ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"DESARROLLO DE PROTOTIPO PARA MONITORIZACION Y DIAGNOSTICO REMOTO DE PARAMETROS DE MOTORES DE TAXIS PARA MANTENIMIENTO PREVENTIVO"

INFORME DE PROYECTO INTEGRADOR

Previo a la obtención del Título de:

INGENIERO EN TELECOMUNICACIONES

DAVID ENRIQUE FOUNES BURGOS.

HOLGER ESTUARDO VELOZ AGUIRRE.

GUAYAQUIL – ECUADOR

AÑO: 2017

AGRADECIMIENTOS

Este proyecto va dedicado de manera especial a nuestros padres quienes han sido nuestro principal cimiento para la construcción de nuestra vida profesional, quienes con sus consejos y apoyo incondicional nos enseñaron que si tenemos un sueño debemos perseguirlo y conseguirlo, a pesar de las dificultades.

A nuestro tutor Ing. RONALD PONGUILLO y demás docentes partícipes, por habernos permitido a acudir a sus conocimientos, enseñanzas y guías haciendo esta tesis posible.

A nuestros hermanos quienes no nos han dejado bajar los brazos en este escalón de nuestras vidas.

TRIBUNAL DE EVALUACIÓN

.....

ING RONALD PONGUILLO.
PROFESOR TUTOR

.....

ING VLADIMIR SANCHEZ.
PROFESOR COLABORADOR

DECLARACIÓN EXPRESA

"La responsabilidad y la autoría del contenido de este Trabajo de Titulación, nos corresponde exclusivamente; y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

David Founes Burgos.

Holger Veloz Aguirre.

RESUMEN

Se planteó como problemática el acceso hacia valores de parámetros obtenidos de sensores de vehículos, es necesario para los propietarios de los vehículos establecer políticas de alertas sobre mantenimiento preventivo y monitoreo de los parámetros de su motor en base a la observación del comportamiento del mismo.

Se decide aplicar una solución multi-tecnológica que integra electrónica y desarrollo para el web, que implementa la idea de IoT y cumple los objetivos de desarrollo. A través del control mediante ejecución de algoritmos se logró obtener, procesar, enviar y recibir datos de utilidad sobre el motor y su comportamiento. Luego presentar a través de una aplicación los resultados obtenidos.

Se obtuvo un elemento terminal que envía información al servidor del sistema, accesible remotamente a través de una dirección de internet. Este elemento se basó en el uso de la placa ARDUINO MEGA y elementos periféricos o módulos, como un módulo GSM/GPRS, un módulo Bluetooth y un módulo GPS para localización. Se realizó también un servicio para almacenamiento y aplicación web con control de credenciales e información estructurada.

ABSTRACT

The access to parameters values obtained from vehicle sensors was raised as a problematic, it is necessary for vehicle owners to establish warning policies on preventive maintenance and monitoring of the parameters of their motor based on the observation of the behavior of the same.

It was decided to apply a multi-technology solution that integrates electronics and development for the web, which implements the IoT concept and meets the development objectives. Through the control and the implementation of algorithms, it was possible to obtain, process, send and receive useful data about the motor and its behavior. Then present the results obtained through a web application.

The result was a terminal element that sends information to the server of the system, remotely accessible through an Internet address. This element was based on the use of the ARDUINO MEGA board and peripheral elements or modules, such as a GSM / GPRS module, a Bluetooth module and a GPS module for location. A service for storage and web application with control of credentials and structured information was also carried out.

ABREVIATURAS

- **OBD:** ON BOARD DIAGNOSTIC**.**
- **GPS:** SISTEMA DE POSICIONAMIENTO GLOBAL.
- **GSM**: GLOBAL SYSTEM FOR MOBILE.
- **DB**: DECIBELIOS.
- **HTTP**: HYPERTEXT TRANSFER PROTOCOL.
- **UART**: UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER.
- **PC**: PERSONAL COMPUTER

ÍNDICE GENERAL

CAPÍTULO 1

1. GENERALIDADES

Muchos de los productos ofrecidos actualmente contienen algún tipo de control electrónico. Es posible acceder a mediciones sobre su estado o realizar control si se facilita los medios para ello. Esta idea se amplía hacia la mayoría de “cosas” como dispositivos, instalaciones eléctricas entre otras. Nuestro proyecto brinda acceso específicamente a lecturas del set de sensores ubicados en un vehículo.

El planteamiento de la solución es desafiante y obliga a integrar múltiples técnicas en el procedimiento obligándonos a utilizar las habilidades y conocimientos adquiridos previamente.

En este capítulo se incluyen los objetivos, problemática, estado del arte y un análisis sobre el alcance y las limitaciones del proyecto. Se realiza una descripción breve de la necesidad encontrada en la comunidad de taxistas de Guayaquil y cómo poner al servicio de la misma el poder integrador y facilitador de la tecnología actualmente disponible.

1.1 Objetivos.

1.1.1 Objetivo General

Implementar un sistema de monitoreo y diagnóstico mediante la lectura de datos obtenidos a través de una interfaz OBD, para el

posterior envió de incidencias a través de la red GSM a un servidor remoto, conectando a una DB en la nube.

1.1.2 Objetivos específicos.

- Adquirir e interpretar los datos obtenidas a través del conector OBD del vehículo.
- Procesar y parametrizar los datos obtenidos por el conector OBD en el microprocesador.
- Almacenar en DB remota a través de peticiones HTTP con los datos obtenidos del conector ODB usando la banda GSM empleando un módulo SIM900.
- Elaborar reportes del vehículo cuando este fuera de los parámetros para su posterior diagnosis.

1.2 Identificación del problema

Se sabe que cuando alguna pieza en el interior del vehículo tiene algún desgaste o falla de algún tipo se puede producir un daño colateral en algún otro componente interno. Esto invita a pensar: ¿por qué no diagnosticar un vehículo antes de que tenga alguna avería o desgaste grave?, ¿por qué esperar que la magnitud del daño impida la movilización del vehículo o cause un perjuicio irreparable?

A través del monitoreo de los diferentes sensores del automotor que brindan la información necesaria, se puede determinar si los valores de

los mismos se encuentran dentro de los rangos óptimos para el correcto funcionamiento del vehículo.

De la misma manera como un individuo asiste a una cita médica preventiva y lleva un historial clínico, es posible de manera autómata y continua almacenar en una base de datos las anomalías que se presenten, esta información se la puede compartir con técnicos automotrices para que puedan asistir a los usuarios en el problema.

1.3 Estado del arte

Al día de hoy el estándar para el diagnóstico de un automóvil es el OBD II por lo cual se dispone de varias herramientas que permiten conocer y diagnosticar un vehículo, ya sea de manera somera o algo más técnico dentro de talleres y centro de servicios con sistemas profesionales, de esta manera se puede realizar una simple clasificación entre sistema de conexión alámbrica e inalámbrica.

Entre los de conexión alámbrica existen los Scan Tool genéricos que nos presentan una información estandarizada del vehículo, así también Scan Tool del fabricante los cuales integran software específico del fabricante y brindan información detallada y técnica.

Los de conexión inalámbrica requieren un soporte para visualizar los datos obtenidos, el cual puede ser un computador, smartphone o algún otro dispositivo móvil que permita la conexión bluetooth o wifi.

El estándar sigue evolucionando y apuntan al OBD III, el cual está orientado a IoT. No solo obtiene el diagnóstico, se pretende llevar una bitácora más evaluaciones continuas, obtener resultados preventivos y que de manera remota se realice la asistencia del mismo.

