

Computación en la nube e Internet de las Cosas. Propuesta de diseño de arquitectura para prototipo de hardware para la extracción de datos de los vehículos con sistema de OBDII.

A. M. Jorge Ernesto, D. G. José Fernando y M. R. Edwin Alberto

Depto. de Maestrías y Postgrados, Universidad Don Bosco, San Salvador, El Salvador
jalmendarez3@gmail.com, jfduran.7@gmail.com, selarom511@gmail.com

Resumen—Este documento presenta el diseño de una arquitectura para la implementación de las tecnologías de Computación en la Nube e Internet de las Cosas, con el fin de automatizar el proceso de extracción de datos de diagnóstico de vehículos que poseen el Sistema de Diagnóstico a Bordo (OBDII). Para dicho diseño, se utilizaron dispositivos que permitieron conectarse con el sistema OBDII en los vehículos y servicios provistos por la plataforma de Amazon Web Services (AWS).

El trabajo da como resultado la transferencia automática y casi en tiempo real de los datos del vehículo hacia un almacén de datos, donde están listos para ser utilizados.

IoT y la computación en la nube pueden ser utilizadas para el monitoreo y control de una infinidad de dispositivos del uso diario.

I. INTRODUCCIÓN

Dado el avance, impacto y crecimiento que ha tenido la Computación en la Nube, y específicamente, las tecnologías de IoT, se ve la oportunidad de aportar acerca del ecosistema alrededor de ellas, de manera teórica, así como también de manera práctica por medio de un diseño arquitectónico y un prototipo funcional.

La comunicación con diferentes dispositivos que forman parte de la vida diaria ha pasado de ser un pensamiento futurista a una realidad. En el año 1999 Kevin Ashton daba vida por primera vez al término “Internet de las Cosas” en una presentación sobre tecnologías RFID, para el cual concibió un sistema de sensores ubicuos que conectan el mundo físico con Internet; aunque las cosas, Internet y la conectividad son los tres componentes principales de IoT, el valor está en cerrar la brecha entre el mundo físico y digital en los sistemas.

Se ve la oportunidad de aportar con la investigación de IoT por la poca exploración en el campo académico en El Salvador, previamente se realizó una investigación bibliográfica en las universidades y se encontró poco avance en este tema, por lo que esta investigación abona un material útil acerca de estas tecnologías que, según muchos autores y conferencistas, tomará un rol importante en nuestras vidas en un futuro cercano.

Con la evolución de las tecnologías se buscan formas de simplificar el desarrollo de actividades y tener un mejor control sobre aspectos del día a día. Con esto en mente se toma como

caso de estudio una empresa salvadoreña dedicada a la comercialización y arrendamiento financiero (leasing) de vehículos automotores, comercialización de repuestos, servicios de reparación de vehículos, entre otros.

Tomando este caso de estudio, la empresa, en el área de arrendamiento financiero, no cuenta con métodos formales o procesos establecidos para obtener la información de manera oportuna del estado de las diferentes unidades arrendadas que se encuentran en circulación. En la actualidad los vehículos traen incorporado un sistema de diagnóstico a bordo también conocido como OBD. Este sistema permite conocer el estado del vehículo y si este posee fallas a nivel mecánico, químico o eléctrico. El sistema realiza un escaneo de todos los sensores del vehículo para determinar su condición; teniendo esta información el sistema lo traduce a códigos estándares que son definidos por el fabricante y pueden ser extraídos por medio del puerto OBD.

II. MARCO TEÓRICO

A. Internet de las Cosas

Internet de las cosas no es un concepto nuevo en la industria, pero aún sigue evolucionando en distintas áreas. Algunos autores definen IoT como escenarios en los que la conectividad de red y la capacidad de cómputo se extienden a objetos, sensores y artículos de uso diario que habitualmente no se consideran computadoras, permitiendo que estos dispositivos generen, intercambien y consuman datos [1], este mismo autor hace referencia que para IoT no existe una definición única y universal, pero algo en los que muchos coinciden es en la conectividad de los dispositivos por medio de internet para lograr uno o varios objetivos específicos.

Según Google Cloud, IoT es un extenso conjunto de tecnologías y casos de uso que no tienen una clara y única definición. Un punto de vista viable enmarca a IoT como el uso de dispositivos conectados a la red, integrados en el entorno físico, para mejorar algunos procesos existentes o para habilitar un nuevo escenario que no era posible anteriormente [1].

