ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

Sistema integrador de información en tiempo real, para la gestión y control de combustible, empleando tecnología IoT implementada a controladores industriales, en la estación de servicio "Eco el Oasis".

Ayala Santamaría, David Sebastián y Ruiz Borja, Mauricio Fernando

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Sistemas e Informática

Trabajo de Titulación, previo a la obtención del título de Ingeniero en Sistemas e Informática

Ing. Delgado Rodríguez, Ramiro Nanac

22 de junio de 2022

Resultados de la herramienta para verificación y/o análisis de similitud de contenidos

TITULACION AYALA_RUIZ_VFINAL.docx

Scanned on: 17:47 May 25, 2022 UTC

Identical Words	243
Words with Minor Changes	12
Paraphrased Words	277
Omitted Words	0

DELGADO RODRIGUEZ, RAMIRO NANAC

CC: 1707019178

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Sistemas e Informática

Certificación

Certifico que el trabajo de titulación: "**Sistema integrador de información en tiempo real, para la gestión y control de combustible, empleando tecnología IoT implementada a controladores industriales, en la estación de servicio "Eco el Oasis"** fue realizado por los señores **Ayala Santamaría, David Sebastián y Ruiz Borja, Mauricio Fernando;** el mismo que cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, además fue revisado y analizado en su totalidad por la herramienta de prevención y/o verificación de similitud de contenidos; razón por la cual me permito acreditar y autorizar para que se lo sustente públicamente.

Sangolquí, 22 de junio de 2022

Firma:

RAMIRO NANAC DELGADO RODRIGUEZ

Ing. Delgado Rodríguez, Ramiro Nanac
C. C 1707019178

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Sistemas e Informática

Responsabilidad de Autoría

Nosotros, **Ayala Santamaría, David Sebastián y Ruiz Borja, Mauricio Fernando,** con cédulas de ciudadanía n° 1725155350 y n° 1724144041 declaramos que el contenido, ideas y criterios del trabajo de titulación: **Sistema integrador de información en tiempo real, para la gestión y control de combustible, empleando tecnología IoT implementada a controladores industriales, en la estación de servicio "Eco el Oasis"** es de nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí, 22 de junio de 2022

Firma

.....

Ayala Santamaría, David Sebastián

C.C.: 1725155350

Firma

.....

Ruiz Borja, Mauricio Fernando

C.C.: 1724144041

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Sistemas e Informática

Autorización de Publicación

Nosotros **Ayala Santamaría, David Sebastián y Ruiz Borja, Mauricio Fernando,** con cédulas de ciudadanía n° 1725155350, n° 1724144041, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **Sistema integrador de información en tiempo real, para la gestión y control de combustible, empleando tecnología IoT implementada a controladores industriales, en la estación de servicio "Eco el Oasis"** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de nuestra responsabilidad.

Sangolquí, 22 de junio de 2022

Firma

.....

Ayala Santamaría, David Sebastián

C.C.: 1725155350

Firma

.....

Ruiz Borja, Mauricio Fernando

C.C.: 1724144041

Dedicatoria

Dedico este trabajo a mis padres, que han estado presentes en cada etapa y en cada paso que he dado, ellos han sido quien me han inculcado valores y principios de dedicación de esfuerzo y de respeto.

David

Agradecimiento

Agradezco a Dios por poner su sabiduría y su conocimiento en mí, ya que sin el nada de esto hubiera sido posible, agradezco a mis padres por su esfuerzo, por su constancia y por su cariño.

Agradezco a la estación de servicio “ECO EL OASIS” por la idea planteada, por el apoyo brindado tanto tecnológicamente como en los conocimientos que fueron de ayuda para la creación del sistema.

Agradezco a mis amigos que estuvieron presentes en mi etapa universitaria y quienes fueron participes de alegrías, tristezas y sobre todo de esas experiencias que nunca se olvidan.

Agradezco a Alejandra, la mujer que me ha acompañado durante todo este camino y que ha sido la que me ha confortado y apoyado para salir en adelante, agradezco su valentía, sus luchas, sus palabras y sobre todo su amor que estuvo presente desde mis inicios en la universidad.

David

Dedicatoria

Dedico el presente trabajo de titulación a mis padres, María Borja y Guido Ruiz, que con su esfuerzo y dedicación desde que di mis primeros pasos en la vida han sido una constante fuente de inspiración y apoyo incondicional.

A mi hermano, Guido, quien fue mi primer amigo, con quien he compartido momentos inolvidables, y que hasta el día de hoy nos seguidos ayudando y apoyando como la familia que somos.

Mauricio

Agradecimiento

En primer lugar, a mi madre, María Borja, quien supo brindarme su amor desde el primer instante, ha sido una fuente de apoyo y ánimo para seguir adelante, logrando ser la persona que hoy soy. Gracias, mamá por ser quien me cuida cada día de su vida, espero me alcance la vida a devolver todo lo que has hecho por mí.

A mi padre, Guido Ruiz, un hombre íntegro que desde mi infancia me ha inculcado valores y principios que aún hoy en día aplico y no los olvidaré, gracias por tu guía, espero te sientas orgulloso de tu hijo tanto como yo lo estoy de ti.

A la estación de servicio ECO EL OASIS por abrirnos las puertas para que este proyecto pueda realizarse y ser llevado a cabo; y a mi director de tesis quien supo guiarnos de la mejor manera en este camino para que pueda ser completado con éxito.

Finalmente, a la Universidad De Las Fuerzas Armadas – ESPE, mi segundo hogar, donde pase días enteros, noches y experiencias inolvidables, que ha sido testigo de haber alcanzado esta meta tan anhelada, gracias a su comunidad, ha sido un honor formar parte de esta respetable institución.

Mauricio

Índice de contenido

Índice de Tablas

Índice de Figuras

Resumen

Las estaciones de servicio son un pilar fundamental para la distribución y comercialización de combustibles, y son una gran fuente de ingreso para el desenvolvimiento económico del país. Por lo tanto, este tipo de organizaciones requieren de una infraestructura tecnológica capaz de adaptarse a los procesos del negocio y ayudar a que el funcionamiento del mismo se desempeñe de una manera ágil, rápida y eficaz.

El presente trabajo se enfoca en la propuesta de un sistema integrador, capaz de gestionar y controlar el combustible en tiempo real, por medio de tecnologías IoT (Internet de las cosas), como lo es el Raspberry, con el fin de reducir los costos de mantenimiento de la actual infraestructura tecnológica que posee la estación de servicio “ECO EL OASIS” y ser una alternativa de uso para la misma, con características de eficiencia mayor o igual a la actual.

La construcción del prototipo software, se basa en la integración de un Surtidor con un controlador Industrial y un dispositivo IoT para los procesos de venta de combustible, y por otra parte la interacción de un sistema de tanques llamado Veeder – Root con un Raspberry para las operaciones de inventario, descargas y alarmas de los depósitos de gasolina. Dichos procesos son posibles de ejecutar, gracias al intercambio de información entre una aplicación local diseñada en Python y una aplicación web desarrollada en Angular en tiempo real.

Palabras clave: iot, transactor, simulador, veeder – root, raspberry pi

Abstract

Service stations are a fundamental pillar for the distribution and commercialization of fuels, and are a great source of income for the economic development of the country. Therefore, this type of organization requires a technological infrastructure capable of adapting to business processes and helping its operation to be carried out in an agile, fast and efficient manner.

The present work focuses on the proposal of an integrating system, capable of managing and controlling the fuel in real time, through IoT (Internet of Things) technologies, such as the Raspberry, in order to reduce fuel costs. maintenance of the current technological infrastructure that the "ECO EL OASIS" service station has and be an alternative use for it, with efficiency characteristics greater than or equal to the current one.

The construction of the prototype software is based on the integration of a Dispenser with an Industrial controller and an IoT device for the fuel sales processes, and on the other hand, the interaction of a tank system called Veeder - Root with a Raspberry for the inventory operations, downloads and alarms of gasoline tanks. These processes are possible to execute, thanks to the exchange of information between a local application designed in Python and a web application developed in Angular in real time.

Key words: iot, transactor, simulador, veeder – root, raspberry pi

Capítulo I

Es fundamental entender los antecedentes históricos acerca de los sistemas y estaciones de servicio, ya que, es importante tener claro cómo ha ido evolucionando esta industria. Este capítulo trata acerca de la problemática existente en este entorno, por lo que es necesario, establecer lineamientos que ayuden como guía durante todo el proceso de desarrollo.

Introducción

Antecedentes

En el Ecuador, desde los años setenta se ha venido efectuando la distribución y comercialización de combustibles a través de compañías estatales y privadas, las cuales han sido fundamentales para el desarrollo económico del país. Las estaciones de servicio, que pertenecen a dichas compañías como: Petroecuador, PyS, Terpel, Primax y más, son las encargadas de la venta de hidrocarburos y requieren de infraestructura tecnológica de punta y adecuada, para un correcto funcionamiento (Chamorro, 2013).

En la actualidad, los mercados e industrias distribuidoras de combustible han optado por implementar arquitecturas tecnológicas innovadoras, debido a que, proveen herramientas, funcionalidades y procedimientos que se caracterizan por tener una conexión entre sí, generando una influencia positiva tanto en la calidad de servicio hacia los usuarios, como para la gestión y el manejo de procesos de venta dentro de las estaciones de servicio (Gabriel, 2019).

Gracias a la evolución y a la transformación tecnológica en la que el mundo y por ende todo tipo de industria se encuentra, la posibilidad de desarrollar nuevos sistemas o de automatizar los ya existentes, es mucho más fácil, puesto que, existen herramientas que permiten establecer una

comunicación inalámbrica, fundamentadas en diferentes técnicas, como lo es el IOT (Internet Of Things), que provee elementos y variables ágiles, dinámicas y sencillas de conectar o integrar (Ortiz, 2020).

El reto principal para la implementación de estas técnicas, como por ejemplo en una estación distribuidora de combustible, es conseguir que un conjunto de componentes como: surtidores de distintos tipos, controladores industriales, switches y servidores, logren integrarse de forma eficiente. La aplicabilidad del Internet de las Cosas es muy extensa y brinda una orientación definida hacia la interconexión de herramientas de procesamiento, de control o de percepción, ya que, conecta tanto elementos lógicos como físicos y gracias a una adecuada arquitectura de software es posible llevar a cabo este tipo de sistemas, con una definición metodológica que identifique cada uno de sus factores y de su forma de comunicación.

Los prototipos o los sistemas IoT (Internet de las cosas), de la vida real deben desarrollarse de manera rápida y eficiente. Cuando esto sucede, dos tareas cobran vida de inmediato: programar los dispositivos de IoT (Internet de las cosas), y organizar un backend que interactúe con estos dispositivos. En ambas tareas, se puede utilizar Python como lenguaje de desarrollo, o bien, se puede hacer uso de una versión completamente funcional y práctica de MicroPython para trabajar en dispositivos con pocos recursos informáticos y, en consecuencia, a un costo bajo.

Python es un lenguaje de programación amigable, que tiene diversas aplicaciones como, por ejemplo, en el campo del desarrollo web, investigaciones científicas y en muchas otras industrias, este lenguaje puede ser aplicado a distintos tipos de tecnologías, ya que, ofrece un sin número de ventajas y beneficios para la creación de sistemas simples o complejos. Una conexión que resulta interesante ya sea para proyectos educativos, o para sistemas industriales es el uso de Python en ordenadores de bajo costo y de placa reducida como Raspberry Pi, puesto que una integración de dichos elementos en un

entorno organizacional, como las estaciones de servicio, puede reducir notablemente los costos de infraestructura tecnológica.

La estación de servicio “ECO EL OASIS” ubicada en Quito, en la Av. Manuel Córdova Galarza Lote 45, dedicada a la venta de combustible para automóviles, motocicletas y vehículos. Empezó sus actividades en el año 2017 y hasta el año 2019 su cuerpo de trabajo se encontraba conformado por 14 personas; según los datos proporcionados por la estación, la infraestructura que se emplea para realizar el procesamiento de información en tiempo real y la gestión de combustible es costosa, debido a sus componentes y herramientas, por lo cual, está en busca de un nuevo sistema que sea ágil y eficiente, pero con componentes de precios más accesibles.

Como antecedentes para la ejecución de este proyecto de investigación se tomaron en cuenta algunos trabajos que poseen una similitud con el tema propuesto como por ejemplo: “Monitoreo del nivel de combustible en la Gasolinera Milagro, utilizando tratamiento digital de imágenes y generación de alarmas”, el cual, se desarrolló en el año 2016 y especifica la realización de un modelo que monitorea la cantidad de combustible de una forma más precisa en tiempo real, con el fin de reducir incidentes o eventualidades no deseadas, permitiendo un dominio más seguro (Espinoza, 2016).

Otro ejemplo que se tomó como base es el “Sistema de control electrónico para el reconocimiento y almacenamiento de combustibles mediante hardware y software libre” realizado en el año 2017, el cual, consideró las dimensiones y tamaños reales de los tanques de almacenamiento de combustible, diseñando de esta manera un prototipo que ayudó a la inspección de la cantidad de gasolina que se encontraba presente en los tanques con gran exactitud (López, 2017).

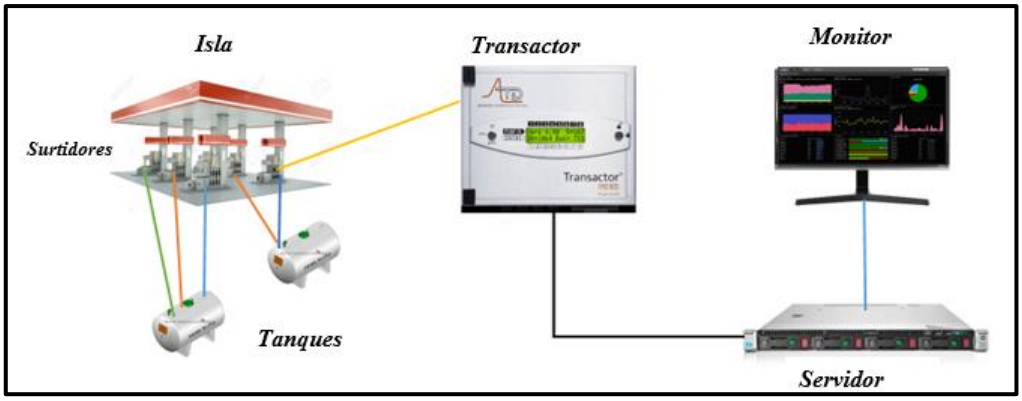
Planteamiento del problema

El petróleo se ha destacado por ser el producto que aporta en mayor cantidad a la sociedad y a la economía ecuatoriana desde el año 1972, puesto que, el uso del mismo se emplea en diversas organizaciones e industrias, posicionado al Ecuador como un territorio petrolero, sin embargo, las fallas e inconsistencias que se presentan tanto en la infraestructura tecnológica como en la tecnificación de procesos, dentro de las diversas comercializadoras de combustible, constituyen un problema para el servicio que se brinda al usuario y para la correcta gestión del combustible (Mateo & García, 2014).