Algunas empresas que proporcionan un producto similar pero no igual al que podría implementarse con el desarrollo de esta idea son: Chevystar, Hunter y Top-Ten

1.4 Diseño de la solución

El sistema propuesto se compone principalmente de 4 etapas

- Adquisición de datos
- Procesamiento de datos
- Transmisión de datos al servidor
- Acumulación de resultados y presentación

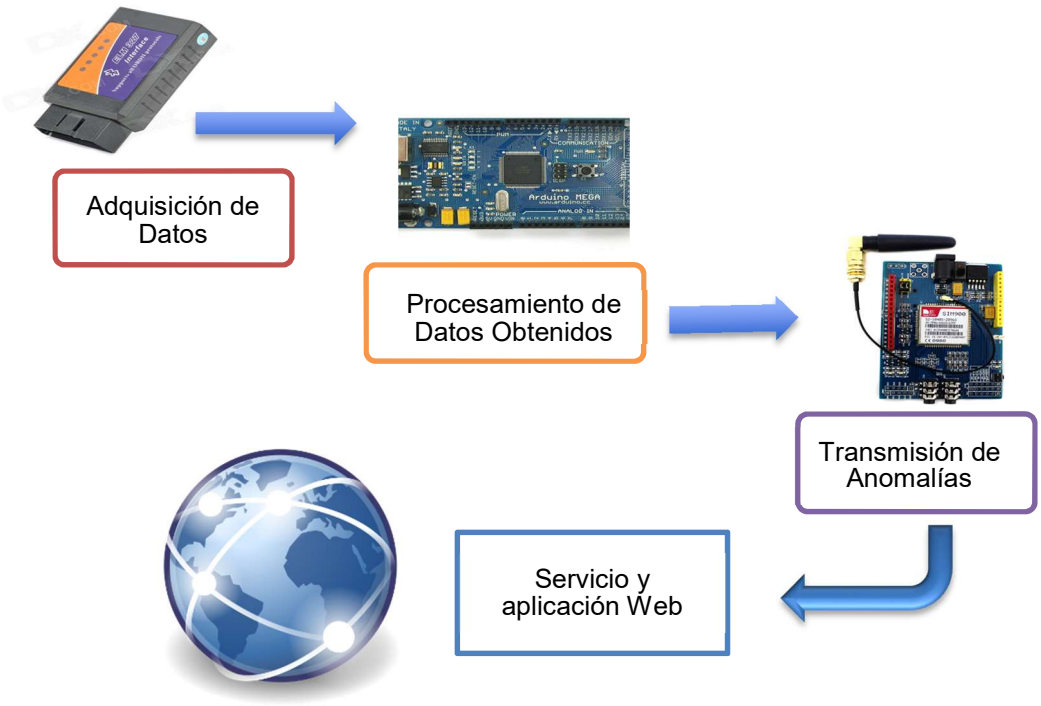

Figura 1.1 Diagrama del Sistema

1.4.1 Adquisición de datos

La interacción con el computador del vehículo se hace a través de la interface OBD II contemplada en el diseño. El uso de dispositivos como este permite abstracción y capacidad para concentrar esfuerzos en los propósitos específicos y relevantes del desarrollo. Esta etapa debe controlarse la correcta configuración del enlace con la interfaz OBD correspondiente.

Esta parte del diseño se la realizará con el algoritmo lector que se implementará sobre la placa ARDUINO.

1.4.2 Procesamiento de datos

Los datos adquiridos deben sufrir un proceso de verificación para realizar el diagnóstico automático de posibles fallas en el vehículo. Esto se efectuará mediante un algoritmo de proceso alojado en la memoria de programa de la placa ARDUINO, realizando la comparación de las lecturas obtenidas con valores referenciales específicos y en rango.

1.4.3 Transmisión de los datos al servidor

En este diseño se explota el módulo GSM para realizar una petición HTTP posteriormente con el dispositivo. Dentro de la petición se enviará un identificador único del vehículo y los valores de las lecturas que incluyen el resultado de la consulta al computador del automotor. Se pretende en líneas generales que mediante un algoritmo se configure y controle el módulo GSM para que haga un requerimiento web con el texto que nosotros le especifiquemos en la trama

1.4.4 Acumulación de resultados y presentación

La petición HTTP que se realiza será recibida en un servidor web listo para procesarla, verificando la integridad de las mismas, extrayendo el valor de las lecturas y almacenando en tablas estructuradas de una DB los valores recibidos.

Estos registros pueden ser accedidos con una interfaz simple y orientada al usuario que mediante la ejecución de consultas tipo SQL nos permitan obtener información clave que pueda ser de interés para los operadores del servicio que estamos brindando, a

los clientes finales o a la gerencia para tomar decisiones administrativas.

La tecnología a usarse para esta etapa se definió como:

- Aplicación web hecha con PHP montada en un servidor (APACHE) alojado en un dominio accesible al usuario final.
- Base de Datos en Motor MySQL.
- Servicio Web escrito en PHP destinado a recoger la lectura de los sensores y grabarlas estructuradamente. Alojado en un dominio accesible al usuario final.

En resumen, para el desarrollo del sistema se creará un prototipo que use ARDUINO como dispositivo de adquisición y proceso de los valores que se obtienen a través de la interface OBD II. Estas lecturas tendrán un retardo configurable. Se preparará una trama que después de la conexión a la red GSM permitirá realizar una petición HTTP hacia un servidor en la nube. El servidor web albergará la DB que alimentará la aplicación encargada de presentar información clave para que el incidente se identifique y sea atendido por el operador pertinente. Además en todo este proceso se deben emitir alarmas al usuario que lo alerten sobre el mal funcionamiento del vehículo.

1.5 Alcance y limitaciones

El proyecto se encuentra enmarcado en adquirir los datos necesarios para el diagnóstico en tiempo real del estado del motor de un vehículo Chevrolet Aveo / Hyunday Accent, estos datos serán las RPM del motor, temperatura

de aceite, carga y presión de aceite, misma información que será adquirida a través de la interfaz OBD.

En ninguno de los casos se pretende diseñar o elaborar alguna clase de sistema embebido que realice el sistema descrito, a partir de módulos y placas comercializadas. Se implementará el sistema expuesto realizando una configuración diferente dándole una tónica orientada a ODB III, de monitoreo y diagnosis remota.

Estos parámetros serán comparados por el microcontrolador, de estar fuera de los rangos o valores referenciales la información será enviada a través de la red GSM al servidor remoto para su posterior análisis.

El servidor remoto recibirá la información, y a manera de bitácora mantendrá un registro de las incidencias que se hayan presentado. Dicha información técnica es importante para una inspección técnica que se realice en un centro automotriz especializado.

La limitante será llegar a proporcionar la información del monitoreo de los parámetros de temperatura, carga y presión de aceite del/los vehículos mencionados, diagnosticar incidencias y emitir alertas en tiempo real automáticamente con el sistema a bordo del vehículo. De ningún modo se pretende dar solución a las averías que pueda sufrir el automotor. Los datos de las incidencias deben ser analizadas por un técnico especializado en la rama automotriz para dar la solución al mismo, así también la plataforma web queda disponible para suministrar los datos necesarios para su análisis.

CAPÍTULO 2

2 MARCO TEÓRICO

A lo largo de la última década se han multiplicado los usuarios de tecnologías móviles basadas en internet. El uso de dispositivos con conexión inalámbrica y la red celular expandida en casi el 100% del territorio, acercan a las personas y ponen a disposición su aporte en el desarrollo comunitario. Nuestra propuesta busca justamente eso, inculcar en la comunidad el uso de sistemas de almacenamiento remoto. Es su vehículo quien le brinda el acceso para ejecutar un diagnóstico basado en sensores, guiado por un conjunto de reglas y conocimientos técnicos relativos al automotor.

Se debió establecer para este propósito los elementos apropiados y compatibles a través de investigación exhaustiva, basando métodos en conocimientos previamente desarrollados en proyectos similares.

En este capítulo se describe brevemente los fundamentos de uso, características técnicas y observaciones sobre los elementos del prototipo. Se hace también referencia de las cualidades del dispositivo que nos orientaron a elegirlo por sobre sus similares.

2.1 Arduino Mega

El ARDUINO MEGA es una placa basada en el microcontrolador ATmega2560. Es una placa diseñada para implementar prototipos. En este proyecto se usa este dispositivo porque nos ofrece 4 puertos seriales UART, ideal para nuestros propósitos.

Se elige este dispositivo por sus especificaciones técnicas como el voltaje de operación, el número de puertos UART, su capacidad en memoria FLASH y memoria SRAM, también la versatilidad que nos permiten sus pines para ser usados como entrada o salida de distintos tipos

La facilidad de uso ha popularizado el empleo de estas placas pues vienen integradas con la tecnología periférica suficiente para ayudarnos a construir el prototipo. La placa ARDUINO se ha vuelto sin lugar a dudas en un elemento común y muy eficaz en el desarrollo de innovaciones tecnológicas, sin embargo su uso en la industria es escaso prefiriendo placas PLC más complejas. En la figura 2.1 podemos observar el ARDUINO MEGA visto desde arriba con sus 4 puertos seriales, alimentación y demás elementos característicos de la placa ARDUINO.

