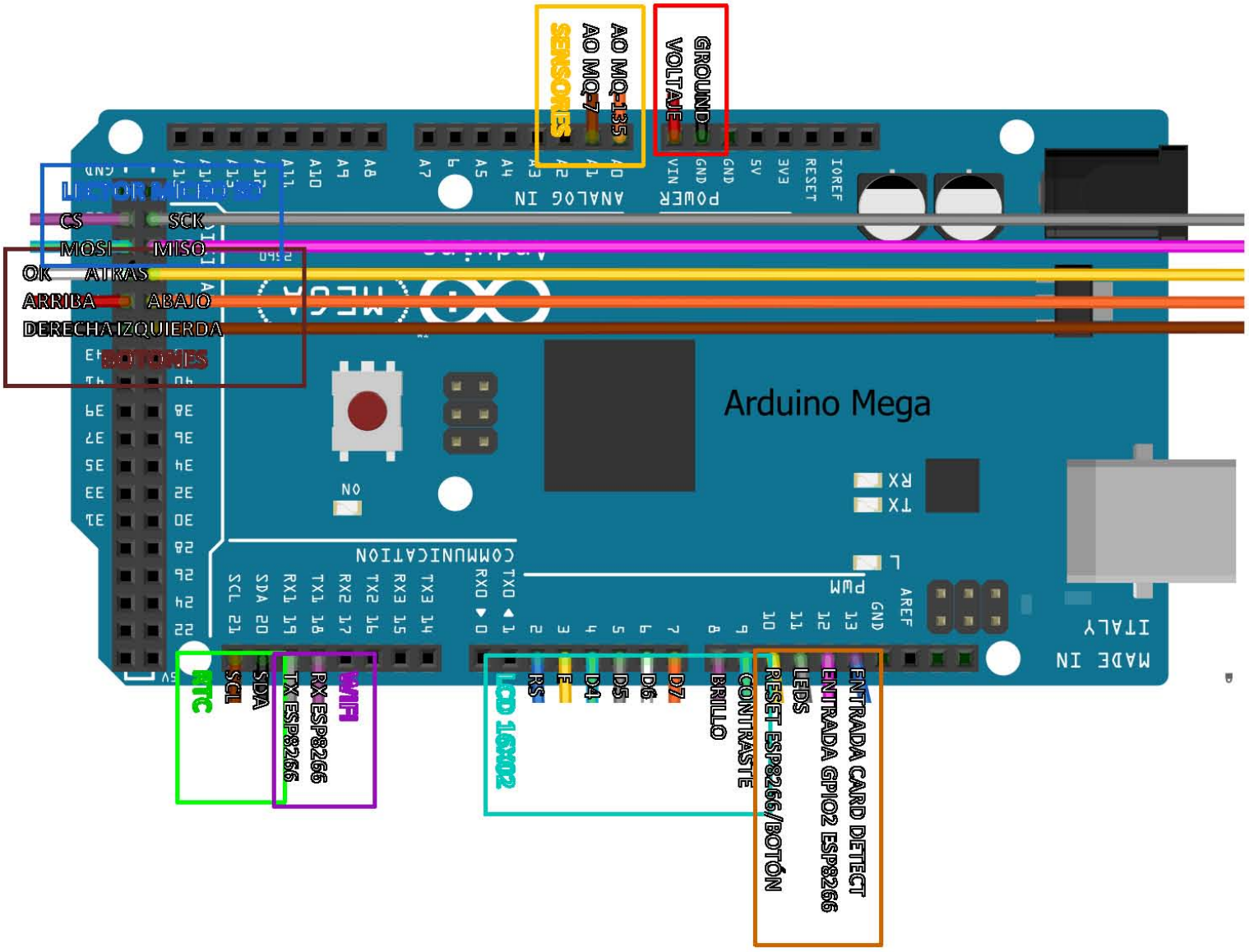


Figura 57. Diagrama de conexiones.