La infraestructura tecnológica de una estación de servicio está conformada por islas, en las cuales se encuentran los surtidores o dispensadores, y estos a su vez, disponen de dos lados, cada lado posee mangueras que están conectadas hacia tanques, los mismos que están ubicados bajo tierra, que es en donde se almacena el combustible. La tecnología que se usa para la gestión de la gasolina y para el procesamiento de información, son controladores industriales (Transactor), los que permiten una comunicación con los dispensadores, la información que se obtiene a través de dicha comunicación se almacena en servidores, los cuales, se encuentran localizados dentro de la misma estación.

Figura 1.

Infraestructura Tecnológica Estación de Servicio

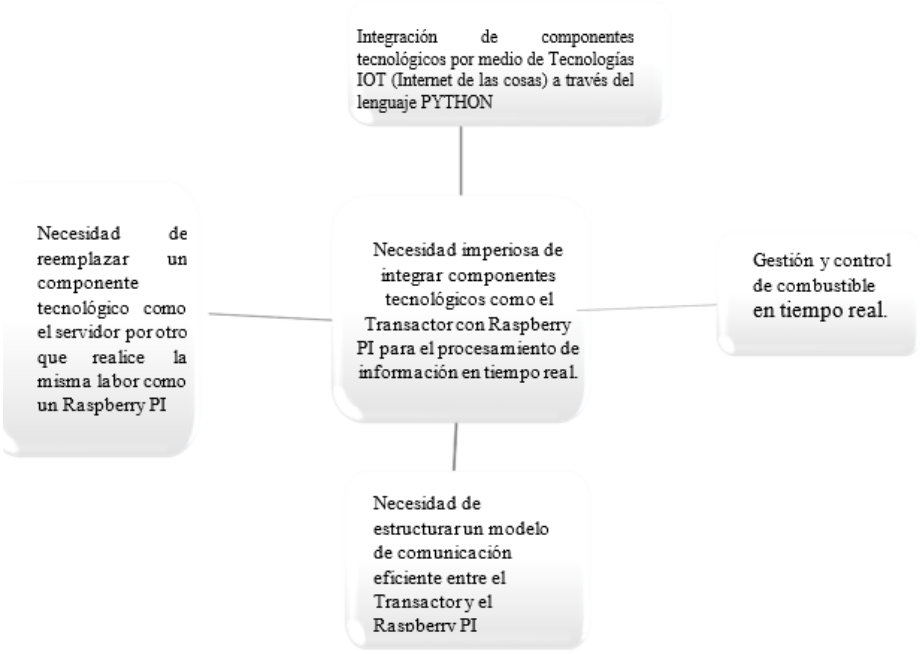


Según la información proporcionada por la estación de servicio “ECO EL OASIS” la infraestructura tecnológica que posee, tiene una similitud con la que se muestra en Figura 1., por lo cual, el costo de implementación para la comunicación entre el surtidor, el controlador industrial (Transactor) y el servidor es demasiado alto además de que, el tiempo en el que se procesa la información desde el momento en el que empieza la venta de gasolina hasta que termina , tiene un delay de aproximadamente 4 segundos, ocasionando que en la pantalla del sistema se visualice un descuadre de 25 ctvs con referencia al precio real.

Lo que se buscó, fue reducir el costo de la infraestructura y realizar un procesamiento de información en tiempo real, reemplazando el servidor por un Raspberry en el cual, se encuentra la base de datos, que es donde se almacenan los datos recogidos que se obtienen durante la comunicación. El diseño que se visualiza en la Figura 1., es el modelo que todas las estaciones de servicio poseen, debido a que, no existe la propuesta ni la creación de otro tipo de infraestructura, por lo cual, no pueden optar por un modelo diferente, que sea de calidad superior y de bajo costo. En la Figura 2. Se establecen los puntos fundamentales que estructuran la problemática del proyecto de tesis.

Figura 2.

Características Árbol de Problema



La centralización fundamental de la formulación del problema está enfocada en la necesidad de reducir los costos de infraestructura tecnológica, reemplazando el servidor que es un componente demasiado costoso, por otro elemento tecnológico como Raspberry PI, que realice la misma labor y procese la información del combustible de mejor manera y en tiempo real, pero con precios reducidos de implementación.

Para esto se ha identificado 3 pilares fundamentales del problema que son:

- ¿Cómo afecta la necesidad de procesar la información en tiempo real en los procesos de control de combustible, en relación con la venta de éste?
- ¿Cuál es la necesidad fundamental de integrar un controlador industrial como el Transactor con un Raspberry por medio del Internet de las Cosas, a través de PYTHON?

- ¿Cuál es la necesidad de reemplazar el servidor por otro elemento tecnológico como lo es el Raspberry PI?

Justificación

Hoy en día, algunas empresas encargadas de la gestión y manejo de combustible tienen carencias en sus procesos de comunicación, ya que, si bien estos son bastante rápidos, no se logra obtener la información de los procesos de carga o descarga de combustible en tiempo real. Como solución se propone un modelo de comunicación que se integre a un sistema tecnológico y automatizado para el procesamiento de información en los procesos de venta de combustible, integrándolo a un componente IoT (Internet Of Things) de bajo costo como un Raspberry Pi, el cual desempeña la función de establecer una conexión eficiente y precisa entre un Transactor (pump controller), que es el encargado de traer la información en tiempo real y los surtidores de combustible, esta información es procesada por medio de una aplicación desarrollada en lenguaje de propósito general como es Python. Todo esto ayuda en los procesos operativos, de manera que sean óptimos, fiables previniendo la perdida de combustible, además de reducir tiempos y costos.

Si bien en la actualidad la arquitectura que posee la estación de servicio “ECO EL OASIS” no presenta un problema de alto riesgo, la intención de este proyecto de investigación es reducir los costos de la infraestructura tecnológica existente y mejorar el tiempo de procesamiento de información sobre los procesos de venta de combustible, implementando una arquitectura basada en tecnología IoT (Internet Of Things), que además de cumplir con su objetivo, entrega la información en tiempo real de manera fiable, permitiendo establecer un nuevo prototipo de comunicación de menor costo tanto de implementación como de mantenimiento.

Es importante tener un control del combustible suministrado por parte del distribuidor, puesto que la carga puede ser ligeramente menor a lo que indica la orden facturada. Por ejemplo, de una entrega de 10,000 litros, solo se recibe 9990 litros generando un margen de error en el momento de cargar el combustible al tanque. La cantidad, calidad y precio, son parámetros de control que son regulados por parte de la Agencia de Regulación y Control Hidrocarburifero (ARCH) en las estaciones de servicio que comercializan combustibles. (Agencia de Regulación y Control Hidrocarburífero, 2011)

La estación de servicio “ECO El OASIS” cuenta con un sistema de gestión de combustible, el cuál proporciona la información de manera fiable y eficiente sin embargo el costo de la instalación y mantenimiento es relativamente alto, como ha sido mencionado por el presidente ejecutivo de la estación el Ing. Fernando Sandoval, ante lo cual se plantea proponer un nuevo prototipo de arquitectura, ya que, como se muestra en la Figura 3 la comunicación es estable y entrega la información en tiempos relativamente cortos, mas no en tiempo real.

Por lo tanto, en la Figura 4 se propuso una arquitectura la cual al obtener la información de los surtidores en bajo nivel, en el Transactor convertirla en alto nivel y posteriormente enviarla a un dispositivo IoT (Internet of Things), esto mediante una adaptación de un RS422 y así obtener una salida USB totalmente funcional para nuestro Raspberry, el cual recibió las tramas enviadas por el controlador industrial, registrándolas en una base de datos y permitiendo mediante un aplicativo enviar estas mismas hasta un servidor alojado por un proveedor en la nube, en el que se almacena una base datos NoSQL totalmente manejable y que interpreta la información en tiempo real los datos en alto nivel que vienen desde el Transactor, y de igual manera enviar comandos en bajo nivel al surtidor.

Figura 3.

Arquitectura Actual Estación De Servicio ECO El Oasis

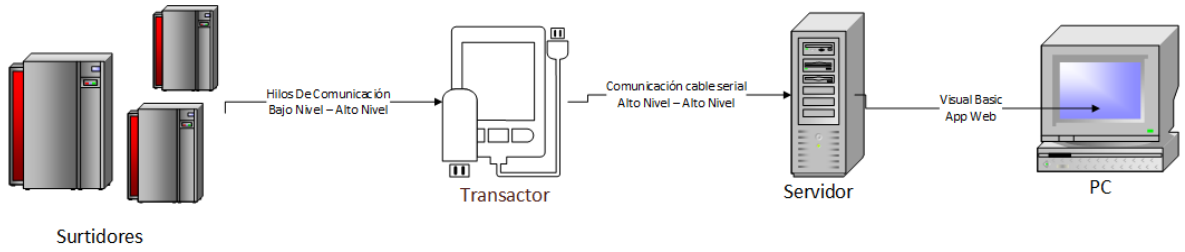
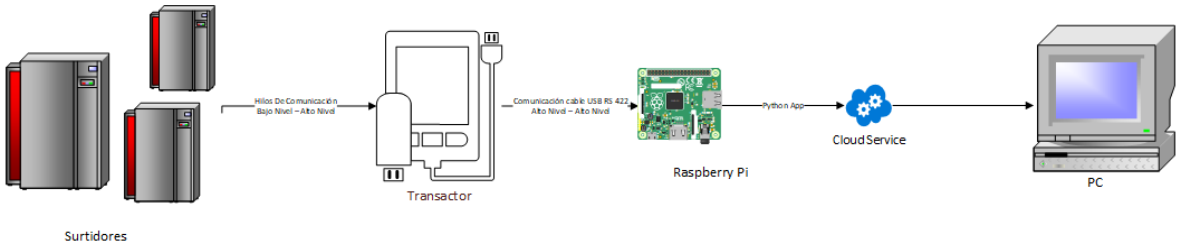


Figura 4.

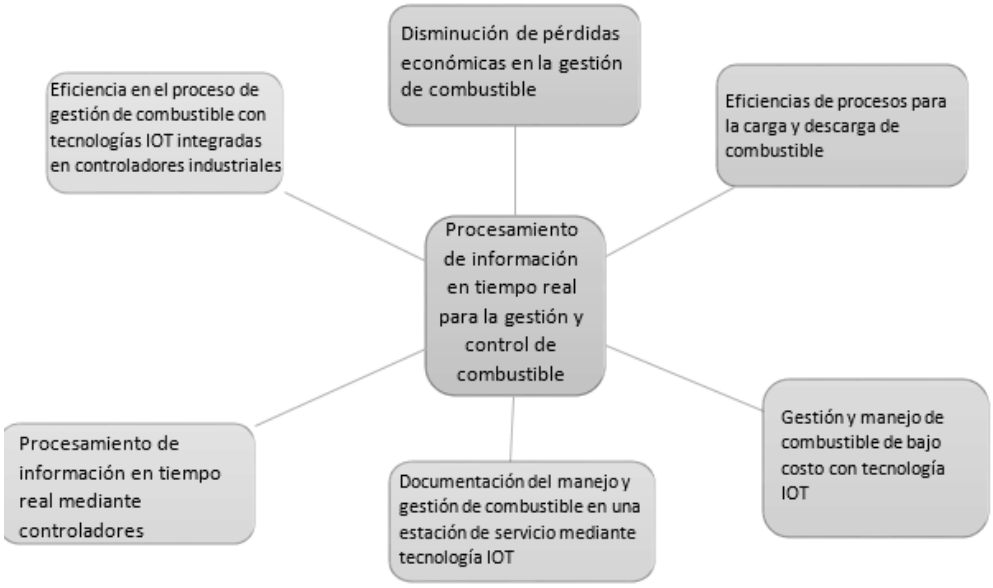
Arquitectura Tiempo Real Con Tecnología IoT



Técnica del árbol de objetivos

Figura 5.

Árbol de Objetivos



Los beneficios de la integración de un sistema de gestión y control para combustible en tiempo real se muestran en la Figura 5 y tienen características como:

- Comunicación en tiempo real para la gestión de combustible con tecnología IoT (Internet de las Cosas) de bajo costo para manejar información en una interfaz entendible por los despachadores.
- Evitar pérdidas económicas difíciles de detectar sin un exhaustivo análisis de costos.
- Tener la información en tiempo real enviada por un controlador industrial, cargada en un servidor en la nube la cual se encuentre disponible en cualquier momento.
- Tener un registro adecuado y continuo del stock tanto de carga y descarga, lo que permite optimizar tiempos de pedidos al distribuidor de combustible

Objetivos

Objetivo general

Desarrollar un sistema integrador de información en tiempo real, entre un controlador industrial (Transactor) y un Raspberry PI, basado en software, para la gestión y control de combustible optimizando el costo de la infraestructura tecnológica, mediante la utilización de tecnologías IoT por medio del lenguaje de programación PYTHON, dentro de la estación de servicio “ECO EL OASIS”.

Objetivos específicos

- Realizar un análisis comparativo, de diseños y modelos de referencia para arquitecturas IoT (Internet Of Things), con el fin de seleccionar un modelo que solvente las necesidades para la integración entre un controlador industrial (Transactor) y un Raspberry PI.
- Establecer un sistema de comunicación, que envíe y reciba información en tiempo real, mediante protocolos definidos entre un controlador industrial (Transactor) y un Raspberry PI.
- Gestionar las tramas generadas desde un controlador industrial (Transactor) en un Raspberry Pi, con el fin de almacenar las mismas, en una base de datos NoSQL y mediante un aplicativo web poder visualizarlas.
- Procesar la información obtenida por la comunicación establecida mediante un modelo de arquitectura IoT (Internet Of Things) y presentarla en una aplicación, desarrollada en un lenguaje de propósito general PYTHON.