Por otra parte, Microsoft define a IoT, en ocasiones llamada Internet de Todo, como una red gigante de objetos que se conectan entre sí e intercambian y analizan datos [3].

* Revista Tecnológica Universidad Don Bosco.

1) Elementos de IoT:

Las IoT deben considerar ciertos elementos para su aplicación; es decir elementos que soportan la comunicación entre los dispositivos e informar de alguna forma al usuario que ha iniciado o finalizado su tarea o función designada. A continuación, se presentan 6 elementos que ImapaTech considera importantes para la aplicación de IoT [4].

- **Transmisión de Datos:** Las aplicaciones de IoT acumulan más información de lo que el procesamiento tradicional puede administrar; tener capacidad para transmitir esta información continuamente es clave para alimentar de forma fiable, y en tiempo real, procesos de negocios y extraer información oportuna.
- **Plataformas IoT:** Una plataforma IoT permite desarrollar, implantar y administrar aplicaciones IoT y M2M: automatizar procesos y conexiones de red, almacenar y administrar datos de sensores, conectar y controlar los dispositivos y analizar la información.
- **Aplicaciones IoT:** Las aplicaciones IoT deben ser capaces de capturar, recolectar, interpretar y actuar en grandes cantidades de información, detectando brechas de conectividad, manejando interrupciones y cumpliendo con requisitos específicos del negocio y la industria.
- **IoT Cloud:** Las soluciones IoT en la nube proveen acceso asequible a redes de datos de alta velocidad para ampliar significativamente el alcance y la usabilidad de sus aplicaciones. También puede ofrecer almacenamiento de datos procesamiento, análisis y administración remota de dispositivos.
- **Datos IoT:** Las tecnologías de gestión de datos IoT garantizan la recopilación de los datos de forma correcta y en el momento oportuno, incluso cuando la conectividad se interrumpe.
- **Seguridad IoT:** Los enormes volúmenes de datos de dispositivos en IoT deben protegerse de robo o alteración desde el momento de su extracción; la seguridad tiene que continuar mientras los datos están en tránsito a través de la red, en reposo en los almacenes de datos y cuando las aplicaciones los utilizan.

2) Cómo funcionan las IoT

IoT funciona básicamente por medio de dispositivos que se conectan a través de WIFI, Bluetooth, 3G/4G u otros protocolos de red. Estos dispositivos están equipados con diferentes sensores de detección como, por ejemplo: sensores de movimientos, cambios de luz, gravedad, sonido, etc. El dispositivo está programado para realizar ciertas actividades o también puede ser capaz de recibir órdenes de tareas a ejecutar por medio de la comunicación inalámbrica [5]. Adicional a esto debe existir un sistema que recolecta esta información y la envía a la nube, donde la información recolectada será

analizada y procesada por los servicios de IoT **¡Error! No se encuentra el origen de la referencia.**

La comunicación de estos dispositivos debe tener ciertas características que deben cumplir como la conectividad entre dispositivos y protocolos de seguridad, esto con el fin de proteger la integridad de la información. A continuación, se muestra un diagrama del funcionamiento y componentes que permiten la concepción de IoT [5].

B. OBDII

Cada vez existen más dispositivos conectados a internet y muchas de las brechas de seguridad no son solventadas y puede llegar ser más críticos con el uso que se le puede dar; según el informe de Julio de 2014 de HP FORTIFY el 80% de los dispositivos tienen fallos en la autenticación y 6 de cada 10 dispositivos con interfaz de usuario son vulnerables [7].

1) Definición:

OBD es un sistema para el monitoreo y diagnósticos de fallas de los vehículos automotores, consiste en recolectar información de los componentes del vehículo y así alertar al usuario de posibles fallas mecánicas, eléctricas, entre otras. El acceso al sistema OBD se realiza por medio de un puerto también conocido por el mismo nombre OBD que en muchos modelos se encuentra en la parte baja del control de mando o bajo el capo del vehículo, este permite el acceso a las fallas detectadas como por ejemplo la temperatura del motor, nivel de aceite, nivel de voltaje de batería, falla en sistemas ABS, emisión de gases, etc.