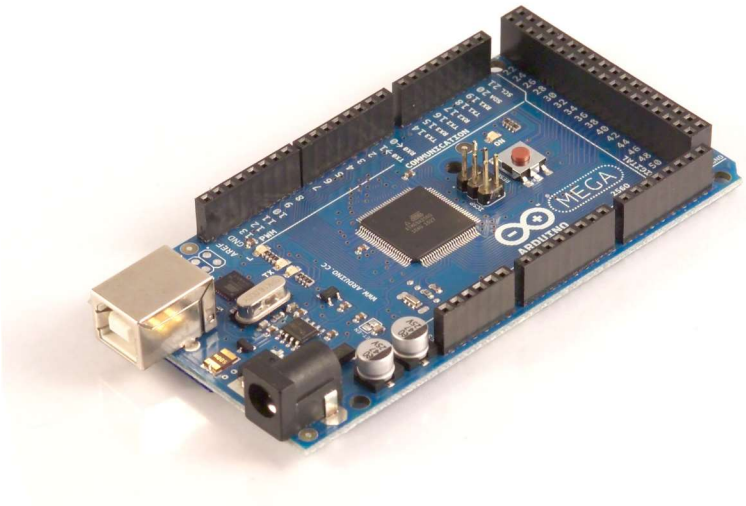

Figura 2.1 ARDUINO MEGA

2.2 Módulo Bluetooth HC-05

Figura 2.2 BLUETOOH HC-05

El **Bluetooth HC-05** puede configurarse tanto como Master que como Slave, dispone de muchos parámetros de configuración y capacidad de interrogación.

Su aspecto externo es bastante similar al modelo HC-06 pero se diferencian en la cantidad de pines mientras el HC-06 consta de 4 pines el HC-05 consta de 6 pines. **(Fig. 2.3)**

El procedimiento con estos módulos, suele ser: conectarlos, ver la configuración y reprogramarlos con nuestras preferencias. Después mantendrá la programación que diseñamos para ejecutar hasta que decidamos cambiarla.

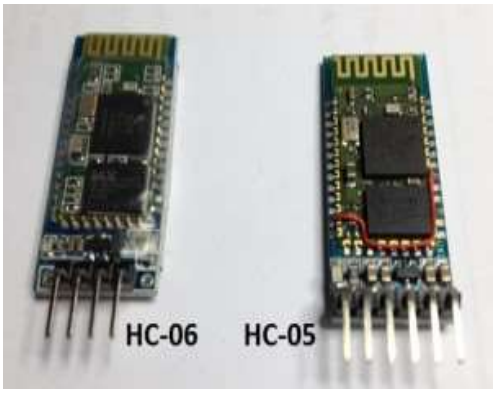

Figura 2.3 DIFERENCIA DE PINES ENTRE HC-06 Y HC 05

CARACTERISTICAS DESEABLES EN EL PROTOTIPO:

- ✓ Puede configurarse como maestro, esclavo, y esclavo con autoconexión
- ✓ Frecuencia: 2.4 GHz, banda ISM
- ✓ Antena de PCB incorporada
- ✓ Alcance 5 m a 10 m
- ✓ Sensibilidad: ≤ -84 dBm a 0.1% BER
- ✓ Seguridad: Autenticación y encriptación (Password por defecto: 1234)
- ✓ Consumo de corriente: 50 mA
- ✓ Voltaje de alimentación: 3.6 V a 6 V
- ✓ Temperatura de operación: -20 ºC a +75 ºC.
- ✓ Utiliza comandos At.

2.3 Módulo GPS GPS6MV2

Figura 2.4 Módulo GPS GPS6MV2

El modelo GY-GPS6MV2 de sensor GPS viene con un módulo de serie U-Blox NEO 6M equipado en el PCB, EEPROM con configuración de fábrica, mas pila de botón para mantener los datos de configuración en la memoria EEPROM, indicador LED y antena cerámica. Posee los pines o conectores Vcc, Rx, Tx y Gnd por el que se puede conectar a algún microcontrolador mediante una interfaz serial. Para que nuestro módulo GPS funcione a la perfección se lo debe ubicar preferentemente en un ambiente abierto o cercano a la ventana para una correcta recepción de la señal satelital.

Las características técnicas deseables de este dispositivo en el prototipo son su voltaje de alimentación, la interfaz UART que implementa comunicación asíncrona y su antena externa de porcelana

2.4 Interface ELM - 327

Figura 2.5 Interface ELM – 327 OBD-II

Casi todos los automóviles producidos hoy en día necesitan por ley, proporcionar una interfaz de la que los equipos de prueba puedan obtener información diagnóstica

La transferencia de datos en estas interfaces sigue varios estándares, ninguno de los cuales son directamente compatibles con PCs o PDAs. El ELM327 está diseñado para actuar como puente entre estos diagnósticos a bordo (OBD) puertos y un interfaz estándar RS232.

El ELM327 se basa en versiones mejoradas de interfaces probadas de ELM320, ELM322 y ELM323 añadiendo siete protocolos de Can a ellos. El resultado es un IC que puede detectar y convertir automáticamente los protocolos más comunes en uso hoy en día. Hay una serie de otras mejoras también velocidad alta en RS232, supervisión del voltaje de la batería, y funciones personalizables mediante parámetros programables, por nombrar sólo algunos.

El ELM327 requiere pocos componentes externos para hacer un circuito completamente funcional.

Como características deseables en el diseño tenemos el amplio número de protocolos que soporta, velocidad de trasmisión en UART, reconoce automáticamente el protocolo que recibe, es totalmente configurable a través de

comandos AT, su diseño permite bajo consumo de potencia y su alimentación coincide con las características del sistema que se está implementando.

2.5 SIM900 (SimCom)

Módulo GSM / GPRS ultra compacto con el SIM900 tipo “Cuatri Banda” en montaje superficial y diseñado con un “Core” AMR926EJ-S, un procesador “single – chip” muy poderoso que permite mayores prestaciones que el promedio de los módulos existentes en el mercado con mayor velocidad de procesamiento y un significativo ahorro de energía en el modo “sleep” en comparación con la media del mercado.

Con una interface estándar del mercado, el SIM900 entrega una alta prestación en los modos GSM / GPRS en las bandas de 850/900/1800/1900 MHZ para voz, SMS, Datos, y Fax. Todo en un factor de forma muy pequeño de tan solo 24 mm x 24 mm x 3 mm y con la facilidad de montaje SMD (superficial) con “pads” en los 4 laterales, lo que lo hace muy práctico para la producción automatizada o semi – manual.

CAPÍTULO 3

3. METODOLOGÍA

Es importante establecer cómo fueron encajando las piezas del rompecabezas, la disposición de los elementos para hacer funcionar cada una de las partes que en su conjunto dieron luz a la solución.

Para ello dividimos la tarea global (medir y almacenar en la nube) en partes de similar dificultad y dispuestas en cascada secuencial una después de la otra. Así se pudo encarar el problema de manera más sencilla, preocupándonos por obtener la solución de aquel pequeño bloque, esto es, empleamos la modularización del problema aplicando el paradigma, “divide y vencerás”

En este capítulo se describe de manera más amplia las técnicas usadas en el desarrollo de la solución. Se describen también algunos de los algoritmos empleados en cada una de las etapas del sistema propuesto

Con esta propuesta tecnológica se plantea la respuesta a la necesidad del taxista ecuatoriano de detectar a tiempo posibles fallas en su vehículo que se debe recordar, no solo es su herramienta de trabajo, también es un medio de transporte en el que se movilizan seres humanos y por lo tanto debe estar en óptimas condiciones. Se logró esto enviando alertas al usuario final para evitar: accidentes por falla de vehículo, colapso total del automóvil por daños irreversibles al motor, días enteros en el taller y percances económicos no previstos.

En este sistema se pueden notar 4 etapas bien definidas. A saber:

ETAPA 1: Conexión con la computadora del PC a través de un adaptador

ETAPA 2: Extracción de lecturas de sensores

ETAPA 3: Enviar información a nube controlando un módulo GSM

ETAPA 4: Proceso y presentación de la información con una Aplicación Web

La implementación de estas 4 etapas se realizó a través de un prototipo que realiza las etapas de la 1 a la 3. La última etapa descrita obligó a la construcción de una plataforma Web y un servicio destinado a la recolección de los datos y su almacenamiento estructurado.

El prototipo tiene un algoritmo implementado en lenguaje C destinado a ejecutarse desde el llamado a dos rutinas que son típicas de ARDUINO. A saber setup() y loop(). setup() se ejecuta una sola vez y loop() continuamente mientras el dispositivo se encuentre encendido. Dentro de setup() se alberga la configuración de los enlaces entre los módulos del prototipo. Luego se realiza el uso de los mismos para los propósitos del diseño. Estas instrucciones se implementaron como rutinas que se describen a continuación en el desglose de cada una de las etapas.