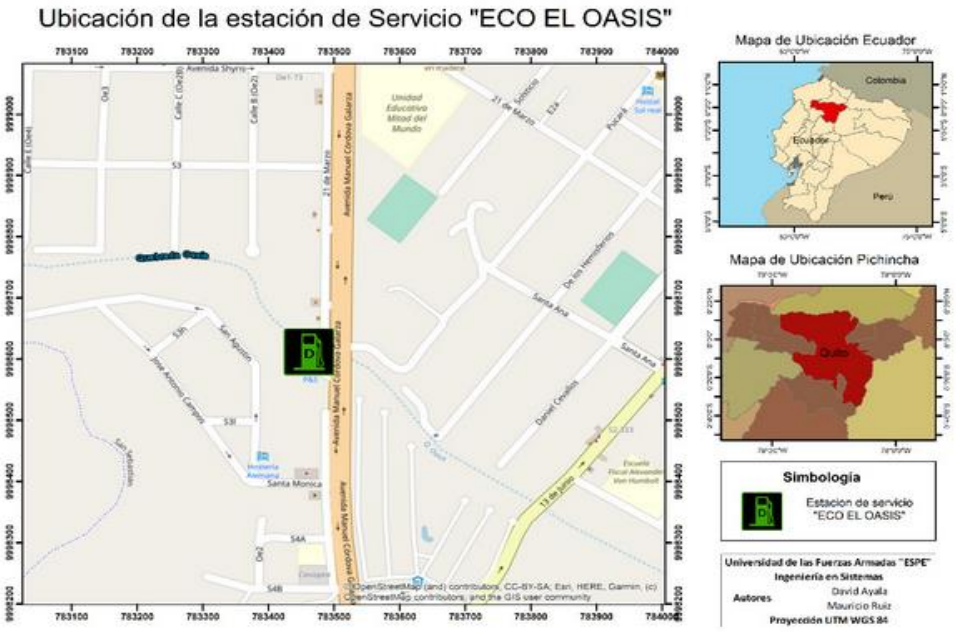
Alcance

En el presente proyecto de investigación, se planteó el desarrollo de un sistema que ayude a la integración de un controlador industrial (Transactor), con un Raspberry PI, mediante tecnologías IoT (Internet Of Things) a través del lenguaje de programación PYTHON para el procesamiento de información en tiempo real, tomando como guía, un modelo basado en arquitectura IoT (Internet of

Things) que ha sido previamente seleccionado, resultado de un análisis comparativo entre los distintos modelos, que brindan una solución óptima, y eficiente a este tipo de proyectos.

Figura 6.

Ubicación estación de servicio "ECO EL OASIS"



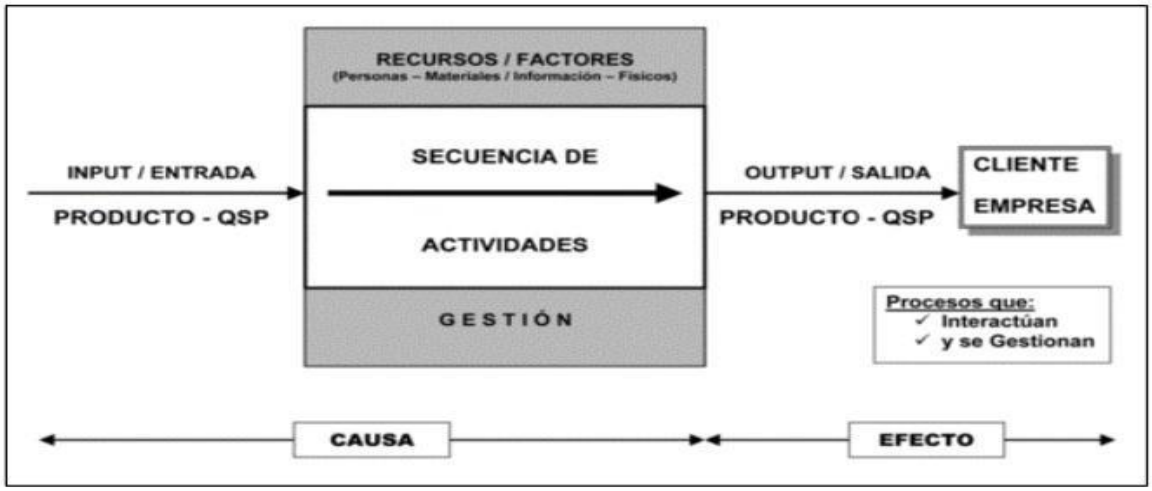
El proyecto de investigación se desarrolló para la estación de servicio “ECO EL OASIS” en la cual, se realizaron las pruebas respectivas, la estación posee todos los elementos tecnológicos, para la creación del sistema integrador. En la Figura 6, se observa la ubicación exacta y la zona geográfica en donde se encuentra situada la estación de servicio.

El sistema integrador abarca una serie de operaciones secuenciales que están destinadas a tomar un input o entrada (son los datos proporcionados por el surtidor) para transformarlo mediante el controlador industrial (Transactor) en una secuencia de actividades en la que se involucran recursos

(personas) o factores (información, factores físicos), y dar como resultado una salida u output, como muestra en la Figura 7, a continuación:

Figura 7.

Procesos



Nota. (Pérez 2010, 53)

Los procesos que han sido identificados por la estación de servicio “ECO EL OASIS” que se tomaron como base para el desarrollo del sistema integrador, son los siguientes:

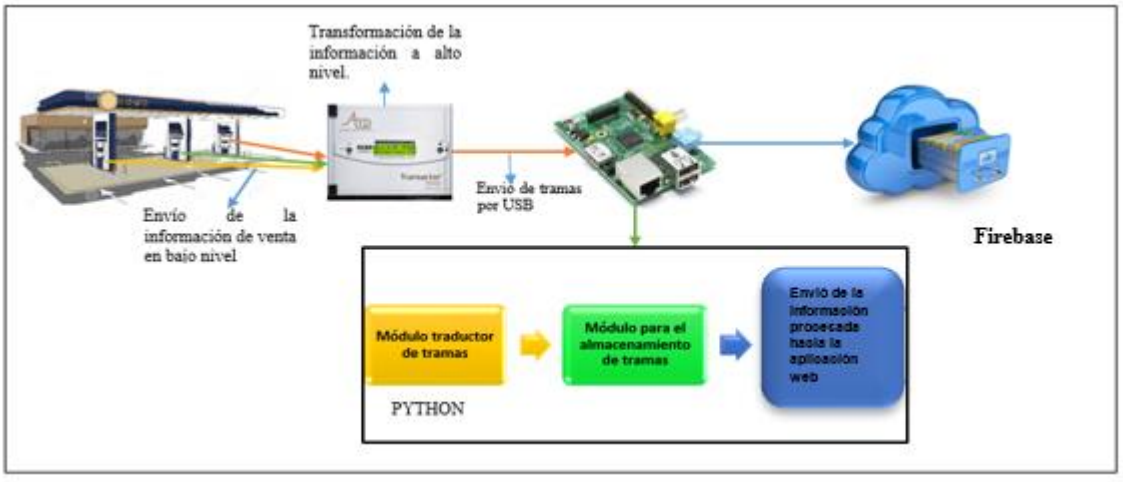
- Integración de los componentes: Surtidor, controlador industrial (Transactor), Raspberry Pi, para crear una comunicación eficiente y un procesamiento de información en tiempo real sobre el proceso de la venta de combustible.
- Creación de un módulo traductor de tramas, el cual es el encargado de leer los datos enviados por el controlador industrial (Transactor), por medio del lenguaje de programación PYTHON.
- Creación de un módulo para el almacenamiento de datos, el cual cumple con la función de almacenar la información de las tramas procesadas en una base de datos no relacional (Firebase).

- Creación de un módulo para la lectura de información almacenada en la base de datos no relacional (Firebase).
- Creación de un Dashboard y un tablero de mando una vez procesada la información que se alberga en una aplicación web.

En la Figura 8 se detalla, la secuencia y los procesos que se desarrollaron a lo largo de este proyecto de investigación:

Figura 8.

Proceso de desarrollo



A través de la investigación y el desarrollo del sistema se generó un prototipo que sirve de ayuda tanto para el administrador como para el servicio que se brinda a los beneficiarios en un tiempo estimado de 5 meses, por medio de tecnologías IoT (Internet Of Things) creando una integración y una comunicación exitosa entre los componentes ya mencionados, de una forma estructurada, organizada y sistematizada.

Hipótesis

Al realizar la interconexión entre los elementos Tokheim, Transactor y Veeder-Root con el Raspberry, por medio de una comunicación serial y a través de la interacción entre los lenguajes Python y Angular, se creó un sistema integrador, el cual, es capaz de controlar y gestionar el combustible en tiempo real, además de ser eficiente, eficaz y distinto a los existentes en las estaciones de servicio actuales, logrando remplazar un servidor por un Raspberry, y permitiendo la reducción de los costos de implementación y mantenimiento de infraestructura tecnológica para la estación de servicio “ECO EL OASIS”.

Capítulo II

Dentro del desarrollo de un proyecto siempre es necesario conocer con qué componentes, aplicativos y herramientas se va a trabajar e interactuar durante todo el proceso que abarca desde la planificación hasta la implementación, es por eso que en este capítulo se especifican todos los términos y componentes involucrados en el desarrollo del sistema de gestión y control de combustible para la estación de servicio “ECO EL OASIS” , todo esto mediante estudios de literatura, gráficas y tablas explicando de la manera más adecuada su función y comportamiento esperado.

Marco teórico

El marco teórico se encarga de mostrar los tópicos que tiene relación con el proyecto de investigación, además de, estar fundamentado en la red de categorías.

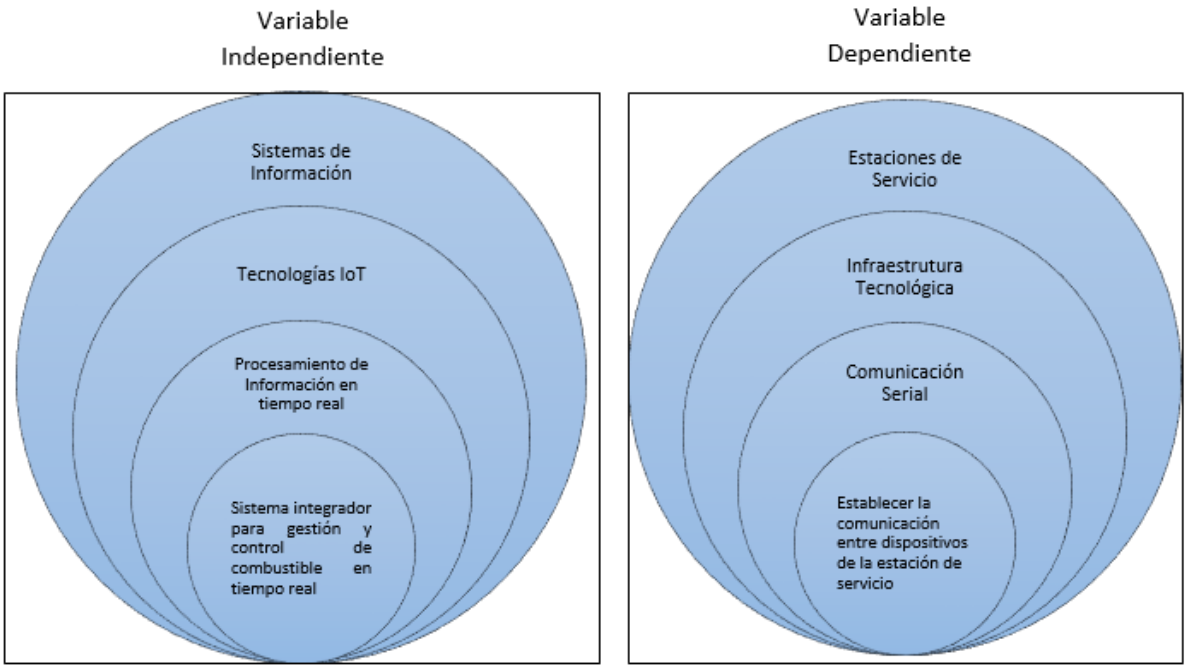
Red de Categorías

La red de categorías otorga la capacidad de estructurar el cimiento teórico, de acuerdo a la temática presentada, basándose en variables independientes y dependientes de la hipótesis que se estructuró en el Capítulo I. En la Figura 9 se muestra la jerarquización de variables tanto independientes como dependientes.

- **VI:** Sistema integrador para la gestión y control de combustible en tiempo real, basado en tecnologías IoT (Internet de las Cosas).
- **VD:** Establecer la comunicación entre los elementos Tokheim, Transactor y Veeder-Root con el Raspberry para la reducción de costos en la infraestructura tecnológica de la estación de servicio “ECO EL OASIS”.

Figura 9.

Jerarquización de variables



Fundamentación Científica de la Variable Independiente

Sistemas de información

Un sistema de información se define como un conjunto de elementos que se relacionan e interactúan entre sí, con la finalidad de lograr un objetivo en común. Independientemente existen varios sistemas de diferentes tipos o en distintos campos de aplicación. Todos los sistemas se encuentran formados de un modelo de cinco bloques: elementos de entrada, elementos de salida, sección de transformación, mecanismos de control y objetivos.

Existe una gran cantidad de definiciones de lo que es un sistema, pero el escenario es totalmente distinto al hablar de la definición de un sistema de información. Sin embargo, basado en

definiciones y matices que son dadas por las organizaciones quienes usan este término se realiza un primer acercamiento de una definición muy acertada, conceptuándola como un conjunto de componentes que interactúan entre sí con una finalidad o intención de conseguir un objetivo en común, satisfacer las necesidades de información de una organización (Laudon & Laudon, 2012).

En el afán de definir de una manera más específica, los sistemas de información son un conjunto de componentes interrelacionados que recopilan, procesan, almacenan y distribuyen información para sugerir o ayudar en la toma de decisiones, el manejo y gestión de una parte o de toda una organización. Además de ayudar al factor humano (dueños, gerentes, empleados) a analizar problemas, visualizar temas complejos y crear soluciones que cubran todas las necesidades que se presenten (Osborne et al., 2012).

IoT (Internet de las cosas)

El internet de las cosas se ha consolidado como un adelanto significativo para el mundo, ya que, existe una gran cantidad de usuarios que se encuentran en contacto con el Internet, ya sea en el campo laboral o en el diario vivir. El IoT (internet of things) constituye un enfoque en el cual, los elementos tecnológicos inteligentes están ligados a una conexión con la red, estableciendo un dominio y un control remoto sobre diferentes tipos de situaciones, sin la necesidad de una manipulación física.