2) Extracción de Códigos PID:

Para poder realizar extracción de datos de los vehículos se debe comprender el funcionamiento del sistema OBDII, este funciona por medio de un conjunto de sensores instalados en el vehículo que están constantemente enviando señales del estado de sus componentes y se puede trabajar en nueve modos, comunes entre todos los vehículos que implementan la tecnología, que van desde extraer datos para su verificación, extraer códigos de averías, borrados de datos y realizar pruebas dinámicas. El modo en que estos datos se presentan se maneja bajo un estándar y de igual forma se maneja el tamaño de trama de consulta OBDII.

Byte #0	Byte #1	Byte #2	Byte #3	Byte #4	Byte #5	Byte #6	Byte #7
---------	---------	---------	---------	---------	---------	---------	---------

Fig. 1. Trama de bytes de consulta OBDII

El contenido de cada "Byte" depende de si la trama es de petición o respuesta y el modo en que se está accediendo, a excepción del #0 que contiene el número de bytes que poseen información.

Este trabajo de investigación utiliza específicamente 2 modos, los cuales son el "modo 01", por medio del cual se pueden consultar parámetros actuales del vehículo y el "modo 03" el cual consulta los códigos de averías detectados por la computadora del vehículo.

- Modo 01 (Datos de diagnóstico): sirve para consultar datos en tiempo real sobre el estado del vehículo, como la temperatura del motor o las revoluciones por minuto. Para realizar una consulta en este modo, se debe enviar en la trama de OBDII el valor PID (por sus siglas en inglés) a extraer; puede consultarse más de un valor PID en la misma trama hasta un máximo de 6.
- Modo 03 (Códigos de diagnóstico): permite extraer todos los códigos de diagnóstico de error (o DTC por sus siglas en inglés) que se encuentran en la computadora del vehículo. La estructura de estos códigos es la mostrada en la Fig. 2. Los códigos que se pueden obtener mediante este modo se pueden categorizar en dos tipos; según el valor del segundo carácter (Tipo de código), estos pueden ser códigos estandarizados o propios del fabricante del vehículo.

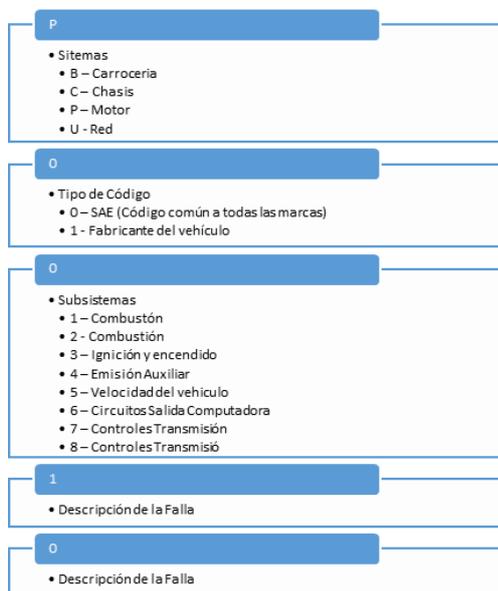


Fig. 2. Estructura de códigos de diagnóstico

C. Computación en la Nube

La computación en la nube es una solución integrada basada en servicios por medio de internet y no es necesario tener dichos servicios dentro de la organización, esto facilita que los recursos sean escalables y de alta disponibilidad desde cualquier lugar con acceso a internet [6].

1) Definición:

Computación en la nube o también conocido como “Cloud computing” es una capa de servicios publicados en internet y está disponible todo el tiempo y en cualquier lugar, el único requisito es tener una conexión a internet para poder acceder a los servicios. Beka Kezherashvili en su investigación del 2011 define la computación en la nube como un modelo de presentación de servicios de negocio y tecnología, que permite al usuario acceder a un catálogo de servicios estandarizados y responder a las necesidades de su negocio, de forma flexible y

adaptativa, en caso de demandas no previsibles o de picos de trabajo, pagando únicamente por el consumo efectuado.

2) Modelos de Servicio de Computación en la Nube:

La computación en la nube incluye varios modelos de servicios, los cuales son ofertados individualmente como servicios en la nube, estos pueden ser Software as a Service, Platform as a Service o Infraestructura as a Service. Estos servicios comúnmente son desarrollados en conjunto para tener un aprovechamiento del aprovisionamiento de software en la nube. A continuación, se muestran algunos ejemplos de los modelos en la nube.

