UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

SISTEMA DE MONITOREO VEHICULAR MEDIANTE IOT

TESIS

Para obtener el grado de

Ingeniero en Telecomunicaciones

PRESENTA

Mondragón Martínez Kevin Alejandro

DIRECTOR DE TESIS

Ing. Javier Ortiz Villaseñor

Ciudad Universitaria, CD. MX, 2021

AGRADECIMIENTOS.

Agradezco y dedico este trabajo de tesis a mi madre y padre. Ellos han sido mi principal inspiración, me han brindado su apoyo incondicional no solo en el aspecto académico, sino en cada aspecto de mi vida. El ser humano que soy ahora se debe a sus enseñanzas y a su esfuerzo incondicional para poder brindarme las herramientas necesarias para cumplir con mis objetivos. Este logro es de ellos.

Agradezco a mis hermanos y amigos por ser siempre un apoyo y que han estado conmigo en esta época de mi vida, siempre abiertos a escucharme y a brindarme su mano cuando es necesario, por compartir risas, penas y demás experiencias que he vivido.

A mi abuelo que me dio la bendición de tener dos padres, enseñarme el valor del trabajo y a brindarme su cariño. A pesar de que ya no esta con nosotros le sigo compartiendo mis logros y pidiendo su apoyo en mis tropiezos.

A Cindy que es un faro de luz que llegó a dispersar la oscuridad en la que estaba durante parte este proceso, funcionando como una brújula que me ayudó a encontrar el camino cuando estaba perdido. Agradeceré siempre por todo lo que me dio, las lecciones que me hizo aprender y lo feliz que hizo.

Finalmente, a la Universidad, profesores y compañeros de trabajo que tuvieron la humildad para poder compartir conmigo parte de los conocimientos que he utilizado para elaborar este trabajo de tesis y en mi vida profesional.

INDICE GENERAL

AGRADECIMIENTOS. ***2***

INDICE GENERAL ***3***

INDICE DE FIGURAS. **4**

INDICE DE TABLAS. **5**

INTRODUCCIÓN. ***6***

1. Definición del problema. 6
2. Objetivos. 6
   a. Objetivo general. 6
   b. Objetivos específicos. 7
3. Relevancia y contribución. 7
4. Estructura de la tesis. 7

CAPITULO I ***8***

1. IoT. 8
   a. INTRODUCCIÓN. 8
   b. CARACTERÍSTICAS. 8
   c. ELEMENTOS IOT. 9
2. REDES DE ACCESO. 11
   a. WWAN. 11
      i. TECNOLOGÍA 3G. 11
      ii. TECNOLOGÍA LTE. 13
      iii. WLAN (Wifi). 17
      iv. LPWAN. 20
      v. Bluetooth Mesh. 25
3. SERVICIOS EN LA NUBE (CLOUD SERVICES). 29
   a. INTRODUCCIÓN. 29
   b. TIPOS DE SERVICIOS EN LA NUBE. 30
   c. RECEPCIÓN DE DATOS. 31
      i. API. 32
      ii. MQTT 32

CAPITULO II ***34***

1. GESTIÓN VEHICULAR. 34
   a. Redes CANbus. 34
   b. CAPA FÍSICA. 36
   c. CAPA DE ENLACE DE DATOS. 37
      i. DATA FRAME. 38
      ii. Remote Frame. 40
      iii. Error Frame. 41
      iv. Overload Frame. 41
   d. Administración de Errores. 41
2. ESTÁNDARES DE CAPAS SUPERIORES. 42
   a. SAE J1939. 42
      i. Formato de Mensajes. 43
      ii. J1939 estandares. 43
   b. OBD2 44
      i. PIDs 44

ii. ESTRUCTURA DE MENSAJES. 44
iii. MODOS 45
iv. LISTA DE PIDs ESTANDAR. 46

CAPITULO III 47

1. DESARROLLO DE UN SISTEMA DE MONITOREO VEHICULAR. 47
   a. ELM327. 47
   b. Creación de un "thing" en AWS. 48
   c. Conectividad a Internet. 53
      i. Router Gateway. 53
      ii. All-in-one. 55
   d. Script para la obtención y envío de datos. 56
   e. Visualización de datos. 56

CAPITULO IV CONCLUIONES. 63

1. ANÁLISIS DE RESULTADOS Y GENERACIÓN DE CONCLUSIONES. 63
2. Trabajo Futuro. 64

Bibliografía y referencias web. 65

Anexos. 66

a. Anexo A. Listado PIDs. 66
b. Anexo B. Standar Fault Codes (DTC) 66
c. Anexo C. Configuración Raspberry Pi Wifi. 66
d. Anexo D. Configuración Router Gateway. 67
e. Anexo E. Configuración Raspberry Pi como Router y AP. 69
f. Anexo C. Python Script para el envío de datos. 72

INDICE DE FIGURAS.

Figura 1. Arquitectura IoT. [1] 10
Figura 2. Uso del espectro 3G y Representación del multicceso con WCDMA. [1] 12
Figura 3. Evolución de la arquitectura del sistema de 3G a LTE. [2] 13
Figura 4. Arquitectura E-UTRAN. [2] 14
Figura 5. Pila de protocolos utilizados para intercambiar datos desde un dispositivo móvil a un servidor externo. [2] 16
Figura 6. Red WLAN Ad hoc. [3] 19
Figura 7. Red WLAN AP-based y topología ESS. [4] 20
Figura 8. Comparativa LPWAN vs otras tecnologías inalambricas. [5] 21
Figura 9. Señales Sigfox de UNB vs una señal inalámbrica GFSK convencional. [6] 23
Figura 10. Arquitectura LoRaWan. [7] 24
Figura 11. Pila del protocolo Bluetooth Mesh. [8] 27
Figura 12. Proveedores de servicios en la nube. [9,10,11] 30
Figura 12. Comunicación MQTT. [12] 33
Figura 13. Comparativa del uso de CAN bus. [1] 35
Figura 14. Comunicación entre nodos CAN. [2] 35
Figura 15. Circuito e impedancias para redes CAN. [3] 36
Figura 16. Niveles de voltaje de las señales de una red CAN. [4] 37
Figura 17. CAN Data frame. [5] 38
Figura 18. Campo Identifier estándar y extendido de una data frame CAN. [6] 39
Figura 19. Campo Control de una data frame CAN. [6] 39

Figura 20. Campo CRC y ACK de una data frame CAN. [6] 40
Figura 21. Formato de mensaje J1939. [7] 43
Figura 22. Estructura de mensaje OBD2. [8] 45
Figura 23. ELM327 47
Figura 24. Consola AWS. 48
Figura 25. Manage "IoT Core". 48
Figura 26. Create Thing. 49
Figura 27. Single Thing. 49
Figura 28. Registro Thing. 50
Figura 29. Certificado. 51
Figura 30. Asociar policy. 51
Figura 31. Crear policy. 52
Figura 32. Test IoT Core. 52
Figura 33. Router Teldat V vista superior. 53
Figura 34. Router Teldat V vista trasera. 54
Figura 35. Conexiones. 54
Figura 36. Router en el vehículo. 55
Figura 38. Conexiones con Dongle LTE. 56
Figura 39. Act IoT Core. 57
Figura 40. Create rule. 57
Figura 41. Action AWS. 58
Figura 42. Crear recursos Analytics. 59
Figura 43. Data Set. 59
Figura 44. Schedule Data Set. 60
Figura 45. QuickSight data. 60
Figura 46. Seleccionar fuente de datos. 60
Figura 47. Seleccionar Data Set. 61
Figura 48. Creación de Analisis. 61
Figura 49. Dashboard AWS QuickSight. 62

INDICE DE TABLAS.

Tabla 1. Velocidades 3G 12
Tabla 2. Características LTE release 10. 17
Tabla 4. Estandares SAE J1939 44
Tabla 5. Comparativa de mercado. 64

INTRODUCCIÓN.

1. Definición del problema.

Los vehículos motorizados han estado evolucionando y adaptándose a las nuevas tecnologías para dar un servicio superior y marcar una diferencia con respecto a la competencia. Pasó de solo ser un vehículo con componentes mecánicos, hidráulicos y eléctricos a ser inteligentes, con un elemento central que se encarga del procesamiento de señales y poder controlar los componentes del vehículo.

El mantenimiento siempre ha sido una actividad de suma importancia para la correcta operación de cualquier máquina, realizándole ajustes para poder funcionar de manera eficiente. El mantenimiento crece en importancia ya que evita contratiempos o accidentes en algún traslado, costos superiores debido a descomposturas prevenibles y alargar el tiempo de vida útil del vehículo.

Los vehículos desde la década de los 90's tienen la capacidad de dar mucha información del estado actual de su funcionamiento, niveles de algunos parámetros de operación como son combustibles, lubricantes, energía e incluso indicar cual es el problema que tiene el vehículo en ese preciso momento. Conectándose mediante una interfaz especial a la conocida "computadora" del vehículo y una herramienta que permita obtener estos datos, es posible extraerlos y realizar un análisis con el cual se pueden tomar decisiones con respecto al estado, mantenimiento, operación, etc.

Muchas empresas u organizaciones que cuentan con flotillas de vehículos utilizan dispositivos para poder obtener datos y reportes de fallas actualmente de manera local, por lo que es necesario que el vehículo se encuentre fuera de operación y dentro del taller de mantenimiento.

2. Objetivos.

a. <u>Objetivo general.</u>

Desarrollo de sistema capaz de realizar la recolección, envió y almacenamiento de datos proporcionadas por vehículos, para realizar un análisis a partir de la información obtenida y poder tomar acciones que permitan optimizar los procesos operación, mantenimiento y monitoreo.

b. Objetivos específicos.

- Utilizar alguno de los servicios en la nube para la recepción de datos.

- Creación de script para la recolección de datos de un entorno físico a través de un Raspberry.

3. Relevancia y contribución.

Este trabajo de tesis propone la creación de una solución permita la monitorización de vehículos, utiliza la tecnología LTE y OBD2. Sin embargo, puede ser fácilmente adaptado para obtener datos de alguna otra fuente diferente a vehículos o la utilización de otro protocolo de comunicación dependiendo del alcance que sea definido. En posteriores trabajos puede utilizar diferentes protocolos y datos con ayuda de las herramientas descritas en esta tesis.

4. Estructura de la tesis.

Esta tesis esta conformada por cuatro capítulos. En el capitulo uno, se describe las tecnologías que pueden ser aplicadas, como funcionan y cuales son sus características a grandes rasgos. Descripción de las tecnologías IoT, que conciben cualquier dispositivo que realice una acción o medición y tenga capacidades de comunicación como un elemento dentro de su estructura; las redes que brindan conectividad dependiendo del tipo de solución a implementar y una descripción de los servicios en la nube.

En el capitulo dos, se describe los estándares y protocolos de comunicación vehicular. La estructura de sus mensajes y los datos que puede ser obtenidos, además del modo de interpretación para poder obtener información útil y valiosa de estos datos.

En el capitulo 3, se hace el desarrollo de una aplicación en Python con ayuda de una Raspberry Pi, donde estará solicitando los datos de un vehículo y enviándolos a un punto central de gestión.

En el capitulo 4, se presentan las conclusiones de este trabajo de tesis y la bibliografía consultada en la investigación de esta tesis.

CAPITULO I

1. IoT.

a. INTRODUCCIÓN.

El Internet de las cosas (IoT por sus siglas en inglés) es una red de objetos físicos o "cosas" incorporados con electrónica, sensores, software y conectividad para intercambiar datos con los fabricantes, operadores y/o otros dispositivos conectados.

IoT se refiere a los dispositivos que tiene capacidades de cómputo y comunicación, principalmente con conectividad a Internet. El objetivo es proveer comunicación de máquina a máquina a través de Internet sin intervención humana. Ejemplos de este tipo de tecnología la podemos encontrar en Fabricas Inteligentes, dispositivos de monitoreo médico, dispositivos del hogar inteligentes, etcétera.

b. CARACTERÍSTICAS.

Los dispositivos que utilizan tecnología IoT son capaces de recolectar datos de su entorno a través de sensores, ejecutar alguna acción o realizar análisis para proveer escenarios futuros. Estos análisis proveen servicios basado en los parámetros definidos por el usuario. Los dispositivos más sofisticados son capaces de "aprender" reconociendo patrones y preferencias de todos los datos capturados anteriormente. El dispositivo se vuelve "más inteligente" dependiendo de la capacidad de ajustar su programación para tener una capacidad de predicción superior.

Los dispositivos IoT se encuentran conectados directamente a Internet, conectados a través de otro dispositivo o incluso contar con ambos métodos de comunicación. La conectividad a la red es usada para compartir datos para su análisis en un punto centralizado e interactuar con los usuarios. Este tipo de conectividad provee acceso remoto al usuario. Por ejemplo, dispositivos inteligentes del hogar gestionables desde cualquier punto a través de un teléfono inteligente.

Las redes, comunicación y protocolos de conectividad usados dependen de la aplicación para lo cual es utilizado. Los protocolos de comunicación para enviar información incluyen MQTT, REST API, CoAP, DTLS, entre otros. La comunicación entre dispositivos con el nodo central se realiza de manera inalámbrica a través de redes LPWAN, la red celular WWAN, Wifi, Ad hoc, Bluetooth (BLE) y NFC principalmente.

c. ELEMENTOS IOT.

Los dispositivos IoT son usados en diferentes sectores del mercado debido a la gran cantidad de funciones que pueden desempeñar. Una red IoT pude estar integrada por los siguientes tipos de elementos:

Sensor: Son dispositivos transductores capaces de detectar y realizar mediciones como la temperatura, humedad, luz, presión, proximidad o magnitudes físicas similares. Posteriormente convertir estos datos a una señal para ser transmitida.

Actuador: Es el dispositivo transductor que recibe una señal de algún punto de la red o desde la herramienta de gestión, para ejecutar una acción.

Gateway: Este dispositivo es el encargado de orquestar la comunicación entre sensores, recibir los datos, enviarlos para su análisis y recibir mensajes desde el punto de gestión para enviarlos a los actuadores. Este dispositivo que brinda conectividad IP con los servicios en la nube, centros de datos o a la red del usuario con los dispositivos anteriormente mencionados, gestión local de los dispositivos y en algunas ocasiones realiza un nivel de análisis localmente. Dentro de la arquitectura general de las soluciones de IoT se encuentra en el nivel de Edge y se comunica con las plataformas en la nube.

Plataformas IoT: Las plataformas IoT integran capacidades y funcionalidades en un punto, esencialmente permitiendo desplegar proyectos y desarrollar aplicaciones de una manera más eficiente. Las plataformas brindan la interfaz con el usuario con la información útil que los datos brindan después de pasar por diferentes tipos de análisis. Generalmente proveen capacidades modulares en donde se adquieren los servicios que el usuario requiere, como son: la gestión masiva de los dispositivos, sistema de alarmas, autenticación y seguridad, Data storage, etc.

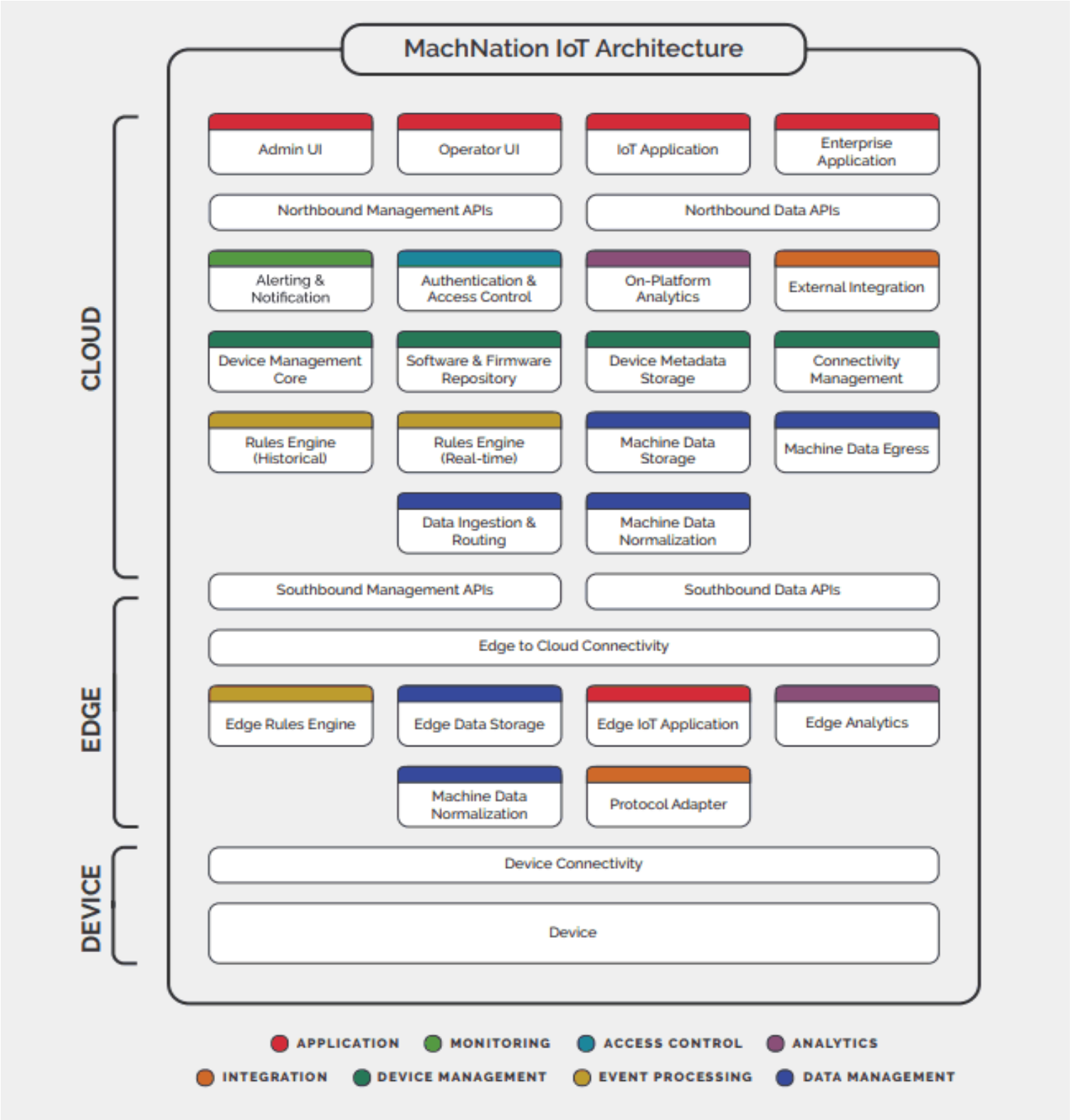


Figura 1. Arquitectura IoT. [1]

1. Functional architecture for IoT platforms. https://www.machnation.com/2017/09/18/functional-architecture-iot-platforms/?utm_source=iscoop

2. REDES DE ACCESO.

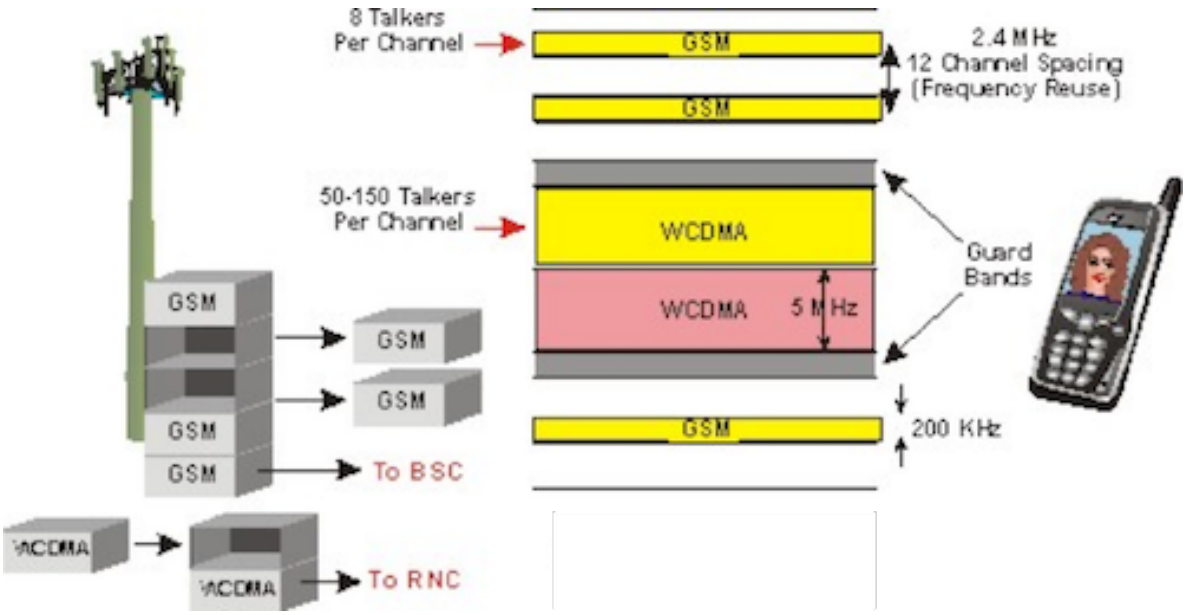
a. WWAN.

Son una forma de redes inalámbricas que proveen servicios de conectividad e Internet. Utiliza las redes de comunicación móvil celular con tecnologías 3G y LTE brindadas por la infraestructura de los ISP móviles.

i. TECNOLOGÍA 3G.

Es tercera generación de las redes móviles desarrollada a finales de la década de 1990. Incluye velocidades de descarga de hasta 42 [Mbps] en los estándares más recientes, roaming global, GPS y una mayor calidad en las llamadas de voz. Basada en paquetes de WCDMA, enviando los datos a través de tecnología de conmutación de paquetes mientras que para la voz utiliza conmutación de circuitos. Utiliza las bandas de frecuencia 1, 2, 5, 6 y 8 con un ancho de banda de 20 [MHz].

Utiliza un nuevo canal de transporte denominado HS-DSCH, que multiplexa las trasmisiones de datos hacia cada terminal de usuario, en donde las variaciones del canal se compensan con modulación y codificación adaptativa, manteniendo la probabilidad de error en el valor apropiado. Mapea a un conjunto de uno a quince canales físicos, todos sobre la misma portadora. Además, multiplexa los canales a través de código, dando como resultado un máximo de 15 códigos paralelos en este canal, asignando cada uno a usuarios durante la transmisión.