En gran parte, la red ha sido utilizada para protocolos enfocados a la integración de aplicaciones, así como SMTP (Simple Mail Tansfer Protocol) y HTTP (Hypertext Transfer Protocol). No obstante, en la actualidad una gran cantidad de herramientas tecnológicas establecen su comunicación con diferentes tipos de sistemas de control, empleando un concepto llamado M2M (comunicaciones de máquina a máquina) (Salazar & Silvestre, 2017). El IoT (Internet de las cosas) posee ventajas en el campo de la gestión de artículos y servicios, proveyendo un incremento en la cantidad de datos, aportando en la

optimización de herramientas y reduciendo el uso de recursos, permitiendo un ahorro en los costos de las industrias u organizaciones (D. Evans, 2011).

Aplicaciones IoT

IoT está definida como una integración de sensores y actores, que tienen la capacidad de enviar y receptar información, para después posicionarla en redes de doble sentido, es decir, existe una devolución de información tanto del emisor como el receptor para establecer una comunicación bidireccional, permitiendo la transmisión de datos, para emplearlos en distintas funciones por diferentes usuarios. Los elementos y componentes que conforman la tecnología IoT (Internet de las Cosas) se muestra en la Figura 10.

Figura 10.

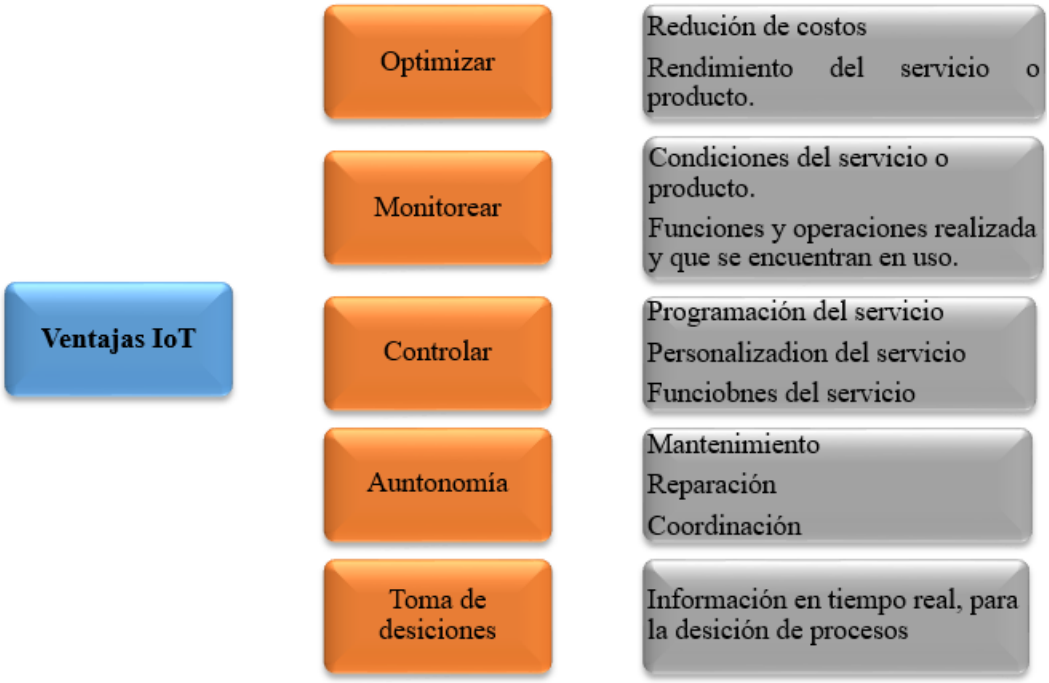
Definición IoT (Internet de las Cosas)

El instante en el que la cantidad de dispositivos conectados a Internet fue mayor que el número de personas enlazadas, se creó un mundo de posibilidades para el desarrollo de software en el campo

de la automatización. La posibilidad de integrar múltiples sensores a una herramienta tecnológica otorga la capacidad de obtener gran cantidad de variables y datos definidos por el entorno en el cual se encuentra una organización, para después ser procesados y enviados hacia la nube, creando de esta manera un modelo de servicio. Las ventajas que posee la tecnología IoT (Internet de las cosas) son diversas y están detalladas en la Figura 11.

Figura 11.

Ventajas IoT (Internet de las Cosas)



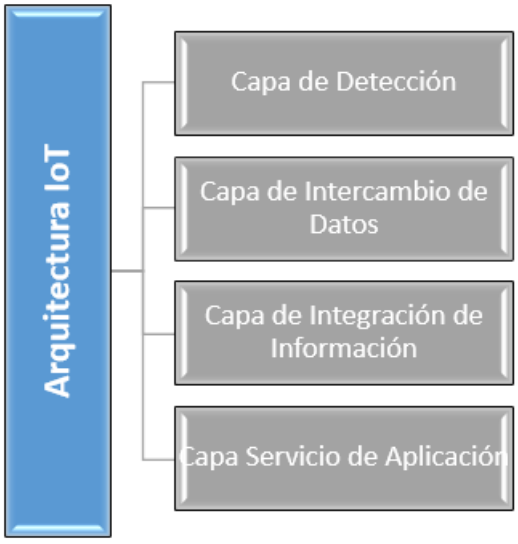
Arquitectura IoT (Internet de las cosas)

Las herramientas tecnológicas inteligentes pueden encontrarse ya enlazadas por medio de Internet tradicional. No obstante, el internet de las cosas posee una arquitectura que está clasificada en

cuatro capas, como se observa en la Figura 12, una de ellas es la capa de detección, la cual disminuye las condiciones que establecen los dispositivos para realizar una interconexión entre sí. Existen sensores que se encargan del consumo de datos y a su vez permiten la comunicación con otros sensores similares por medio de la capa de integración, la cual es la encargada de las transacciones y de la comunicación, a lo largo de los diferentes procesos de integración de dispositivos nacen nuevos retos, enfocados en el intercambio, la interconexión y el filtro de información, la estructuración de servicios innovadores para el usuario final y el aumento de la dificultad de arquitecturas de red. Por otro lado, uno de los beneficios del internet de las cosas es, el diseño y generación de productos y servicios innovadores, brindado una alta productividad de la mano de la disminución de costos ya sea de infraestructura tecnológica o de la obtención y uso de servicios existentes.

Figura 12.

Arquitectura IoT (Internet de las Cosas)



Procesamiento de información en tiempo real

La información en tiempo real está relacionada estrechamente con sistemas informáticos que suelen evolucionar externa y ágilmente con una temporalidad física, esto dependiendo de su funcionalidad o naturaleza (Martínez, 2010). Los cuales, dan acceso a que exista dinamismo con los beneficiarios y tienen como objetivo principal brindar una respuesta al instante dentro de los límites de tiempo que se hayan determinado. Lo que permite Angular es el desarrollo de diferentes aplicaciones que se basan en un patrón de arquitectura, el cual también ha sido establecido en JavaScript y HTML (Loarte Cajamarca & Maldonado Soliz, 2019).

Modelo de procesamiento de información en tiempo real

Un sistema en tiempo real es un conjunto de procesos, los cuales deben satisfacer restricciones explícitas (acotadas) de tiempo de respuesta o arriesgarse a sufrir graves consecuencias, incluido el fracaso, en algunos casos, el procesamiento de información en tiempo real se puede referir a la apariencia de una respuesta instantánea, sin embargo, existe un mínimo retraso, aunque este puede que sea imperceptible. (Thoen et al., 2022)

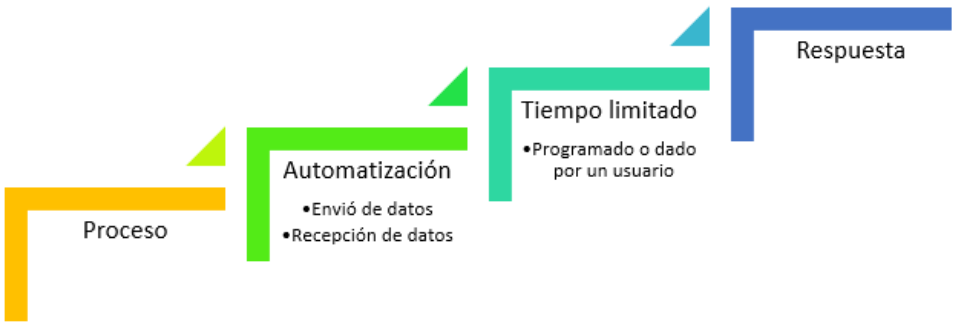
¿Cuándo se considera un sistema que es en tiempo real?

Los sistemas en tiempo real suelen ser sistemas reactivos o empotrados, aquellos en los que la programación está impulsada por la interacción continua con su entorno; por ejemplo, un sistema de control de incendios reacciona a los botones pulsados por un piloto. Los sistemas integrados son los que se encuentran en un sistema que no es en sí mismo un ordenador. Por ejemplo, un automóvil moderno contiene muchos ordenadores integrados que controlan la inyección de combustible, el despliegue del airbag, el frenado, el control del clima, etc.

Los sistemas de procesamiento de información en tiempo real parten de la especificación de un sistema de procesos concurrentes e intentan automatizar la asignación de datos o envíos a un único procesador. Un modelo de procesamiento que se adapta bien al soporte de restricciones de concurrencia y de tiempo, junto con modelos de ejecución flexibles es aquel que cubre todas las necesidades y maneja un conjunto de tareas o procesos con un mínimo de tiempo establecido o limitado. Las etapas de un modelo de procesamiento de información en un sistema de tiempo real se muestran en la Figura 13.

Figura 13.

Modelo de procesamiento de información en un sistema de tiempo real



Sistema integrado de gestión y control de combustible

Antes de la avanzada evolución tecnológica, los administradores de unidades o flotas habituaban ingresar la adquisición de combustible de forma manual, es decir con hoja y pluma, todo este proceso requería de un tiempo elevado ocasionando ineficiencia, tanto para las organizaciones como para el medio ambiente. Gracias a la creación y transformación de los sistemas informáticos, nacieron softwares que otorgaban la posibilidad de gestionar el combustible de manera mucho más ágil,

rápida y eficiente, además de almacenar los datos y procesar la información para tomar acción sobre medidas que ayuden a la mejora de compra y venta de combustible.

La diversificación de softwares que se encuentran en el mercado referentes a la gestión y control de combustible, varían en funciones específicas, basándose en las necesidades de las instituciones o estaciones de servicio; independientemente de los procedimientos o de los valores agregados que posea cada sistema, su operatividad y modelo de negocio por lo general es el mismo, debido a que, su principal enfoque es facilitar el manejo de los datos y la información referente al combustible, ya sea la cantidad vendida o la cantidad que se necesita para que los tanques operen con normalidad. Una de las variables más representativas para el diseño y uso de estos sistemas, son los datos del vehículo y del conductor, ya que, permite que el software recoja información del usuario y el combustible se despache de forma mucho más ágil, además de ofrecer una configuración específica dependiendo del tipo vehículo y del tipo de cliente (Bridgestone Company, 2015).

Sistema de Control

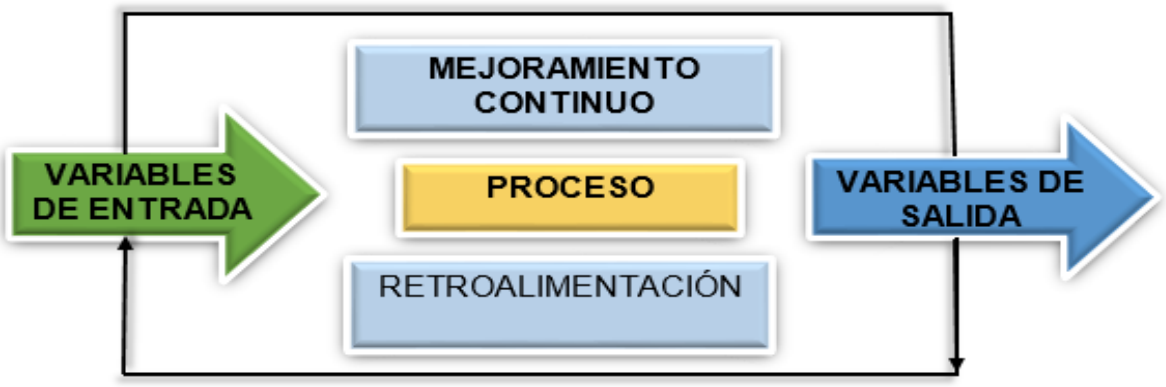
Conceptualmente se puede definir como una entidad que recepta atributos de entrada o procedimientos externos, y cuya función es emitir una contestación o replica a la entrada que recibió, denominando la respuesta como atributo o variable de salida como se muestra en la Figura 14. Los fenómenos que ocurren desde el exterior se clasifican en dos conjuntos, el primero las llamadas alteraciones, en las cuales, no existe la posibilidad de intervención y las segundas, conocidas como variables de control sobre las cuales se puede ejercer un control o manejo de estas. La definición de sistema de control se determina por la existencia de componentes específicos, los cuales permiten la intervención en el trabajo del sistema; el objetivo de estos sistemas es obtener a través de operaciones sobre los componentes de control, una influencia en los atributos de salida, de manera que, estas

adquieran elementos ya establecidos, y para alcanzar su objetivo, el sistema de control debe poseer las siguientes características (Huamancayo & Saavedra, 2020):

- Ofrecer consistencia, equilibrio, solidez, y fundamentalmente, poseer robustez ante alteraciones y fallos en los entornos.
- Lograr una eficiencia tanto como sea asequible, mediante juicios y razonamientos preestablecidos, naturalmente dichos valores o criterios deben tener el objetivo de que las funciones de inspección sobre los atributos de entrada sean posibles y ejecutables, omitiendo procesos ilusorios.
- Tener la capacidad de una implementación, fácil, ágil y rápida, con el fin de una manipulación en tiempo real, por medio de una Pc u otro dispositivo electrónico.

Figura 14.