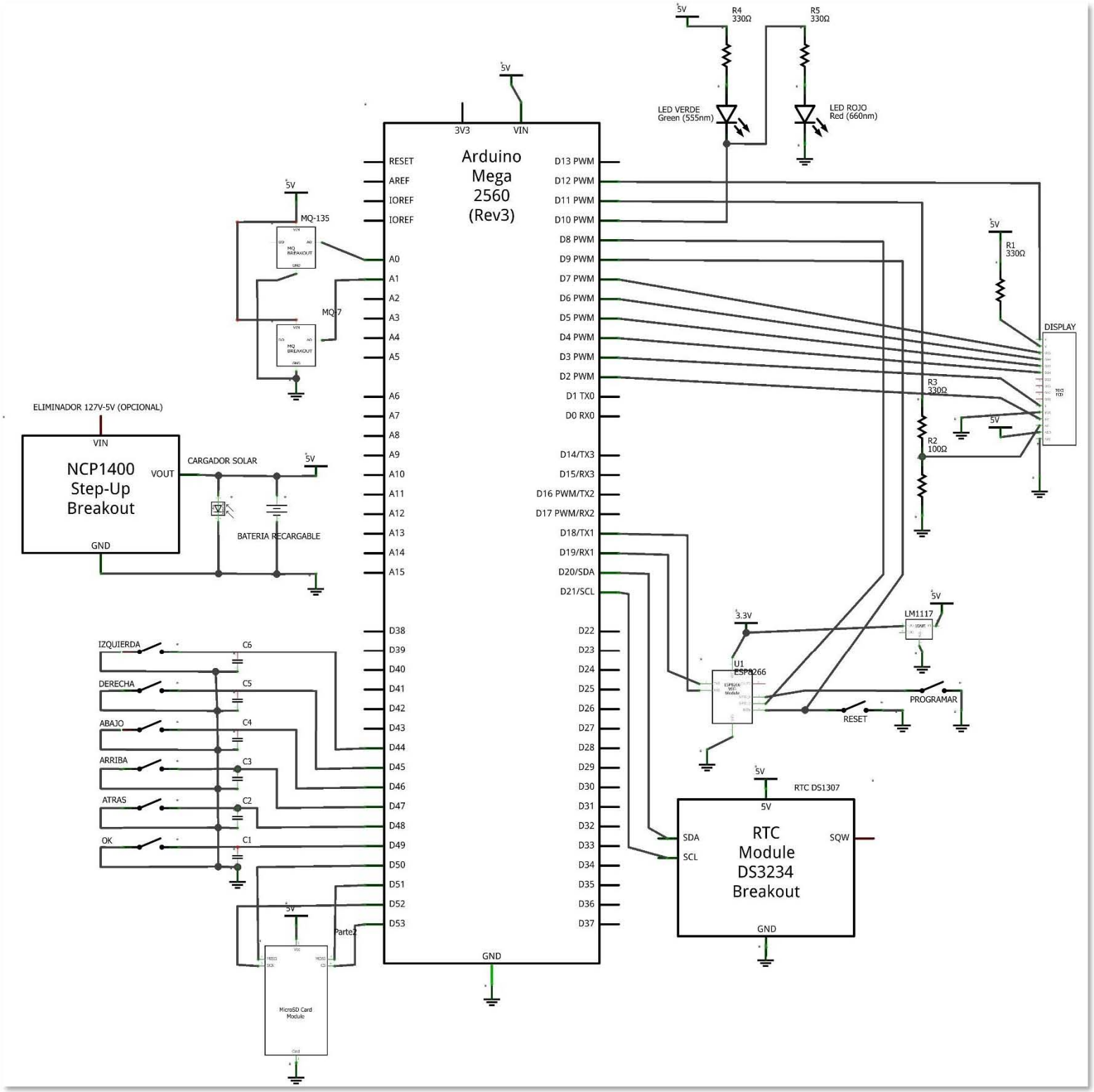


Figura 58. Diagrama esquemático.