La implementación para el internet se realizó en php y se publicó usando un servicio de hosting gratuito (www.000webhostapp.com). El motor de DB escogido es MySQL. Entre los motivos para la elección de esta tecnología tenemos:

- Bajos costos de implementación
- Escalabilidad
- Tecnología Liberada
- Capacidades razonables respecto a nuestro propósito

3.1 ETAPA 1: Conexión al ECU a través de Bluetooth

Como se explicó anteriormente, la conexión al ECU se produce a través de una interface basada en el chip ELM-327 que realiza el puente de comunicación desde OBD que implementa el estándar del vehículo que deseamos diagnosticar hacia un otra interfaz RS-232 preparada para ser leída por la mayor parte de dispositivos a través de

comunicación UART. Además este dispositivo en particular brinda conectividad a través de Bluetooth lo que ofrece una importante ventaja a la hora de ubicar estratégicamente el dispositivo para que el mismo no sea localizado fácilmente.

El uso de este canal de comunicación inalámbrica se logra manipulando la librería HardwareSerial que permite abstracción sobre el uso de comunicación serial simple, en este caso usamos el puerto serial 1 del ARDUINO MEGA al que se define con el nombre de variable “bt”. “ConfBT()” implementa la configuración inicial destinada a ejecutarse una sola vez cuando se la llama a través de setup(). En ella se observa la comunicación con comandos AT y retardos hacia y desde el módulo HC-05 usando la rutina EnvATbt(). Entre otras cosas se lo configura como MASTER, se inicializa la taza de baudios para UART y se empareja con el OBDII

```
1.  void ConfBT(){
2.
3.    bt.flush(); // limpia el buffer de salida
4.    delay(500);
5.    digitalWrite(EnBT, HIGH);
6.  //Ingreso al modo AT del modulo HC-05
7.    delay(500);
8.    bt.begin(38400);
9.  //Configuracion por defecto de la tasa de baudios (38400) del del modulo HC-05
10.   delay(500);
11.
12.   EnvATbt("AT+RESET");
13. //Envio AT+RESET, reinicia\r\n
14.   delay(1000);
15.   EnvATbt("AT+ORGL");
16. //Envio AT+ORGL, propiedades originales
17.   //poner el baud rate de mode comunicacion
18.   //EnvATbt("AT+UART=38400,0,0");
19.   EnvATbt("AT+ROLE=1");
20. //Envio AT+ROLE=1, configuracion como rol master
21.   EnvATbt("AT+CMODE=0");
22. //Envio AT+CMODE=0, configuracion en modo de una especifica direccion
23.   EnvATbt("AT+BIND=001D,A5,000DD3");//98d3,31,8094f6
24. //Envio AT+BIND=001D,A5,000DD3, direccion MAC del dispositivo OBDII bluetooth
25.   EnvATbt("AT+INIT");
26. //Envio AT+INIT, comando que inicializa la vinculacion
27.   delay(1000);
28.   EnvATbt("AT+PAIR=001D,A5,000DD3,20");//98d3,31,8094f6
29. //Envio AT+PAIR=001D,A5,000DD3,20, empareja con la direccion del OBDII
30.   delay(1000);
31.   EnvATbt("AT+LINK=001D,A5,000DD3"); // 001D,A5,000DD3
32. //Envio AT+LINK=001D,A5,000DD3, enlace con la direccion del OBDII
33.   delay(1000);
34.   // ?? ENTRA AUTOMATICAMENTE A MODO COMUNICACION? LUEGO DE VINCULAR??
35.   bt.flush();
36.   delay(500);
37. }
38.
39. void EnvATbt(char *command)
40. {
41.   char rec;
```

```
42.   char str[2];
43.   int i;
44.   boolean OK;
45.     OK=false;
46.     while (!(OK)){
47. //Mientras la bandera OK sea falsa, HC-05 no continua con los comandos
48.        if(strlen(command) > 1)
49.          bt.println(command); pc.println(command);
50. //Envio el comando al HC-05
51.       while (bt.available()<=0);
52. //Espero mientras no halla datos
53.       if (bt.find("OK")) OK=true;
54.     } pc.println("OK");
55.   delay(500);
56.    }
```

3.2 ETAPA 2: Extracción de lecturas de sensores

Para este objetivo se debe configurar y usar comunicación con el ECU utilizando comandos OBD específicos. Se utiliza una rutina propia llamada EnvCmdOBD para enviar estos comandos y esperar por la respuesta para finalmente reconocer su carácter de terminación de frase (“>”) y devolver al PC la frase OK

La configuración inicial a ser ejecuta en setup() se realiza con la rutina ConfOBD y la lectura con CalcParam

```
1. void ConfOBD(){
2.
3.   EnvCmdOBD("ATZ");       //Envio a OBD ATZ, Restauracion
4.   EnvCmdOBD("ATSP0");     //Envio ATSP0, Auto Protocolo
5.   EnvCmdOBD("0100");      //Envio 0100, disponibilidad de los PID 00-19
6.   EnvCmdOBD("0120");      //Envio 0120, disponibilidad de los PID 20-39
7.   EnvCmdOBD("0140");      //Envio 0140, disponibilidad de los PID 40-??
8. }
9.
10.    void EnvCmdOBD(char *obd_cmd){
11.      char buff[32];
12.      int i=0;
13.      boolean prompt;
14.        prompt=false;
15.    while(!prompt){
16.          bt.print(obd_cmd); bt.print("\r"); pc.print(obd_cmd);

17.          while (bt.available() <= 0);
18.          //necesario un retardo para que el obd responda
19.          pc.println(" OK");
20.
21.          if (bt.available() > 0) {
22.              i = bt.readBytesUntil('>', buff, 30);  // Numero de elementos de la
cadena hasta el caracter '>'
23.              buff[i+1] = '\0';                          // Termino de cadena String
24.           }
```

```
25.          if (buff[i] = '>') prompt=true;
26.        } pc.print(buff); pc.println("OK");delay(1500); //tiempo necesario para que
responda el OBD
27.    }
28.
29.
30.
31.    int CalcParam(char *obd_cmd){   // no es necesario que sea una funcion tipo entero
32.      char buff[60];
33.      int i=0;
34.      boolean prompt;
35.        //bt.begin(38400);  esto esta de mas
36.        prompt=false;
37.
38.          while(!prompt){
39.          bt.print(obd_cmd); bt.print("\r"); pc.print(obd_cmd);

40.          while (bt.available() <= 0);
41.          //necesario un retardo para que el obd responda
42.          pc.println("OK");
43.
44.              if (bt.available() > 0) {
45.                  i = bt.readBytesUntil('>', buff, 58);  // Numero de elementos de la
cadena hasta el caracter '>'
46.                  buff[i+1] = '\0'; // pc.println(buff);                          //
Termino de cadena String
47.               }
48.              if (buff[i] = '>') prompt=true;
49.            }
50.            pc.print(buff);pc.println("OK"); delay(2000); // Retardo importante
ERROR(0) no recibe datos
51.
52.            if ((buff[6]=='4') && (buff[7]=='1') && (buff[9]=='0') && (buff[10]=='4'))
{
53.            EngLoa[0]=buff[12]; EngLoa[1]=buff[13]; EngLoa[2]='\0'; pc.println(EngLoa)
;//hasta aqui bien presenta en hexadecimal las lecturas
54.
55.            }
56.            if ((buff[6]=='4') && (buff[7]=='1') && (buff[9]=='0') && (buff[10]=='5'))
{
57.            EngTmp[0]=buff[12]; EngTmp[1]=buff[13]; EngTmp[2]='\0';pc.println(EngTmp);
58.            //return (strtol(tmp, 0, 16)-40); //EngTmp=
59.            }
60.            if ((buff[6]=='4') && (buff[7]=='1') && (buff[9]=='0') && (buff[10]=='B'))
{
61.            ManPre[0]=buff[12]; ManPre[1]=buff[13]; ManPre[2]='\0';pc.println(ManPre);
62.            }
63.            if ((buff[6]=='4') && (buff[7]=='1') && (buff[9]=='0') && (buff[10]=='C'))
{
64.            EngRpm[0]=buff[12]; EngRpm[1]=buff[13]; EngRpm[2]=buff[15]; EngRpm[3]=buff
[16]; EngRpm[4]='\0';pc.println(EngRpm);
65.
66.            }
67.    }
68.
```

Notar que la función CalcParam realiza la operación necesaria para extraer el valor real del parámetro y también se encarga de almacenar en el correspondiente buffer el valor de la lectura que se desea obtener.