Sistema de Control



Nota. (Villalta, 2010)

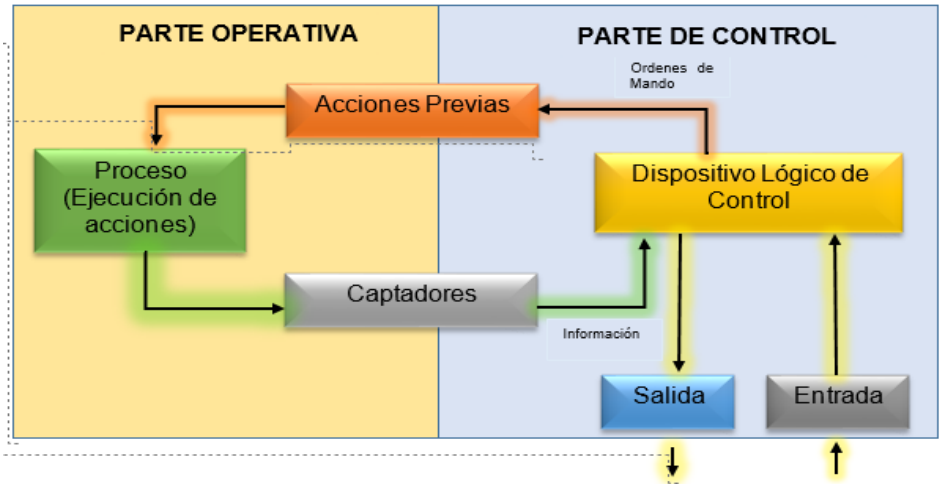
El sistema de control determinado para la gestión de combustible tiene variables de entrada específicas como el surtidor, los tanques en los cuales se almacenan los distintos tipos de hidrocarburos, las mangueras que se encuentran conectadas al surtidor y el dispositivo Transactor, que es el que trabaja directamente con el dispensador y obtiene la información de despacho.

Sistema de Gestión Automatizado

Se caracteriza por tener un entorno informático automatizado, el cual posee la facultad de inspeccionar y gozar de un dominio sobre las funciones de un procedimiento específico, con el fin de controlar el medio en el cual se implemente, de manera continua y sistematizada, partiendo de tecnologías que otorguen la capacidad de ejecutar procesos sin la ayuda de un ser humano. La tarea de automatizar procedimientos es una labor que debe ser llevada a cabo por procesos metodológicos, para obtener un sistema eficiente y eficaz, este trabajo se lo realiza en conjunto con los sistemas de gestión de calidad, el cual es acogido por instituciones y organizaciones, debido a que, por medio de este método se intenta afirmar la satisfacción de los usuarios finales, dicho esto, es imprescindible que la institución organice y estructure un plan de procedimientos y actividades poniendo como objetivo principal el mejoramiento continuo. En la Figura 15, se observan las características básicas de un sistema automatizado (Camones Justiniano, 2020).

Figura 15.

Esquema Sistema Automatizado



Nota. (Villalta, 2010)

Fundamentación Científica de la Variable Dependiente

Estaciones de servicio

Las estaciones de servicio se han consolidado como un pilar y un elemento fundamental para la evolución y transformación de un estado, la configuración de estas organizaciones por lo general está enfocada en la expedición de dos categorías de hidrocarburos, como lo son el diésel y la gasolina, que son provenientes de un procesamiento llamado refinación o depuración de crudo. No obstante, existen algunos establecimientos que ofertan diferentes clases de combustible como:

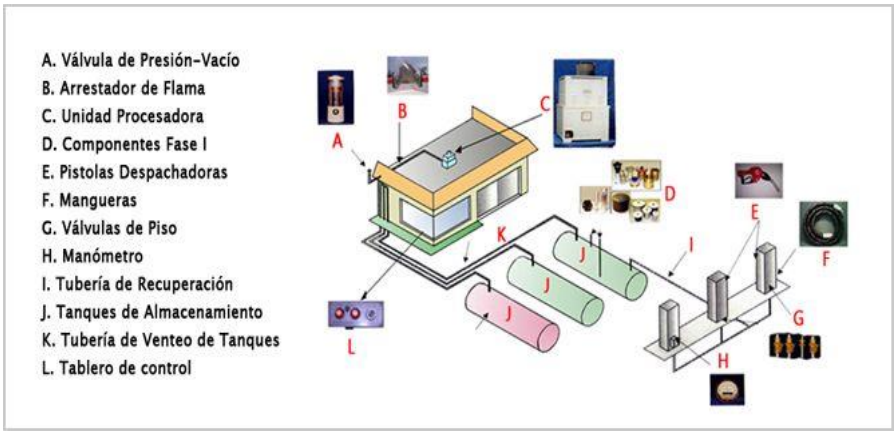
- Gas Natural
- Biodiesel
- Electricidad
- Etanol

Estructura física de las estaciones de servicio

Además de expender servicios de compra y venta de combustible, poseen distintos servicios, como: aprovisionamiento de bebidas y comida, servicios de aseo, sistema de lavado, mangueras de aire y dependiendo de la ubicación o del estado en el que se encuentre la estación, cambios de aceite. Pero fundamentalmente la organización estructural y los elementos que componen un establecimiento de este estilo, son los dispensadores de combustible, los cuales, deben regirse a las normas requeridas por el tamaño de la estación, así como, los tanques, en los cuales se almacena el combustible para su distribución. En la Figura 16, se observa la infraestructura que por lo general la mayoría de estaciones de servicio poseen (Norma Argentina GE-N1-118, 2016).

Figura 16.

Infraestructura Estación de Servicio



Nota. (World Energy Trade, 2019)

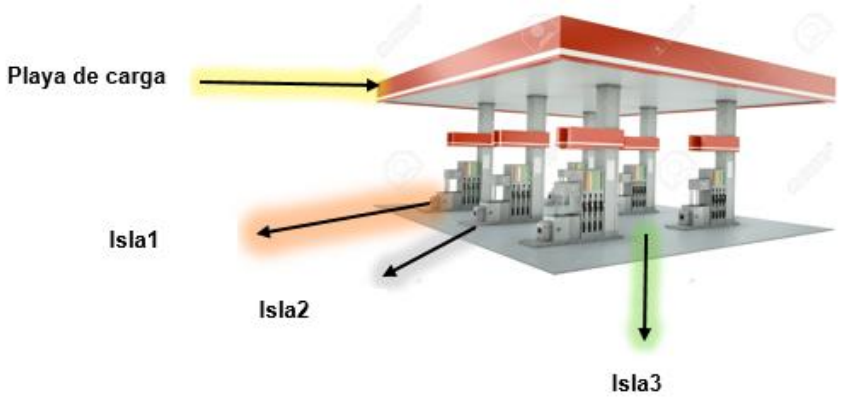
Playa de carga. Conceptualizada como la sección ubicada para el movimiento de vehículos y para la expedición de combustible.

Límites de playa. Definidos como los componentes tangibles que definen la zona reservada a la conducción y movilidad de los automóviles en la playa.

Isla de dispensador o Surtidor. Sección salvaguardada por la playa de carga, en este sector no está permitido la movilidad automovilística, además de contener el surtidor o dispensador de venta de combustible, conteniendo las mangueras y válvulas de aislamiento (Norma Argentina GE-N1-118, 2016), en la Figura 17, se observa la distribución y ubicación de los componentes playa e isla pertenecientes a una estación de servicio.

Figura 17.

Playa e Isla



Nota. (Merriam Webster, 2021)

Surtidor. El dispensador cumple la función de llenar el tanque de combustible del vehículo que está realizando la compra, este mecanismo está conformado por un componente de inspección electrónico, que se encuentra al interior de la embocadura, controlando el procedimiento de bombeo y mostrando en la pantalla del surtidor, compuesto también por válvulas que ayudan a verificar el paso de la gasolina. La cantidad de combustible que ha sido enviado hacia el automotor es medida a través de paletas que mientras van girando van enviando choques o impulsos eléctricos. Las mangueras que se

encuentran conectadas al surtidor y estas a través de ductos conectadas hacia los tanques por donde se envía el combustible, poseen un sistema que ayuda en la detección de llenado, y otro dispositivo permite el cierre evitando el derrame o desperdicio de gasolina. A continuación, en la Figura 18, se visualiza la interconexión existente entre un surtidor y una manguera.

Figura 18.

Surtidor - Manguera

Nota. (Merriam Webster, 2021)

Almacenamiento de combustible

El almacenamiento de combustible correspondiente a las estaciones de servicio, requiere de un lugar ubicado en el subsuelo, el cual cumpla la función de un depósito, para que pueda almacenar combustible y después ser vendido, así como se visualiza en la Figura 19. Existen normas determinadas para poder realizar este tipo de procedimientos como la NTE INEN 2251, 2003, en la que se pide como requisitos, el medio de transporte, el lugar y la forma de almacenamiento, el tipo de manejo, y venta en

las estaciones de servicio para expendio de combustible. Una imposición, la cual deben regir los tanques de almacenamiento es que, posean los siguientes elementos:

- Tanques
- Mecanismos de llenado
- Mecanismos de medición
- Mecanismos para captación de vapores o gases
- Bomba sumergible
- Mecanismo para tubería de ventilación
- Dispositivos de control
- Ingreso para el personal, hacia el tanque, con una abertura mínimo 60cm.

Comúnmente los tanques, se encuentran fraccionados en dos o tres sectores, esto varía según la cantidad de tipos de combustibles que maneje la estación de servicio. Los tanques tienen un diseño estructurado para su adecuado funcionamiento, la primera característica es que la forma del tanque tiene que ser cilíndrica, y su disposición horizontal; debe poseer una capa doble en sus paredes, además de contar con un sistema que permita monitorear posibles fugas, su componente de fabricación tiene que estar basado netamente en acero, el diámetro interior y el espesor de sus capas son definidos por medio de la cantidad de almacenamiento y por su material de fabricación así como se visualiza en la Tabla 1 (Espín Ramos, 2016).

Tabla 1.

Dimensiones de Tanques

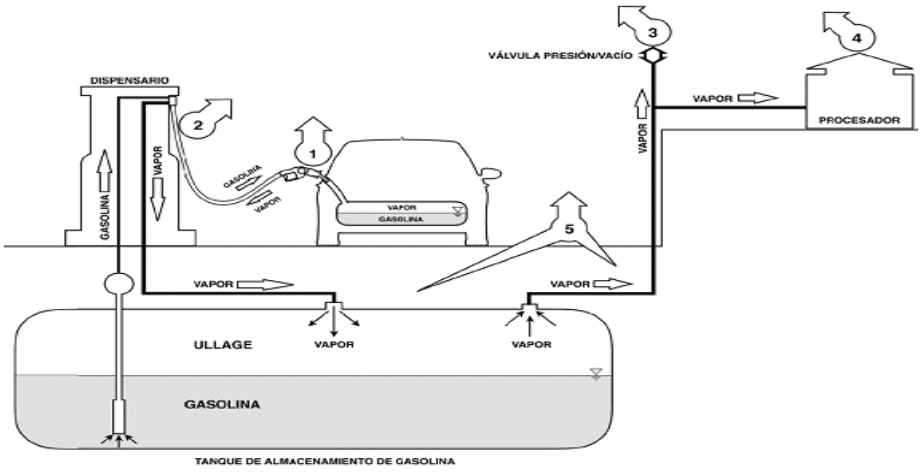
Espesor(mm)	Capacidad (dm^3)	Diámetro interior (m)
1,70	Máximo 1078	1,07
2,36	1082 a 2120	1,22
3,12	2124 a 4164	1,63
4,24	4168 a 15142	2,13
6,00	15145 a 45425	3,20
7,67	45429 a 75708	3,66
	75712 a 189270	3,66

Nota. (Instituto Ecuatoriano de Normalización, 2006)

Las dimensiones y datos de la Tabla 1, tiene como fin determinar los valores que poseen los tanques de combustible, ya que debe existir una congruencia y una proporcionalidad con la cantidad de combustible y el tamaño de la estación de servicio.

Figura 19.

Almacenamiento de Combustible



Nota. (DOF, 2017)

Infraestructura Tecnológica

La infraestructura tecnológica son los componentes electrónicos necesarios para la construcción del sistema integrador que permite monitorear y gestionar el combustible en tiempo real.

Sistemas Empotrados

Es un sistema informático relacionado al diseño de funciones que en su mayoría ocurren dentro de un sistema en tiempo real, la funcionalidad primordial de este diseño es la de ejecutar tareas de control. La diferencia entre una computadora personal y un sistema empotrado radica en que, la PC cumple con un rango de necesidades, mientras que estos sistemas fueron realizados con el objetivo de cumplir necesidades específicas (UPV, 2022). Además, tienen la capacidad de programar en el lenguaje ensamblador del propio microprocesador o empleando compiladores específicos. Una de las características principales de un sistema empotrado es que tiene un costo bajo y consumo de potencia, esto se debe a que estos sistemas suelen producirse en miles de unidades y el costo unitario es primordial en la fase de diseño (Lifelong Learning, 2011).

Simulador Tokheim

El simulador Tokheim hace las veces de un dispensador de combustible real, permitiendo la realización de pruebas en el proceso de venta de gasolina, siendo el componente de campo de la suite de operación de este laboratorio tecnológico y el actor inicial de la comunicación con el equipo Transactor, antes de implementarlo en un ambiente productivo. Este simulador proporciona información acerca del estado de las pistolas, situación del despacho de combustible, si está autorizado o rechazado, si la venta ha sido completada, si no hay conexión o si la venta se está realizando. Dicho esto, este componente fue de gran ayuda para experimentar con el modelo de comunicación que se creó, sin alterar el funcionamiento de la estación de servicio “ECO EL OASIS”, este equipo tecnológico fue proporcionado por la misma estación, así como se observa en la Figura 20.

Figura 20.