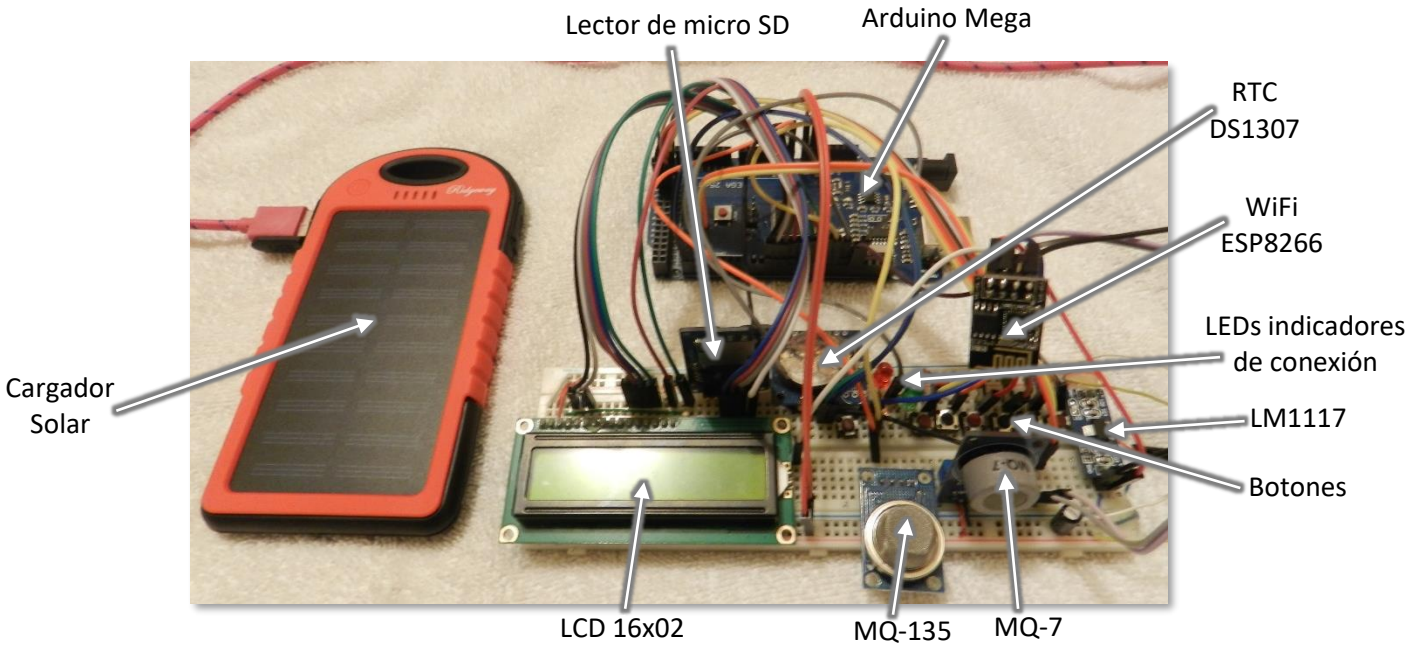


Figura 59. Fotografía con los nombres de los componentes del nodo.

Figura 60. Fotografías de un nodo ya conectado y funcionando.

Referencias

[1] Wang, Y. y Zhang, X. (19 de agosto de 2012). *Internet of Things*. Obtenido de http://link.springer.com/book/10.1007/978-3-642-32427-7

[2] *Revolutionary new weapon in air pollution fight*. (s.f.). Obtenido de https://www.rmit.edu.au/about/our-locations-and-facilities/facilities/research-facilities/rmit-microscopy-and-microanalysis-facility/research/revolutionary-new-weapon-in-air-pollution-fight/

[3] *Sensores personales para ver en el móvil la contaminación del aire.* (27 de diciembre de 2012). Obtenido de http://sociedad.elpais.com/sociedad/2012/12/27/actualidad/1356636990_397497.html

[4] Monks, K. (25 de noviembre de 2015). *Sensores personales: un lector de contaminación podría salvar vidas*. Obtenido de CNN: http://cnnespanol.cnn.com/2015/11/25/sensores-personales-un-lector-de-contaminacion-podria-salvar-vidas/#0

[5] Arenas, M. (12 de noviembre de 2015). *Si quieres conocer el estado del aire que respiras, te decimos cómo puedes hacerlo.* Obtenido de Xataka: http://www.xataka.com/ecologia-y-naturaleza/si-quieres-conocer-el-estado-del-aire-que-respiras-te-decimos-como-puedes-hacerlo

[6] Herzl, D. (31 de julio de 2014). *Mapping How Our Cities Live and Breathe.* Obtenido de Aclima: http://insights.aclima.io/

[7] *Usan en Jalisco sistema de monitoreo para rediseñar control de emisiones*. (03 de diciembre de 2014). Obtenido de Respira México: http://respiramexico.org.mx/2014/12/usan-en-jalisco-sistema-de-monitoreo-para-redisenar-control-de-emisiones/

[8] Méndez, E. (24 de abril de 2016). *Fotomultas, para quien contamine; Semarnat analiza su aplicación en julio*. Obtenido de http://www.excelsior.com.mx/comunidad/2016/04/26/1088859#view-1

[9] Buettrich, S. y Escudero Pascual, Alberto. (2005). Topología e Infraestructura Básica. En *Guías para construir una Red Inalámbrica* Comunitaria (cap. 4). Obtenido de http://www.itrainonline.org/itrainonline/mmtk/wireless_es/files/04_es_topologia-e-infraestructura_guia_v02.pdf

[10] Buettrich, S. y Escudero Pascual, Alberto. (2005). Topología e Infraestructura Básica. En *Guías para construir una Red Inalámbrica* Comunitaria (cap. 4). Obtenido de http://www.itrainonline.org/itrainonline/mmtk/wireless_es/files/04_es_topologia-e-infraestructura_guia_v02.pdf

[11] Cano, J. y Manzoni, P. (2001). *Encaminamiento en las Redes Inalámbricas Ad-hoc*. Obtenido del sitio web de la Universidad Politécnica de Valencia, Departamento de Informática de Sistemas y Computadores: http://www.grc.upv.es/papers/docs/jucano_paper_nov%e1tica.pdf