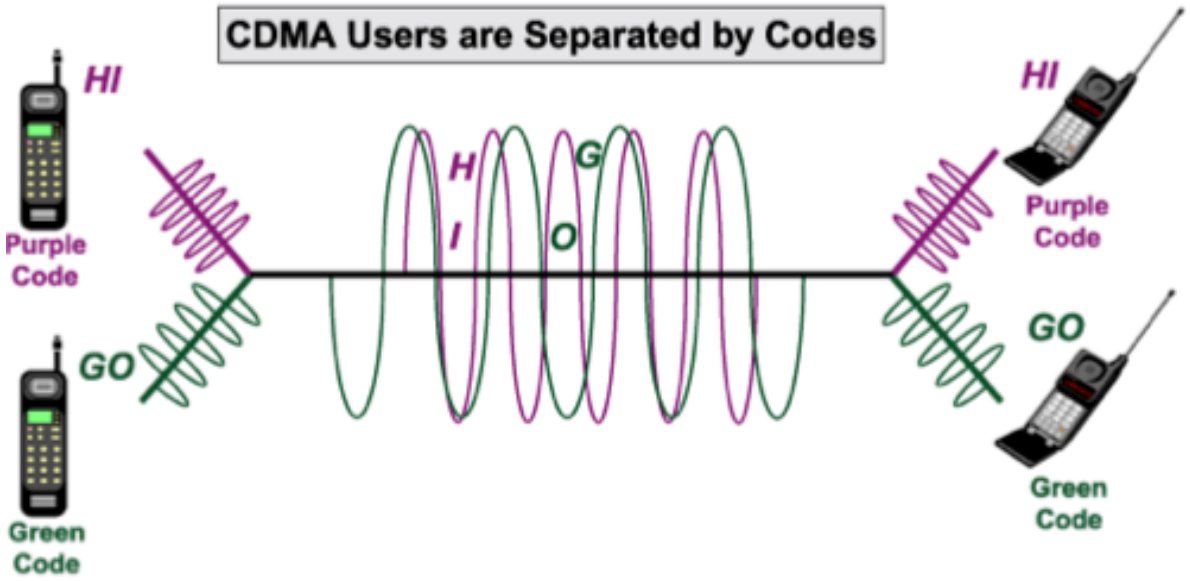


Figura 2. Uso del espectro 3G y Representación del multicceso con WCDMA. [1]

Las velocidades de transmisión dependiendo de la tecnología 3g son las siguientes:

Packet data Service		Máximo teórico [Mbps]	Banda (Frecuencia [MHz])	Estándar	Diversidad
UMTS	Upload	0.384	Banda 1 (2100)	3GPP Release 5	Si
	Download	0.384			
HSUPA	Upload	5.76	Banda 2 (1900)	3GPP Release 6	
HSDPA	Download	14.4	Banda 5 (850)		
HSPA+	Upload	5.76	Banda 6 (800)	3GPP Release 7	
	Download	21			
DC-HSPA	Upload	12	Banda 8 (900)	3GPP Release 8	
	Download	42			

Tabla 1. Velocidades 3G

1. Tecnologías para el desarrollo de aplicaciones: https://mastermoviles.gitbook.io/tecnologias2/conectividad-movil.-tecnologias-3g-4g-wifi-y-bluetooth

ii. TECNOLOGÍA LTE.

En la concepción de esta tecnología por la 3GPP tenía como objetivo mantener sus sistemas de comunicación móvil por 10 años, entregando las altas velocidades de datos y las bajas latencias que requerirían los futuros usuarios. En la figura 3 vemos como evolucionó la arquitectura de red a partir de UMTS (3G).

Se requería que LTE entregara una velocidad de 100 [Mbps] de downlink y 5 [Mbps] de uplink. También se buscaba una mejora respecto a movilidad y cobertura. LTE fue desarrollado para soportar células mayores a 5 [km], con desempeño degradado hasta 30 [km]. Se buscó optimizar la movilidad hasta velocidades de 15 [km/h], trabaja en alto desempeño hasta 120 [km/h] y puede soportar velocidades de hasta 350 [km/h]. Estos requerimientos se superaron eventualmente en las versiones más recientes.

SAE

La 3GPP desarrolló el Evolución de Arquitectura del Sistema (System Architecture Evolution SAE), que define una nueva red de núcleo de paquetes capaz de soportar IPv4, IPv6 o ambas, conocida como núcleo de paquetes evolucionado (EPC). La EPC permite tener siempre una conexión IP a diferencia de tecnologías anteriores donde solo se podía tener conectividad IP mediante solicitud y esta se eliminaba una vez que ya no es requerida.

LTE también conocida como Evolved Packet System (EPS) tiene tres principales componentes llamados User Equipment (UE), la Evolved UMTS Terrestrial Radio Access Network (E-UTRAN) y el EPC. Dentro de esta arquitectura el EPC comunica la red móvil con el resto del mundo a través de Internet o a redes corporativas.

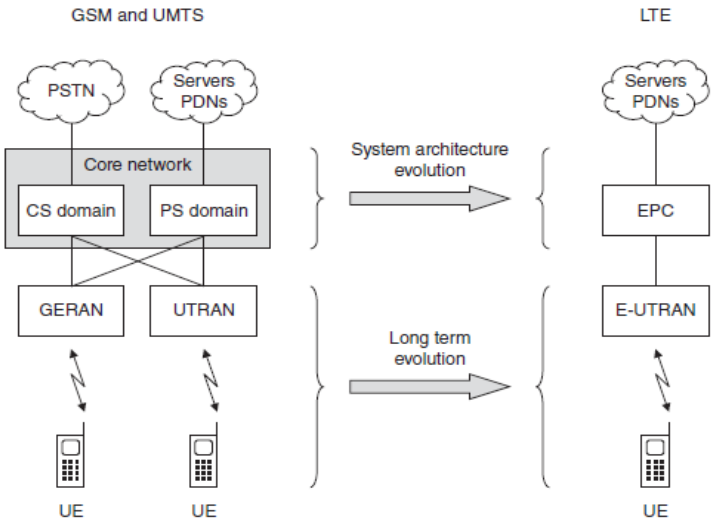


Figura 3. Evolución de la arquitectura del sistema de 3G a LTE. [2]

2. "An Introducction to LTE, LTE Advance, SAE, VoLTE and 4G Mobile communications"; Christopher Cox, Wiley, UK.

User Equipment.

Está compuesto por tres elementos distintos llamados Mobile Termination (MT) que maneja todas las funciones de comunicación, por ejemplo, un dongle LTE; el Terminal Equipment (TE) que es donde finalizan y originan los flujos de datos; y el Universal Integrated Circuid Card (UICC) conocida normalmente como la tarjeta SIM, que ejecuta un programa coincido como Universal Suscriber Identity Module (USIM) que contiene los datos específicos del usuario para poder realizar la conexión con la red.

E-UTRAN

Es la encargada de la comunicación entre los UE y el EPC y solo tiene un elemento funcional llamado Evolved Node B (eNB).

Las eNB corresponde a una radio base que controla la comunicación de los UE en una o más células. Estas tienen dos funciones principales, la primera consta de enviar y recibir los mensajes de los UE e intercambiar sus datos con la red; la segunda permite el control de los comandos de señalización relacionados a las transmisiones de radio, como la asignación, cambio y liberación de los recursos en las radio bases.

Cada uno de estos elementos cuenta con dos interfaces lógicas, la definida como S1 que conecta con la EPC y la interfaz X2 que comunica las eNBs para enviar principalmente señalización y paquetes durante el handover.

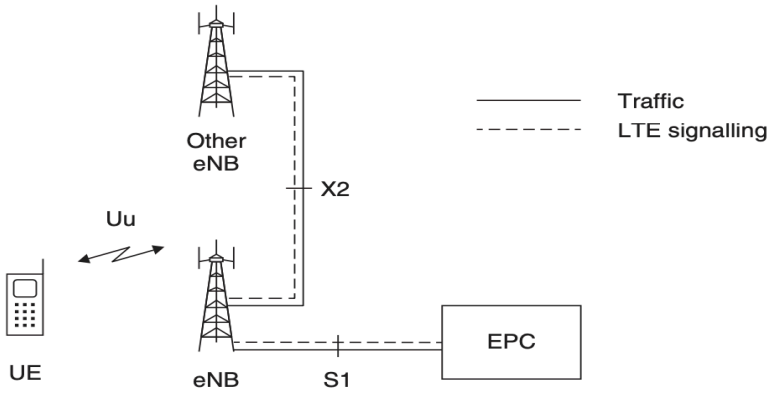


Figura 4. Arquitectura E-UTRAN. [2]

Las eNB no se encuentran conectadas directamente entre si y con el EPC, por lo que utilizan una red de transporte basada en IP.

EPC

El EPC está compuesto por distintos dispositivos con funciones especificas, estos son:

2. "An Introducction to LTE, LTE Advance, SAE, VoLTE and 4G Mobile communications"; Christopher Cox, Wiley, UK.

- Home subscriber server (HSS). Es la base de datos que contiene toda la información del usuario de la red del operador móvil.

- Packet data network Gateway (P-GW). Es el punto en donde se comunica con uno o más dispositivos de redes del mismo tipo PDN y redes externas como Internet. Cada PDN es diferenciado por un Access point name (APN), que las operadoras usan para diferenciar clientes o servicios.

- Serving Gateway (S-GW). Se desempeña como un router que envía los datos de las radio bases a el P-GW.

- Mobility management entity (MME). Controla las operaciones móviles enviando mensajes de señalización para cuestiones de la seguridad y administración de los flujos de datos. También controla a los otros elementos dentro del PDN enviando mensajes de señalización.

Protocolos de Comunicación.

Cada interfaz de comunicación esta asociada a un conjunto de protocolos, estos se encuentran en dos planos de comunicación. El plano de usuario que se encarga de gestionar los datos interesantes para el usuario, mientras que en el plano de control se gestionan datos de los dispositivos de red.

Existen tres tipos de protocolos, de señalización que definen la comunicación entre dos dispositivos que intercambien mensajes, de plano de usuario que manipulan los datos del usuario que envían a través de la red y los de transporte que llevan los mensajes de señalización de un punto a otro.

Protocolos de transporte.

Interfaz aérea.
La interfaz aérea se encuentra entre el dispositivo móvil y la estación base. Esta interfaz utiliza protocolos de las siguientes capas del modelo OSI.
En la capa física se encarga de la modulación, codificación de canal, etc.; por lo que utiliza OFDM para el acceso al medio de comunicación. De manera inicial, esta tecnología utiliza anchos de banda de entre 1.4 [MHz] a 20 [MHz] que permiten velocidades desde aproximadamente 6 [Mbps] hasta a 100 [Mbps] respectivamente. Esto ya que en cada asignación dinámica de timeslots de 1 [ms] se permite utilizar las subportadoras moduladas en QPSK, 16QAM o 64QAM.

En la capa dos se utilizan los protocolos MAC, Radio link control (RLC) y el Packet data convergence protocol (PDCP).

Interfaces fijas.
De la misma manera que la interfaz de aire utiliza protocolos dentro de las capas del modelo OSI. En estas interfaces se pueden utilizar diferentes protocolos, como puede ser Ethernet dentro de una red MPLS que comunica los elementos de la red con otra red IP asociada.

Protocolos del plano de Usuario.

El plano de usuario contiene mecanismos para poder establecer comunicación entre los UE y el P-GW, que responde rápido ante los cambios de la locación móvil, se utiliza un protocolo conocido como GPRS Tunneling Protocolo User Part (GTP-U). Mientras que para comunicar el S-GW con el P-GW se utiliza como alternativa túneles GRE.

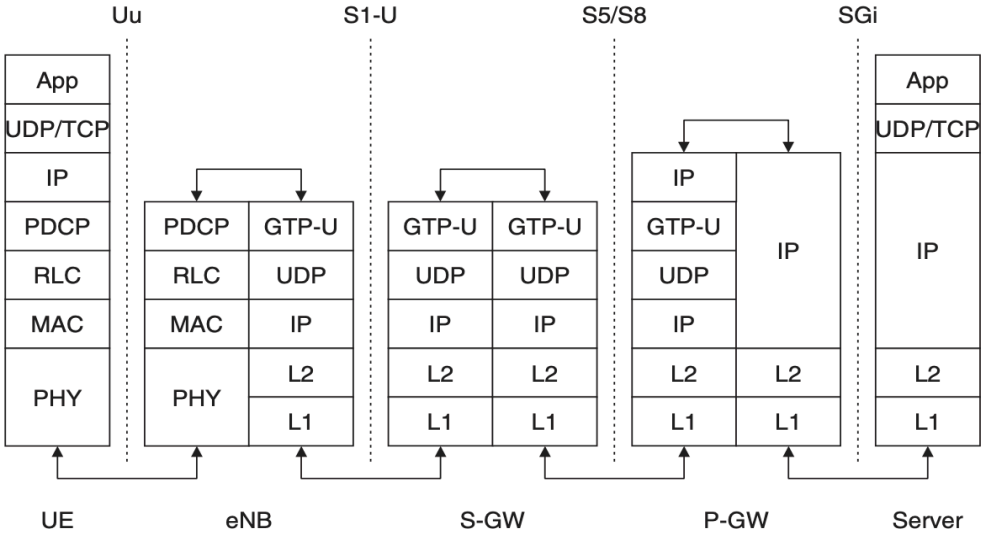


Figura 5. Pila de protocolos utilizados para intercambiar datos desde un dispositivo móvil a un servidor externo. [2]

Protocolos de señalización.

Se utilizan diferentes tipos de señalización, donde se utiliza prácticamente uno distinto por cada intercambio de mensajes entre cada elemento del EPC, eNB y UE.

En la interfaz aérea se utiliza el Radio Resource Control (RRC), que realiza la señalización entre la UE y eNB para gestionar los recursos de radio, así como la petición y asignación de canales de radio, mediciones e información de broadcast. Los dispositivos MME se comunican con los eNB que pertenezcan a una misma área utilizando el S1 Application Protocol (S1-AP), mientras que dos eNB se comunicarán con el protocolo X2-AP. Los dispositivos dentro del EPC también utilizan túneles para la parte de control GTP-C.

2. "An Introducction to LTE, LTE Advance, SAE, VoLTE and 4G Mobile communications"; Christopher Cox, Wiley, UK.

Características.

En su versión operativa más reciente, el release 10 (ya que los posteriores están diseñados para otro tipo de aplicaciones) cuenta con las siguientes características:

LTE release 10	
Data rate	DL 3 [Gbps] UL 1.5 [Gbps]
Eficiencia espectral	30 [bps/Hz]
Carrier Aggregation	5 portadoras para un BW de 100 [MHz]
MIMO	hasta DL 8x8 UL 4x4
Performance cell Edges	con DL 2x2 al menos 2.40 [bps/Hz/cell]
Modulación DL	QPSK, 16QAM, 64QAM, 128QAM
Modulación UL	QPSK, 16QAM, 64QAM, 128QAM
Bandas	450-470, 698-862, 790-862, 2300-2400, 3400-4200, 4400-4990 [MHz]

Tabla 2. Características LTE release 10.

iii. WLAN (Wifi).

Con el éxito de las redes LAN cableadas se buscó emularlo de manera inalámbrica con las WLAN. Utiliza bandas sin licencia aproximadamente en los 2.4 [GHz] y 5 [GHz]. Dependiendo del estándar y el fabricante del dispositivo, las redes WLAN pueden proveer desde 11 [Mbps] hasta casi 5 [Gbps] de velocidad en los últimos estándares (802.11ax).

El comité 802.11 de la IEEE es el encargado de realizar los estándares de WLAN o Wifi, cada uno con sus características y mejoras para soportar una mayor cantidad de dispositivos conectados, distancia de cobertura y mayores velocidades.

Algunas ventajas de utilizar esta tecnología son:

- Movilidad. Permite acceder al usuario a la información sin importar su posición.
- Menor Infraestructura. Evita que se tengan que instalar dispositivos locales para conectarse a la red o realizar adecuaciones a la estructura de los edificios u oficinas.
- Reducción de costos. Se requiere una menor cantidad de infraestructura y cableado para brindar conectividad lo que reduce costos y tiempos de instalación por usuario, principalmente en ambientes dinámicos.

Las desventajas de utilizar WLAN son:

- Interferencia y confiabilidad. Ya que se utiliza un medio compartido es susceptible a interferencias de otras redes vecinas y señales electromagnéticas de otro origen (radiación de microondas, por ejemplo). Se emplean mecanismos como el ARQ y FEC para poder aumentar la confiabilidad.

- Security. Las señales de radio no se encuentran confinadas dentro de los edificios o construcciones, por lo que la red puede ser escuchada por dispositivos no deseados. Para poder proveer un nivel de privacidad a los datos se utilizan técnicas de encriptación.

- Consumo de Energía. Las redes WLAN generalmente están ligadas a aplicaciones móviles y brindar una alternativa de conectividad a redes cableadas, por lo que estos dispositivos consumen gran cantidad de energía. Los dispositivos deben ser diseñados para ser energéticamente eficientes.

- Throughput. Para soportar múltiples transmisiones simultaneas, se utilizan técnicas para aprovechar el espectro radioeléctrico y brindar conectividad a todos los usuarios lo que disminuye el throughput.

Dispositivos WLAN.

Existen tres tipos de dispositivos básicos para redes WLAN. Estos son:

- Adaptador de Red. Permite la comunicación entre el sistema operativo del dispositivo o también llamada estación (STA). Los adaptadores inalámbricos están hechos con el mismo principio de la contraparte cableada, pueden ser mediante USB, PMCIA, Card bus o PCI.

- Puntos de Acceso. Los puntos de acceso o APs son el equivalente a los Hubs cableados. Recibe y transmite los mensajes entre la red cableada y un grupo de STAs. Cada AP generalmente se comunica con la red a través de Ethernet y entre ellos de diferente manera dependiendo el fabricante. Similar a lo que ocurre en la red celular, los APs pueden soportar el handover y handoff para permitir mayor movilidad y cobertura a los usuarios.

- Outdoor Bridges. Los outdoor bridges se utilizan para conectar otro edificio con la misma red LAN. Cuando el costo de realizar este cableado y adecuaciones es muy elevado o existe algún obstáculo que dificulte realizar la instalación, la tecnología WLAN es una opción para que de manera inalámbrica quedan comunicarse. Algunos APs son capaces de realizar esta función.

Topologías WLAN.

Las redes WLAN pueden ser implementadas en las siguientes topologías:

- Peer-to-peer (Ad-hoc).

Los dispositivos en un mismo rango de cobertura o célula se comunican directamente unos con otros. En el estándar 802.11 esta topología se le llama Independent Basic Service Set (IBSS), esta red generalmente tiene un corto periodo de vida y están diseñadas para un propósito en especifico.

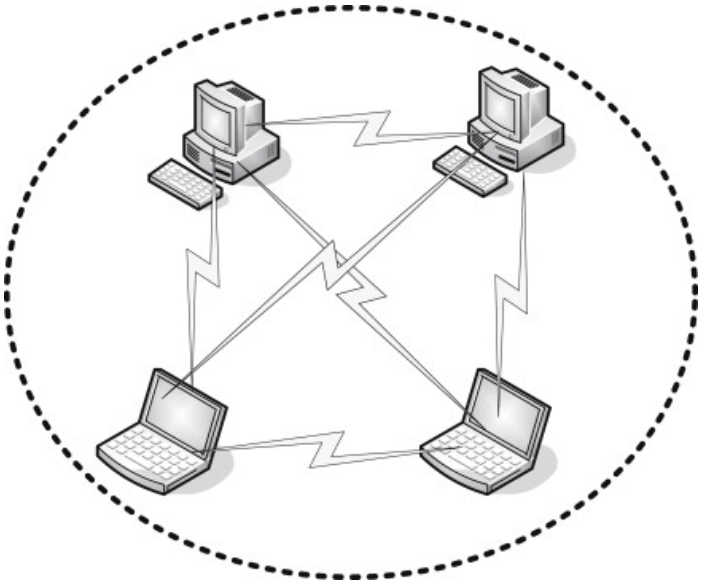


Figura 6. Red WLAN Ad hoc. [3]

- AP-based.

Permite a un cliente inalámbrico comunicarse con cualquier otro dispositivo en la red inalámbrica o cableada. Esta es la topología más comúnmente utilizada en donde solo se extiende la conectividad de la red cableada a los dispositivos inalámbricos.

Dentro del estándar 802.11 define el Based Service Set (BSS) cuando un AP actúa como servidor de un canal o célula WLAN, así que actúa como bridge cuando se comunican dos STA conectadas al mismo BSS.

La topología ESS consiste en un conjunto de BSS interconectados inalámbricamente o mediante un medio cableado llamado sistema de distribución.

3. "Wireless Networked Video", Anthony C. Caputo, in Digital Video Surveillance and Security (Second Edition), 2014

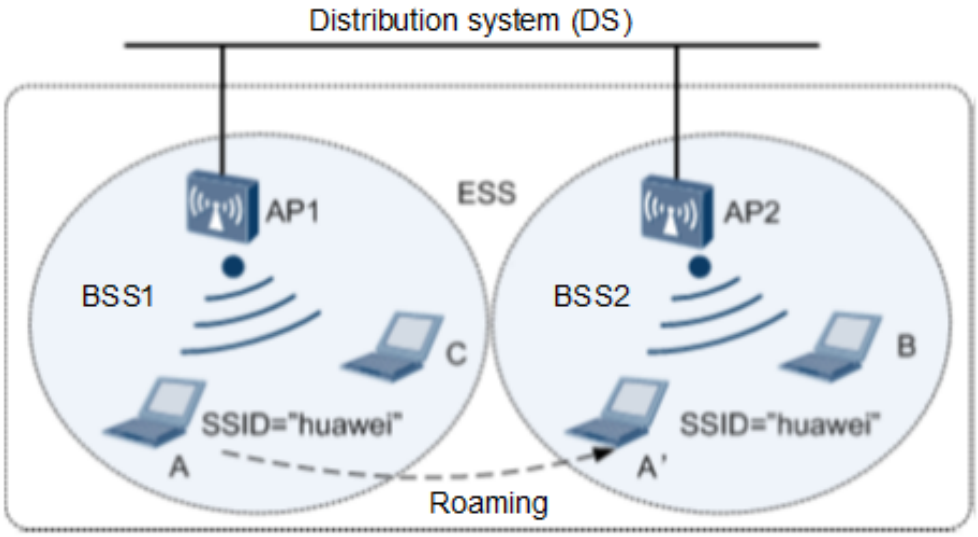


Figura 7. Red WLAN AP-based y topología ESS. [4]

iv. LPWAN.