Se denotó también durante el desarrollo del proyecto, que era necesario poder ofrecer como información adicional, una lectura más de las que ya brindaba el sistema, un parámetro que no va alineado con el diagnóstico remoto pero que sin embargo nos ofrece información de utilidad para brindar el servicio que se pretende implementar. La localización no es solo un parámetro de alto interés, también puede ser crucial si se trata de robos, atención en carretera, rastreo de ruta, cálculo de tiempos estimados de llegada, etc. Para ello se decidió emplear un módulo GPS para ARDUINO que realice las veces de sensor GPS y cuya lectura pueda brindar esta información.

Es de importancia mencionar que se intentó utilizar la funcionalidad de la tarjeta SIM-900 pero debido a políticas de las operadoras en nuestro país (ECUADOR), el servicio de localización a través de triangulación con las estaciones base no es abierto para los usuarios y su utilización debe ser desbloqueada por la operadora para ser tarifada adecuadamente.

El módulo anteriormente mencionado se controla con el puerto serial 3 del ARDUINO MEGA, es usado con la librería HardwareSerial y se lo nombró en el programa como “gps”.

Su manipulación se implementa mendiante la rutina GPS, en donde se realiza la respectiva lectura, se convierte en unidades y se llena el buffer adecuado para su posterior subida a internet. En el caso de suscitarse algún error se devuelve la frase “Dispositivo no encontrado”.

```
1. void GPS(){
2.   int k,j;
3.   char graLat[3], minLat[3], segLat[8], oLat, graLon[4], minLon[3], segLon[8], oLon;
4.   char bu [508]="", c ='°';
5.     j=0;
6.   while((gps.available()>0) && (j<249)){
7.     bu[j]=gps.read(); j++;
8.     }pc.println(bu);
9.
10.        if(bu[0]=='$' && bu[1]=='G' && bu[2]=='P' && bu[3]=='R' && bu[4]=='M' && bu[5]
=='C', bu[17]=='A'){
11.            graLat[0]=bu[19]; graLat[1]=bu[20]; graLat[2]='\0';
12.            minLat[0]=bu[21]; minLat[1]=bu[22]; minLat[2]='\0';
13.            segLat[0]='0'; segLat[1]='.';segLat[2]=bu[24]; segLat[3]=bu[25];segLat[4]=
bu[26]; segLat[5]=bu[27]; segLat[6]=bu[28];segLat[7]='\0';
14.            oLat=bu[30];
```

```
15.            j=60*atof(segLat);
16.            graLon[0]=bu[32]; graLon[1]=bu[33]; graLon[2]=bu[34]; graLon[3]='\0';
17.            minLon[0]=bu[35]; minLon[1]=bu[36];  minLon[2]='\0';
18.            segLon[0]='0'; segLon[1]='.'; segLon[2]=bu[38]; segLon[3]=bu[39];segLon[4]
=bu[40]; segLon[5]=bu[41]; segLon[6]=bu[42];segLon[7]='\0';
19.            oLon=bu[44];
20.            k=60*atof(segLon);
21.
22.            sprintf(ubiGPS,"%s%c%s'%d\"%c,%s%c%s'%d\"%c",graLat,c ,minLat ,j,oLat, gra
Lon,c ,minLon ,k,oLon);
23.            pc.println(ubiGPS);
24.            pc.println("OK");
25.
26.
27.            if(oLat=='S') oLat='-';else oLat='+';  if(oLon=='W') oLon='-
';else oLon='+';
28.            char tmp[10];
29.            tmp[0]=bu[21];tmp[1]=bu[22];tmp[2]=bu[24];tmp[3]=bu[25];tmp[4]=bu[26];tmp[
5]=bu[27];tmp[6]=bu[28];tmp[7]='\0';
30.            j=atoi(tmp)/60;
31.            tmp[0]=bu[35];tmp[1]=bu[36];tmp[2]=bu[38];tmp[3]=bu[39];tmp[4]=bu[40];tmp[
5]=bu[41];tmp[6]=bu[42];tmp[7]='\0';
32.            k=atoi(tmp)/60;
33.            sprintf(lat,"%c%s.%d", oLat, graLat, j);
34.            sprintf(lon,"%c%s.%d", oLon, graLon, k);
35.
36.         }
37.        else{
38.         pc.println("Dispositivo no encontrado");
39.         sprintf(lat,"0.0000");
40.         sprintf(lon,"0.0000");
41.         }
42.        delay (1000);//Serial.println(bu);
43.     }
44.
```

3.3 ETAPA 3: Enviar información a Internet controlando un módulo GSM

Para esta etapa del diseño se utilizó el módulo SIM-900. Algunas configuraciones tuvieron que ser efectuadas. Destacan la rutina ConfBP para configurar el acceso a internet por GPRS, GETHTTP para realizar el request hacia el servicio web y EnvATgsm para el envío de comandos AT, entre otras subrutinas usadas recursivamente en el algoritmo. La configuración inicial se realiza en setup() a través del encendido del módulo y la rutina de configuración, el request es labor de GETHTTP. Muy importante el uso de retardos adecuados

```
1.  void EnvATgsm(char *command)
2.  {
3.    boolean OK;
4.      OK=false;
5.      while (!(OK)){                                  //Mientras la bandera OK sea
    falsa, HC-05 no continua con los comandos
6.         if(strlen(command) > 1)
7.           gsm.println(command); pc.println(command);  //Envio el comando al HC-05
8.        while (gsm.available()<=0);                    //Espero mientras no halla
    datos
9.        if (gsm.find("OK")) OK=true;
```

```
10.     } pc.println(gsm.read());
11.   delay(500);  //3000
12. }
13. void ConfBP(){
14.   gsm.flush();                                          //Limpeza del buffer
15.   //delay(5000);
16.   EnvATgsm("AT+SAPBR=3,1,\"Contype\",\"GPRS\"");                  //Configure
    bearer profile 1
17.   EnvATgsm("AT+SAPBR=3,1,\"APN\",\"internet.movistar.com.ec\"");  //Configuracion
    del APN del operador
18.   EnvATgsm("AT+SAPBR=3,1,\"USER\",\"movistar\"");                 //Configuracion
    del USER
19.   EnvATgsm("AT+SAPBR=3,1,\"PWD\",\"movistar\"");                  //Configuracion
    del PASSWORD
20.   EnvATgsm("AT+SAPBR=1,1");                                       //To open a GPRS
    context
21.   EnvATgsm("AT+SAPBR=2,1");                                       //To query the
    GPRS context
22.   }
23.
24. // HTTP GET Method
25. void GetHTTP(char *buf){  //char *prm1, char *prm2char *par1, char *par2
26.   boolean OK;
27.   gsm.flush();                    //Limpia el buffer salida
28.   EnvATgsm("AT+HTTPINIT");        //Inicia la sesion HTTP
29.   EnvATgsm("AT+HTTPPARA=\"CID\",1");
30.   EnvATgsm(buf);
31.   gsm.println("AT+HTTPACTION=0"); pc.println("AT+HTTPACTION=0");  //Envio el
    comando AT+HTTPACTION=0 al SIM900
32.       while (gsm.available()<=0);                               //Espero
    mientras no halla datos
33.       OK=false;while(!OK){if (gsm.find("OK")) OK=true;}         //Mantengo en el
    ciclo hasta hallar la cadena "OK"
34.       OK=false;while(!OK){if (gsm.find("200")) OK=true;}        //Mantengo en el
    ciclo hasta hallar la cadena "200"
35.       pc.println(gsm.read());
36.   EnvATgsm("AT+HTTPREAD");        //No es necesario pero retornara los datos
    enviadosvoid OnOffGSM(){
37.
38.     digitalWrite(EnGSM, HIGH);  //Encendido por software
39.     delay(500); gsm.begin(19200);               //se envia un pulso para el
    encendido
40.     digitalWrite(EnGSM, LOW);   //como apagado del SIM900
41.     delay(500); gsm.begin(19200);
42. }
43.
44. void EncGSM(){
45.     OnOffGSM();
46.     boolean OK;
47.     OK=false;
48.     while (!(OK)){                                  //Mientras la bandera OK sea
    falsa, HC-05 no continua con los comandos
49.       while (gsm.available()<=0);                    //Espero mientras no halla
    datos
50.       if (gsm.find("Ready")) OK=true;
51.     } pc.println(gsm.read());
52.   }
53.
54.   EnvATgsm("AT+HTTPTERM");        //Cierra la sesion HTTP
55.   digitalWrite(LedHTTP, HIGH);delay(200);digitalWrite(LedHTTP, LOW);
56. }
```