[12] Navarro Gavira, S. (2006). *Algoritmos Cross-Layer para la Optimización de las prestaciones del TCP en Redes Wireless Ad-hoc* (tesis de licenciatura). Universidad de Sevilla: Escuela Técnica Superior de Ingenieros, Sevilla, España.
[13] Performance Analysis of Three Routing Protocols in MANET (2017). En J. Hamilton Ortiz y A. Pachón de la Cruz, *Ad Hoc Networks* (pág. 48). Rijeka, Croacia: InTech.
[14] Data-Gathering and Aggregation Protocol. (2017). En J. Hamilton Ortiz y A. Pachón de la Cruz, *Ad Hoc Networks* (pág. 4). Rijeka, Croacia: InTech.
[15] Misra, P. (1999). *Routing Protocols for Ad Hoc Mobile Wireless Networks*. Obtenido de http://www.cse.wustl.edu/~jain/cis788-99/ftp/adhoc_routing/
[16] Gutiérrez Reina, D. (s.f.). *Diseño de redes móviles ad hoc en aplicaciones de transporte sobre entornos NS-2*. Obtenido de http://bibing.us.es/proyectos/abreproy/70218
[17] Vega Garabitos, E. (2014). *Desarrollo de protocolo de enrutamiento híbrido basado en zonas para redes MANET* (tesis de maestría). Universidad de Sevilla: Escuela Técnica Superior de Ingenieros, Sevilla, España.
[18] Sivakumar, R.; Sinha, P. y Bharghavan, V. (1999). *CEDAR: A Core-Extraction Distributed Ad hoc Routing algorithm*. Obtenido de https://www.cs.cornell.edu/people/egs/615/sivakumar99cedar.pdf
[19] Ramasubramanian, V.; J. Haas, Z. y Gün Sirer, E. (2003). *SHARP: A Hybrid Adaptive Routing Protocol for Mobile Ad Hoc Networks*. Obtenido de https://www.cs.cornell.edu/people/egs/papers/sharp.pdf
[20] Introduction and overview of wireless sensor networks. (2007). En K. Sohraby, D. Minoli y T. Znati, *Wireless Sensor Networks: Technology, Protocols, and Applications* (pág. 1). Nueva Jersey, EUA: John Wiley & Sons, Inc.
[21] *¿Qué es una Red de Sensores Inalámbricos?* (22 de abril de 2009). Obtenido de National Instruments: http://www.ni.com/white-paper/7142/es/
[22] *Arduino Mega.* (11 de octubre de 2016). Obtenido de https://www.arduino.cc/en/Main/arduinoBoardMega
[23] *Sensor Calidad Aire MQ135*. (9 de octubre de 2016). Obtenido de http://www.naylampmechatronics.com/sensores-gas/73-sensor-calidad-aire-mq135.html
[24] *Technical data mq-7 gas sensor*. (11 de octubre de 2016). Obtenido de https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf
[25] *WiFi Module - ESP8266*. (9 de octubre de 2016). Obtenido de https://www.sparkfun.com/products/13678
[26] *Specification for LCD Module 1602A-1 (V1.2).* (10 de octubre de 2016). Obtenido de https://www.openhacks.com/uploadsproductos/eone-1602a1.pdf
[27] [CNLohr]. (21 de septiembre de 2014). *ESP8266 Wifi Range/Distance Tests (Wi07C)* [Archivo de video]. Obtenido de https://www.youtube.com/watch?v=7BYdZ_24yg0
[28] International Telecommunication Union. (25 de enero de 2007). *Definition of Quality of Experience (QoE)*. Obtenido de https://www.itu.int/md/T05-FG.IPTV-IL-0050/en
[29] *Sensor Calidad Aire MQ135*. (9 de octubre de 2016). Obtenido de http://www.naylampmechatronics.com/sensores-gas/73-sensor-calidad-aire-mq135.html
[30] *Earth's CO2 Home Page*. (agosto 2017). Obtenido de https://www.co2.earth
[31] *Technical data mq-7 gas sensor*. (11 de octubre de 2016).

Obtenido de https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf
[32] Moretton, J. (1996). *Contaminación del aire en la Argentina*. Argentina: Ediciones Universo.
[33] Enciso, L.; Quezada, P.; Fernández, J. y Figueroa, B. (noviembre de 2015). *Análisis de parámetros de desempeño de los protocolos de enrutamiento Ad Hoc aplicado a un entorno urbano.* Obtenido de https://www.researchgate.net/publication/310488189_Analisis_de_parametros_de_desempeno_de_los_protocolos_de_enrutamiento_Ad_Hoc_aplicado_a_un_entorno_urbano
[34] Espíritu Castro, J.; Estrada Burgos, M. y Vázquez González, J. (2012). *Evaluación del desempeño de los protocolos de enrutamiento AODV y DSDV en una red ad hoc* (tesis de licenciatura). Instituto Politécnico Nacional: Escuela Superior de Ingeniería Mecánica y Eléctrica, Ciudad de México, México.