Las redes de área amplia de baja potencia o LPWAN por sus siglas en inglés, se usa para referirse a cualquier red de datos que se comunica de manera inalámbrica con una potencia más baja que las redes de comunicación convencional. Además, son capaces de comunicarse a mayores distancias que otras redes de baja potencia como las redes Bluetooth o NFC.

Las redes LPWAN tienen un muy limitado ancho de banda por lo que no es una opción funcional para la mayoría de las aplicaciones tradicionales como la transmisión de voz, video o mensajería; por esto, se considera una tecnología complementaria a las que rindan comunicación inalámbrica actualmente y solo se emplea en soluciones de IoT en donde se necesiten enviar y recibir datos de sensores o en soluciones M2M.

4. Fundamentos de WLAN, Sección 6 Conceptos básicos en WLAN: https://forum.huawei.com/enterprise/es/de-principiante-a-experto-fundamentos-de-wlan-sección-6-conceptos-básicos-en-wlan/thread/485665-100239

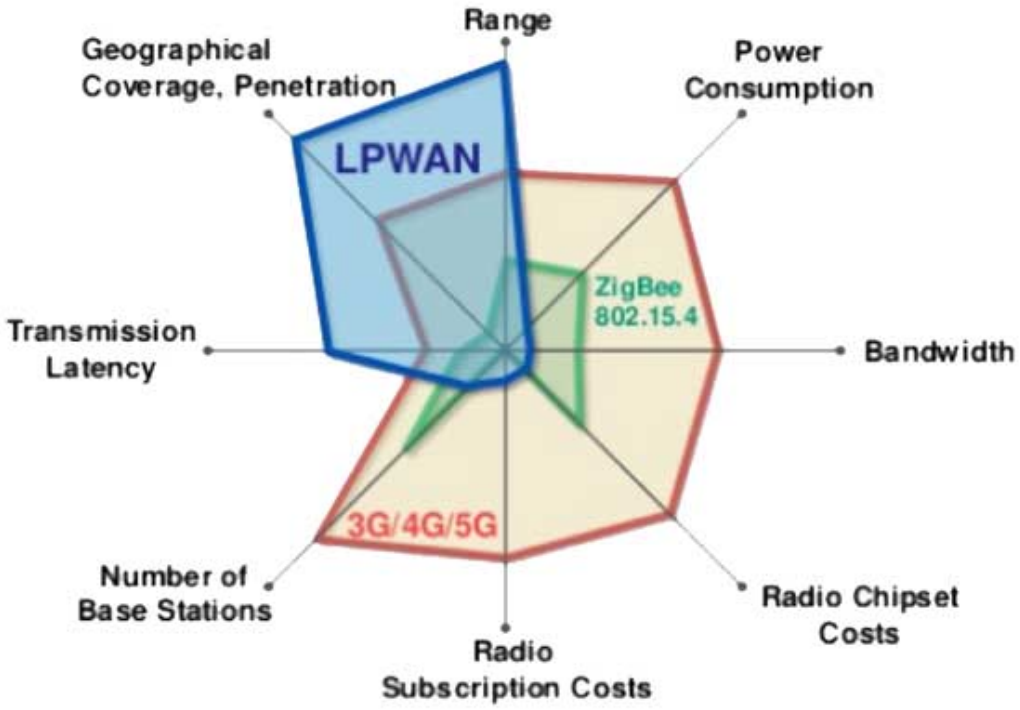


Figura 8. Comparativa LPWAN vs otras tecnologías inalambricas. [5]

Las redes LPWAN deben cumplir con las siguientes características básicas:

- Bajo consumo. Está destinado a minimizar el coste de energía para los sistemas de IoT a diferencia de las redes móviles tradicionales. Esto permite a los sensores y actuadores funciones durante un periodo muy grande de tiempo (idealmente, años) sin necesidad de cambiar las baterías.

- Amplia cobertura. Las LPWAN tienen un radio de cobertura que varía de algunos pocos kilómetros en zonas urbanas hasta varias decenas de kilómetros en áreas sin mucha interferencia.

- Bajo ancho de banda. La mayoría de los dispositivos IoT, envían cantidades de información muy pequeñas solo unas pocas veces al día, debido a las mediciones y acciones que ejecutan. Las redes LPWAN son diseñadas para transportar mensajes pequeños, lo que reduce los costos del sistema.

- Bajo costo. La mayoría de las soluciones LPWAN emplean un modelo de suscripción en donde se cobra por cada dispositivo conectado a la red o por la cantidad de mensajes permitidos por dispositivo. Además, generalmente se utilizan bandas de frecuencia libres, en donde se puede reducir costos ya que no es necesario una licencia.

5. ¿Qué es una red IOT (LPWAN)?: https://osiberia.org/que-es-una-red-iot-lpwan/

En este nuevo mercado aún no se encuentra definido un estándar o alguna tecnología que se deba seguir, es por eso por lo que los proveedores que existen brindan sus propia soluciones y características. Algunos de los proveedores más importantes son los siguientes:

Sigfox.

Sigfox en una de las redes LPWAN con mayor cobertura que existen hasta ahora en el mercado, propiedad de una empresa francesa. Utiliza las bandas sin licencia por debajo de 1 [GHz] (902 [MHz] para México) con una topología de estrella, donde cualquier estación base dentro del rango de transmisión de los dispositivos puede recibir cualquier transmisión. Para poder utilizar esta red es necesario utilizar un dispositivo certificado por Sigfox o "Sigfox ready" y contratar el servicio a través de un operador (WND en México). Cualquier fabricante de Hardware puede desarrollar nuevos dispositivos para utilizar esta tecnología, cumpliendo con los requisitos establecidos por Sigfox.

Los servicios de Sigfox incluyen:

- Recepción y transmisión de mensajes. Se tiene permitido que los dispositivos puedan enviar un máximo de 140 mensajes de 12 bytes diarios, mientras que para enviar información a los dispositivos Sigfox garantiza 4 mensajes de 8 bytes al día.

- Gestión en la nube. Servicio también llamado Backend, permite la gestión en la nube en donde podemos conocer el estado actual de los dispositivos registrados, enviar y visualizar los datos recibidos de los sensores y actuadores. Permite la creación de realizar funciones llamadas Callbacks con las que es posible reenviar los datos a algún punto para su análisis y enviar nueva configuración a los dispositivos. También tiene el servicio de API REST para poder solicitar los datos de los dispositivos a petición de alguna aplicación del usuario.

- Servicio de geolocalización y rastreo. Cada mensaje enviado puede tener incluida la información de la posición actual del dispositivo.

Funcionamiento.

Sigfox utiliza la Ultra Narrow Band (UNB) para enviar sus mensajes lo que hace que sea muy resistente a interferencias propias de la banda sin licencia, colisiones y a señales maliciosas como pueden ser las producidas por Jammers; también le permite de transmitir a distancias muy grandes de hasta 50 [km].

Cada mensaje es modulado en D-BPSK con un ancho de banda de solo 100 [Hz], posteriormente se envía de manera aleatoria dentro de la banda de los 902 [MHz]. Estos mensajes no se sincronizan, lo que permite optimizar su uso ya que se evita un intercambio de mensajes de sincronía, evita la necesidad de agregar encabezados y favorece a la movilidad de los dispositivos.

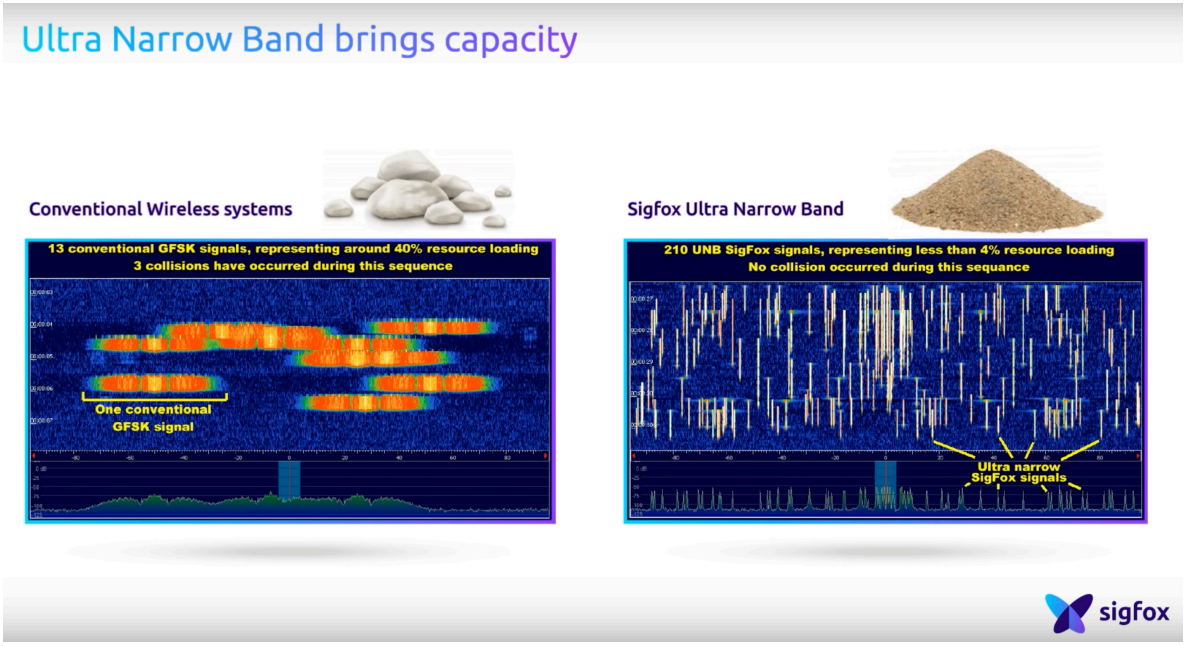


Figura 9. Señales Sigfox de UNB vs una señal inalámbrica GFSK convencional. [6]

Todas las estaciones base se encuentran escuchando permanentemente el espectro para poder interpretar las señales UNB en el momento en que sean recibidas. Además, ya que está pensado en una topología de estrella, se busca que todas las transmisiones sean escuchadas por al menos tres estaciones base al mismo tiempo, dando una diversidad espacial y evitar los mensajes en error.

Los dispositivos se identifican mediante un ID y un token que vienen integrados de fábrica, brindando seguridad a los datos enviados. Sin estos parámetros no es posible decodificar las señales y poder leer los datos de cada sensor o actuador.

LoRaWAN.

Es un protocolo de red LPWAN abierto desarrollado por LoRa Alliance, que permite a los usuarios crear sus redes propias (a diferencia de Sigfox). Está basada en la técnica de modulación Long Range (LoRa), patentada por Semtech, lo cual limita la capacidad de desarrollo de nuevos dispositivos. Esta modulación utiliza un enfoque de modulación de pulsos de espectro extendido o Chirp dentro de la banda ISM de frecuencia libre por debajo de 1 [GHz]. Esta modulación provee a los mensajes de una alta resistencia a las interferencias y un mayor alcance de transmisión.

La banda ISM para América del norte esta definida entre 902-928 [MHz], LoRaWan define 64 canales de subida con un ancho de banda de 125[kHz] y 8 canales de bajada con un ancho de banda de 500[kHz], lo que permite un data rate de 980[bps] a la uplink y 21.9[kbps] de downlink.

6. Sigfox Technology: https://www.sigfox.com/en/what-sigfox/technology

En esta banda la potencia máxima de transmisión es de +30[dBm], pero para la mayoría de los dispositivos es suficiente +20[dBm].

La arquitectura de la red LoRaWan presenta una topología de estrella en donde todos lo sensores y actuadores se conectan con un Gateway, este retransmite los mensajes a los servidores de Backend a través de comunicaciones tradicionales (3G/LTE, WIFI, Ethernet, etc.). LoRaWan utiliza dos capaz de seguridad, una en la red y otra en la aplicación. La seguridad en la aplicación evita que alguien no autorizado tenga acceso a los datos del usuario, mientras que la seguridad en la red garantiza la autenticidad de los nodos o elementos de la red. Encripta la información con AES haciendo un intercambio de llaves utilizando un identificador IEEE EUI64.

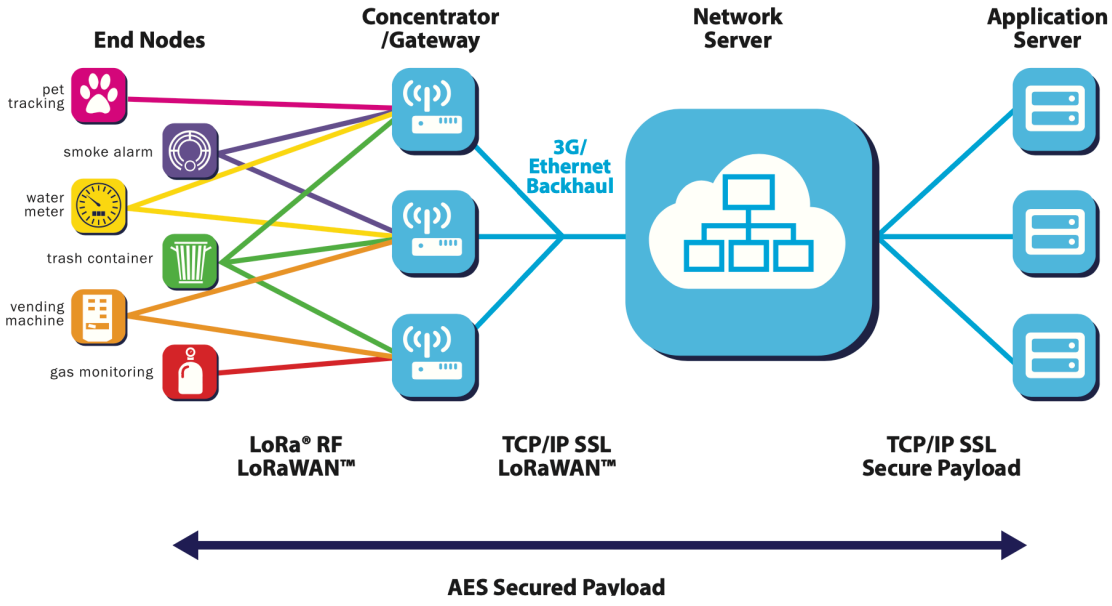


Figura 10. Arquitectura LoRaWan. [7]

Tipos de dispositivos y servicio.

Los dispositivos finales se utilizan para aplicaciones muy diversas, por lo que LoRaWan optimiza los dispositivos dividiéndolos en clases, dependiendo del servicio que se requiere.

- Dispositivos finales bidireccionales (Clase A). Permiten la comunicación bidireccional cuando realiza una transmisión ascendente, seguida de dos ventanas en donde puede recibir una transmisión descendente. Los slots de transmisión son programados por el dispositivo final dependiendo de sus necesidades de comunicación con una variación aleatoria (similar a ALOHA). La clase A en la que tiene menor consumo de energía, ya que solo es capaz de recibir señales por un periodo pequeño de tiempo después de realizar una transmisión. Aquellas

7. What is LoRaWAN?: https://lora-alliance.org/resource-hub/what-lorawanr

transmisiones generadas en el servidor tendrán que esperar la próxima ventana de recepción programada.

- Dispositivos finales bidireccionales con slot de recepción programada (Clase B). Adicionalmente a las ventanas de tiempo aleatorias de la Clase A, los dispositivos de Clase B abren ventanas de recepción a una hora especifica. Al abrir dicha ventana recibe del Gateway un mensaje de sincronización, esto permite al servidor saber cuando el dispositivo esta escuchando.

- Dispositivos finales bidireccionales de recepción máxima (Clase C). Los dispositivos de clase C se encuentran recibiendo continuamente, únicamente dejan de recibir cuando transmite.

v. Bluetooth Mesh.

Bluetooth SMARTH o también conocido como Bluetooth Low Energy (BLE) se refiere a las especificaciones 4.0, 4.1, 4.2 y su mejora 5.0 de la tecnología Bluetooth.

BLE fue diseñado para transmitir pequeñas cantidades de datos y así poder tener un muy bajo consumo de energía. Esto permite a los dispositivos solo estar activos cuando se les soliciten datos o requieran enviarlos.

Arquitectura.

BLE puede soportar 3 tipos diferentes de topología que depende mucho de la cantidad de dispositivos a utilizar y de la aplicación para la cual se necesite, estas pueden ser de tipo Estrella, P2P o Mesh. La que se aplica generalmente a IoT es la topología Mesh.

El Mesh permite establecer comunicación many-to-many entre los dispositivos y permite reenviar mensajes a dispositivos que no se encuentran en el rango del dispositivo origen. De esta manera, una red de tipo Mesh puede ocupar un área física muy amplia y tener una gran cantidad de dispositivos. Esta topología esta diseñada para aplicaciones de IoT en donde es necesario que los sensores y actuadores compartan información entre si, se requiera enviar datos a través de área extensas en donde la conectividad es limitada y evitar tener una gran cantidad de dispositivos centrales. El mesh utiliza como tecnología de comunicación inalámbrica a BLE. Bluetooth mesh es considerado un protocolo de networking.

Bluetooth Mesh tiene características que favorecen su implementación en escenarios de IoT. Las cuales son:

- Cobertura en áreas muy amplias.
- Capacidad de gestionar un gran numero de dispositivos.
- Optimización a un bajo consumo de energía.
- Uso eficiente de los recursos de radio, favoreciendo la escalabilidad de la red.
- Compatibilidad con dispositivos actuales del mercado como smartphones, tablets y computadoras.
- Estándar, con seguridad de grado gubernamental.

Comunicación.

Utiliza un sistema de mensajes de publish/suscribe. El publishing corresponde al envío de mensajes a direcciones cuyos nombres y datos tengan sentido en capaz superiores para que el usuario los pueda entender. El suscribing corresponde a la capacidad de los dispositivos de ser configurados para recibir mensajes que fueron enviados de direcciones en particular. Cuando un dispositivo publica un mensaje, todos los demás dispositivos que se encuentran suscritos a esa dirección reciben una copia de este mensaje, lo procesan y reaccionan de alguna manera.

Bluetooth mesh utiliza el mecanismo de Relay para retransmitir mensajes de otros dispositivos. Con esto, los dispositivos son capaces de comunicarse con otros que no se encuentran en el rango de radio donde se publicó originalmente. Los mensajes pueden ser retransmitidos en muchas ocasiones, a lo que se conoce como “hop”. El máximo de hops permitido es de 127, suficiente para cubrir un área muy amplia.

El “Estado” es un concepto clave para su funcionamiento. Cada dispositivo tiene un conjunto de valores de estado independientes, que representan una condición del dispositivo. Modificando el valor de los estados, se modifica una condición física del dispositivo. Los mensajes, estados y la respuesta de los dispositivos se encuentran definidos en especificaciones conocidas como “Modelos”.

Para optimizar el uso de energía, existen dispositivos que fueron diseñados para que la batería dure por muchos años, estos son llamados Low Power Nodes (LPN). Los LPN establecen una comunicación con nodos designados como “Friends”, estos a su vez guardan mensajes que van dirigidos a LPN y los entregan una vez que estos preguntan por los mensajes. La relación entre estos dispositivos es llamada “friendship”.

El mesh especifica un rol de dispositivo llamado “Proxy Node”, que es el encargado de enviar y recibir datos dentro o fuera de la red Bluetooth. Estos dispositivos pueden ser smartphones o PCs ya que estos dispositivos son capaces de soportar BLE.

Dispositivos.

Dentro de esta red se encuentran distintos tipos de dispositivos. Los más característicos dentro de la red, son:

- Nodo: Los nodos son todos los elementos que componen la red.
- Low Power Node: Es un nodo que soporta y tiene habilitada la función de Low Power y opera en conjunto con otros nodos “friend”.
- Proxy Node: Es el nodo encargado de enviar y recibir mensajes al exterior de la red y viceversa.

Protocolo.

Bluetooth mesh define una nueva pila de protocolo, en donde cada una de sus capas tiene una función en especifico.

Figura 11. Pila del protocolo Bluetooth Mesh. [8]

- BLE

Bluetooth mesh utiliza BLE en una capa similar a la capa física del modelo OSI. Trabaja en la banda de frecuencias ISM, 40 canales de comunicación de 2 [MHz] (3 canales de advertizing), modulación GPSK que permite tasas de transmisión de datos de has 2 [MHz]. Utiliza topologías de P2P (que incluye las Piconets), Broadcast y Mesh.

- Bearer layer.

La capa de Bearer define como los PDUs son transportados a través de la capa inferior de BLE. Existen dos tipos de PDUs definidos, el “Advertising bearer” y el “GATT bearer”.

8. An Intro to Bluetooth Mesh Part 2: https://www.bluetooth.com/blog/an-intro-to-bluetooth-mesh-part2/?utm_campaign=mesh&utm_source=internal&utm_medium=blog&utm_content=an-intro-to-bluetooth-mesh-part-1

- Network layer.

La capa de red define varios tipos de direccionamiento de mensajes y formatos de mensajes de red. Los comportamientos de Proxy y Relay son implementados por esta capa.

- Lower Transport Layer.

La capa baja de transporte se encarga de la segmentación y el ensamble de los PDUs cuando es requerido.

- Upper transport layer.

La capa superior de transporte es responsable de la encriptación, desencriptación y autenticación de los datos de aplicación que se reciben y envían a la capa de acceso. También es responsable de mensajes de control de transporte como los heartbeats y los relacionados a la relación de “friendship”.

- Access layer.

La capa de acceso es responsable del formato de los datos de aplicación, definir y controlar la encriptación y desencriptación realizada en la capa superior de transporte y verificar que los datos recibidos correspondan a la red y aplicación correcta antes de enviarlo a capas superiores.

- Fundation models.

La capa de modelos de fundación es responsable de la implementación de estos modelos relacionados con la configuración y gestión de la red.

- Models.

La capa de modelos se encarga de la implementación de los modelos, comportamientos, mensajes, estados y aspectos adicionales.

Seguridad.

La seguridad es el punto central del diseño del mesh y su uso es obligatorio. Cada paquete está encriptado y autenticado. Los ataques de repetición se evitan mediante el uso de los números de secuencia. Se esta protegido contra ataques de man-in-the-middle con el uso de criptografía asimétrica durante procedimientos importantes. Se proporciona protección contra trash-in-can, que explotan dispositivos desechados. Las claves de seguridad se actualizan cuando es necesario.