- *Software as a Service (SaaS)*: es una forma de distribución de software en la nube, con un régimen de pago normalmente mensual o anual, aunque existen otros que son muy utilizados para este tipo de modelo, donde el proveedor contiene sus aplicaciones en servidores propios permitiendo a sus clientes acceder por medio de internet, estando disponible en todo momento y desde cualquier lugar.
- *Platform as a Service (PaaS)*: son herramientas de software que son utilizadas para el desarrollo de aplicaciones y que está alojada en la nube, permitiendo a los desarrolladores ingresar desde cualquier computadora y comenzar a construir, analizar, desarrollar, probar, documentar y hasta poner en marcha la aplicación, sin tener que instalar algún software en el equipo local.
- *Infrastructure as a Service (IaaS)*: se refiere a la virtualización de servidores. Usualmente la contratación de estos servicios se realiza en base a procesador, memoria y espacio de almacenamiento; siendo escalable según el negocio lo requiera y no es necesario adquirir equipos o actualización de estos.

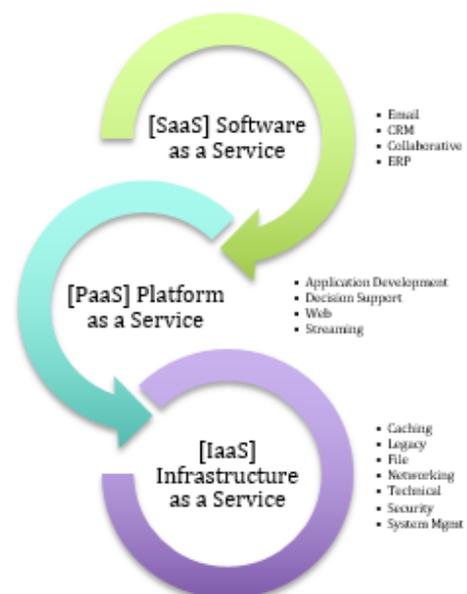


Fig. 3. Modelo de servicios en la nube.

III. METODOLOGÍA

La metodología comprende las siguientes fases para el cumplimiento de cada uno de los objetivos propuestos:

A. Fase 1

Investigación bibliográfica. Esta fase tiene por objetivo enriquecer los conocimientos y estos a la vez nos brindaran un punto de partida en el marco conceptual que se enfocara en las herramientas de IoT, sus características, proveedores que brindan servicios integrales; también información técnica de componentes que utilizan los prototipos de hardware a evaluar con el fin de comprender y tomar criterios de evaluación según los requisitos funcionales y no funcionales.

B. Fase 2

Seleccionar un prototipo de hardware para la extracción de información del vehículo. Se debe presentar un prototipo de hardware que cumpla con un mínimo de características necesarias para la recolección de información de diagnóstico del vehículo y la transferencia de estos datos hacia la API. Para cumplir con este objetivo se realizarán las siguientes actividades:

- Investigar prototipos en el mercado y realizar un modelo comparativo.
- Describir la información de los componentes que utilizan los prototipos de hardware y así determinar el que más se adapta a las necesidades de la investigación.

C. Fase 3

Diseñar y desarrollar un algoritmo de extracción para el prototipo de hardware que permita la extracción de la información de diagnóstico de vehículos automotores por medio del puerto OBDII. Este será el interlocutor lógico entre el hardware y la computadora central del vehículo y que estará programado para obtener información del vehículo y ser enviada a la API de manera que los datos puedan ser utilizados posteriormente para ellos se realizarán las siguientes actividades:

- Seleccionar el protocolo de intercambio de datos para crear la API, en base a la investigación realizada.
- Crear una estructura de datos a utilizar.
- Diseñar un modelo para el registro de datos.
- Realizar el diseño de la API en base a los requisitos funcionales y no funcionales.
- Integrar los componentes involucrados (sistemas, datos, dispositivo, protocolos).
- Crear un algoritmo de extracción de la información con pasos lógicos de extracción y envío de la información hacia la API publicada en los servicios de IoT.

D. Fase 4

Diseñar una API para la administración de los datos. La función principal de la API de administración el traslado de la información extraída del vehículo hacia un repositorio central. Para lograrlo se realizarán las siguientes actividades:

- Generar pruebas de configuración de los servicios de comunicación.
- Generar casos de uso.
- Realizar un diseño de los servicios de comunicación.