Simulador Tokheim

Transactor

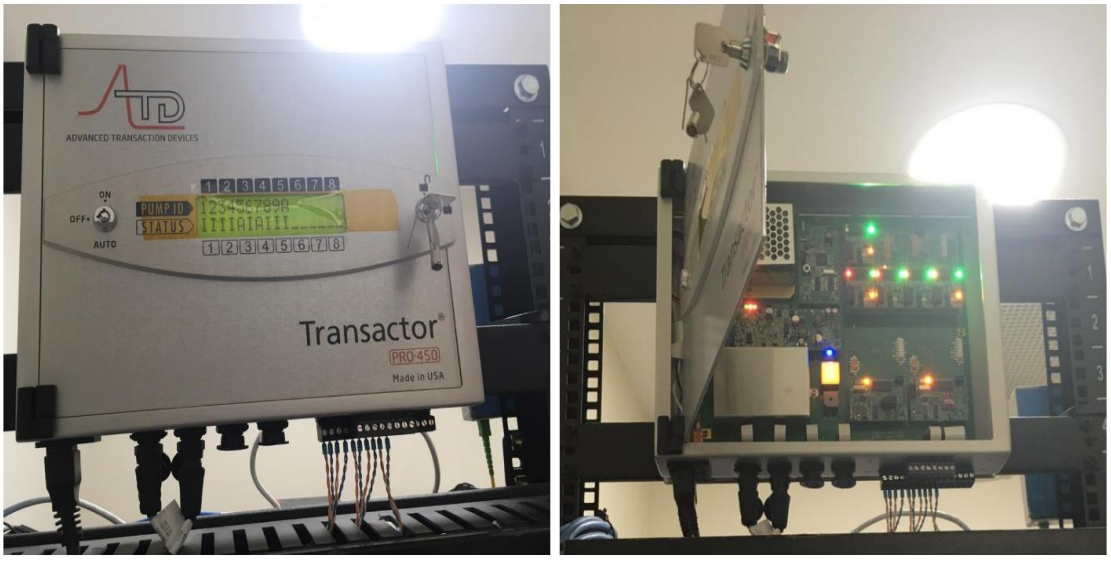
El controlador industrial (Transactor) es un dispositivo encargado de establecer una comunicación de bajo nivel con los surtidores o puntos de despacho, independientemente de la marca del dispensador, con el fin de transformar la información de bajo a alto nivel, la cual es mostrada en el display del Transactor además de, ser interpretada, permitiendo una comunicación fiable y relativamente inmediata, más no en tiempo real. La información que proviene de cada dispensador está representada por letras, las cuales, indican diferentes situaciones sobre la comunicación establecida como, por ejemplo:

- La letra “I”: está estable (estado correcto – pistolas en el puesto)
- La letra “A”: Autorizar Despacho (se levantó la pistola)
- La letra “C”: Venta completada (si el lado se quedó en C el sistema no procesa la venta)
- La letra “X”: No hay conexión con el surtidor.
- La letra p: Se está despachando combustible.

El Transactor permite una comunicación con todos los puntos de venta al mismo tiempo, además de que su velocidad es 20 veces mayor con respecto a los diferentes controladores de otras marcas existentes. Este dispositivo posee una estructura modular que facilita la sustitución de sus componentes excluyendo el uso de elementos adicionales; existen dos tipos de controladores como el PRO-450, que está diseñado para instituciones encargadas de la venta de combustible que posean hasta 16 surtidores, y el PRO-100 que se enfoca a organizaciones de un único surtidor.

Figura 21.

Transactor



Raspberry Pi

Principios fundamentales de Raspberry como dispositivo de almacenamiento y control

La Raspberry Pi es una placa de ordenador de bajo costo, pequeña y portátil. Se puede utilizar para conectar el monitor del ordenador o la televisión, el teclado, el ratón, el pendrive, etc. La Raspberry Pi lleva incorporado un software como Scratch que permite a los usuarios programar y diseñar animaciones, juegos o vídeos interesantes. Además, los programadores también pueden desarrollar

scripts o programas utilizando el lenguaje Python, que es el lenguaje principal del sistema operativo Raspbian. Raspberry Pi B+ es la evolución del modelo B. El lenguaje Python se ha utilizado en este trabajo para escribir el script para la comunicación cliente/servidor.

Además, hay mejoras como la adición de más PIN de cabecera GPIO, más puertos USB, menor consumo de energía, etc. Se recomienda utilizar el modelo B+ para el aprendizaje en la escuela porque ofrece más flexibilidad que el modelo A, especialmente para proyectos embebidos, y requiere un bajo consumo de energía, además de proporcionar más puertos USB en comparación con el modelo B. (Gareth, 2018).

Raspberry como tecnología IoT (Internet de las cosas)

Hay miles de sensores en una industria con diferentes usos, como transmisores de presión, medidores de flujo, transmisores de temperatura, transmisores de nivel, etc. Las redes cableadas se utilizan principalmente para transferir datos a la estación base mediante la conexión de sensores. Esto supone una ventaja, ya que proporciona un sistema de comunicación fiable y estable para los instrumentos y controles. Sin embargo, el coste de los cables necesarios es muy elevado.

En el modelo cliente/servidor, el servidor de archivos actúa como un nodo padre que permite que varios nodos hijos se conecten a él. Es responsable del almacenamiento central y de la gestión de los datos para que otros ordenadores puedan acceder al archivo en la misma red. Wai Zhao et al en (2015) logró demostrar el uso de Raspberry Pi como un servidor en el que varios ordenadores portátiles están conectados a él para copiar, almacenar y eliminar archivos a través de la red. A continuación, se muestra en la Tabla 2, los diferentes modelos de Raspberry existentes con sus respectivas características.

Tabla 2.

Modelos de Raspberry

	Zero 2 W	**3B+**	**Zero WH**	**4B**
Procesador	Cortex-A53 64-bit	Cortex-A53 64-bit	ARM1176JZF-S	Cortex-A72 (ARM v8) 64-bit
Número de Núcleo	4	4	1	4
RAM	512MB	1GB DDR2	512MB	1 GB, 2 GB, 4 GB, 8GB LPDDR4
Velocidad del CPU	1 GHz	1.4 GHz	1 GHz	1.5 GHz
Wi-Fi	802.11n	2.4GHz and 5GHz 802.11 b/g/n/ac	802.11n	2.4GHz and 5GHz 802.11 b/g/n/ac
Bluetooth	4.2, BLE	4.2, BLE	4.1 BLE	5.0

Comunicación serial

En telecomunicaciones e informática, la comunicación serial o las comunicaciones en serie son el proceso de envío de datos de un bit a la vez, de forma secuencial, a través de un canal de comunicaciones o bus informático. Esto contrasta con las comunicaciones en paralelo, en las que todos los bits de cada proceso se envían juntos. Las comunicaciones se utilizan en todas las líneas de larga distancia y en la mayoría de las redes informáticas, donde el coste del cable y las dificultades de sincronización hacen que las comunicaciones en paralelo sean poco prácticos. Los buses informáticos en serie son cada vez más comunes, ya que permite transferir datos a mayor velocidad de manera sincronizada.

UART (Universal Asynchronous Receiver-Transmitter)

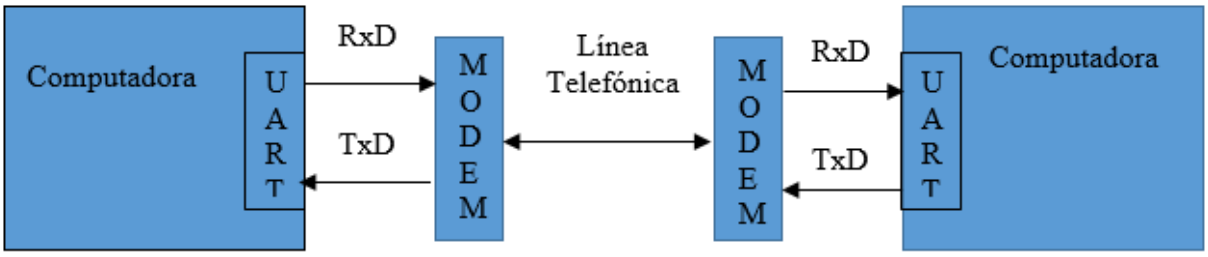
Un UART (Receptor-Transmisor Asíncrono Universal) es el microchip programado que controla la interfaz de un ordenador con sus dispositivos en serie conectados, el mismo se representa en la Figura 22. En concreto, proporciona al ordenador la interfaz RS-232C de equipo terminal de datos (DTE) para

que pueda comunicarse e intercambiar datos con módems y otros dispositivos en serie. Como parte de esta interfaz, León, F. (2021), explica que un UART también:

- Convierte los bits que recibe del ordenador a lo largo de los circuitos paralelos en un único flujo de bits en serie para la transmisión de salida.
- En la transmisión de entrada, convierte el flujo de bits en serie en bytes que pueden ser interpretados por el ordenador
- Añade un bit de paridad (si ha sido seleccionado) en las transmisiones salientes y comprueba la paridad de los bytes entrantes y descarta el bit de paridad
- Añade delineadores de inicio y de parada en las transmisiones de salida y los elimina de la entrada
- Maneja las interrupciones del teclado y del ratón (que son dispositivos en serie con puertos especiales)
- Puede manejar otro tipo de interrupciones y gestión de dispositivos que requieran coordinar la velocidad de funcionamiento del ordenador con la de los dispositivos.

Figura 22.

Comunicación Serial mediante UART



.

Comunicación de dispositivos por medio de aplicaciones

Python como lenguaje de propósito general

Python es un lenguaje de programación interpretado de alto nivel y de propósito general, la filosofía de diseño de este lenguaje se enfatiza en la legibilidad del código con el uso de una sangría significativa. Sus construcciones de lenguaje, así como su enfoque orientado a objetos, tiene como objetivo simplificar y redactar la escritura de código para los programadores, siendo claro y lógico, en proyectos de cualquier ámbito (Challenger et al., 2014).

Python resulta ser sorprendentemente versátil en el contexto de lograr integrar aplicaciones con controladores a nivel de industria, ya sea para crear un prototipo de una sola cinta transportadora o para domar un edificio lleno de robots. Python se puede usar para interactuar con dos dispositivos comunes de automatización industrial: un escáner de código de barras y un controlador lógico programable (PLC). Un ejemplo es lo que se ha logrado en el artículo definido por Montgomery & Nakroshis en 2019, usar Python para tomar una base de datos del PLC y traducirla a un montón de archivos EXCEL. Lo cual, a manera de consideración, es mejor trabajar en algo que no sea la base de datos de trabajo para conseguir esas traducciones correctas o sensatas.

Protocolos de comunicación entre Raspberry Pi y dispositivos

El Raspberry Pi puede ser usado como un servidor web ya que, esté tiene la capacidad de brindar un ambiente liviano, cuando pocos usuarios pretenden acceder al mismo. Entre sus características, el Raspberry Pi a pesar de, relativamente con otros ordenadores, contar con poca memoria y poco poder de procesamiento, el bajo consumo de energía y su modo de ejecución, lo hacen un componente ideal para servir en sistemas dentro de una red local o incluso fuera de internet. Al lograr usarse como un servidor web este funciona de igual manera que otro servidor LAMP (Linux, Apache, MySQL y PHP). En este esquema Linux es el encargado de proveer el sistema operativo o la distribución de este, Apache es

el servidor web, MySQL funciona para el manejo de base de datos y PHP se trata de un lenguaje de scripting para páginas dinámicas.

Socket IO

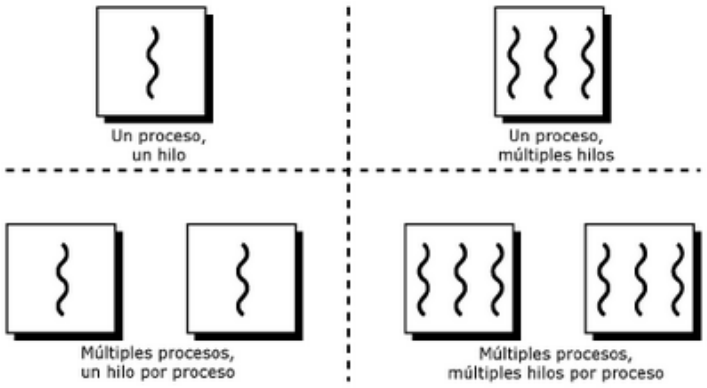
Se asigna el termino socket a una interfaz o mecanismo compuesto de datos que permiten que exista un proceso de emisión y recepción de información, lo cual lleva a una intercomunicación entre procesos sin importar que estos se encuentren en el mismo hardware, es la manera que dos programas distintos se trasfieran datos entre sí, hablando en lenguaje de programación un socket puede ser denominado también como un “fichero” que debe ser desplegado de una forma determinada (Martín Santana & Martínez Santana, 2007) . Mientras que el Socket IO se considera un instrumento residente en una web homóloga y su objetivo principal es optimizar y dar viabilidad al empleo de websockets, los mismos que deben brindar funcionalidades más avanzadas como namespaces (Programación Web, 2022).

Threads

La palabra thread podría ser definida como una aglomeración de un pedazo de programa unido a ciertos registros determinados que tiene un procesador (RasterSoft, 2022). Se conoce también como la característica encargada de que un programa o aplicación pueda ejecutar varias tareas al mismo tiempo. Threads constituye un concepto más amplio y global de lo que se conoce como multitarea, lo cual hace referencia a que una aplicación se ejecute continuamente, pero instantáneamente surgió la necesidad de que una misma aplicación sea capaz de realizar varios procesos a la vez (Microcontroladores, 2022). Como se observa en la Figura 23.

Figura 23.

Ilustración del funcionamiento de los threads



Nota. (Microcontroladores, 2022)

Bases de datos no relacionales

Las bases de datos representan un papel primordial en el cual se hallan introducidos sistemas informáticos, abriendo camino hacia el almacenamiento de enormes conjuntos de datos referentes a organizaciones y a consumidores, de forma que la información guardada posibilite la gestión de datos. Previamente a la existencia de las bases SQL (Structured Query Language) y No SQL (“Not Only SQL”), los datos se almacenaban en registros fijos, debido a que, la información era supervisada a través de un entorno de procesamiento tradicional, en donde existía la necesidad de estructurar una diversificación de aplicaciones, para la obtención de registros, así como para la anexión a los documentos proporcionados.

Las bases de datos son una asociación de elementos constituidas en registros, tales que, son unificados en un entorno informático asequible por medio de un computador. Con el fin de obtener una idea más clara y específica acerca de su funcionamiento y su contenido, es indispensable diferenciar la

concepción de dato e información, ya que, estos compendios son fundamentales para el diseño y estructuración de las bases de datos. Un dato es una incorporación de caracteres con una connotación específica, dentro de estos se encuentran los alfanuméricos, alfabéticos o numéricos, y es concebido como la unidad más ínfima de información.

Estructura de una Base de Datos No SQL

NOSQL término utilizado a finales de la década de los 90s, con su significado de no únicamente SQL, ya que no seguiría los mismos patrones que el SQL, sin embargo, con el mismo modelo relacional. En el año 2009, se retomó la expresión NOSQL con el fin de promover el nacimiento de un nuevo sistema de almacenamiento de datos distribuidos no relacionales. Por lo tanto, es una estructura que provee un dispositivo en el cual se puede almacenar y recuperar datos dentro de varios medios y diferentes a los modelos relaciones. Una gran parte de las bases de datos NOSQL comparten varias características, obviamente, ya que el termino abarca muchas cosas, dentro de estas descripciones pueden existir ciertas excepciones. Para mejorar nuestra concepción sobre las bases de datos NOSQL a continuación algunas de sus características:

- Flexibilidad: Estos mecanismos han sido modificados para que puedan trabajar en clusters, con el objetivo de ofrecer una mejora capacidad en una mayor cantidad de equipos y también disminuyendo los costos y los recursos.
- Distribuidas: Son sistemas los cuales están conformados por diferentes mecanismos que se ayudan entre sí para proveer información a los usuarios.
- Propiedades Base: Están basadas en un estándar denominado consistencia eventual, permitiendo la lectura de información pasada en favor de optimizar su utilidad. Refiriéndose a clientes encaminados a la satisfacción del usuario.