La "separación de preocupaciones" es un principio importante del Bluetooth mesh. La seguridad de la red y de las aplicaciones, son independientes entre sí. Se utilizan diferentes claves de seguridad para proteger las operaciones de la capa de red y proteger el contenido de mensajes específicos de la aplicación. El resultado de esto es, un nodo tiene acceso total a los datos en los mensajes transmitidos que pertenezcan a la misma aplicación por lo que tienen la clave; pero si bien un nodo puede transmitir a los pertenecientes a otra aplicación,

no puede ver el contenido de esos mensajes. Todos los mensajes son encriptados con AES128.

El proceso llamado "Provisioning", se utiliza cuando se hace el intercambio de llaves para poder encriptar la comunicación, se agrega el nuevo dispositivo a la red y a la aplicación correspondiente.

3. SERVICIOS EN LA NUBE (CLOUD SERVICES).

a. INTRODUCCIÓN.

Los servicios en la nube son una tecnología que permite el acceso remoto a servidores, almacenamiento, bases de datos, redes, análisis, inteligencia y otros servicios de TI a través de Internet. Por lo que no es necesario que una empresa implemente estos servicios de manera local.

Se evita la inversión en hardware para la implementación de nuevos servicios y destinar recursos a administrar y monitorear estos sistemas, lo que permite una reducción en costos operativos, implementaciones más eficaces, ambientes seguros y solo pagar por lo utilizado.

Características de los servicios en la nube.

Los proveedores de este tipo de servicios son los dueños del hardware, pero deben de cumplir con algunas características generales, las cuales son ventajas que debe tener el hacer uso de estas tecnologías. Las ventajas son:

- Costo. Los usuarios evitan hacer la inversión en infraestructura TI, además de eliminar la necesidad de personal capacitado para operar dicha infraestructura.

- Multiplataforma: Solo es necesaria un dispositivo con conexión a Internet para poder hacer uso de estos servicios.

- Flexibilidad: Estos servicios permiten dimensionar los recursos necesarios con base en la utilización y el presupuesto de los usuarios.
- Global. Permiten la asignación de recursos cuando se necesiten y en la ubicación geográfica conveniente. Es decir, se utiliza la distribución geográfica de estos servicios en función de la ubicación y necesidades de los usuarios.

- Confiabilidad. Facilita y reducen los costos al realizar copias de seguridad de los datos y la recuperación ante desastres, ya que los proveedores de servicios deben proveer respaldos y la distribución de estos datos.

- Servicio medido: Los recursos utilizados se pueden determinar de manera anticipada, creando perfiles de presupuestos de utilización y de costos. Permite crear reportes y análisis de la utilización por periodos de tiempo que permiten conocer el uso real del servicio.

- Seguridad. Los servicios en la nube son bastante seguros en comparación con la computación tradicional. Debido a la cantidad de usuarios que se encuentran en posible riesgo, los proveedores de servicio implementan un robusto sistema de seguridad por encima de los estándares gubernamentales, además cuentan con mayores recursos para desarrollar sistemas de detención, eliminación y detección de amenazas de lo que podría tener la mayoría de los usuarios de estos servicios. Es responsabilidad del proveedor las actualizaciones de seguridad.

b. TIPOS DE SERVICIOS EN LA NUBE.

Los servicios en la nube pueden estar divididos en dos grupos, en el tipo de nube contratada para desplegar los servicios o en el tipo de servicio. Existen distintos proveedores de servicios en la nube, en donde destacan AWS, Azure, Google Cloud, IBM Cloud o Oracle Cloud.

Figura 12. Proveedores de servicios en la nube. [9,10,11]

Tipo de nube.

- Nube pública. Las nubes publicas son propiedad de los proveedores de servicios, por lo que ellos son los encargados de administrar y operar sus recursos. Los clientes de este tipo de nube comparten la infraestructura, por lo que tiene que ser una plataforma multitenant.

- Nube privada. Es propiedad de una sola organización dentro de su infraestructura sin un punto de acceso público. La nube privada es aquella que se mantienen dentro

9. Amazon Web Services: https://aws.amazon.com/training/
10. Azure: https://azure.microsoft.com/es-mx/overview/what-is-cloud-computing/
11. Google Cloud: https://cloud.google.com/why-google-cloud

de una red privada y puede encontrarse dentro de los centros de datos propios de una compañía.

- Nube híbrida. Las nubes hibridas combinan las privadas y publicas aportando mayor flexibilidad, seguridad, optimización de la infraestructura y mayor soporte para el desarrollo.

- Nube comunitaria. Es aquella nube en donde diferentes organizaciones comparten recursos dentro de una misma instancia en la nube para resolver un problema común.

Tipos de servicio.

- Infraestructura como servicio (IaaS). Se alquila infraestructura de TI (dispositivos de red, servidores, sistemas operativos, etc.) y se paga por su uso durante un periodo de tiempo. Los usuarios son responsables de administrar el funcionamiento de los dispositivos (hardware y sistemas operativos).

- Plataforma como servicio (PaaS). Este servicio elimina la necesidad de las compañías y organizaciones de administrar la infraestructura (normalmente hardware y sistemas operativos) y les permiten solo concentrarse en el desarrollo, implementación y la administración de sus aplicaciones.

- Software como servicio (SaaS). El software como servicio brinda la posibilidad a las organizaciones de obtener un producto completo listo para ser ejecutado, donde el usuario solo hace uso de el y lo administra dentro de su propia organización, evitando gastos en el desarrollo, implementación y gestión de toda la infraestructura necesaria para ello.

c. RECEPCIÓN DE DATOS.

Los proveedores de servicios en la nube reciben información desde los dispositivos de sus clientes que capturan datos y los envías a las distintas plataformas para su almacenamiento y posterior análisis. Todos ellos utilizan generalmente dos tecnologías presentes en el mercado, el API que esta enfocado para cualquier dispositivo inteligente con conexión a Internet y el protocolo MQTT.

Para poder tener un control de los dispositivos de los cuales recibir información y tener una capa de seguridad, los proveedores necesitan tener un registro de los dispositivos que estarán enviando datos y se les asigna una clave segura o certificado.

El registro se realiza dentro de la plataforma en la nube, donde se definirán las reglas correspondientes para poder agrupar los dispositivos que pertenezcan a una misma aplicación y poder hacer un análisis de sus datos. Las claves o certificados se instalan en los dispositivos que se estarán comunicando con el backend de la plataforma de IoT para proveer un canal de comunicación seguro, ya que se encriptan los datos para conservar su integridad durante su transporte.

i. API.

Aplication Programing Interface (API) es un conjunto de definiciones y protocolos permitiendo la comunicación entre dos aplicaciones de software.

Cada API está diseñada en un lenguaje de programación en concreto y con atributos que la definen, donde pueden estar variables, rutunas y estructuras de datos. Las API permiten que otros puedan interactuar con funciones un software sin tener que conocer todo el código.

Debido a que se utilizan a través de Internet, la mayoría de las API están diseñadas de acuerdo con los estándares web.

Hay dos tipos de API web habituales:

- SOAP (Simple Object Access Protocol). Un protocolo estándar de intercambio de datos en XML.

- REST (Representational State Transfer). Usa el protocolo HTTP, por lo que los mensajes de control y error son los mismos. Las API RESTfull son las más utilizadas.

Las API RESTfull tienen una arquitectura de cliente servidor en donde el estado de las sesiones se quedan en posesión del cliente. Permite el almacenamiento en cache de cierta información y así evitar algunas interacciones redundantes. Utiliza mensajes auto descriptivos que informan como procesar el mensaje y sistemas de capas que ofrece funciones de seguridad, caché compartido o equilibrar la carga de los mensajes.

ii. MQTT

El Message Query Telemetry Transport, es un protocolo para establecer una comunicación publish/suscribe entre distintos dispositivos. Este protocolo es usado por las soluciones IoT debido a que una de sus principales características el tamaño pequeño de sus mensajes, lo que permite la implementación de hardware de características limitado y redes de bajo ancho de banda y alta latencia.

Se ha convertido en una alternativa a las API-REST HTTP cuando la red tiene características limitadas y un bajo desempeño ya que no se requiere transmitir una gran cantidad de datos.

El protocolo MQTT define dos entidades en la red: un bróker y los clientes.

Bróker.

Bróker es un servidor que tiene como función recibir los mensajes de los clientes y redirigir estos mensajes a un destino relevantes. Un cliente es aquel miembro en la red que puede interactuar con el bróker y recibir mensajes. Un cliente puede ser un sensor o actuador de IoT o una aplicación en un centro de datos que procesa los datos.

Los clientes se conectan a un Bróker y se suscriben a los "topic" a los que están interesados. Esta conexión puede ser una conexión TCP/IP simple o una conexión TLS cifrada para mensajes con información que deba ser protegida. Los clientes también se conectan al Bróker para publicar mensajes en un topic, donde el bróker reenvía los mensajes a todos los clientes que estén suscritos a el mismo topic.

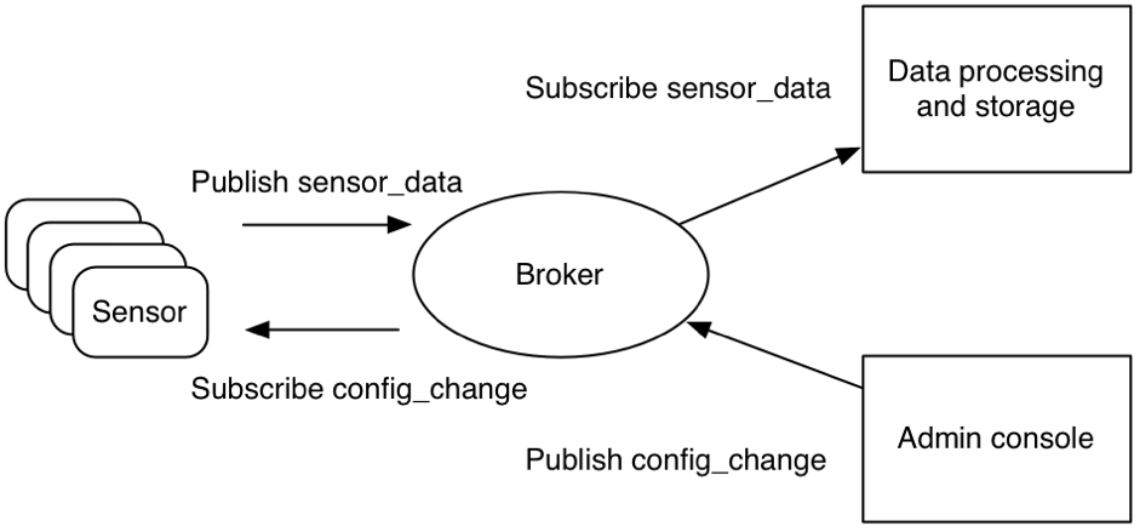


Figura 12. Comunicación MQTT. [12]

Topic y suscripciones.

Los mensajes en MQTT son publicados en los topics y eso es suficiente para crearlos, no es necesario realizar una configuración de ellos. Los topics tienen una estructura jerárquica en donde se crean "rutas", similares a los de los sistemas de archivos, en donde los topics pertenecerán a un nivel. Los clientes pueden recibir mensajes creando suscripciones. Una suscripción puede ser a un topic explícito o puede incluir comodines para poder enviar o recibir a varios topics.

12. Conociendo MQTT: https://www.ibm.com/developerworks/ssa/library/iot-mqtt-why-good-for-iot/index.html

Estructura del mensaje.

Uno de los componentes importantes de MQTT es la estructura de sus mensajes, en donde radica una de sus fortalezas como la ligereza. Se compone de tres partes:

- Cabecera fija. Compuesto de 2-5 bytes obligatorios, donde en el primer byte se encuentra el encabezado de control que especifica el tipo de mensaje o solicitud, y a partir del segundo byte se define la longitud del payload.

- Cabecera variable. Sección opcional del encabezado en donde se agrega información adicional para algunos mensajes.

- Payload. El contenido real del mensaje. Puede tener un máximo de 256 [Mb].

CAPITULO II

1. GESTIÓN VEHICULAR.

a. Redes CANbus.

CAN es un protocolo de comunicaciones desarrollado por la compañía Robert Bosch GmbH, basado en una topología de bus para el envío de mensajes distribuidos, debido a la necesidad de conectar cada vez más dispositivos electrónicos, sensores y ECUs en los automóviles (los automóviles actuales pueden llegar a tener más de 70 ECUs).
Este protocolo fue presentado en febrero de 1986 en el congreso de la SAE. Debido a las características del protocolo se ha incorporado también a otro tipo de vehículos desde trenes, barcos hasta robots industriales.

La aplicación de este protocolo de comunicación provee una reducción en las conexiones entre dispositivos que eran necesarias anteriormente, esto debido a que el protocolo emplea solo un único bus de comunicaciones y evita la necesidad de establecer una conexión punto a punto con cada uno de ellos.

Figura 13. Comparativa del uso de CAN bus. [1]

Alguno de los beneficios que brinda la comunicación CAN Bus son:

- Las ECUs se comunican mediante un solo bus CAN, en lugar de una línea análoga conectada directamente con cada dispositivo, reduciendo errores, cableado y peso.
- El sistema de comunicación CAN permite un diagnóstico de errores y configuración centralizado para todas las ECU. Tiene la capacidad de discernir entre errores puntuales en la transmisión, o errores producidos por el fallo de un nodo, en cuyo caso, tiene la capacidad de desconectarlo para evitar que el error sature la red.
- El sistema es inmune a interferencias electromagnéticas lo que lo vuelve ideal para los vehículos.
- Priorización de mensajes y baja latencia en la entrega de estos. Este protocolo utiliza una categorización mediante IDs.

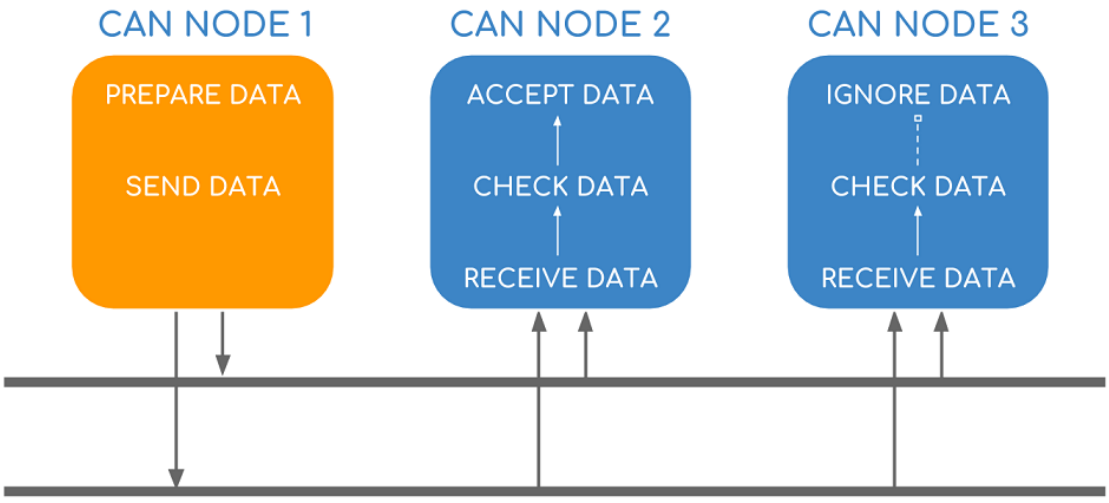


Figura 14. Comunicación entre nodos CAN. [2]

1. Introducción a CAN bus: Descripción, ejemplos y aplicaciones de tiempo real. Proyecto fin de máster, Universidad Politécnica De Madrid.
2. CAN BUS EXPLAINED - A SIMPLE INTRO (2019): https://www.csselectronics.com/screen/page/simple-intro-to-can-bus

b. CAPA FÍSICA.

La capa física describe los aspectos físicos de la comunicación entre los nodos de la red a través del Bus de datos. Esta se encuentra dividida en tres subcapas PLS (señalización física), PMA (Medio físico adjunto) y MDI (Interfaz dependiente del medio). La capa PMA y MDI son distintas dependiendo del país, la industria y las especificaciones del fabricante. El más común se encuentra definido en la ISO 11898.

Las especificaciones del estándar ISO 11898 están dadas por una tasa máxima de señalización de 1 [Mbps] con una longitud de Bus de 40 [m] y un máximo de 30 nodos, también se recomienda una distancia máxima de 0.3 [m] para las líneas stub no terminadas en carga.

La tasa de señalización es inversamente proporcional a la distancia del bus, esto debido principalmente a la variación del tiempo, produciendo interferencia inter-símbolo. Para poder minimizar la degradación en los mensajes deben terminar las líneas con la carga debida. El cable especificado es el par de cobre blindado o sin blindaje con una impedancia característica de 120 [ohm], con la línea terminada con cargas de la misma impedancia para evitar reflexiones y la degradación de la señal. Existen dos modelos recomendados para terminar las líneas, con solo una resistencia de carga RL o utilizando dos resistores RL/2 acoplando un capacitor CL, este último frecuentemente utilizado para filtrar armónicos de alta frecuencia en el bus.

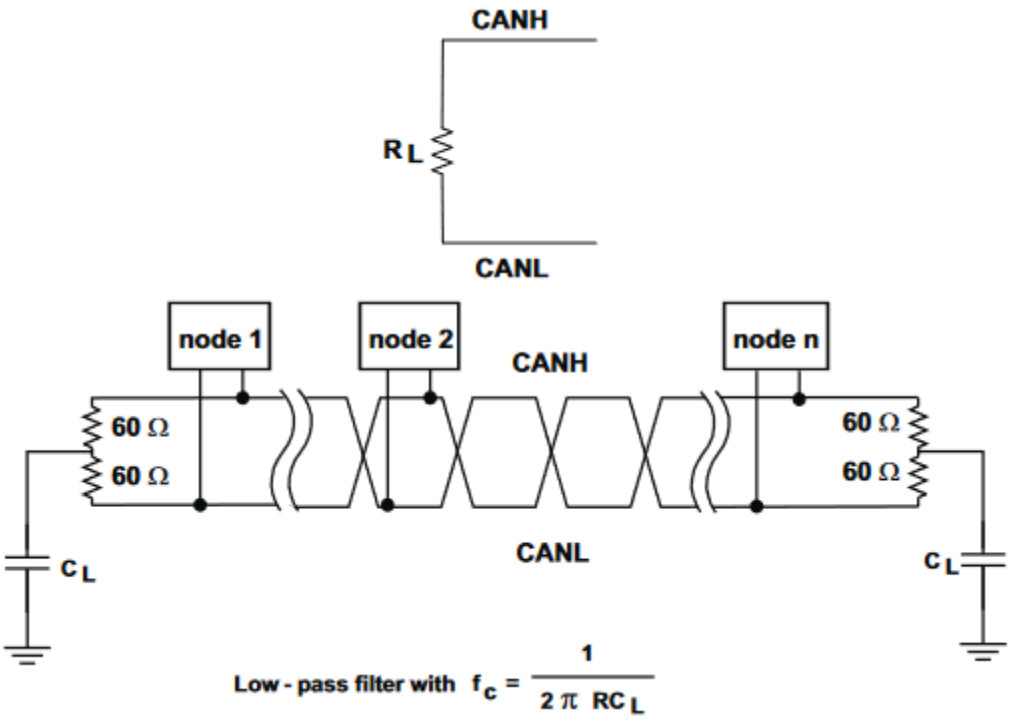


Figura 15. Circuito e impedancias para redes CAN. [3]

3. CAN BUS EXPLAINED - A SIMPLE INTRO (2019): https://www.csselectronics.com/screen/page/simple-intro-to-can-bus

Las redes CAN cuentan con un bus diferencial, donde cada cable cuenta con un voltaje. EL valor entre el cable CAN_H y el CAN_L determina el 1 o 0. Los valores son los siguientes:

- Bit dominante (0): CAN_H = 3.5 V, CAN_L = 1.5 V.
- Bit recesivo (1): CAN_H = 2.5 V, CAN_L = 2.5 V.

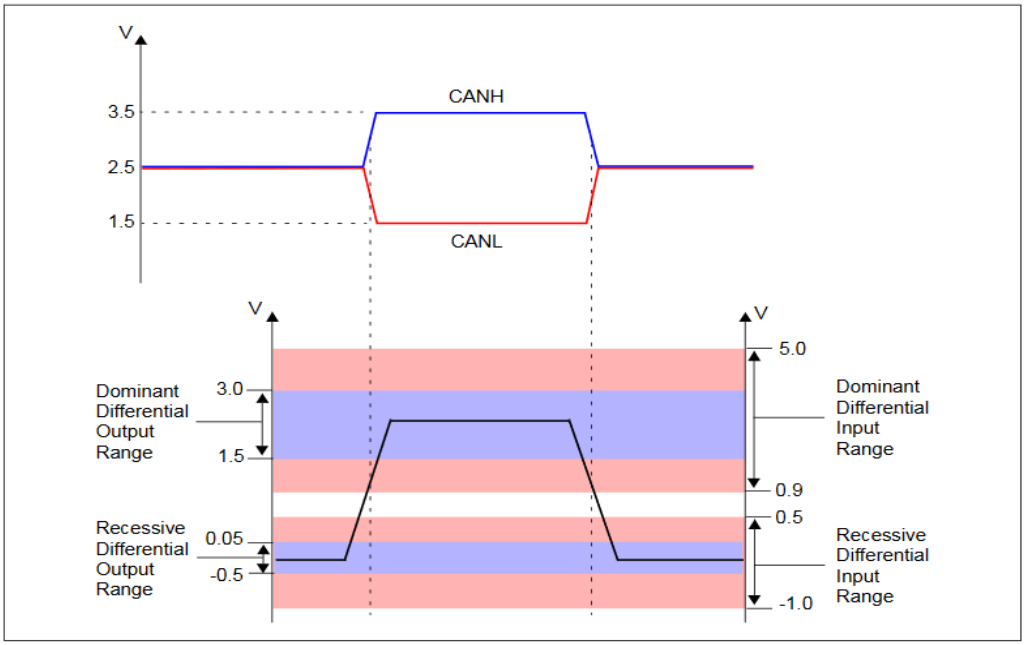


Figura 16. Niveles de voltaje de las señales de una red CAN. [4]

No existe un conector especificado en el estándar, solo debe contar con la impedancia característica para que no haya afectación en los parámetros mínimos de operación. Los protocolos de nivel superior como lo son el CANopen o OBD2 definen el hardware necesario para su implementación, incluyendo los conectores.

c. CAPA DE ENLACE DE DATOS.