E. Fase 5

Diseñar la arquitectura IoT de comunicación para el registro y consumo de la información de diagnóstico proveniente del prototipo de hardware en un repositorio central. Para realizar este objetivo se realizarán las siguientes fases:

- Evaluarán proveedores de IoT que más se adapten a las necesidades.
- Evaluar proveedores de Cloud Computing.
- Evaluar los componentes que se necesitan para generar una arquitectura funcional que cumpla con los requisitos.
- Diseñar una base de datos que almacene de forma central los registros extraídos de los vehículos.
- Diseñar un modelo de autenticación y autorización para el acceso a la información.
- Generar los diagramas necesarios, que complementen el diseño de la arquitectura IoT.

Para lograr el diseño de un sistema automatizado para la extracción y centralización de los datos de los vehículos, se requiere la ejecución de una serie de pasos en un orden lógico. Cada objetivo específico para este proyecto tiene una metodología a seguir y para ello, se cuenta con las siguientes herramientas:

Entrevistas realizadas a personal operativo y gerencia en la empresa de estudio. Estas incluyen información como:

- Descripción del proyecto.
- Requisitos funcionales y no funcionales.
- Perfiles de expertos.

IV. DISEÑO

Se realizó la elaboración de los diseños necesarios para el desarrollo de la arquitectura los cuales están analizados en base a los requisitos funcionales y no funcionales que se generaron en base a entrevistas con los usuarios de Gx. El diseño consta de diagramas UML de clases, secuencia, actividades, etc. Estos describen los componentes físicos y lógicos que son necesarios para el correcto funcionamiento de la arquitectura; desde la extracción de la información por medio del prototipo de hardware hasta el registro y consulta de la información en Cloud Computing.

A. Descripción de servicios

- API Gateway: es un servicio completamente administrado que facilita a los desarrolladores la

creación, la publicación, el mantenimiento, el monitoreo y la protección de API a cualquier escala. Con tan solo unos clics en la consola de administración de AWS, puede crear API REST y API WebSocket que actúen como "puerta delantera" para que las aplicaciones obtengan acceso a datos, lógica de negocio o funcionalidades desde sus servicios backend.

- Kinesis Firehose: es un servicio que ofrece una manera sencilla de cargar datos de streaming de manera fiable en almacenes de datos y herramientas de análisis.
- AWS S3: es un servicio de almacenamiento de objetos creado para almacenar y recuperar cualquier volumen de datos desde cualquier ubicación: sitios web y aplicaciones móviles, aplicaciones corporativas y datos de sensores o dispositivos von IoT.
- Kinesis Data Analytics: ofrece una manera sencilla de procesar datos de streaming en tiempo real con SQL estándar sin tener que aprender a usar lenguajes de programación ni marcos de procesamiento nuevos.
- Kinesis Data Stream: es un servicio de streaming de datos en tiempo real con un alto nivel de escalabilidad y durabilidad. Puede registrar de manera continua gigabytes de datos por segundo de cientos de miles de orígenes, como transmisiones de clics de sitios web, transmisiones de eventos de bases de datos, transacciones financieras, fuentes de redes sociales, registros de TI y eventos de seguimiento de ubicaciones. Los datos recopilados se encuentran disponibles en milisegundos para posibilitar los casos de uso de análisis en tiempo real, como paneles en tiempo real, detección de anomalías en tiempo real y precios dinámicos, entre otros.
- Dynamo DB: es una base de datos de documentos y valores clave que ofrece un rendimiento en milisegundos de un solo dígito a cualquier escala. Se trata de una base de datos completamente administrada, en varias regiones y multimaestra con seguridad, copias de seguridad y restauración integradas y almacenamiento en memoria caché de aplicaciones a escala de Internet.
- AWS Lambda: permite ejecutar código sin aprovisionar ni administrar servidores. Se paga por el tiempo informático que consume. Sin costo cuando el código no es ejecutado. [8]