Por último se tiene la rutina NotSms que brinda la capacidad de enviar un SMS personalizado a un número celular válido. En este caso se configuró el SMS para que su contenido y destinatario fuera constante.

```
57. /* Depende del cod se enviara un mensaje personalizado*/
58. void NotSms(){ //char *cod
59.     EnvATgsm("AT+CMGF=1");
60.     gsm.println("AT+CMGS=\"+593995348013\""); pc.println("AT+CMGS=\"+593992879609
    \""); delay(500);   //Envio el comando OBD 992879609
61.     gsm.print("ALERTA DE MAL FUNCIONAMIENTO DE SU AUTOMOVIL");
62.     gsm.println((char)26); // Enviar el menzaje y terminar con el Ascci CTRL + Z
    ASCII DE CONTROL Z 0x1A  26 decimal
63.     pc.print("ALERTA DE MAL FUNCIONAMIENTO DE SU AUTOMOVIL");
64.     gsm.println();delay(4000);  // Esperamos un tiempo para que envíe el SMS
65.   }
```

3.4 ETAPA 4: Proceso y presentación de la información con una Aplicación Web

Se realizó esta última etapa mediante la implementación de un servicio web y una plataforma para visualizar información, observando la necesidad de que existan usuarios, vehículos y cooperativas asociados y relacionados entre sí, los datos almacenados respecto a esta entidades sumados a una selección de archivos en el servidor FTP permitieron brindar una interfaz comprensible e intuitiva para uso principalmente de profesionales del volante.

Para la creación de la plataforma vimos la necesidad de crear una página para login (identificación), una página para darse de alta (logout) y redirecciónar, una página de visualización de contenido con sistema de credenciales. Al hacer request solamente un usuario identificado puede obtener la información de el/los vehiculo(s). Esta funcionalidad fue posible gracias al objeto SESSION que no es borrado al recargar la página. Más adelante se ofrece una descripción más detallada de la lógica implementada en estas páginas dentro del apartado correspondiente.

3.5 Diagrama de Flujo de código del prototipo:

3.5.1 setup()

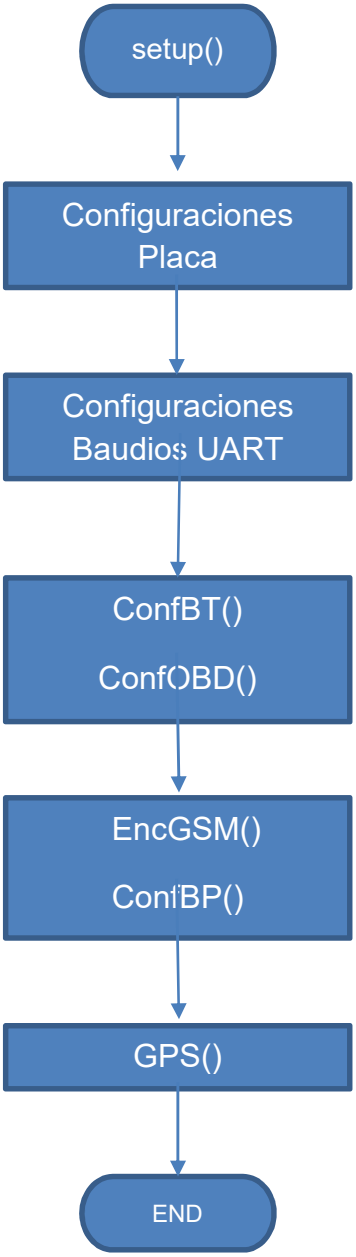

Figura 3.1 Diagrama de Flujo de setup()

3.5.2 loop()