La capa de enlace de datos en para redes de tipo CAN se encuentra definido en el estándar ISO 11898-1, ISO 11898-2 y ISO 11898-3. Las redes CAN utilizan comunicación de tipo Broadcast, esto debido a la topología de bus que manejan en donde se permite transmitir y recibir a todos los elementos dentro del mismo domino, posteriormente cada nodo en la red se encarga de filtrar cada uno de los mensajes para aceptarlo o no. El protocolo CAN permite el acceso simultáneo al bus desde diferentes nodos por lo que es necesario un arbitraje. El método de acceso al bus utilizado en CAN es una arbitración no destructiva y de bit-wise, llamado Carrier Sense Multiple Access with Collision Detection and Arbitration on Message Priority (CSMA / CD + AMP). La prioridad del mensaje se decodifica en el identificador CAN.

4. "Controller Area Network Physical Layer Requirements", Texas Instruments. http://www.ti.com/lit/an/slla270/slla270.pdf

FORMATO DE TRAMAS.

En las redes de tipo CAN existen 4 tipos diferentes de tramas, dependiendo de su función y contenido. Estas son:

- DATA FRAME. Contiene información de una fuente con múltiples posibles destinos.
- REMOTE FRAME. Son utilizadas para solicitar información de una DATA FRAME en específico (tienen el mismo identificador).
- ERROR FRAME. Es transmitida cuando un nodo en la red detecta algún error.
- OVERLOAD FRAME. Es usada para realizar control de flujo, para solicitar tiempo adicional antes de transmitir una DATA o REMOTE FRAME.

i. DATA FRAME.

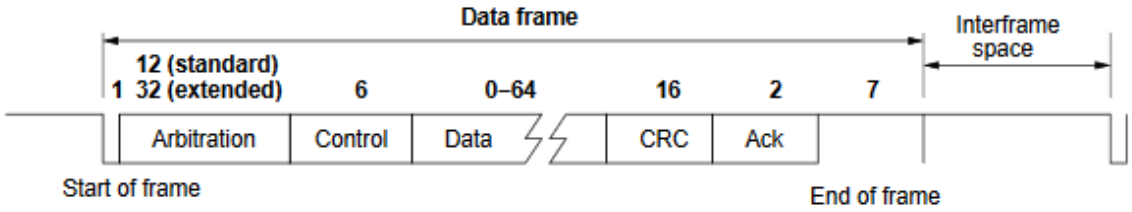


Figura 17. CAN Data frame. [5]

Las Data Frames son utilizada para enviar información desde un nodo origen a uno o varios nodos destino. Las tramas CAN no especifican direcciones destino, pero cada nodo define los mensajes basados en su información, la cual esta contenida en el campo “Identifier” de cada trama. Existen dos formatos de trama, el estándar donde las tramas son definidas por el campo Identifier de 11bits, y las extendidas el mismo campo de 29 bits.

La que se muestra en la figura 17, la longitud de cada uno de los campos de los que está compuesta se encuentra expresados en bits. Todas las tramas comienzan con un solo bit dominante llamado Start of Frame (SOF), que interrumpe el estado recesivo del bus. El campo de arbitraje este compuesto por el “Identifier” y algunos parámetros que dependen del formato de trama. Los otros campos son: el de “Control” conteniendo la información del tipo de mensaje; el campo de “Data” que contiene la información a ser transmitida, puede ser de un tamaño máximo de 8 bytes; el “Checksum” (CRC) que se utiliza para revisar la integridad del mensaje; el “Acknowledge” (ACK) para un dar un acuse de recepción; el Delimitador de trama (EOF) usado para delimitar el final de la trama.

Campo Identifier.

Este campo consiste en 11 bits para el formato estándar y 29 en el formato extendido como se muestra en la figura 18. En ambos casos este campo comienza con 11 bits del identificador base, seguido del bit RTR (Remote Transmition Request) en el formato

5. “A CAN Physical Layer Discussion”, Microchip. http://ww1.microchip.com/downloads/en/appnotes/00228a.pdf

estándar y por el SRR (Substitute Remote Request) en el formato extendido. El RTR define si es una trama de datos o de solicitud remota (Remote Request Frame), siendo dominante para tramas de Datos y recesivo para las tramas de solicitud remota. El SRR solo es un marcador, siempre con valor recesivo, que se utiliza para garantizar la resolución entre tramas estándar y extendidas.

Las tramas extendidas continúan con solo bit de IDE (IDentification Extension), que es siempre de valor recesivo, seguido de los 18 bits menos significativos del Identificador y finalmente el bit RTR.

El bit IDE en las tramas estándar tienen siempre un valor dominante.

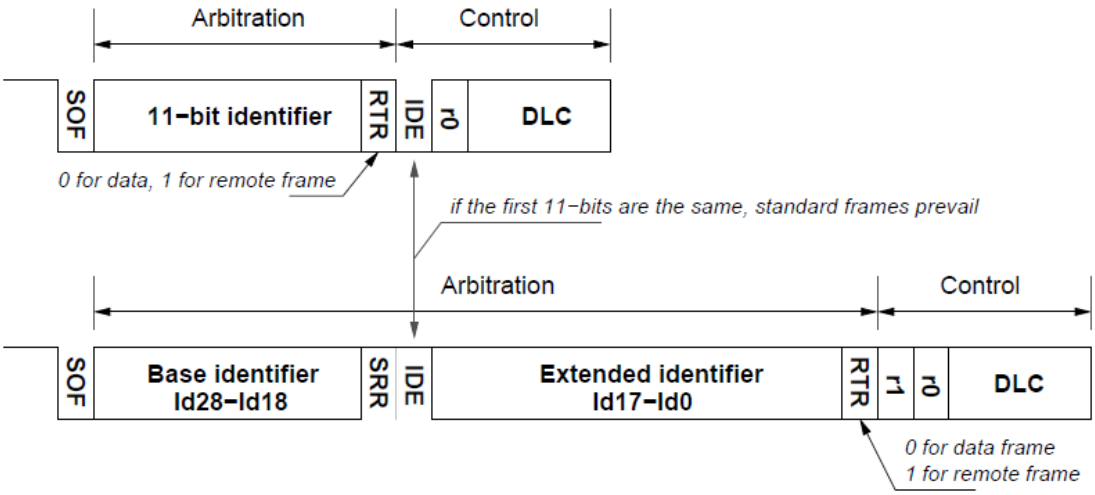


Figura 18. Campo Identifier estándar y extendido de una data frame CAN.[6]

Campo Control.

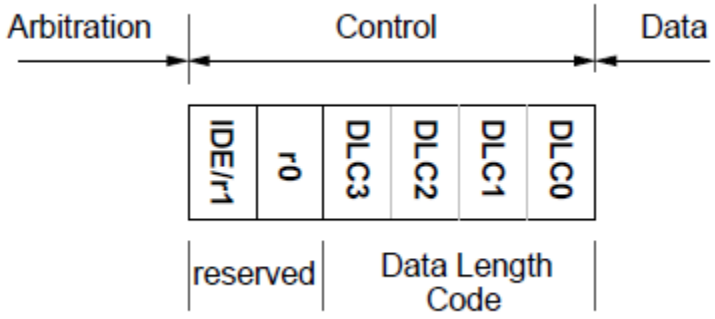


Figura 19. Campo Control de una data frame CAN.[6]

El campo de Control cuenta con seis bits como se representa en la figura 19. En las tramas estándar el primer bit es el IDE seguido de un bit reservado, para las tramas extendidas los

6. "Understanding and Using the Controller Area Network Communication Protocol", Di Natale, M., Zeng, H., Giusto, P., Ghosal, A., Springer-Verlag New York.

dos primeros bits son reservados. El estándar especifica que estos bits reservados deben ser enviados con un valor recesivo, sin embargo, los receptores pueden aceptar cualquier valor.

Los siguientes cuatro bits corresponden el valor de longitud del contenido de datos (Data lenght Content o DLC) en bytes. El valor dominante es interpretado como un 1 lógico, y el valor recesivo como 0.

Campo CRC y Acknowledge.

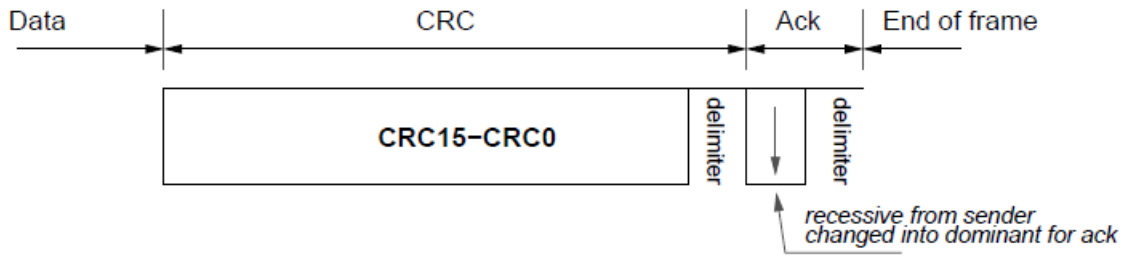


Figura 20. Campo CRC y ACK de una data frame CAN. [6]

EL capo de CRC contiene el código con el cual se hace la comprobación de la integridad del mensaje.

El campo de ACK está compuesto por dos bits. Un bit tiene la función de guardar los ACK de los receptores (ACK SLOT), el otro es solo un delimitador con valor recesivo. El ACK SLOT es cambiado de un valor recesivo a dominante por el nodo receptor para poder informar al transmisor que se ha recibido el mensaje.

Espacio Interframe.

Las tramas de Datos son separadas por el espacio Interframe con 7 bits de valor recesivo en el bus. Este valor es revisado por cada nodo para poder analizar correctamente cada trama.

ii. Remote Frame.

La Remote Frame es usada para solicitar la transmisión de un mensaje desde un nodo remoto. La Remote Frame tiene el mismo formato que la Data Frame con las siguientes características:

- El campo Identifier es usado para indicar el idetifier del mensaje solicitado.
- El campo Data siempre se encuentra vacío (0 bytes).

6. "Understanding and Using the Controller Area Network Communication Protocol", Di Natale, M., Zeng, H., Giusto, P., Ghosal, A., Springer-Verlag New York.

- El campo DLC indica la longitud de mensaje solicitado, no del transmitido.
- El bit RTR en el campo de Arbitration, siempre tiene valor recesivo.

iii. Error Frame.

La Error Frame no es en realidad una trama, sino el resultado de un error de señalización y recuperación.

iv. Overload Frame.

La Overload Frame, así como la Error Frame son de forma fija y no esta codificado por el método de completar de bits.

d. Administración de Errores.

El protocolo CAN esta diseñado para proveer una transmisión de datos y mecanismos para la detección de errores, señalización y auto-diagnostico, incluyendo mediciones para la contención de fallas, previniendo así afectaciones a la red.

Estas mediciones se basan en la capacidad de cada nodo para detectar los mensajes de broadcast recibidas o enviadas y generar una señal de condición de error. Cada nodo que detecte algún error marcará a los mensajes corruptos, estos mensajes son interrumpidos y retransmitidos automáticamente.

Existen 5 diferentes tipos de errores:

- Bit error. Cada transmisor monitorea el valor de la señal transmitida. Si el valor del bit que se escucha en el Bus es diferente al transmitido, un error de Bit es generado.
- Stuff error. Cuando se detectan 6 bits con el mismo nivel de manera consecutiva dentro de un campo que es sujeto a ser completado, un error de Stuff es generado. Existe un mecanismo para detectar este error en el cual después de detectar cinco bits con el mismo nivel, un sexto es agregado con el nivel opuesto y el receptor es capaz de remover el bit adicional.
- CRC error. EL CRC calculado es diferente al que se encuentra en la trama, entonces se genera un error de CRC.
- Form error. Se genera este error cuando existe algún valor invalido para un bit de los campos definidos por el protocolo.
- Acknowledgment error. Detectado por el transmisor si un bit recesivo está presente en el ACK SLOT.

2. ESTÁNDARES DE CAPAS SUPERIORES.

Existen diferentes estándares y protocolos que definen las capas superiores como la capa de red o aplicación, cada uno de estos han sido desarrollados con un propósito en especifico. Todos ellos van sobre el estándar ISO 11898 o CAN Bus, que define las capas inferiores, entre estos se encuentran los estándares J1939, OBD2 o CANOpen.

CANOpen esta desarrollado para comunicar los elementos de robots dentro de una línea de ensamblaje, dispositivos médicos como máquinas de rayos X, máquinas de diálisis y resonancias o maquinaria dentro de las industria agrícola, transporte o marítima. Define un diccionario de objetos que contiene todos los parámetros que describen el comportamiento de un dispositivo en esta red, los cuales son estandarizados.

OBD2 es el estándar que define la estructura de mensajes, interpretación de datos y eventos que se pueden obtener de un automóvil tradicional. Existen algunas variaciones de este estándar que dependen del origen, fabricante y tipo de vehículo. Es obligatorio que los automóviles soporten comunicación de este tipo dentro de algunos países.

J1939 es un estándar similar al OBD2 definido por la sociedad de ingenieros automotrices (SAE), enfocado principalmente para vehículos de uso pesado (camiones, autobuses o tractores). A diferencia de OBD2 es necesario contar con un archivo para decodificar los datos obtenidos del vehículo y sea posible interpretarlos. Este archivo debe de tener una licencia por parte de SAE.

a. SAE J1939.

J1939 es un protocolo de capa superior basado en CAN. Provee una comunicación serial entre las ECUs en cualquier tipo de vehículo de uso pesado. Los mensajes intercambiados llevan los datos de la velocidad, torque, temperatura, entre otros valores.

Todo lo que hace CAN se basa en hacerlo con la máxima fiabilidad y rendimiento posible, no solo en la robustez eléctrica, sino también debido a los requisitos de alta velocidad para un sistema de comunicación en serie. CAN fue diseñado para estar lo más cerca posible de las aplicaciones en tiempo real. Este nivel de rendimiento no es necesario para J1939.

J1939 aprovecha las características de CAN como:

- Máxima fiabilidad.
- Excelente detección de errores y confinamiento de fallas.
- Arbitraje de bus libre de colisión.

i. Formato de Mensajes.

CAN admite identificadores en los mensajes de 11 y 29 bits. CAN también está diseñado de una manera en que el nodo emisor no tiene que ver que nodo recibe los datos. A su vez, un nodo receptor no le importa quién envió los datos.

Por el contrario, J1939 usa solo el identificador de 29 bits, lo usa para identificar la fuente y, en algunos casos, el destino de los datos en el bus.

El identificador CAN es dividido en un campo de prioridad, un Número de grupo de parámetros (PGN) para identificar el contenido del campo de datos y la dirección de origen. Una prioridad de mensaje "0" indica la prioridad más alta y una prioridad de mensaje de "7" indica la prioridad más baja. Por lo general, se asignan altas prioridades a los datos de tiempo crítico.

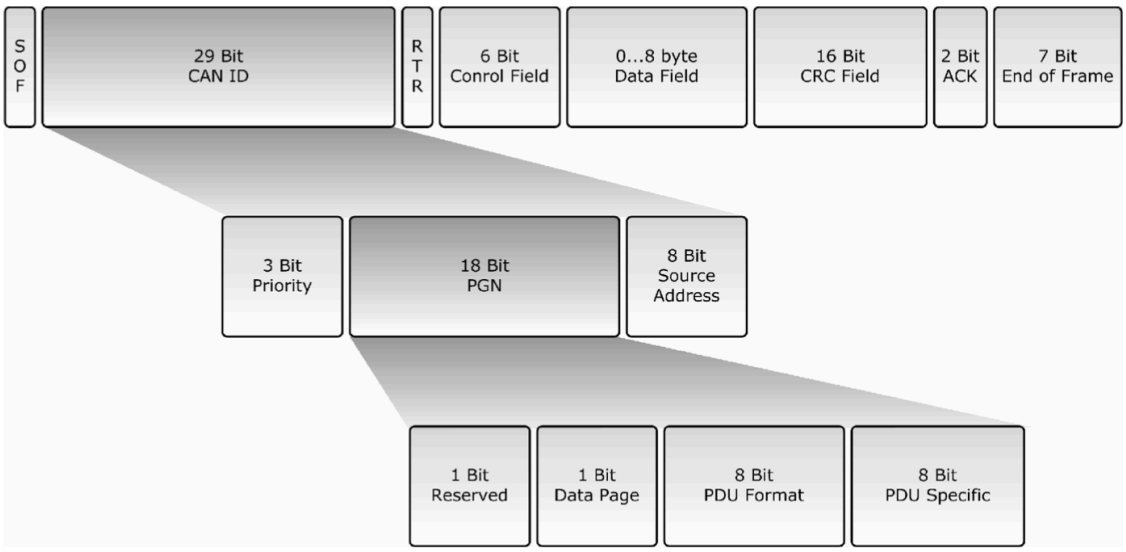


Figura 21. Formato de mensaje J1939. [7]

ii. J1939 estandares.

La colección de estandares del protocolo J1939 fue diseñado para seguir el modelo de referencia OSI. Cada capa corresponde a un documento y estadares especificos.

7. "A Comprehensible Guide to J1939"; Wilfried Voss; Copperhill Technologies Corporation; Greenfield, MA.

Capa modelo OSI	Estandar SAE J1939
7. Aplicación	J1939/7x
6. Presentación	J1939/6x
5. Sesión	J1939/5x
4. Transporte	J1939/4x
3. Red	J1939/3x
2. Enlace de datos	J1939/2x
1. Fisica	J1939/1x

Tabla 4. Estandares SAE J1939

b. OBD2

OBD es un sistema de diagnóstico a bordo del vehículo. Actualmente se encuentra en la mayoría de los vehículos actuales con algunas variantes. Comenzó como una normativa de la Comisión de Recursos del Aire de California para vehículos vendidos en California a partir de los modelos del año 1988, solo se monitorizaba parámetros relacionados a las emisiones como la sonda Lambda, el módulo EGR y el módulo de control ECM. Su evolución, el OBD2 implementada a partir de 1996, permite la detección de fallos eléctricos, químicos y mecánicos que pueden afectar el nivel de emisiones del vehículo.

Los DTC (Diagnostic Trouble Code) se introdujeron a esta nueva versión para poder tener un registro re los fallos y condiciones en los que ocurrió (FFD), si algún sistema o componente ocasiona que se supere el umbral de emisiones por lo que no opere dentro de las especificaciones, un DTC será almacenado y la lámpara de MIL (Malfuntion Indicator Light) se encenderá.

i. PIDs

Los PIDs (Parameter IDs) son los identificadores de los datos a monitorizar y eventos ocurridos en el vehículo. OBD2 soporta una amplia cantidad de PIDs estandarizados que pueden ser utilizados en una gran cantidad de vehículos. Esto permite obtener información en tiempo real del vehículo en forma leíble de los valores de operación del vehículo como son RPM, velocidad, temperatura, etc. También identifican y permiten determinar la causa de los DTC y FFD (Freeze Frame Data)

ii. ESTRUCTURA DE MENSAJES.

La estructura de los mensajes a través de OBD2 pueden llegar a ser de hasta 75 bits de longitud y está compuesto por un identificador y los datos. El campo de datos se puede dividir en el capo de longitud en #bytes, Modo de la solicitud, PID y los datos expresados en forma hexadecimal como se muestra en la siguiente figura:

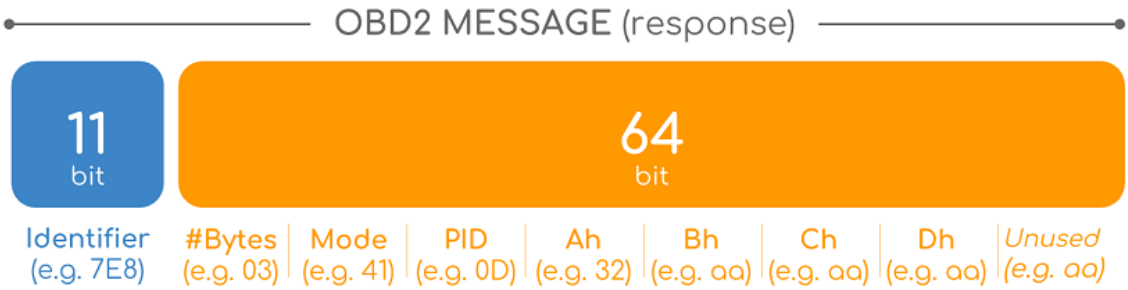


Figura 22. Estructura de mensaje OBD2. [8]

- Identificador: Para mensajes OBD2, el identificador es un estándar de 11-bits que es usado para distinguir los mensajes de "request" (ID 7DF) y los mensajes de "respond" (ID 7E8 al 7EF).

- Longitud: Este campo solo da la información de la longitud que resta del mensaje. Puede ir de 03 a 06.

- Modo: Existen 10 modos definidos en donde en cada uno brinda una información distinta, en los mensajes de "request" este campo puede ir de 01 a 0A mientras que para las respuestas el 0 es remplazado por un 4. El modo 1 muestra la información de los datos actuales en tiempo real como son la velocidad, RPM, etc. Los otros modos son usados para mostrar o limpiar los DTC y para mostrar la FFD.

- PID: Para cada modo existe una lista de PIDs estándar. Cada PID tiene una descripción y algunos especifican los valores máximos, mínimos y fórmulas de conversión.

- Ah, Bh, Ch Y Dh: Son los bytes de datos expresado en hexadecimal. Los datos deben ser convertido a decimal antes de aplicar las fórmulas de cálculo. El ultimo byte de datos no tiene uso.

iii. MODOS

- Modo 1. Muestra los parámetros disponibles y regresa los valores de sensores comunes, como lo son: velocidad del vehículo, RPM, Sensores de oxígeno, sensores de combustible, etcétera.

- Modo 2. Muestra los datos de los FFD. Cuando una falla ocurre y es detectada por la ECM, esta guarda los datos del sensor en el momento especifico que ocurre la falla.

- Modo 3. Muestra los DTC guardados. Estos códigos se pueden dividir en 4 categorías estándar:

8. OBD2 EXPLAINED - A SIMPLE INTRO (2019): https://www.csselectronics.com/screen/page/simple-intro-obd2-explained

- P0xxx: Fallas ligadas al Powertrain (motor o transmisión).
- C0xxx: Fallas ligadas al chasis.
- B0xxx: Fallas ligadas al cuerpo.
- U0xxx: Fallas en la red de comunicación.

Si el segundo digito es diferente de 0 (por ejemplo: P1xxx), este DTC no es estándar y solo es válido para fallas descritas por el fabricante.

- Modo 4. Funciona para limpiar los DTC y apaga la MIL. Si se borra un DTC antes de solucionarlo, la MIL se encenderá nuevamente durante el próximo ciclo de manejo.