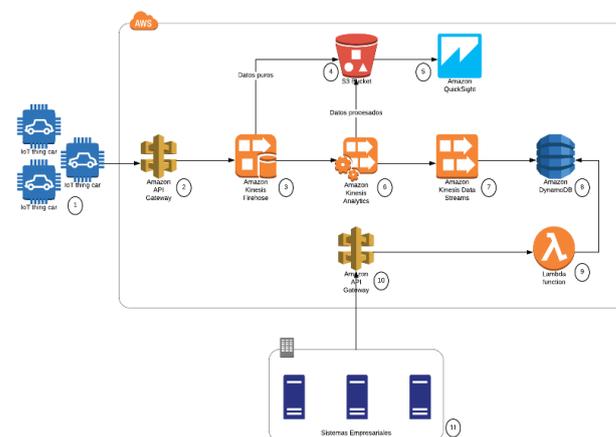


Fig. 3. Diagrama de componentes de Cloud Computing

El flujo de datos a través de los diferentes servicios de AWS, tal como se muestra en la ilustración, comienza por la extracción de los datos del vehículo por medio del prototipo de hardware (1), siendo este el único elemento externo a la infraestructura en la nube. Los datos extraídos son enviados a la API REST (2), punto de entrada de los datos hacia el sistema. Una vez los datos ingresan a la API, son enviados al servicio de streaming de alta disponibilidad (3), donde son guardados en el almacén (4) y enviados al servicio de streaming de procesamiento de datos en tiempo real (6). Cuando los datos han sido procesados, son enviados al servicio de streaming (7) que los inserta en la base de datos (8).

El flujo previamente descrito permite tener disponibles los datos procesados y consultarlos en cualquier momento. Los datos pueden ser extraídos por medio de la API REST (10), la cual utiliza funciones de extracción (9) para obtener la información de la base de datos (8). La API REST (10) permite tener la información e integrarla a los sistemas empresariales existentes (11).

B. Comunicación con la API

La comunicación entre todos los componentes que conforman la arquitectura IoT es por medio de servicios web, teniendo como punto de partida la extracción de la información del vehículo por medio del prototipo de hardware conectado directamente al puerto OBDII del vehículo y que este tiene la funcionalidad de transmitir información vía GPRS por medio de protocolo HTTP. En la capa de cloud computing se mantendrán activos servicios de registros de información que se definieron en el diagrama de servicios visto previamente en la ilustración 12. Este servicio registrará cada petición realizada en el almacén de datos en la nube.

Con la información almacenada los sistemas empresariales de Gx u otros sistemas que Gx autorice, podrán acceder a la información registrada por medio de los servicios web de consulta para el análisis o procesamiento de la información obtenida por la arquitectura IoT.

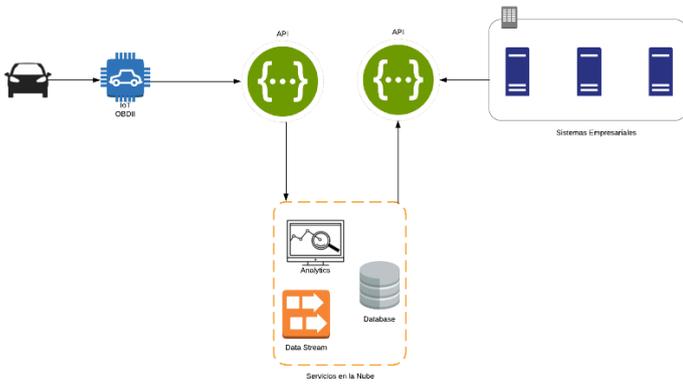


Fig. 4. Diagrama de comunicación entre el prototipo de hardware y Cloud Computing

V. RESULTADOS

Esta etapa se muestran los resultados de las pruebas realizadas por el equipo con el prototipo seleccionado y realizando extracciones directas de vehículos imprimiendo en consola los resultados como las primeras pruebas y luego evolucionando con pruebas piloto con toda la arquitectura propuesta funcionando.

A. Pruebas iniciales: Algoritmo de extracción

Luego de realizar los desarrollos correspondientes según el diseño planteado se obtuvieron resultados satisfactorios. En las siguientes imágenes se muestran los pasos que se realizaron para cargar el algoritmo de extracción en el prototipo de hardware por medio de "Arduino Studio" utilizando puerto USB.



Fig. 5. Carga del algoritmo de extracción en el prototipo de hardware (puerto USB).