- Diseñado para volúmenes mayores: Gran cantidad de sistemas NOSQL se crearon con el fin de ofrecer un procesamiento y un almacenamiento ágiles con respecto a números exorbitantes de datos.
- Datos no estructurados: Los datos tienen la facilidad de que se pueden ingresar en lugares y momentos diferentes. Sin la necesidad de detallar un diseño anticipado. Tienen varios atributos en común, y debido a esto se omiten valores nulos dentro de los atributos.

A continuación, se muestra en la Tabla 3, las diferencias entre las bases de datos relacionales (SQL) y las bases de datos no relacionales (No SQL).

Tabla 3.

Comparación Bases de Datos

Bases de Datos SQL	Bases de Datos No SQL
Baja Escalabilidad	Alta Escalabilidad
Alta Fiabilidad	Baja Fiabilidad
Utilización de propiedades ACID	Utilización de propiedades BASE
Costo alto en la implementación	Costo moderado en la implementación
Seguridad Alta	Seguridad muy baja
Bajo Rendimiento	Alto Rendimiento
Velocidad baja en el procesamiento de datos	Velocidad alta en el procesamiento de datos

El objetivo de las bases de datos No SQL no es eliminar ni sustituir a las bases de datos relacionales (SQL), más bien tiene el objetivo de servir de alternativa en el desarrollo de aplicaciones en el cuál existan requerimientos en donde se vaya a manejar grandes tamaños de datos y la escalabilidad y procesamiento sean sumamente importantes. Sin embargo, se conoce que las bases de datos se las

utiliza para almacenar información, también cuentan con un límite de almacenamiento y registros máximos, esto no se debe a que ya ha llegado a su límite sino a que los tiempos de espera se han incrementado al punto de tomarlos en cuenta. Las bases de datos No SQL en cambio nos ofrecen mayor capacidad de almacenar datos hasta Terabytes, pero sacrifica el ordenamiento de estos.

Metodología de investigación

Para el proyecto de investigación se utilizó un conjunto de procesos basados en el método científico relacionado directamente al IoT (Internet of Things), adquiriendo los conocimientos requeridos y volviendo posible su aplicación hoy en día. En base a la serie de problemas que se presentan en las estaciones de servicio, ya sea respecto al costo o a la cantidad de gasolina que se despacha, la idea es que no quede brecha para que se pueda producir un mal servicio al consumidor.

Marco Metodológico

Para obtener un modelo de arquitectura IoT adecuado, se realizó un análisis de artículos científicos los cuales se encuentran disponibles en sitios web académicos y científicos. Refiriéndose al desarrollo de software bajo las tecnologías IoT, principalmente relacionados con base de datos no relacionales y conceptos de tiempo real. Se excluyeron artículos cortos, incompletos, ponencias, resúmenes de presentaciones entorno a la temática de requisitos.

Preguntas de investigación

En esta sección se planteó encontrar un modelo de arquitectura IoT que solvente las necesidades del sistema de gestión de combustible en la estación “ECO EL OASIS”, para lo cual, se establecieron las siguientes preguntas:

- ¿Existen modelos de arquitectura IoT bien definidos que puedan interactuar simultáneamente con un desarrollo de software enfocado en conceptos de tiempo real?

- ¿Cómo se gestiona el intercambio de información masivo entre dispositivos en un modelo de arquitectura IoT?

Selección de artículos

Una vez que se definió el objetivo del análisis, se eligieron los siguientes artículos científicos que fueron revisados en base a las preguntas detalladas anteriormente:

The Internet of Things: Insights into the building blocks, component interactions, and architecture layers

En este artículo, los autores tratan la tecnología IoT como prometedora y un pilar indispensable para el futuro de los sistemas inteligentes. Para entender los conceptos de IoT este documento plantea el estudio de conocimientos de cuatro bloques de construcción con IoT (cosas, dispositivos, red e infraestructura en la nube), tres componentes principales (“cosas” con sensores, almacenamiento de información masiva y de datos) junto con una capa de la arquitectura (tres capas, cinco capas y siete capas). Se presenta una interacción entre todos estos componentes de IoT integrados con cada etapa de una manera sistemática y detallada (Kumar & Mallick, 2018).

Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges

En este artículo se detalla como el internet está cambiando y evolucionando continuamente. Se hace énfasis en que la principal forma de comunicación de la Internet actual es la humana. El internet de las cosas (IoT) se considera como la evolución del internet, que realiza el aprendizaje de máquina a máquina (M2M). Así, el IoT proporciona conectividad para todos y para todo (Khan et al., 2012).

El IoT incorpora cierta inteligencia en los objetos conectados a la red para que se comuniquen, intercambien información, tomen decisiones, invoquen acciones y proporcionen servicios sorprendentes. Este documento aborda las tendencias de desarrollo existentes, la arquitectura genérica

de IoT, sus características distintivas y las posibles aplicaciones futuras. Se prevé los principales retos asociados al desarrollo de la IO, y se detalla como el IoT está adquiriendo una creciente popularidad para el mundo académico, la industria y el gobierno que tiene el potencial de aportar importantes beneficios personales, profesionales y beneficios económicos.

Arquitectura de análisis de datos generados por el internet de las cosas IoT en tiempo real

En este artículo se detalla la manera en que el internet de las cosas (IoT) está revolucionando la forma en que se percibe todo con lo que el ser humano puede interactuar en el mundo físico. Se detalla la gran cantidad de objetos que en la actualidad se encuentran interconectados y son capaces de brindar funciones automatizadas. Con este análisis se procede a detallar las metodologías y técnicas tradicionales que presentan fallos para la gestión de datos que se recolectan constantemente (Sarabia, 2020).

Este apartado se lo trata como BigData y se analiza que la integración con IoT, tanto como sus metodologías, técnicas y soluciones para estos entornos, lo que afecta directamente a todo el potencial que podría aportar la tecnología IoT. Se plantea una gestión adecuada de BigData dentro de estos entornos, para lo cual se establece una arquitectura diseñada utilizando requerimientos estructuradas por el Instituto Nacional de Estándares y Tecnologías (NIST) para la implementación de ecosistemas IoT y la interoperabilidad de frameworks de BigData.

Construcción de cadena de búsqueda

De los artículos previamente estudiados se extraen palabras clave para establecer cadenas de búsqueda

Tabla 4.

Palabras Clave

Título	Palabras Clave
The Internet of Things: Insights into the building blocks, component interactions, and architecture layers	Internet of Things, architecture, architecture layers IoT, components and adaptability of IoT
Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges	Evolution of Internet, IoT applications, IoT model and layers, connectivity of an IoT architecture
Arquitectura de análisis de datos generados por el internet de las cosas IoT en tiempo real	IoT arquitectura, análisis de capas IoT, arquitectura IoT con software en tiempo real, integración de un modelo IoT

Estas palabras se concatenarán con la ayuda de operadores lógicos (and, or) para buscar artículos técnicos. Estas cadenas se pueden aplicar en Google Scholar, Researchgate o ACM Digital Library, tomando en cuenta los criterios de inclusión y exclusión.

- (Internet of things) and architecture.
- (Internet of things) or (implementation of internet of things) and architecture
- (Internet of things) or (internet of things model) and architecture
- (Internet of things) or (internet of things model) and architecture
- (Internet of things) and (internet of things model)
- (Internet of things) or (implementation of internet of things)

- (Internet of things) and (implementation of internet of things)

Conclusión del análisis de artículos

Después de haber realizado un análisis de artículos enfocado en modelos y arquitecturas de IoT, se llegó al resultado de que esta nueva tecnología no ha sido explorada ni explotada en su totalidad , por lo cual, se determinó que el articulo con mayor definición es “The Internet of Things: Insights into the building blocks, component interactions, and architecture layers” dado que establece conceptos básicos de IoT, además, brinda tablas comparativas entre modelos de tres, cinco y siete capas, las cuales se pueden comparar detenidamente en el siguiente análisis específico entre modelos.

El análisis realizado muestra que, si existen modelos IoT bien definidos, con características, componentes y campo de aplicación distintos entre ellos, sin embargo, no todos están enfocados a conceptos de tiempo real y tampoco son capaces de complementarse simultáneamente en un mismo software, debido al número y forma de interacción entre sus capas.

El manejo de información masiva es una característica que solo con el modelo de siete capas es posible de abarcar, ya que, entre la capa tres y la capa 4 cuatro se gestiona el intercambio de información, por medio de protocolos UART, que permiten el envío de datos desde uno o varios dispositivos hasta una unidad de persistencia de forma constante y asíncrona, sin que esta se vea alterada o interrumpida por otros procesos.

Análisis comparativo entre modelos IoT

IoT como modelo de arquitectura tiene la característica de interactuar sin intervención humana, las tecnologías de IoT en la actualidad están en una fase experimental; sin embargo, se han producido muchos avances en la integración de objetos con sensores en Internet. El desarrollo de IoT implica muchas cuestiones, como la infraestructura, las comunicaciones, las interfaces, los protocolos y los estándares. Para el desarrollo del proyecto en la estación de servicios “ECO EL OASIS”, se eligió un modelo capaz de satisfacer las necesidades del proceso de gestión de combustible, se consideró el modelo de 3, 5, y 7 que son mostrados en las Figuras 24, 25 y 26.

Figura 24.

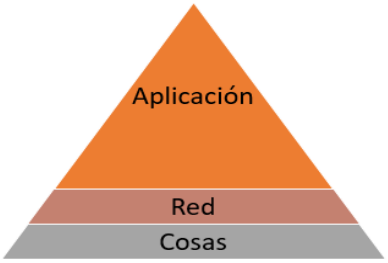
Modelo IoT de 7 capas o Modelo IoTWF

Figura 25.

Modelo IoT de 5 capas o Modelo IoT Simple

Figura 26.

Modelo IoT de 3 capas o Modelo Barton



Una arquitectura que describe una solución IoT incluye aspectos físicos (cosas u objetos) y aspectos virtuales (servicios y protocolos de comunicación). “Al momento de adoptar una arquitectura con múltiples niveles o capas, se debe considerar la complejidad y los aspectos más importantes que se integran dentro de la aplicación. Este enfoque modular ayuda a gestionar la complejidad la solución IoT” (Kumar & Mallick, 2018).

Tabla 5.

Tabla comparativa entre modelos de arquitectura IoT

Componente De Aplicación IoT / N° Capas	3	5	7
Base de datos NoSQL	✓	✓	✓
Integración aplicación Python	✓	✓	✓
Integración aplicación Angular	✓	✓	✓
Monitoreo en tiempo real	×	×	✓
Conectividad Raspberry	✓	✓	✓
Almacenamiento cloud	×	×	✓
Obtención de información de "cosas"	✓	✓	✓
Minería de datos	×	×	✓
Integración y consumo de servicios	×	✓	✓

Nota. (Kumar & Mallick, 2018)

Como resultado de la Tabla 5 y según la Figura 26, para el diseño de arquitectura IoT del proyecto se utilizó un modelo de 7 capas, en beneficio de los componentes, prestaciones, funciones y alto nivel de detalle en cada etapa del modelo, a la vez que estos presentan un buen desenvolvimiento con intercambios masivos de información usando conceptos de tiempo real, a continuación, se especifica cada una de las capas y la relación que tiene con cada dispositivo o software que integró parte del diseño:

Capa 1: Cosas

En esta capa se encuentran todos los dispositivos que recogen y detectan toda la información de la estación de servicio, específicamente el simulador Tokheim, el cual se encarga de emular el funcionamiento de un surtidor de combustible, enviando información referente a precio, tipo de gasolina, cantidad, información de descargas, despachador, entre más datos que dependiendo de lo que se solicite en otra capa del modelo este dispositivo se encarga de enviarla.

Capa 2: Conectividad

En esta capa del modelo se encuentra todo el proceso y componentes que ayudaron a transferir la información detectada por un dispositivo disponible o equipado en la capa de previa a la capa de infraestructura global y viceversa a través de redes como la inalámbrica y LAN. Para que todo esto pueda ser posible dentro de la estación de servicio se usó la misma red del establecimiento, la cual estuvo conectada directamente con un Raspberry Pi, el mismo que se encargó además de tener una conexión serial con el controlador industrial (Transactor).

Capa 3: Infraestructura global

En esta capa se crean y albergan las tecnologías orientadas a servicios, todo lo referente a cloud computing, los datos de licitación, además de la computación cognitiva, lo cual se llevó a cabo y controlador de su correcto funcionamiento por el servicio que ofrece Google con su base de datos no relacional, firebase, la misma que alberga grandes cantidades de información en la nube, con la capacidad de tenerla disponible en cualquier momento sea para obtener, guardar o actualizar cualquier registro que sea recibido desde el controlador industrial, sin antes haber pasado por un proceso de parseo que se ha desarrollado en capas posteriores.

Capa 4: Almacenamiento de información

Esta capa del modelo se encarga de detectar o recoger la información de los objetos que se enfocan en la capa de infraestructura global. Aquí se empleó un sistema de monitorización de datos que permiten albergar y manejar todo tipo de información, el mismo que al igual que en la capa anterior está encargado de ser controlado por Firebase, que con la ayuda de una aplicación diseñada en lenguaje Python que parsea la información enviada por el controlador industrial y se encarga de gestionar la misma en el motor de base de datos.

Capa 5: Análisis de información

En esta capa la información albergada por etapas previas del modelo es usada para mostrar e interpretar datos que son enviados desde el simulador, con el fin de usar esta información para generar funciones estadísticas, gráficas, tablas, respecto al proceso de envío desde la capa 1 hasta la interacción con el negocio.

Capa 6: Aplicación

En esta capa se recolecta la información sobre las distintas tareas que han sido requeridas por la estación de servicio, las cuales son presentadas en un aplicativo web diseñado en angular, el cual consume el mismo motor de base de datos Firebase de la capa de 3 y 4, para su interpretación y futuras tomas de decisiones respecto al negocio.