- Modo 5. Muestra los resultados de los autodiagnósticos hechos por los sensores de oxigeno. Solo aplica para vehículos sin comunicación CAN por lo que se encuentra prácticamente en desuso y ha sido remplazado por el modo 6.

- Modo 6. Muestra los resultados de los autodiagnósticos del sistema. Solo aplica para vehículos con comunicación CAN.

- Modo 7. Muestra los DTC que no han sido confirmados. Los DTC detectados durante el ciclo de manejo actual aparecerán en este modo y son los mismos que el modo 3, esto lo hace útil para determinar si no vuelve a aparecer el problema una vez reparado.

- Modo 8. Muestra los resultados de las operaciones de control de los componentes a bordo.

- Modo 9. Muestra información específica del vehículo como el número de identificación del vehículo o los valores de calibración.

- Modo 10 o modo A. Muestra los códigos de falla permanentes, son los mismo que en el modo 3 y 7. Estos no pueden ser borrados con el modo 4, solo pueden ser borrados una vez que han sido solucionados y no se vuelve a presentar durante varios ciclos de manejo.

iv. LISTA DE PIDs ESTANDAR.

La lista de códigos de PIDs se encuentran en el Anexo A y los códigos DTC en el Anexo B.

CAPITULO III

1. DESARROLLO DE UN SISTEMA DE MONITOREO VEHICULAR.

Para poder desarrollar este proyecto se utilizó un dispositivo Raspberry Pi, en donde se ejecutó un script Python con ayuda de las librerías OBD con la cual se obtienen los datos del vehículo a través de un dispositivo ELM327 y AWSIoTPythonSDK.MQTTLib que es la librería estándar con la que se realiza la comunicación con AWS a través del protocolo MQTT.

a. ELM327.

El dispositivo ELM327 es una herramienta de diagnóstico vehicular compatible con OBD2-CAN en vehículos fabricados a partir de 1996. Esta interfaz permite la visualización de los valores de sensores y actuadores del vehículo en funcionamiento, leer y borrar códigos DTC (la MIL o también llamada luz de “Check Engine”).

Este dispositivo tiene las ventajas de ser mucho más económico comparado con otros dispositivos de diagnóstico, tiene una gran velocidad de lectura y puede ser utilizada en conjunto con otros dispositivos (smartphones, pc) ya que se puede comunicar a través de USB, Bluetooth o wifi.

Permite leer prácticamente todos los parámetros de los vehículos como puede ser la velocidad, revoluciones por minuto, sensores de voltaje, etc.

Figura 23. ELM327

b. Creación de un “thing” en AWS.

Dentro de AWS una vez creando una cuenta e ingresando al portal, nos encontraremos en la consola principal de AWS en donde se enlistan todos los servicios que proveen. Se debe seguir los siguientes pasos:

1. Dentro de la consola de AWS se selecciona el servicio llamado “IoT Core”.

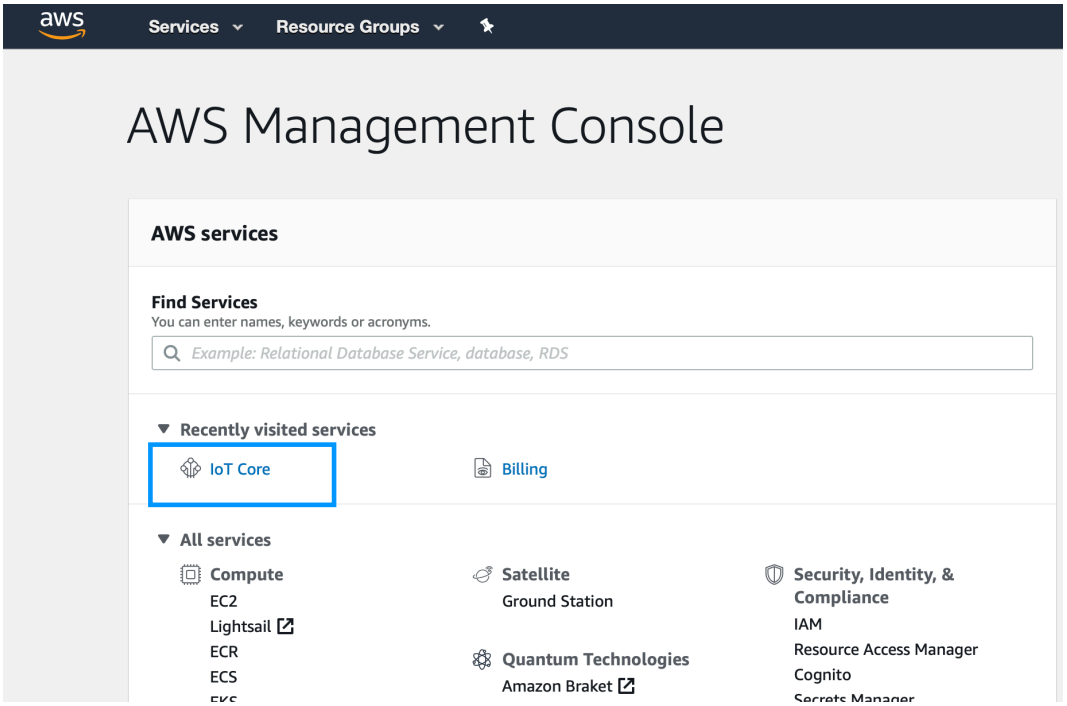


Figura 24. Consola AWS.

2. Dentro de IoT Core, en las opciones de la izquierda se selecciona la opción “Manage” y “Things”. En la parte superior izquierda se debe hacer click en “Create”.

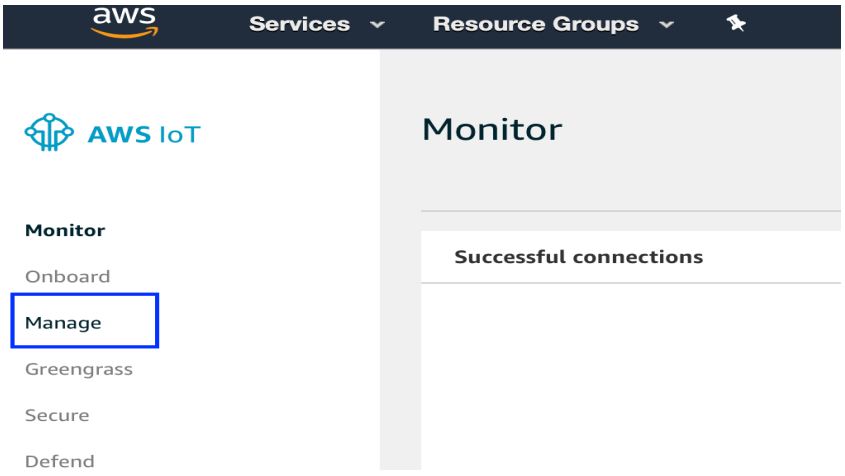


Figura 25. Manage “IoT Core”.

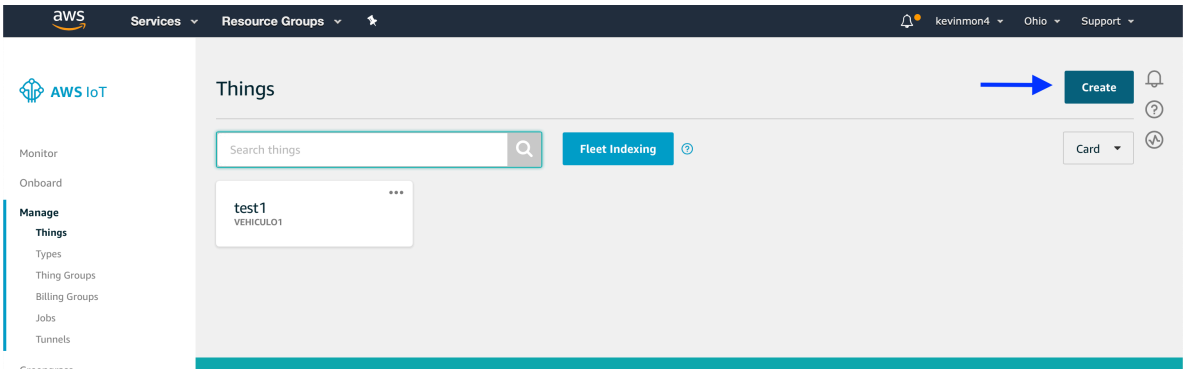


Figura 26. Create Thing.

3. Aparecerá el wizard para crear un nuevo “thing”, seleccionaremos la opción de crear solo uno o en caso de que se tenga un archivo con los datos de todos los objetos a crear se selecciona la otra opción. Se define el nombre, si pertenece a algún grupo de objetos o si tiene atributos específicos.

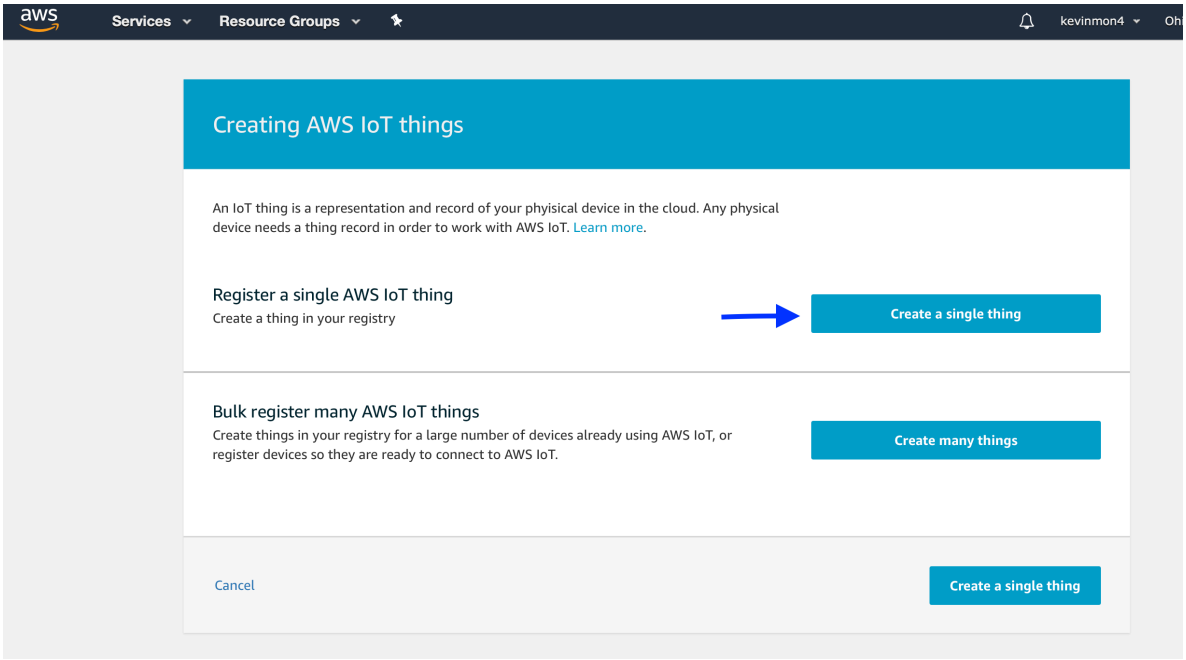


Figura 27. Single Thing.

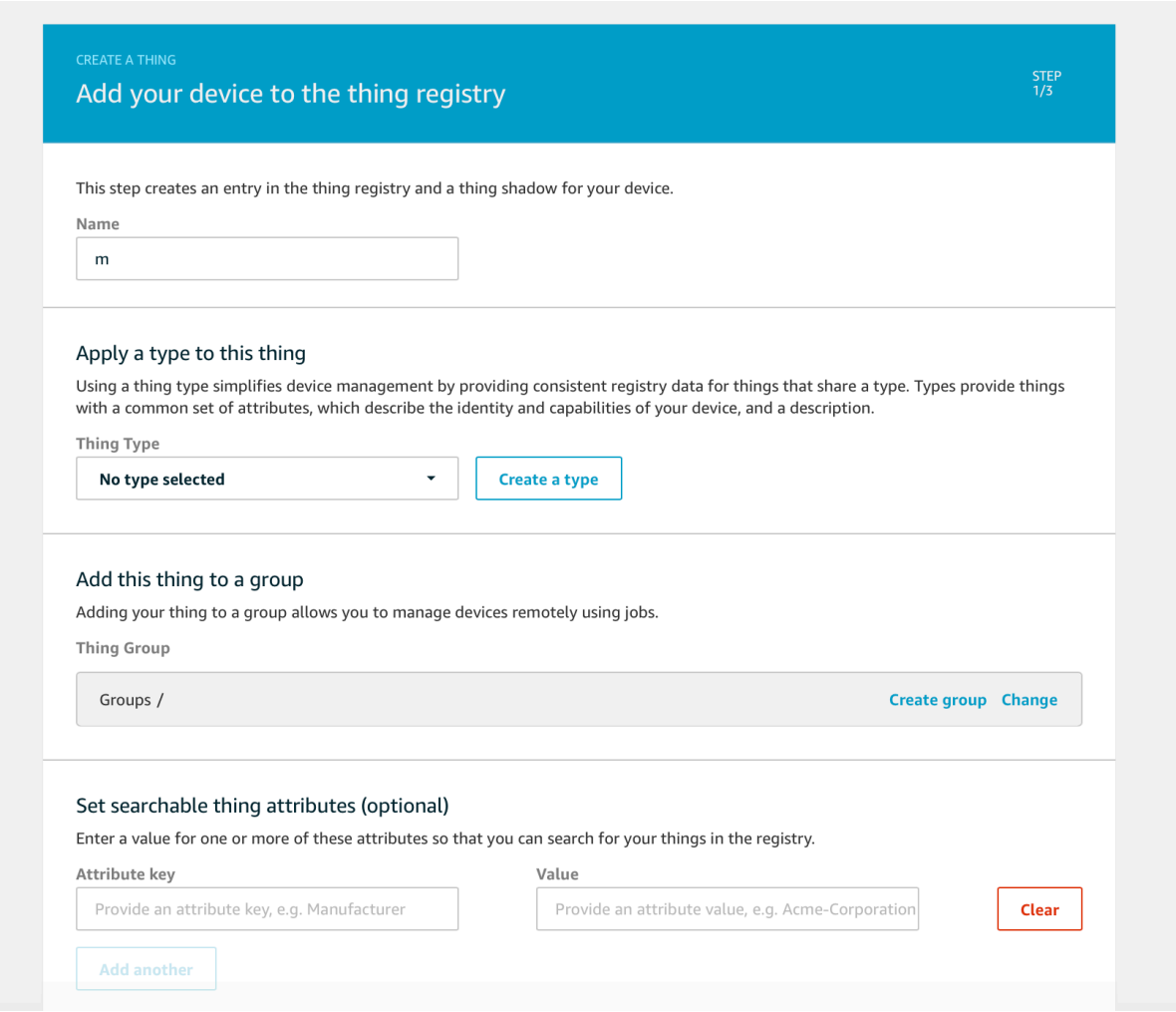


Figura 28. Registro Thing.

4. Se deberá asociar un certificado a el Thing creado para poder tener una comunicación segura. Por lo que se seleccionará la forma en la que se creará.

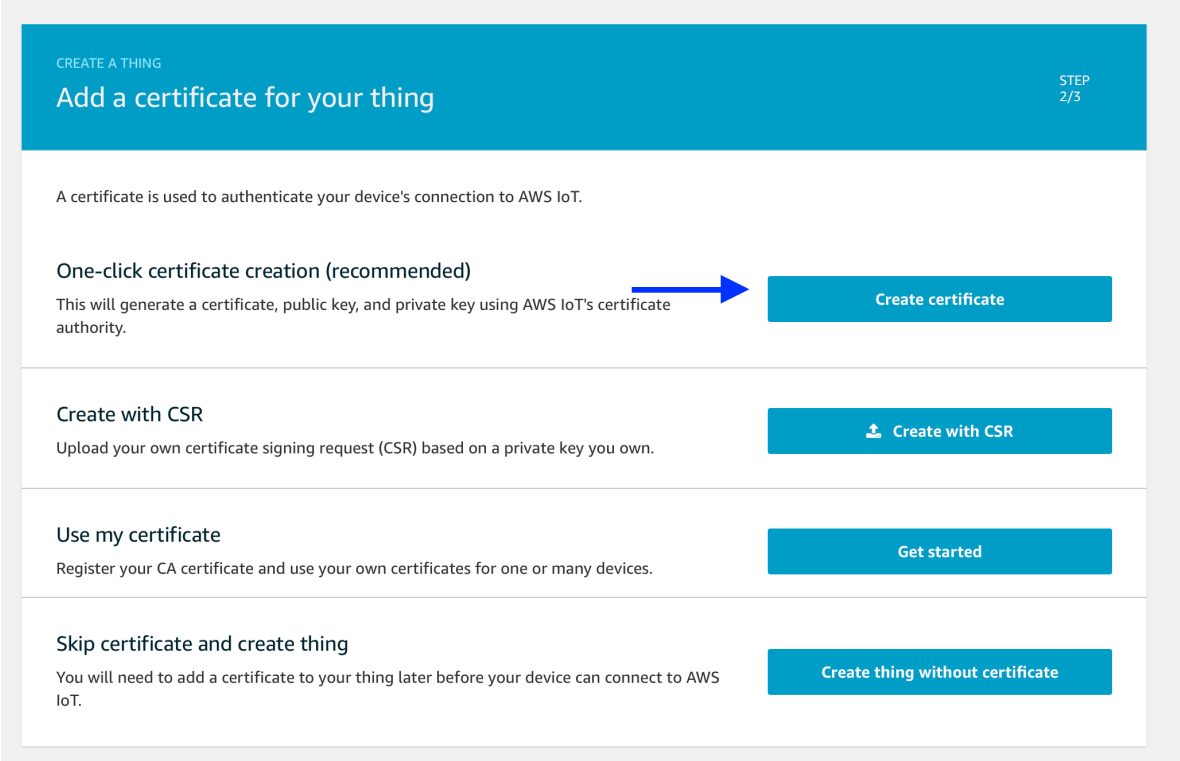


Figura 29. Certificado.

5. Finalmente se deberá asociar una política, la que le brindará los permisos necesarios para comunicarse a algún topic para publicar o suscribirse. Si no se agrega una política se observará un error al tratar de comunicarse. En caso de no haber ninguna crear una nueva en donde se especifique el topic, las acciones permitidas y los permisos.

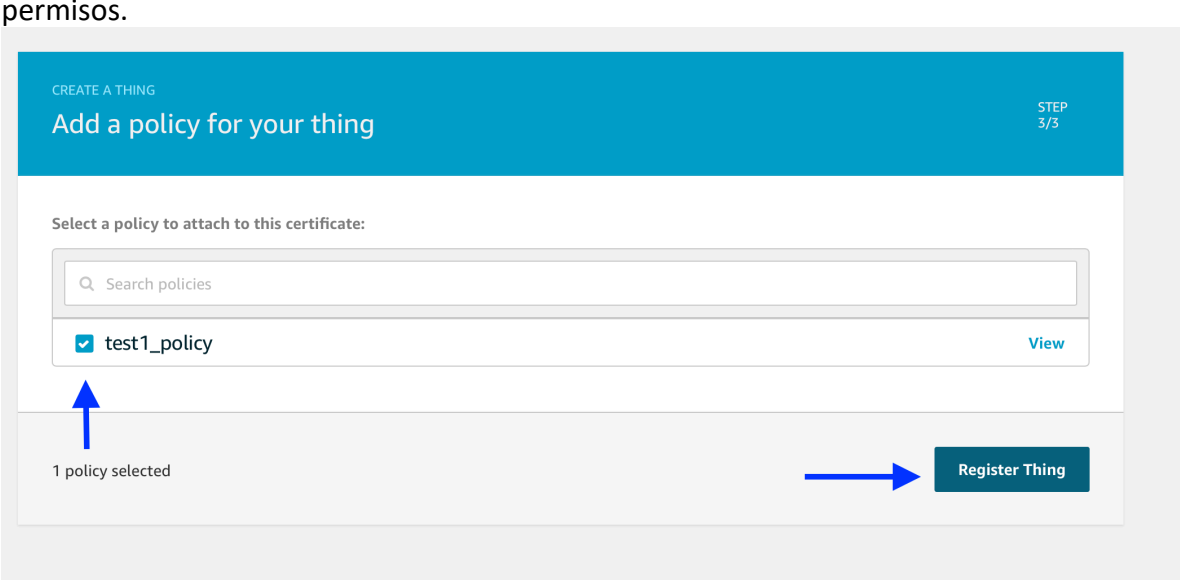


Figura 30. Asociar policy.

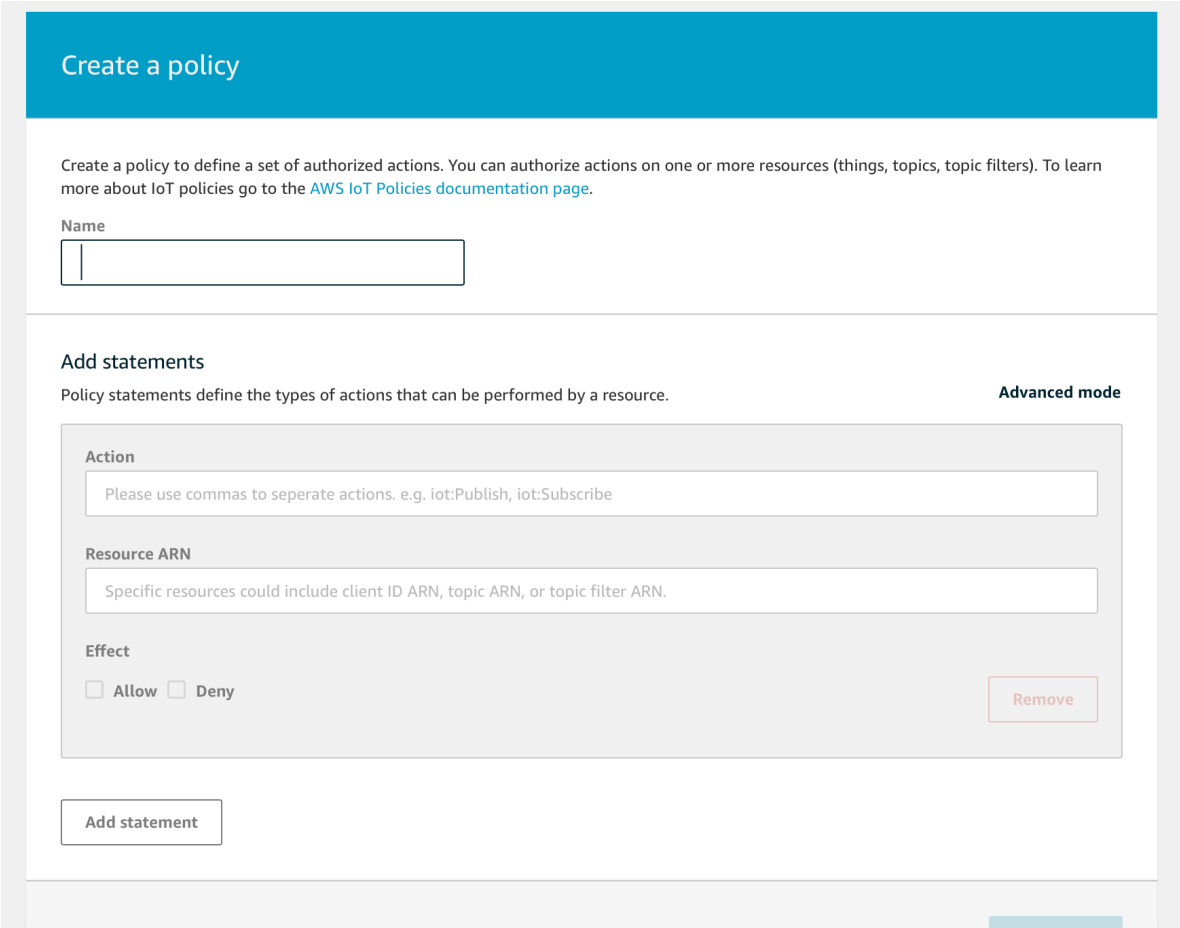


Figura 31. Crear policy.