Nota: Mod es el pin 7 de la placa

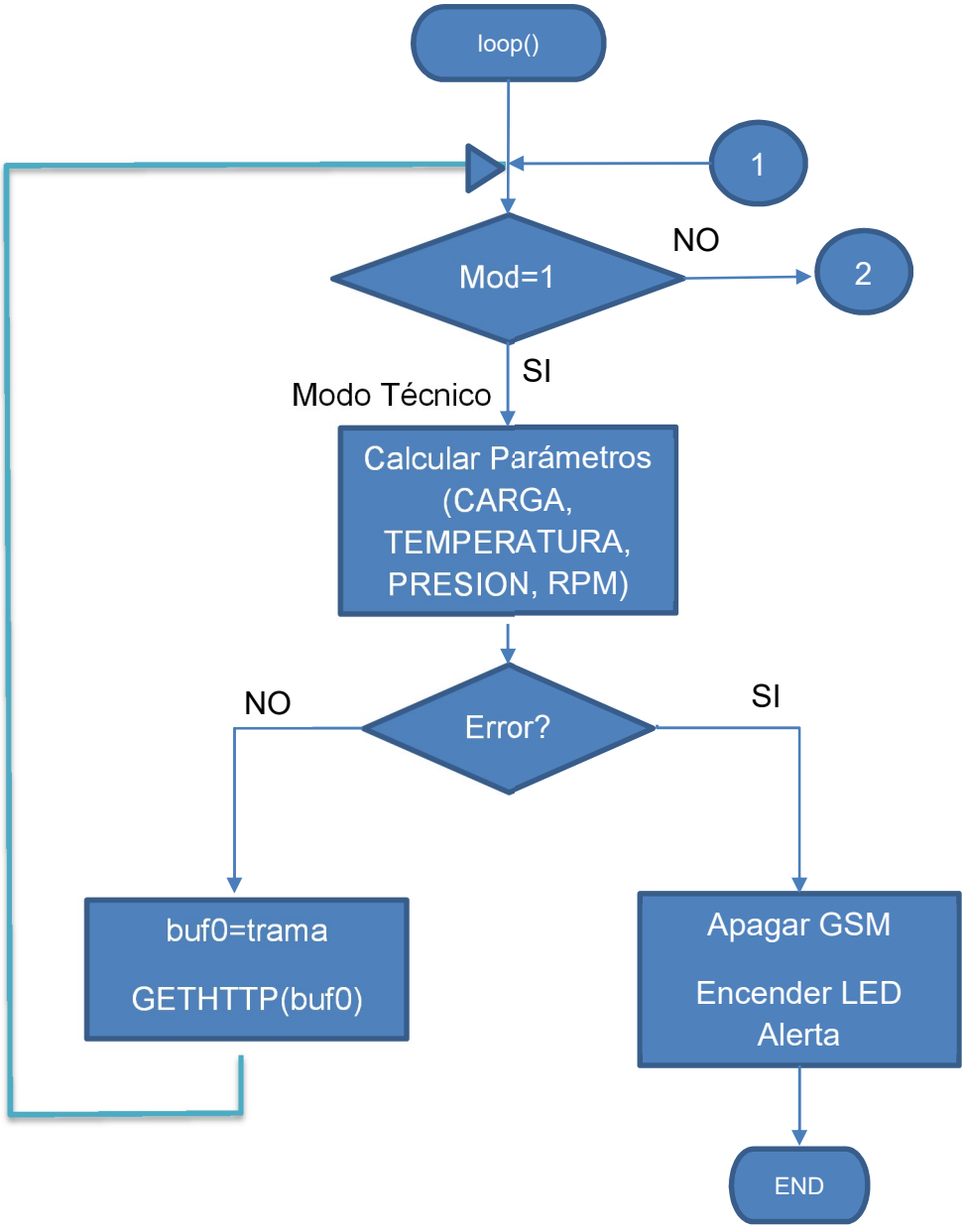

Figura 3.2 **Diagrama de Flujo de loop() parte 1**

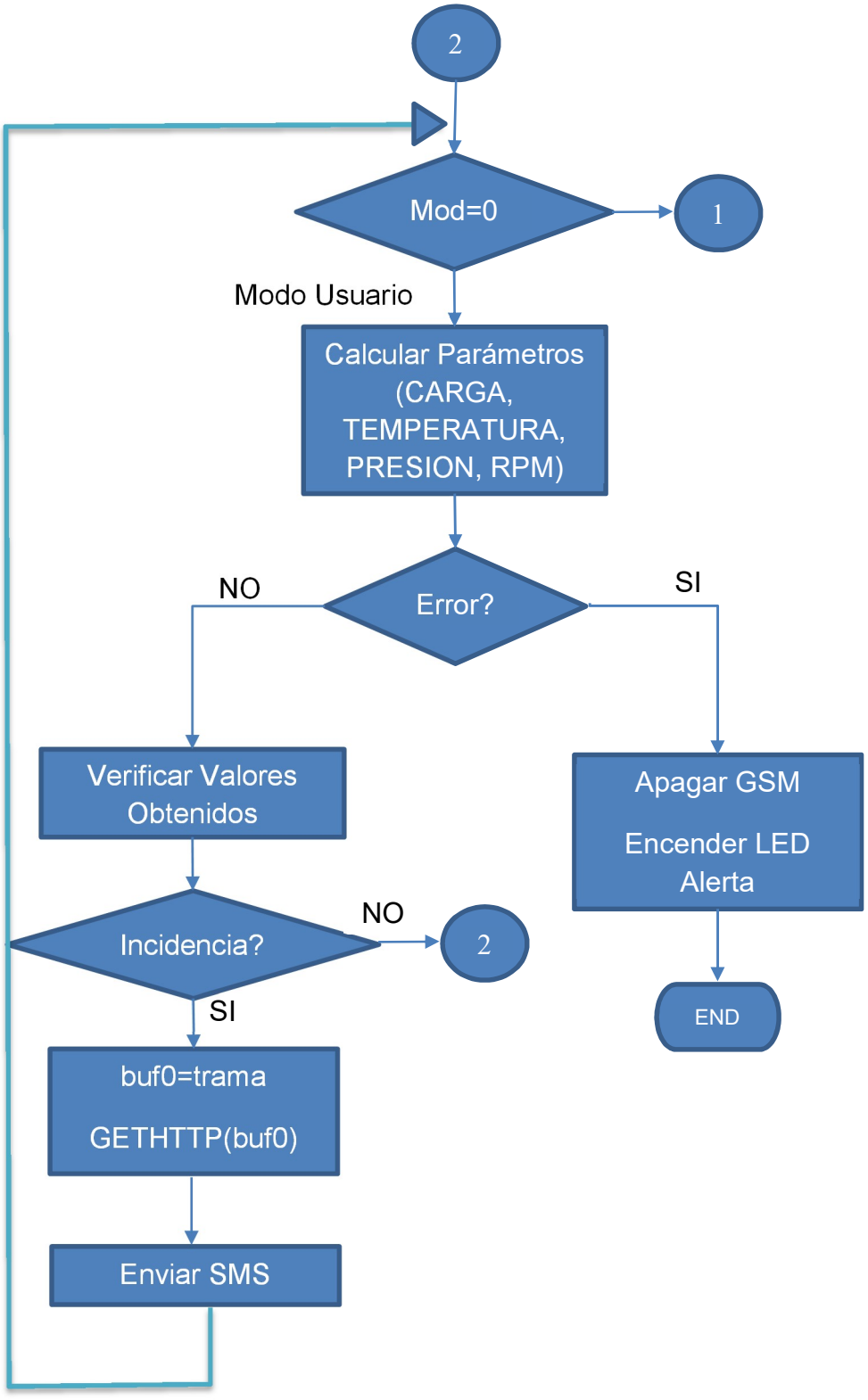

Figura 3.3 Diagrama de Flujo de loop() parte 2

3.6 Diagrama de Flujo de nueva.php

Nota: la función de contar se realiza a través de escritura en un archivo .txt que se encuentra en el mismo directorio de publicación de la aplicación