Utilizando Arduino Studio se realiza la carga del sketch de forma satisfactoria.

```

telnetlogger_sim900_udp | Arduino 1.8.7
File Edit Sketch Tools Help

telnetlogger_sim900_udp [upload]

}

bool CTeleClientSIM900::netSetup(const char* apn, unsigned int timeout)
{
  uint32_t t = millis();
  bool success = false;
  netSendCommand("ATE0");
  do {
    success = netSendCommand("AT+CREG?");
    Serial.print(" ");
  } while (!success && millis() - t < timeout);
  if (!success) return false;
  do {
    success = netSendCommand("AT+CGATT?");
  } while (!success && millis() - t < timeout);
  sprintf(m_buffer, "AT+CSTT=\"%s\"", apn);
  if (!netSendCommand(m_buffer)) {
    return false;
  }
  netSendCommand("AT+CIICR");
  return success;
}

const char* CTeleClientSIM900::getIP()
}

Sketch uses 15326 bytes (41%) of program storage space. Maximum is 32256 bytes.
Global variables use 135 bytes (64%) of dynamic memory, leaving 893 bytes for local variables.

```

Fig. 6. Carga del algoritmo de extracción en el prototipo de hardware (Sketch).

A continuación, se muestra la imagen donde se conecta el prototipo de hardware en el puerto OBD de un vehículo; este comienza a funcionar y enviar información la nube al instante de conectarse en dicho puerto. En la imagen se demuestra que sigue conectado por medio de un cable micro USB hacia una computadora esto con el propósito de monitorear los datos extraídos desde el vehículo por medio de monitor que Arduino Studio nos proporciona.



Fig. 7. Prototipo de hardware conectado al vehículo.

Resultado de los datos que se extraen por medio del puerto OBD y se envía directamente a la nube utilizando las redes GPRS del chip previamente configurado en el prototipo de hardware.

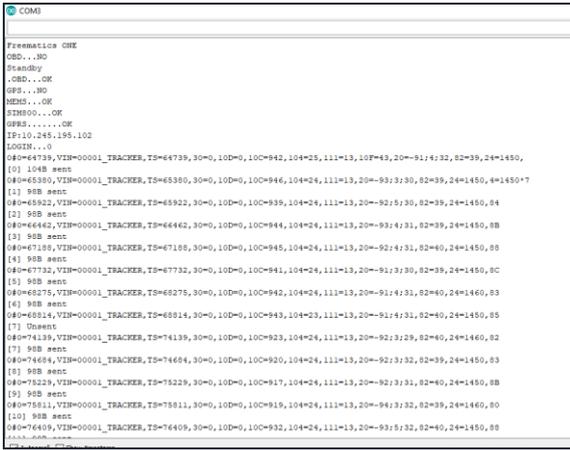


Fig. 8. Registros en consola de Arduino Studio.

- Instalar Serverless Framework

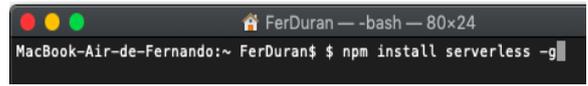


Fig. 12. Comando de instalación de Serverless Framework

- Abrir el proyecto serverless-iot-obd

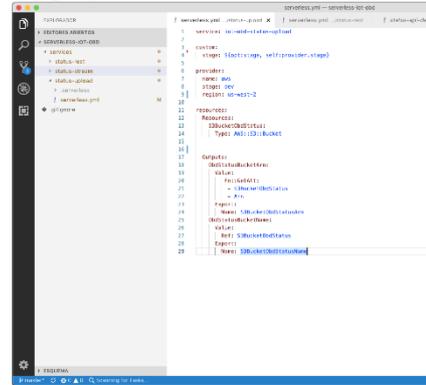


Fig. 13. Proyecto Serverless

- Desplegar el proyecto por medio del Serverless Framework.

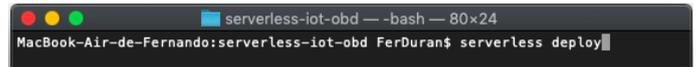


Fig. 14. Comando de despliegue de proyecto Serverless en AWS.

B. Pruebas iniciales: Computación en la Nube

Para las pruebas iniciales en la nube se utilizó AWS como proveedor, una de las principales razones fue que gracias a su capa gratuita se pudo generar un prototipo funcional de la arquitectura sin costo. Por otra parte, se utilizó el paradigma de “Infraestructura como Código” por medio de “Serverless Framework”, el cual, como su nombre lo indica, permite utilizar código para definir la infraestructura a utilizar en la nube; esto permite la replicación casi inmediata de la infraestructura necesaria para este proyecto.

A continuación, se describen brevemente los pasos para desplegar el proyecto:

- Crear cuenta AWS.