Para poder validar que todo se a creado correctamente, podemos ir a “Test” y publicar o suscribirnos a un topic. Aquí observaremos los mensajes que se están recibiendo desde el dispositivo remoto.

Figura 32. Test IoT Core.

c. Conectividad a Internet.

Para los alcances de este trabajo de tesis se definieron dos métodos de conectividad a través de la red celular ambos cumpliendo con su función primordial de enviar los datos obtenidos del vehículo.

Se prefirió la red celular por sobre las demás ya que reduce los costos y tiene una amplia cobertura dentro del país, en cambio las tecnologías LPWAN aun no tienen suficiente cobertura en México y se tienen que utilizar dispositivos certificados por las diferentes tecnologías LPWAN lo que sube los costos.

Son métodos los siguientes:

i. Router Gateway.

Se instala del vehículo un dispositivo que funciona como Gateway y le brinda conectividad a Internet a la Raspberry Pi. Adicional a esto este dispositivo puede brindar servicios de Wifi y posicionamiento GPS.

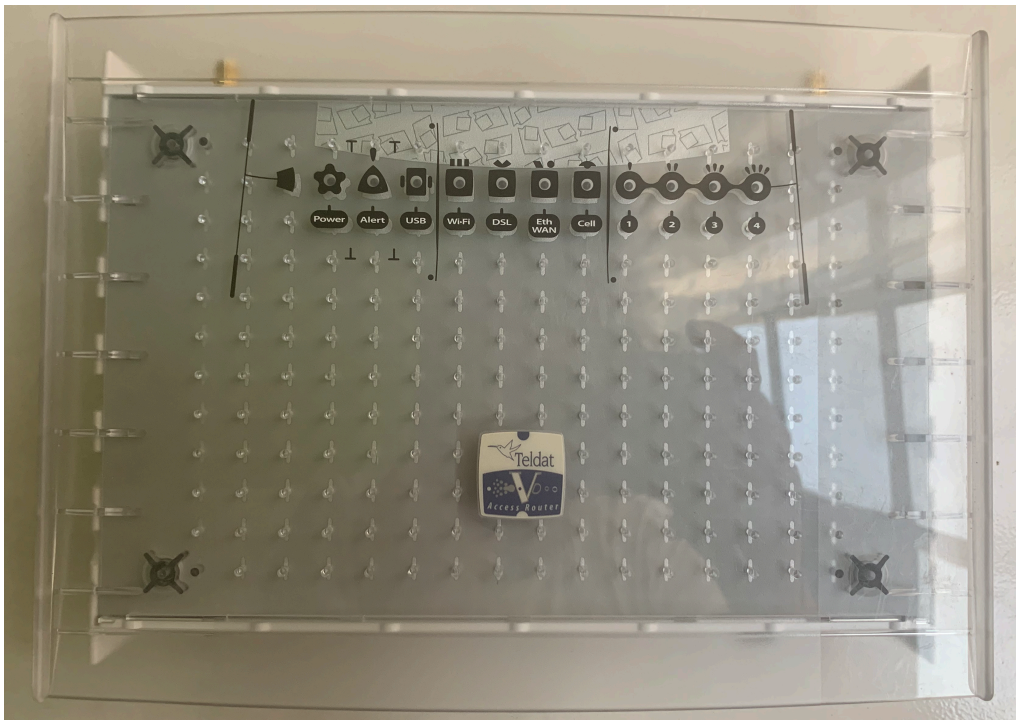


Figura 33. Router Teldat V vista superior.

Figura 34. Router Teldat V vista trasera.

Las conexiones son las siguientes:

Figura 35. Conexiones.

Figura 36. Router en el vehículo.

La Raspberry Pi, se conecta mediante su interfaz WLAN al SSID que brinda el Router y salir a Internet a traves de esta conexión. La configuración de la Raspberry Pi para que se conecte de forma automatica al SSID del vehiculo se encuentra en el Anexo C.

La configuración del Router para brindar conectividad a Internet y el servicio de WiFi se encuentra en el Anexo D.

ii. All-in-one.

Este método de conectividad, utiliza las funciones integradas de la Raspberry Pi para poder brindar conectividad a Internet a traves de un dungle celular y utilizar su interfaz WLAN para radiar un SSID y realizar finciones de Access Point.

Figura 37. Dongle LTE.

Para definir los parámetros de la conexión a la red celular se siguen los pasos del configurador para ingresar los datos del operador móvil que se este utilizando.

La configuración para propagar una SSID y brindar conectividad a Internet se encuentran en el Anexo E.

Figura 38. Conexiones con Dongle LTE.

d. Script para la obtención y envío de datos.

El script con el cual se obtienen los datos y se envían a la nube se encuentra en el Anexo F.

e. Visualización de datos.

Para la visualización de datos hay que crear algunos elementos dentro de AWS para poder realizar el análisis y poder observarlos. Se realizó la prueba con el servicio de Analytics en donde se deben seguir los siguientes pasos:

1. Dentro del servicio de “IoT Core” seleccionamos en la parte izquierda “Act”, se crea una regla para poder enviar los datos a un servicio de almacenamiento y análisis.

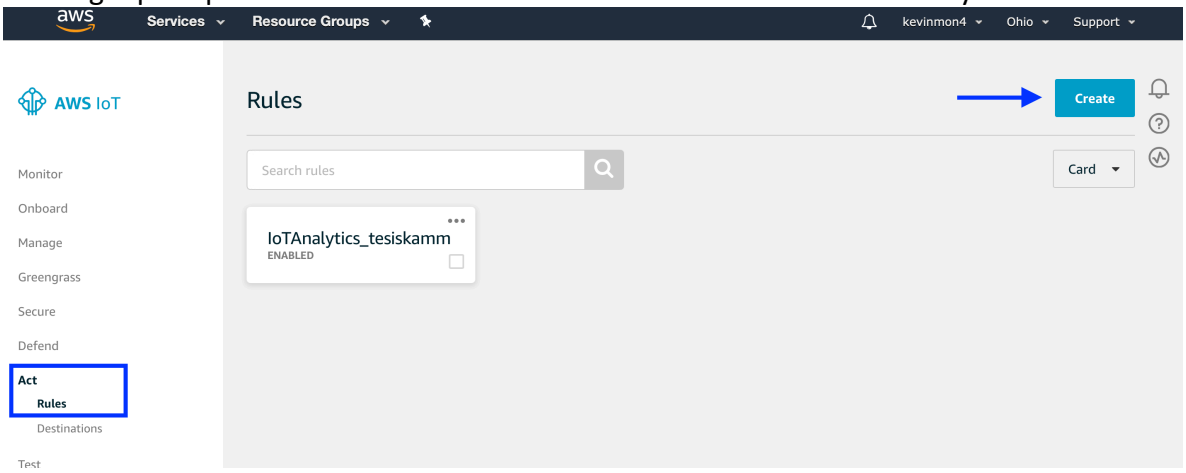


Figura 39. Act IoT Core.

2. Al crear la regla es necesario nombrarla y definir la selección, donde podemos colocar el rango de datos y los topic de donde se obtendrá.

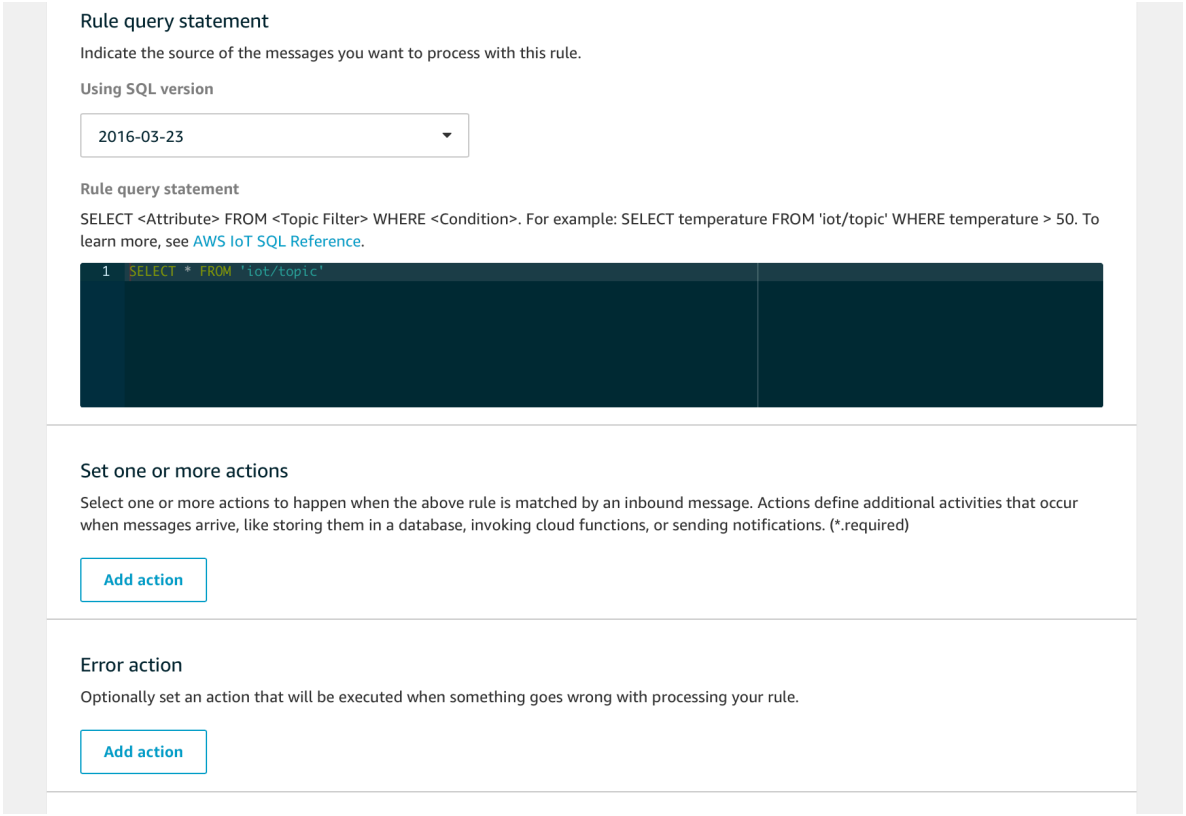


Figura 40. Create rule.

3. Posteriormente se selecciona una acción. Esta acción es a que servicio se enviaran los datos. Para esta ocasión se seleccionó “IoT Analytics” que permite filtrar y crear operaciones con los datos. Este servicio requiere que se creen elementos para poder realizar el procesamiento, el wizard permite crearlo en automático.

Figura 41. Action AWS.

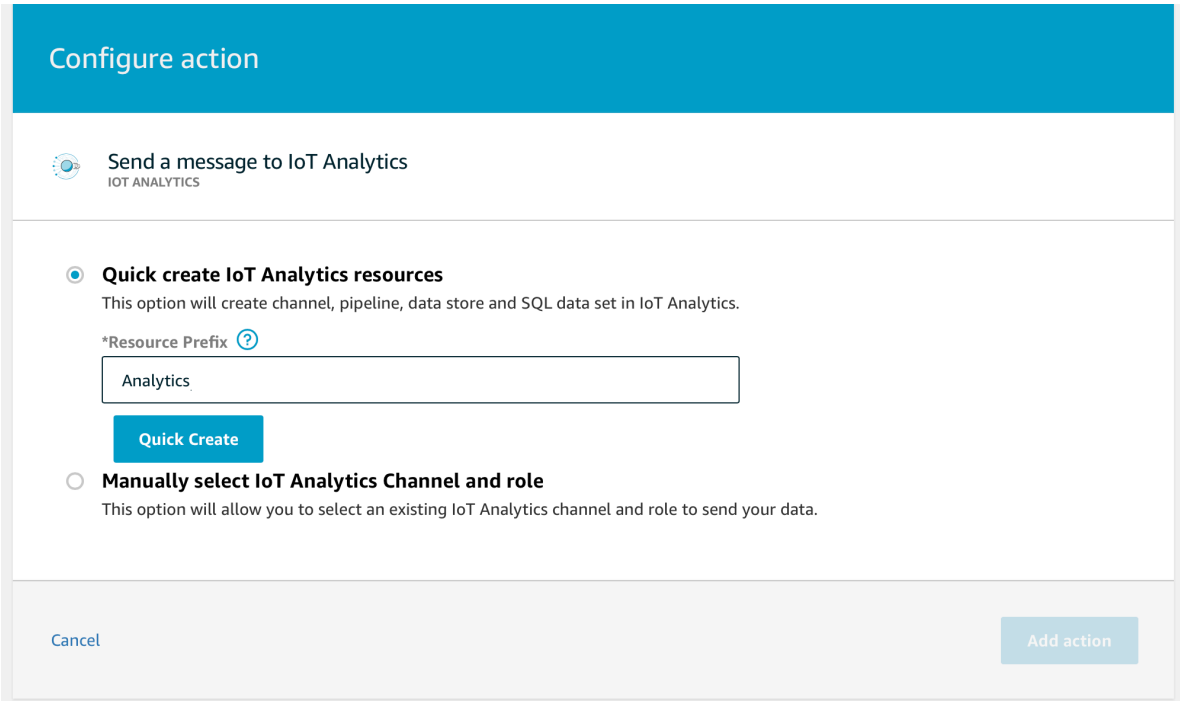


Figura 42. Crear recursos Analytics.

4. Dentro de estos elementos creados, es necesario modificar el llamado “Data Set”. Regresamos a la consola principal de AWS y seleccionamos el servicio “IoT Analytics”. Dentro del dataset en la pestaña de “Content” realizaremos la acción “Run Now” lo que nos permitirá traer los datos recibidos. Esta opción se puede automatizar dentro de “Details” donde se definirá la frecuencia de importación de datos.

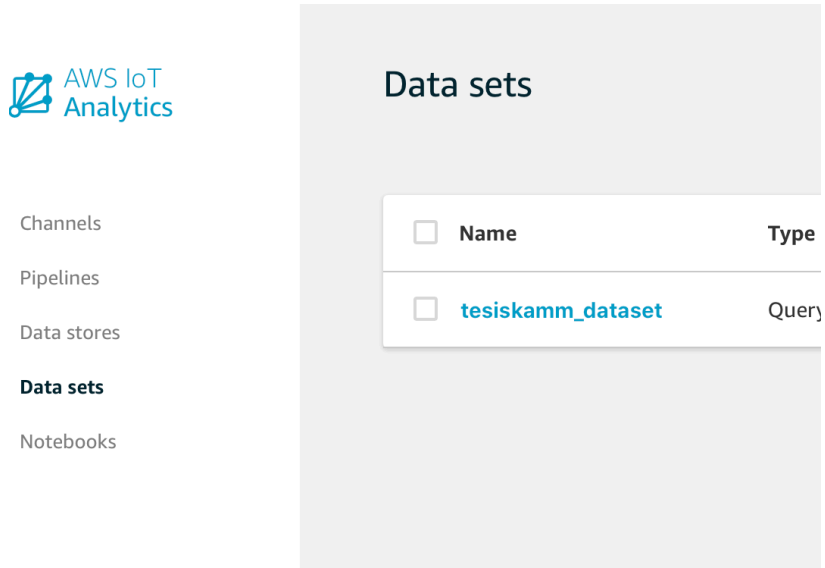


Figura 43. Data Set.

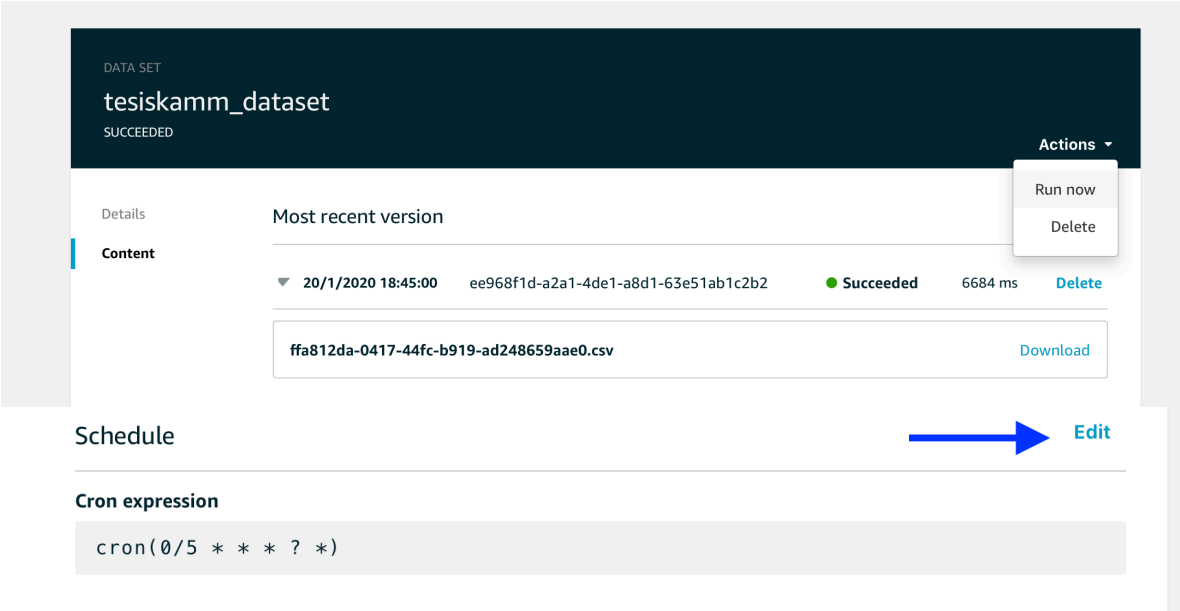


Figura 44. Schedule Data Set.

5. Para visualizar estos datos se utilizará el servicio llamado QuickSight. Dentro de la consola principal de AWS seleccionaremos QuickSight, este nos pide que nos autentiquemos. En la pantalla principal seleccionaremos "Manage Data".

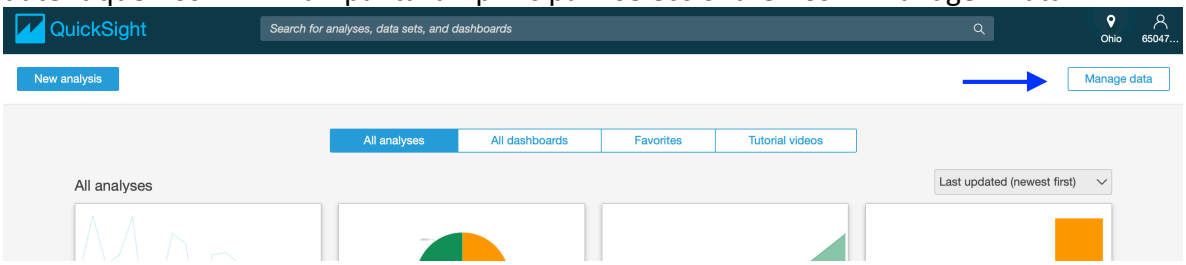


Figura 45. QuickSight data.

6. Se requiere dar click en el "New data set", lo que nos pedirá que definamos la fuente de los datos, para este ejemplo es IoT Analytics y nos mostrará los Data Set de los que podemos importar los datos.

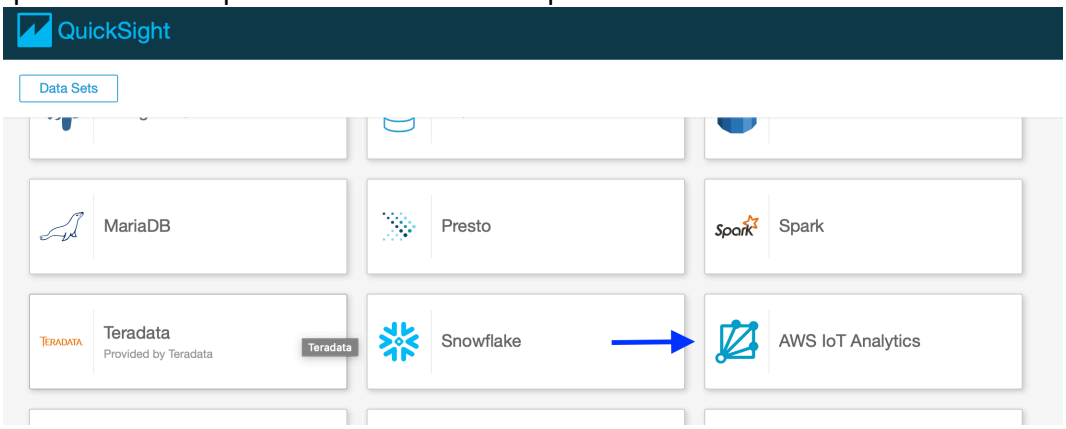


Figura 46. Seleccionar fuente de datos.