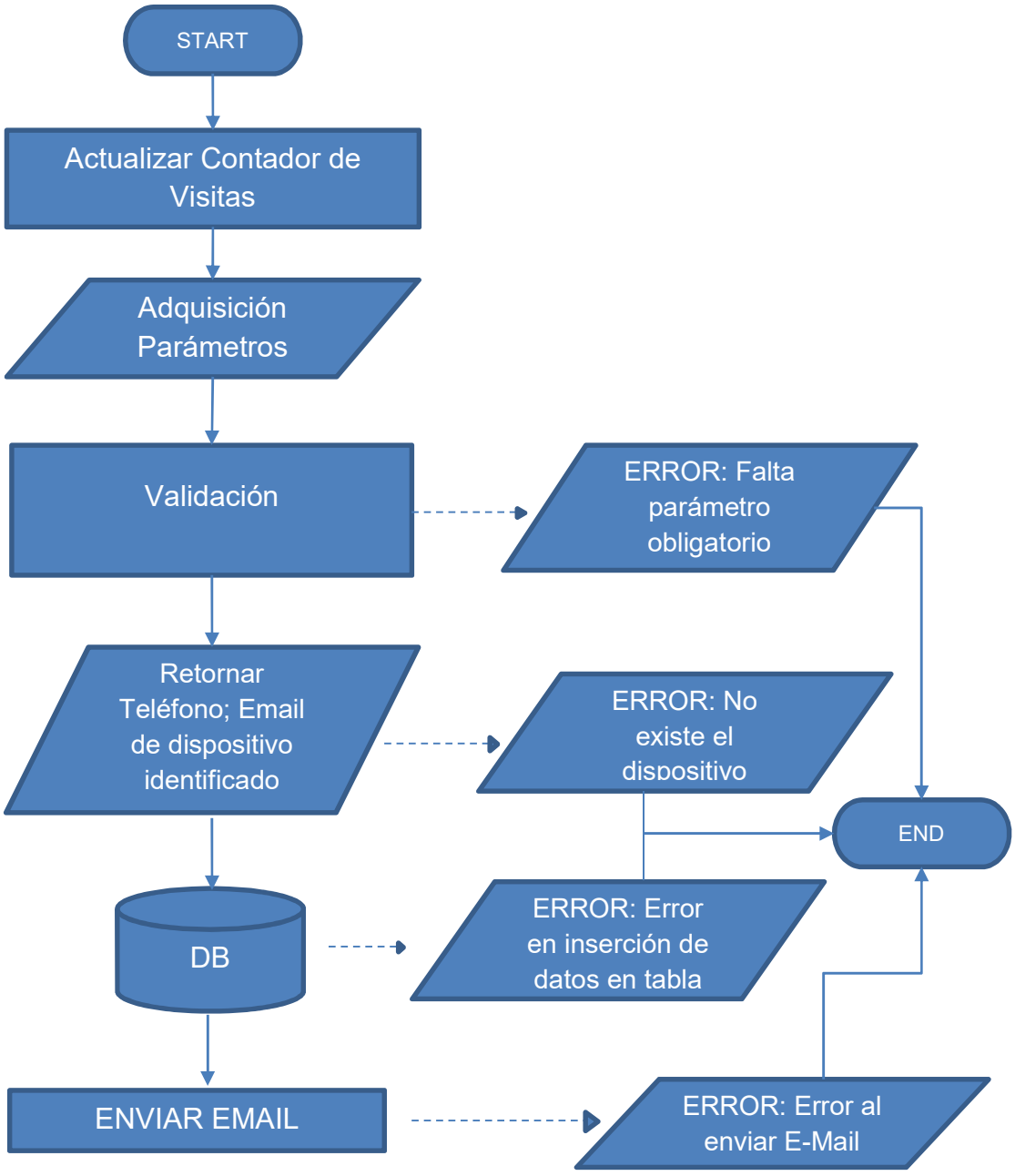

Figura 3.1 **Diagrama de Flujo de nueva.php**

3.7 Diagrama de Flujo de prinboard.php

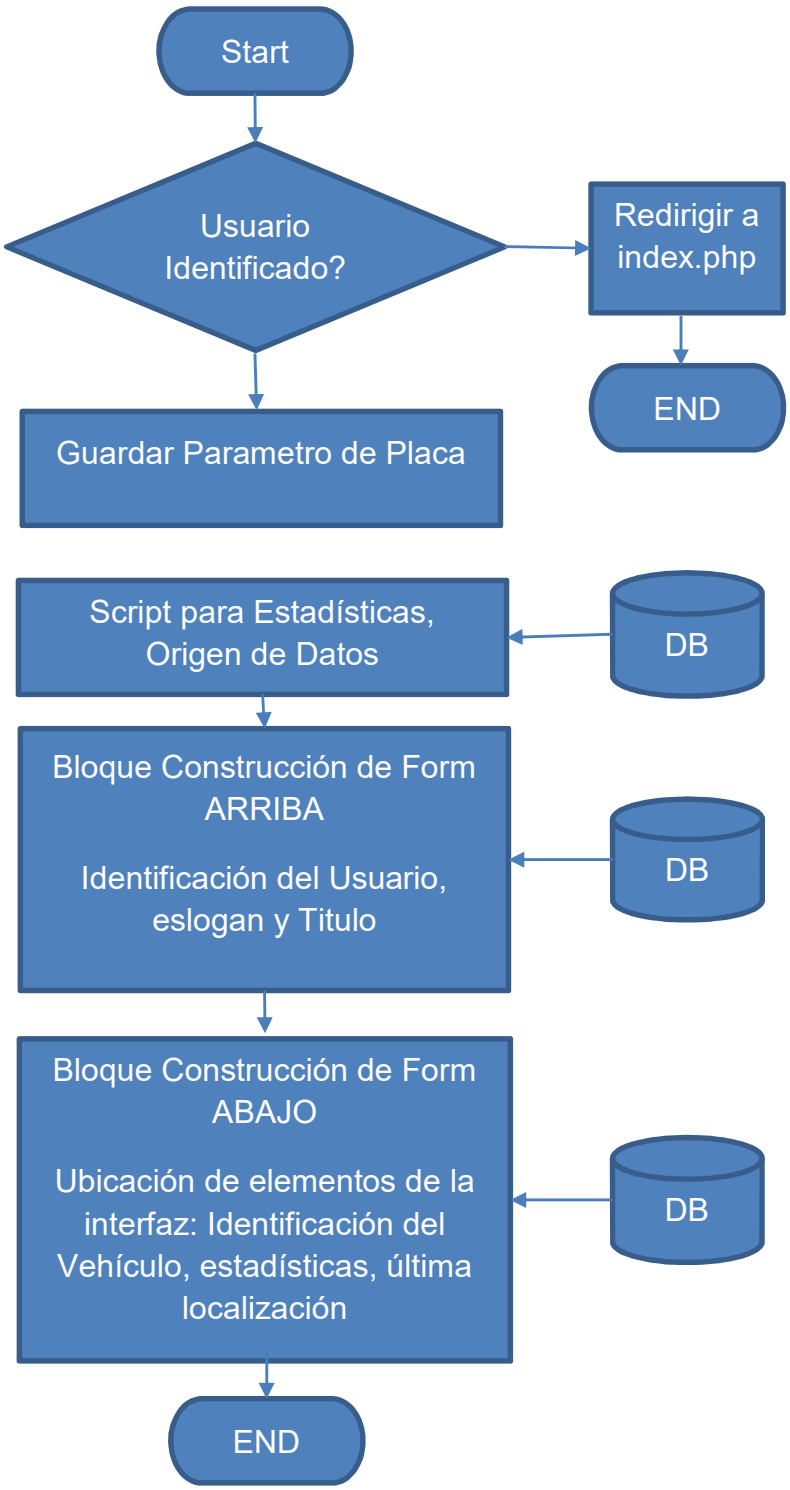

Figura 3.1 **Diagrama de Flujo de prinboard.php**

CONCLUSIONES Y RECOMENDACIONES

Se realizó la adquisición e interpretación de respuestas obtenidas a través del intérprete ODB, se extrajeron 4 parámetros a través del ECU y se establecieron comparaciones de nivel para identificar posible mal funcionamiento

Se procesaron los datos obtenidos a través del anterior proceso, se extrajo la información útil de la respuesta, se reservó esta lectura en los respectivos buffer para luego ser destinada a la Web. Se usó la conectividad móvil por medio de la banda GSM para realizar un requerimiento web con la información que se desea almacenar

Se estableció un servicio web que almacenó información en un servidor con hosting gratuito. Se definieron estructuras para el almacenamiento de dicha información y se presentó mediante aplicación web como resultados y estadística histórica.

Se realizó la implementación completa de un sistema de lectura de datos de diagnóstico a bordo del vehículo, se identificaron incidencias y se reportaron las mismas hacia una base de datos remota, se emitieron alertas hacia quien aparece registrado como propietario del vehículo, se presentó la información correspondiente a los usuarios debidamente autorizados

Se puede ahondar en la búsqueda de una propuesta de negocio viable mediante un estudio de mercado. Con este resultado podríamos establecer precio de venta, ganancia y demás información de utilidad.

El objetivo de este proyecto es la presentación de un prototipo que busca únicamente demostrar que las herramientas tecnológicas en la actualidad hacen posible la ejecución de nuestra propuesta. Para la presentación de un producto terminado se debe considerar

el uso de otros elementos más especializados e integrar hasta conseguir un dispositivo lo más compacto y económico posible.

Las limitaciones propias del hosting restringen la capacidad operativa en 4000 usuarios simultáneos. Esto debido al ancho de banda disponible bajo plan gratuito. Sin embargo con un costo muy razonable se puede adquirir capacidades virtualmente ilimitadas que permitirían multiplicar la cantidad de usuarios posible.

BIBLIOGRAFÍA

[1] PROMETEC. El modulo Bluetooth HC-05 [Online]. Disponible en: https://www.prometec.net/bt-hc05/#

[2] ElectonicosCaldas. Modulo Bluetooth HC-05 [Online]. Disponible en: http://www.electronicoscaldas.com/modulos-rf/452-modulo-bluetooth-hc-05.htm

[3] NAYLAMP. Tutorial Módulo GPS con Arduino[Online]. Disponible en: http://www.naylampmechatronics.com/blog/18_Tutorial-M%C3%B3dulo-GPS-con-Arduino.html

[4] ELEMON. Comentario Técnico de SIM-900[PDF]. Disponible en: http://www.elemon.com.ar/media/catalogos/z%20Boletines%20tecnicos/SIM900%20Nuevo%20m%C3%B3dulo%20GSM-GPRS.pdf

[5] H. Guerra, "DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA DE LOCALIZACIÓN Y SEGURIDAD VEHICULAR CON COMUNICACIÓN GPS Y GSM, BASADO EN HARDWARE Y SOFTWARE LIBRE", Tesis de Grado, Facultad Ingeniería en ciencias aplicadas. Universidad Técnica del Norte, IBARRA, Ecuador 2016.

[6] M. A. Rangel, "DISEÑO, IMPLEMENTACIÓN E INSTALACIÓN DE LÁMPARAS LED ALIMENTADAS CON SISTEMA FOTOVOLTAICO CON MONITOREO REMOTO VIA GPRS EN LA UNIDAD EDUCATIVA FRANCISCANA SALITRE", Tesis de Grado, Facultad de Ingenierías. Universidad Politécnica Salesiana, Guayaquil, Ecuador 2017.

[7] J. B. Zambrano, "DESARROLLO DE UN SIMULADOR ELECTRÓNICO DE UNA ECU Y SU DIAGNÓSTICO SOBRE CAN Y ODB - II", Tesis de Grado, Dep. Ingeniería Electrónica, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla, Sevilla, España 2015.

ANEXOS

1. Hoja de encuesta propuesta para el estudio de mercado

Encuesta de Interes General										SIMDER *Siempre contigo...*
Nombre										
Edad		Cooperativa								
Teléfono		email								
TAXI		Tipo								
PARTICULAR		Ocasional		Ejecutivo		Ambulante		Informal		
Con que frecuencia asiste usted a un taller para revisar los sensores de su vehículo?										
Vehículo Propio		Si		No						
Nivel de Estudios										
Usa usted Internet		SI		No						
Usa actualmente algun servicio de localizacion para su vehículo?										
Valor Mensual		Si		No						
Conoce usted acerca del escaneo que realizan en su taller de confianza para verificar lecturas										
de ciertos sensores?		Si		No						
De lo siguiente escoja que infomacion de su vehículo le es más util calificandola con 5 como de alto interes y 0 para totalmente indiferente										
RPM		TEMPERATURA				INYECTORES				
PRESION DE ACEITE		ALERTA REVISAR MOTOR				ARRANQUE DE VEHICULO				
CARGA DEL MOTOR		ALERTA CAMBIO DE ACEITE				LOCALIZACION				
Le interesa USAR de un servicio de escaneo y localizacion para su vehículo con una inversión mínima y completamente al alcance de su bolsillo?										
Cuanto considera usted es un precio razonable para este servicio (mensual)										
Usaría este servicio con su vehículo particular (No taxi)										
Usaría y recomendaría estos servicios que protegen a su motor de posibles daños y que le brindan la seguridad de localizar su vehículo?										
						SI		NO		