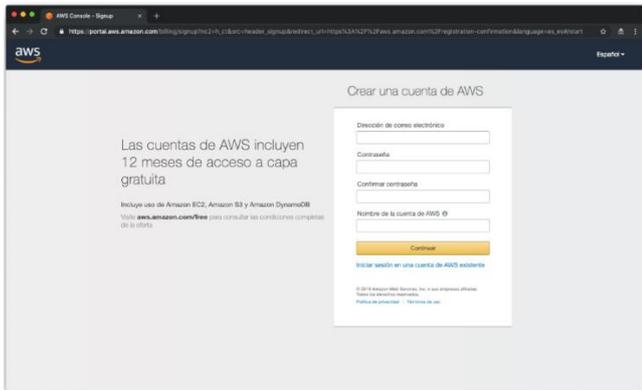


Fig. 9. Formulario de cuenta AWS.

- Descargar AWS CLI.

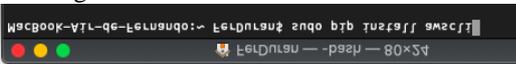


Fig. 10. Comando de instalación de consola AWS

- Ingresar las credenciales de usuario AWS en AWS CLI

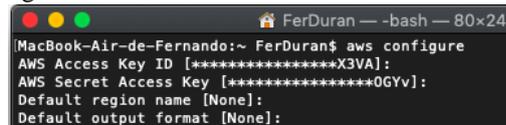


Fig. 11. Comando de configuración de credenciales AWS

VI. CONCLUSIONES

En esta investigación se ha presentado una propuesta a partir de un caso de estudio que tiene la necesidad de automatizar los procesos de recolección de información de los vehículos, por lo tanto, se ha realizado el diseño de una arquitectura basada en tecnologías IoT y computación en la nube, que incluye la extracción de información de los vehículos por medio del sistema OBDII en tiempo cercano al real y que a su vez dicha información es registrada en la nube. Esta propuesta contribuye a superar cada objetivo descrito al inicio de la investigación.

Basado en el objetivo “selección de un prototipo de hardware para la extracción de información del vehículo, a partir de una evaluación de los prototipos existentes en el mercado”, se seleccionó un prototipo de hardware según la evaluación realizada a múltiples prototipos, tomando en cuenta las características requeridas a cumplir.

Basado en el objetivo “diseñar un algoritmo de extracción para el prototipo de hardware que permita obtener la información de diagnóstico de vehículos automotores por medio del puerto OBDII”, se diseñó el algoritmo para recolectar la información generada por los vehículos automotores para posteriormente enviar la información a servicios en la nube.

Basado en los objetivos “diseñar la arquitectura IoT de comunicación para el registro y consumo de la información de diagnóstico proveniente del prototipo de hardware en un almacén de datos” y “diseñar la arquitectura en la nube de los servicios de recolección, procesamiento y revisión de datos”, se diseñó la arquitectura en la nube basada en los servicios de Amazon AWS para la recolección de los datos, exponiendo

servicios de recopilación, transformación, procesamiento y almacenamiento de la información proveniente de los prototipos de hardware instalados en los vehículos.

Además, se realizó la documentación del proceso de selección del prototipo de hardware y la documentación del diseño de la arquitectura propuesta basado en UML.

REFERENCIAS

- [1] K. Rose, S. Eldridge y L. Chapin (2015, Oct). Internet de las cosas [En línea]. Disponible: <https://www.internetsociety.org/wp-content/uploads/2017/09/report-InternetOfThings-20160817-es-1.pdf>
- [2] Google Cloud, 2019, Google. <https://cloud.google.com/solutions/iot-overview>
- [3] Azure, 2018, Microsoft. <https://docs.microsoft.com/en-us/azure/iot-fundamentals/iot-introduction>
- [4] ImayaTech, 2018. <http://www.imayatech.com/it-management/iot.html>
- [5] Amidata S.A. ¿Qué es el Internet de las Cosas? [En línea] Disponible: <http://es.rs-online.com/web/generalDisplay.html?id=i/iot-internet-of-things>
- [6] SAP. (19 de 01 de 2018). SAP Company. Obtenido de <https://www.sap.com/products/enterprise-management-erp.html>
- [7] M. A. Arenas, "Internet de las Cosas: Ciberseguridad", OWASP, Sucre, Bolivia, 2016
- [8] Amazon Web Services, 2019, AWS. Obtenido de: <https://aws.amazon.com/>