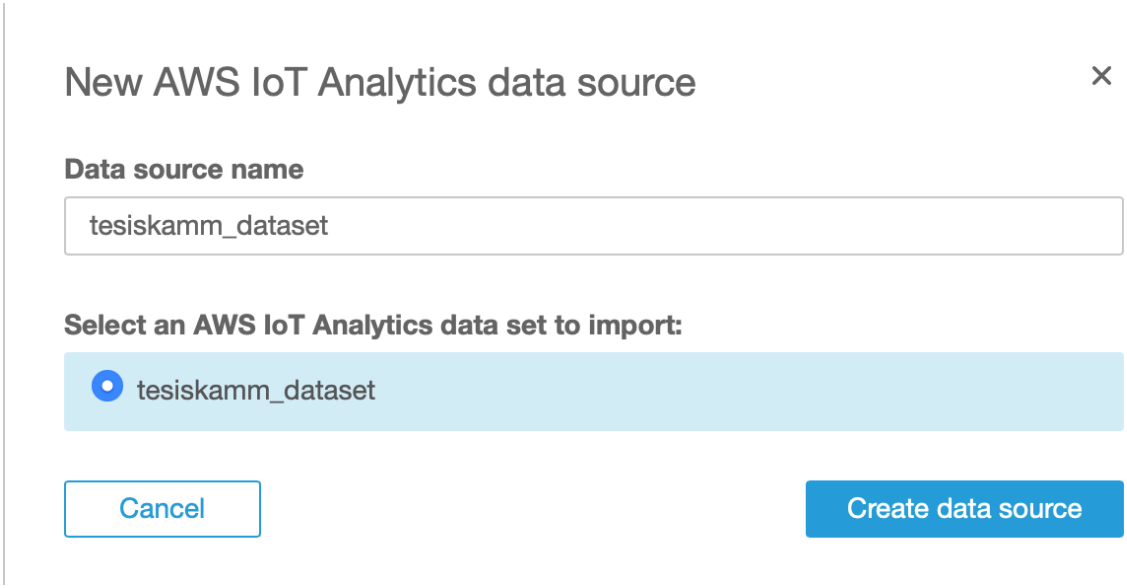


Figura 47. Seleccionar Data Set.

7. Finalmente regresando a la pantalla principal, crearemos un nuevo análisis y seleccionar la fuente de datos que creamos en un paso anterior. En este punto nos dejara crear el dashboard con diferentes tipos de grafica (barras, líneas, circulares, etc.) y modificar los datos que se corresponden a cada eje. Este se puede publicar, enviar periódicamente vía correo o compartir por correo.

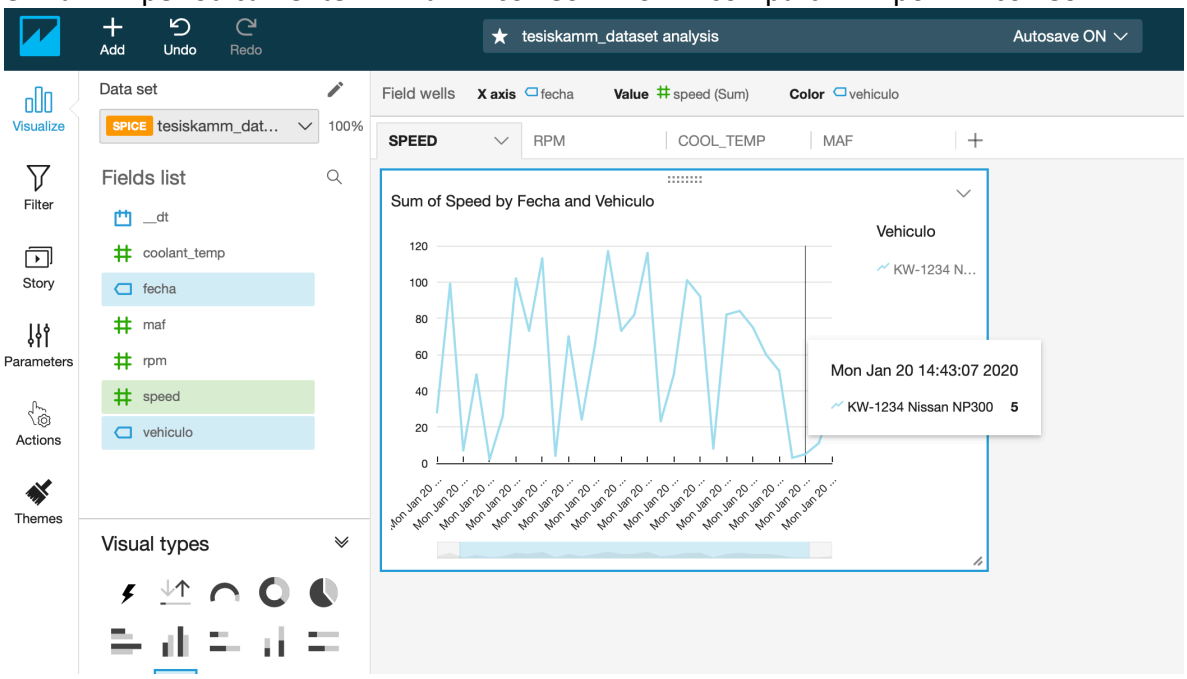


Figura 48. Creación de Analisis.

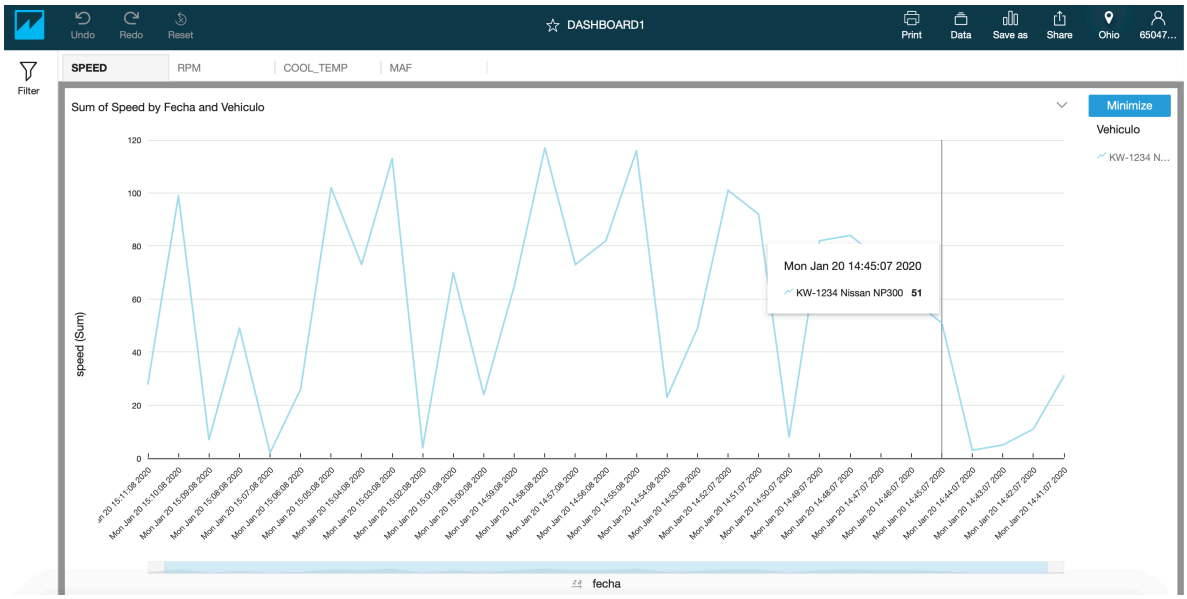


Figura 49. Dashboard AWS QuickSight.

CAPITULO IV CONCLUIONES.

1. ANÁLISIS DE RESULTADOS Y GENERACIÓN DE CONCLUSIONES.

Una vez realizado el proyecto vemos la facilidad para poder obtener datos de un vehículo para su posterior envío a un punto de gestión, solo es necesario que se tenga una conexión a Internet que puede ser provista a través de cualquier red de acceso (WWAN, WLAN, Cableada, etcétera). Una vez con los datos en un punto centralizado donde cualquier persona perteneciente a la institución propietaria de la solución pueda acceder y hacer uso de ellos, para realizar un análisis exhaustivo o solo tener visibilidad de ellos.

Estos datos se pueden enviar a alguno de los servicios descritos en este trabajo de tesis, como los servicios en la nube de cualquier proveedor que convenga o en dado caso solo implementar un servidor que pueda recibir mensajes MQTT o API Rest. En estos servicios no solo se permite visualizar los datos, también que brinden una información útil y sencilla de leer, con la capacidad de exportarlo y crear reportes personalizado, dependiendo de el uso que se quiera dar.

Esta solución tiene algunas comparativas reales con las soluciones que actualmente se encuentran en el mercado, ya que generalmente no integran todas las características que se trabajaron durante la elaboración de esta tesis. Algunos de los más famosos son:

- Fleetrackers

Cuentan con un producto que tiene la capacidad de leer parámetros del vehículo al conectarse a la interface OBDII, representando la información del comportamiento del vehículo en forma gráfica y su ubicación geográfica por medio de GPS.

- Trackimo.

Cuenta con diversas soluciones de rastreo a través de GPS-GMS, Wifi o Bluetooth. Provee alertas de movimiento, velocidad, geolocalización y un histórico de 5 años. No lee datos del vehículo.

- AWAREGPS

Cuenta con una solución de rastreo vehicular a través de GPS+GSM, además de un sistema de monitoreo en donde se muestra el seguimiento, reportes y seguimiento de ruta. Adicionalmente cuenta con alertas de mantenimiento basado en la distancia recorrida por los vehículos. No lee datos del vehículo.

- Escáner automotriz especializado.

Este tipo de escáneres cuentas con muchas más funciones que el ELM327, con lo cual se pueden obtener valores de los vehículos específicos y poder manipular valores no estándares. Estos son de uso local, que dependiendo del valor y las características del dispositivo cuentan con capacidades para poder exportar las lecturas.

Tomando en cuenta los costos básicos en dólares por la implementación de cada vehículo, la solución descrita en este trabajo de tesis fue más económica que la mayoría de los productos en el mercado. Para tener una vista más acercada a la realidad, se debe analizar los costos de fabricar hardware especializado (en lugar de utilizar una Raspberry Pi con periféricos conectados), una infraestructura central en caso de no utilizar servicios en la nube, la operación y ganancias. La comparativa de los costos aproximado se describen en la siguiente tabla:

Solucion	Costo Inicial	Costo Recurrente (mensual)
Fleetrackers (solo GPS)	$160	$18
Trackimo	$280	$5
Awaregps	$40	$19.50
Escáner	$100 – $1000	N/A.
Solución Tesis AWS *	$76	$13
Solucion Tesis Local *	$76	$9

Tabla 5. Comparativa de mercado.

2. Trabajo Futuro.

Existe la posibilidad de ampliar lo desarrollado en este trabajo de tesis como puede ser los siguientes:

- Implementación con API Rest.
- Desarrollo de un análisis amplio que permita la descripción y predicción de comportamientos a través de los servicios de Machine Learning que tienen los proveedores de servicios en la nube.
- Implementación privada mediante con premisas propiedad del usuario, a través de redes privadas VPNs y un servidor centralizado que permita la recepción de mensajes y su visualización.

Bibliografía y referencias web.

Capítulo I:

- An Introducction to LTE, LTE Advance, SAE, VoLTE and 4G Mobile communications; Christopher Cox, Wiley, UK.
- https://www.ietf.org/topics/iot/
- https://www.machnation.com/2017/09/18/functional-architecture-iot-platforms/?utm_source=iscoop
- https://www.i-scoop.eu/internet-of-things-guide/iot-technology-stack-devices-gateways-platforms/
- https://forum.huawei.com/enterprise/es/de-principiante-a-experto-fundamentos-de-wlan-sección-6-conceptos-básicos-en-wlan/thread/485665-100239
- https://www.sigfox.com/en
- https://lora-alliance.org
- https://www.bluetooth.com/blog/an-intro-to-bluetooth-mesh-part1/
- https://aws.amazon.com/training/
- https://azure.microsoft.com/es-mx/overview/what-is-cloud-computing/
- https://cloud.google.com/why-google-cloud
- https://www.ibm.com/developerworks/ssa/library/iot-mqtt-why-good-for-iot/index.html

Capítulo II:

- Introducción a CAN bus: Descripción, ejemplos y aplicaciones de tiempo real. Proyecto fin de máster, Universidad Politécnica De Madrid.
- Understanding and Using the Controller Area Network Communication Protocol, Di Natale, M., Zeng, H., Giusto, P., Ghosal, A., Springer-Verlag New York.
- "A Comprehensible Guide to J1939"; Wilfried Voss; Copperhill Technologies Corporation; Greenfield, MA.
- "A Comprehensible Guide to Controller Area Network", Wilfried Voss, Copperhill Technologies Corporation, 2005
- https://www.csselectronics.com/screen/page/simple-intro-to-can-bus
- "Controller Area Network Physical Layer Requirements", Texas Instruments. http://www.ti.com/lit/an/slla270/slla270.pdf
- "A CAN Physical Layer Discussion", Microchip. http://ww1.microchip.com/downloads/en/appnotes/00228a.pdf
- https://www.csselectronics.com/screen/page/simple-intro-obd2-explained

Capítulo III:

- http://profesores.elo.utfsm.cl/~agv/elo322/1s17/projects/reports/ControlAreaNetwork_CAN.pdf
- http://www.simmasoftware.com/j1939-presentation.pdf

- https://imt.mx/archivos/Publicaciones/PublicacionTecnica/pt474.pdf
- http://www.inp.nsk.su/~kozak/canbus/candll.pdf
- http://www.inp.nsk.su/~kozak/canbus/canphy.pdf
- https://copperhilltech.com/blog/sae-j1939-vs-can-bus-physical-layer-and-higher-layer-protocol-hlp/
- http://ww1.microchip.com/downloads/en/appnotes/00228a.pdf
- https://www.sis.se/api/document/preview/919965/

Capítulo IV:

- https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md
- https://www.outilsobdfacile.com/obd-mode-pid.php
- http://obdcon.sourceforge.net/2010/06/obd-ii-pids/
- https://python-obd.readthedocs.io/en/latest/
- https://www.teldat.com/es/
- https://www.cyd.conacyt.gob.mx/?p=articulo&id=315

Anexos.

a. Anexo A. Listado PIDs.

El listado de los PIDs de OBD2 se pueden encontrar en las siguientes ligas de referencia:

- https://python-obd.readthedocs.io/en/latest/Command%20Tables/
- http://obdcon.sourceforge.net/2010/06/obd-ii-pids/

b. Anexo B. Standar Fault Codes (DTC)

El listado de DTC se pueden encontrar en las siguientes ligas de referencia:

- https://www.obdadvisor.com/most-popular-obd2-codes/
- https://www.obd-codes.com/trouble_codes/

c. Anexo C. Configuración Raspberry Pi Wifi.

Se ejecutan el siguiente comando en la consola:

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Nos arrojará un archivo con la siguiente información (puede variar con la versión de Sistema Operativo).

```
country=GB
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
```

Se agregan los datos de la red a la cual se conectará, basta con el nombre del SSID y el password de la red:

```
country=GB
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
        ssid="Nombre_de_la_red"
        psk="Clave"
}
```

Se guarda y cierra el archivo para finalmente reiniciar la Raspberry Pi con el siguiente comando:

```
sudo reboot
```

d. Anexo D. Configuración Router Gateway.

Se utiliza un Router Teldat, la configuración es la siguiente:

```
   log-command-errors
   no configuration
   set hostname Gateway
;
   add device bvi 0
   add device direct-ip 1
   set data-link at cellular0/0
   set data-link at cellular0/1
   set data-link at cellular1/0
   set data-link nic cellular1/1
   global-profiles dial
; -- Dial Profiles Configuration --
      profile LTE default
      profile LTE dialout
      profile LTE 3gpp-apn <APN>
      profile LTE 3gpp-pdp-type ipv4v6
;
   exit
```

```
;
;
;
   network wlan<x/y>
; -- Wireless LAN Interface. Configuration --
;
      bss <SSID_name>
         privacy-invoked
         rsn wpa2
         cipher aes-ccmp
         akm-suite psk
         wpa-psk passphrase plain <password>
      exit
;
   exit
;
;
   network bvi0
; -- Bridge Virtual Interface configuration --
      ip address 192.168.1.1 255.255.255.0
;
   exit
;
;
   network direct-ip1
; -- Generic Direct IP Encapsulation User Configuration --
      ip address dhcp-negotiated
;
      base-interface
; -- Base Interface Configuration --
         base-interface cellular1/1 link
         base-interface cellular1/1 profile LTE
;
      exit
;
      direct-ip
; -- Direct IP encapsulator user configuration --
         address dhcp
      exit
;
   exit
;
;
;
;
   protocol asrt
; -- ASRT Bridge user configuration --
```

```
      bridge
      irb
      port ethernet0/0 1
      port wlan<x/y> 2
      no stp
      route-protocol ip
   exit
;
;
;
;
   protocol ip
; -- Internet protocol user configuration --
      route 0.0.0.0 0.0.0.0 direct-ip1
;
      rule 1 local-ip direct-ip1 remote-ip any
      rule 1 napt translation
      rule 1 napt tcp-adjust-mss mss_clamping
;
   exit
;
   protocol dhcp
; -- DHCP Configuration --
      server
; -- DHCP Server Configuration --
         enable
;
;
         subnet RED 0 network 192.168.1.0 255.255.255.0
         subnet RED 0 range 192.168.1.2 192.168.1.254
         subnet RED 0 dns-server 192.168.1.1
         subnet RED 0 router 192.168.1.1
;
      exit
;
   exit
;
;
   dump-command-errors
   end
```

e. Anexo E. Configuración Raspberry Pi como Router y AP.

Para que la Raspberry Pi funcione como Router y AP hay que instalar los paquetes de software DNSMasq y HostAPD, para ello se ejecutan los siguientes comandos:

```
sudo apt install dnsmasq hostapd
```

Se debe detener los siguientes servicios para poder modificar los archivos de configuración:

```
sudo systemctl stop dnsmasq
sudo systemctl stop hostapd
```

Servidor DHCP.

Uno de los primeros pasos es configurar la IP fija de la Raspberry Pi y habilitar el servidor DHCP. Modificamos el siguiente archivo:

```
sudo nano /etc/dhcpcd.conf
```

Al final se agrega lo siguiente:

```
interface wlan0
    static ip_address=192.168.4.1/24
    nohook wpa_supplicant
```

Se guarda el archivo donde se define la IP y se reinicia el servicio asociado:

```
sudo service dhcpcd restart
```

Posteriormente se crea el servidor DCHP. Se abre el archivo de configuración y se modifica con las siguientes líneas:

```
sudo nano /etc/dnsmasq.conf

     interface=wlan0
     dhcp-range=192.168.4.2,192.168.4.20,255.255.255.0,24h
```

Después de guardar el archivo se inicia dnsmasq:

```
sudo systemctl start dnsmasq
```

Access Point Host.

Creamos un nuevo archivo de configuración con el siguiente comando y se agregan las líneas siguientes (aquí se indica el nombre del SSID y el password de la red):

```
sudo nano /etc/hostapd/hostapd.conf

          interface=wlan0
          driver=nl80211
```

```
ssid=RED_RASPBERRY
hw_mode=g
channel=7
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=Raspberrytesis
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

Se le debe indicar al sistema donde encontrar este nuevo archivo de configuración modificando el siguiente archivo:

```
sudo nano /etc/default/hostapd
```

Buscar la línea que diga #DAEMON_CONF y remplazarla por la siguiente:

```
DAEMON_CONF="/etc/hostapd/hostapd.conf"
```

Finalmente habilitamos e iniciamos hostapd:

```
sudo systemctl unmask hostapd
sudo systemctl enable hostapd
sudo systemctl start hostapd
```

NAT.

Para habilitar la funcionalidad de Nat en la Raspberry Pi, se debe modificar este archivo y eliminar el comentario (símbolo # al inicio de la línea) de la siguiente línea:

```
Sudo nano /etc/sysctl.conf

     net.ipv4.ip_forward=1
```

Agregar y guardar la siguiente línea para permitir el trafico de salida por una interfaz determinada:

```
sudo iptables -t nat -A  POSTROUTING -o wwan0 -j MASQUERADE
sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

Se modifica el archivo /etc/rc.local para que el dispositivo inicie con estas reglas agregando justo arriba de “exit 0” lo siguiente:

```
iptables-restore < /etc/iptables.ipv4.nat
```

Finalmente se reinicia la Raspberry Pi y se verifica que siga funcionando correctamente. Algunos comandos que pueden ayudar a verificarlo son los siguientes:

```
sudo systemctl status hostapd
sudo systemctl status dnsmasq
```

f. Anexo C. Python Script para el envío de datos.

```
import obd
from AWSIoTPythonSDK.MQTTLib import AWSIoTMQTTClient
import time
import json

host = "a3hjbzslooloc8-ats.iot.us-east-2.amazonaws.com"
caPath = "/home/pi/aws_certs/ca.crt"
cert = "/home/pi/aws_certs/1849a92a69-certificate.pem.crt"
key = "/home/pi/aws_certs/1849a92a69-private.pem.key"

connection_obd = obd.OBD()
r = connection_obd.status()
print(r)


myMQTTClient = AWSIoTMQTTClient("test1")
myMQTTClient.configureEndpoint("a3hjbzslooloc8-ats.iot.us-
east-2.amazonaws.com", 8883)
myMQTTClient.configureCredentials(caPath, key, cert)
myMQTTClient.configureOfflinePublishQueueing(-1)   # Infinite
offline Publish queueing
myMQTTClient.configureDrainingFrequency(2)
myMQTTClient.configureConnectDisconnectTimeout(10)
myMQTTClient.configureMQTTOperationTimeout(5)
#connect and publish
myMQTTClient.connect()
myMQTTClient.publish("test/monitoreo", "connected", 0)


while 1:
    fecha = time.strftime("%c")

    r = connection.query(obd.commands.RPM)
    d_rpm = r.value
```

```
    r = connection.query(obd.commands.SPEED)
    d_speed = r.value

    r = connection.query(obd.commands.COOLANT_TEMP)
    d_cool = r.value

    r = connection.query(obd.commands.MAF)
    d_maf = r.value

    datos = {'Fecha': fecha, 'RPM': d_rpm, 'SPEED': d_speed,
'COOLANT_TEMP': d_cool, 'MAF': d_maf}
    payload = json.dumps(datos)
    print payload
    myMQTTClient.publish("test/monitoreo", payload, 0)
    time.sleep(10)
    print (".")

connection_obd.close()
```