Capa 7: Personas y procesos

La última capa del modelo brinda al usuario la potestad de evaluar el resultado del sistema IoT analizando especialmente el modelo del negocio y en beneficio de una mejor toma de decisiones en un futuro, permitiendo a la estación innovar y avanzar en términos tecnológicos, de tiempo y en la forma de prestar el servicio de despacho de combustible en un futuro, con componentes de menor costo y mayor velocidad.

Capítulo III

Diseño de Modelo de Solución

En este capítulo se establece uno de los pasos fundamentales para empezar con el desarrollo del prototipo propuesto, el cual, es la especificación de requerimientos, en este apartado se estructuraron dichas especificaciones partiendo de los pedidos de la estación de servicio “ECO EL OASIS”, además del diseño de los modelos de casos de uso, que ayudaron como guía y constancia en el documento de la ejecución del proyecto, basados en los protocolos y metodologías dispuestas para los dispositivos de los cuales se hicieron uso como el Transactor, Raspberry, Simulador y el medidor de tanques.

Problemática y modelo de solución

Línea base y procesamiento estadístico del sistema de gestión y control actualmente usado por la estación

En la actualidad la estación de servicio ECO El OASIS cuenta con un sistema de gestión y control de combustible, el cual, se encarga de llevar la información enviada por un surtidor al momento que despacha combustible para un auto, esta es llevada hacía un controlador industrial (Transactor Pro 450), el cual, transforma datos de bajo nivel entregados por el dispensador a información de alto nivel, los cuales, son entregados a un servidor ubicado dentro de las instalaciones de la estación. En lo que va del año 2020 se realizó una consulta SQL a la base de datos de la estación y se encontró que se realizaron 312429 transacciones desde el primero de enero de 2020 hasta el 31 de diciembre de 2020, entre las mismas 52919 son de combustible “DIESEL PREMIUM”, 22902 son de “SUPER” y 236608 son de “EXTRA”. A continuación, en la Tabla 6 y en Figura 27, se detallan los datos de la consulta.

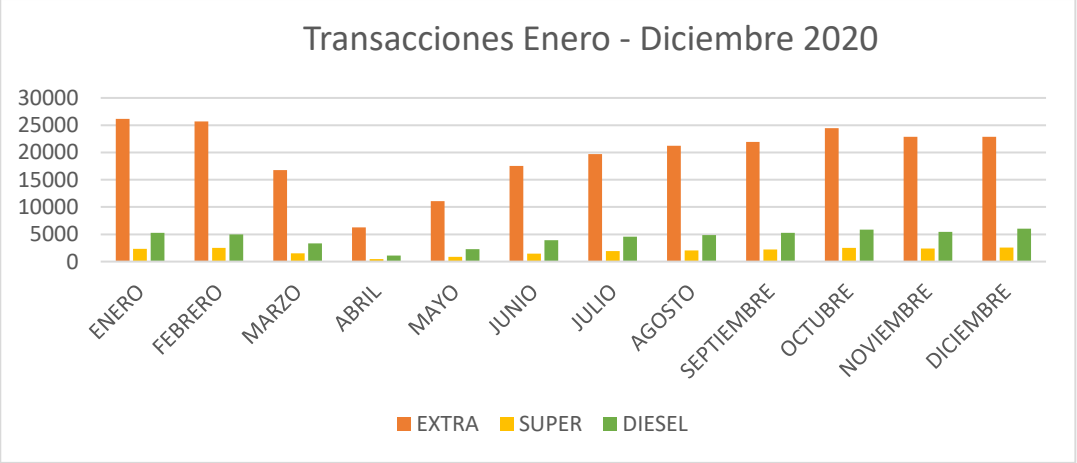
Tabla 6.

Transacciones del año 2020

2020	EXTRA	SUPER	DIESEL PREMIUM
ENERO	26164	2340	5292
FEBRERO	25675	2500	4987
MARZO	16760	1501	3316
ABRIL	6266	488	1124
MAYO	11098	879	2282
JUNIO	17547	1465	3946
JULIO	19700	1952	4566
AGOSTO	21256	2072	4837
SEPTIEMBRE	21942	2248	5252
OCTUBRE	24450	2508	5871
NOVIEMBRE	22875	2368	5440
DICIEMBRE	22875	2581	6006
	236608	22902	52919

Figura 27.

Gráfica de ventas del año 2020



Especificación de requerimientos Software

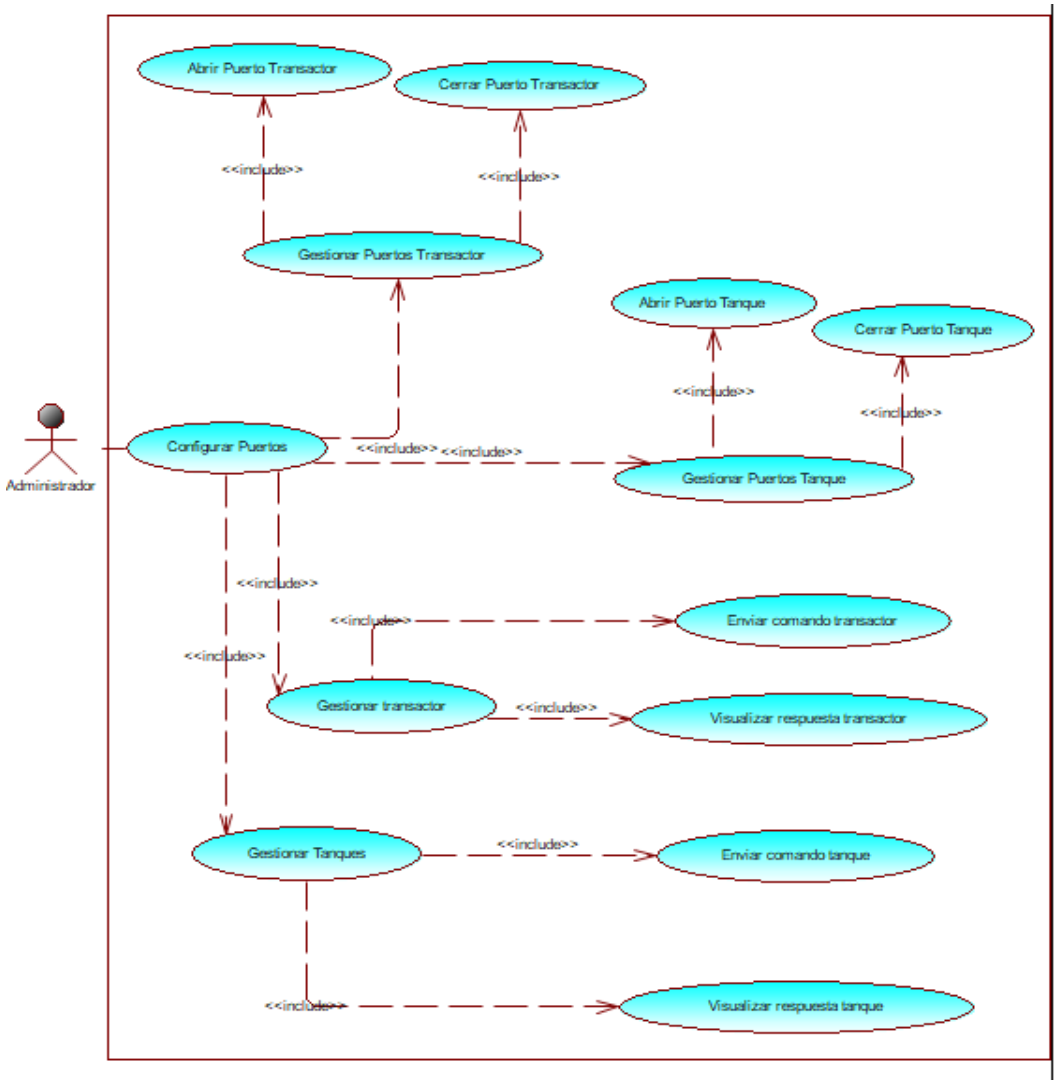
En esta sección se visualiza la especificación de requisitos, que tuvo como objetivo mostrar una representación de los procesos y del funcionamiento del sistema, además de la participación e interacción de los actores con el mismo. En este caso, el sistema se encuentra dividido en dos partes, la primera, la cual está instalada localmente en la estación de servicio “ECO EL OASIS”, y la segunda, la cual, la cual permite visualizar toda la información de los dispositivos tanto Transactor como el medidor de tanques, a través de una aplicación web.

Modelo de Casos de Uso Aplicación Local

El sistema permite el ingreso del administrador a la pantalla de inicio, la cual se dirige a la pantalla de configuración de puertos, donde puede gestionar los baudios y puertos para el Transactor o para el sistema de tanques, donde tiene las opciones de abrir o cerrarlos, una vez realizado el proceso de apertura del puerto puede entrar a las ventanas de comandos de cualquiera de los dispositivos donde tiene la capacidad de enviar comandos, una vez que el controlador o sistemas de tanques responda se visualizó sea cual sea la información que se envíe mediante el puerto serial. A continuación, se muestra en la Figura 28, el diagrama de casos de uso general para la aplicación de escritorio diseñada en Python.

Figura 28.

Diagrama de caso de uso general aplicación local



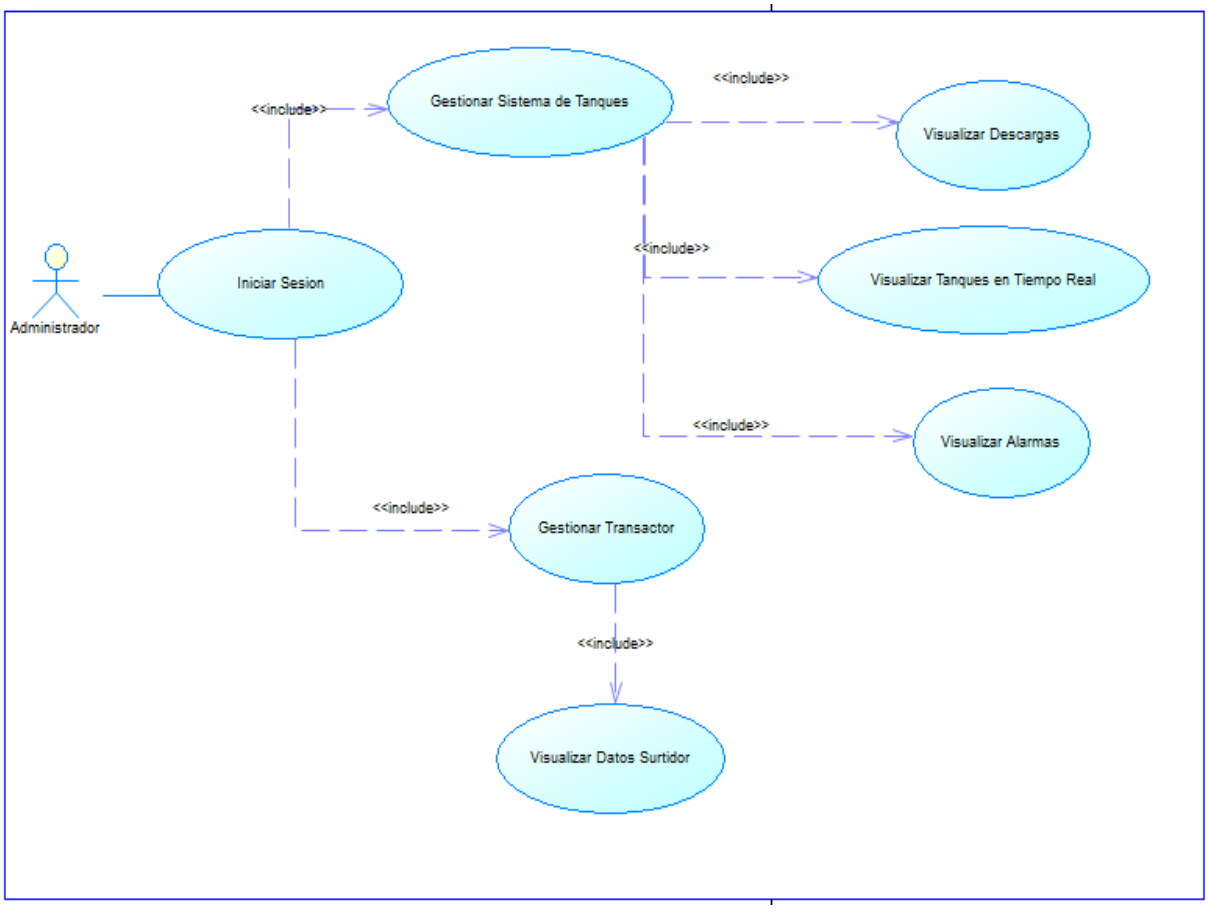
Modelo de Casos de Uso Aplicación Web

El sistema permite el ingreso del administrador por medio de la validación del nombre de usuario y la contraseña, una vez validados los datos, la aplicación da acceso al uso de las funciones y a la información que se obtiene a través de consultas hacia la base de datos, dando como resultado la

visualización de los datos en tiempo real ya sea del sistema de tanques o del dispositivo Transactor. A continuación, se detalla en la Figura 29, el diagrama de casos de uso de la aplicación web desarrollada en Angular.

Figura 29.

Diagrama de caso de uso general aplicación web



Requerimientos Funcionales

Para el sistema “AdvancedOne” que se encuentra localmente en la estación de servicio “Eco El Oasis” se han identificado los siguientes requerimientos funcionales.

Tabla 7.

RFL-01 – Abrir puerto para el Transactor.

Id. Requerimiento	**RFL -01**	
Actores	Administrador	
Descripción	Abrir una conexión serial entre un puerto del Raspberry y el controlador industrial (Transactor)	
Entradas	• Baudios • Nombre del puerto • Bit de paridad • Tiempo de conexión	
Salidas		
Proceso	**Actor** 1. Solicita abrir puerto 3. Selecciona el puerto adecuado 4. Selecciona el número de baudios respectivo (115200) 5. Confirma los datos ingresados 6. Presiona el botón de abrir puerto	**Sistema** 2. Despliega formulario con los siguientes campos: “Puerto”, “Baudios”. 7. Verifica que el puerto esté disponible y baudios sean los adecuados. 8. Abre el puerto
Precondiciones	Haber ingresado a la opción de configuración de puerto	
Post condiciones	El administrador puede ingresar a la opción de comandos de Transactor	
Flujo alterno	Línea 6.1: Si el puerto o los baudios son incorrectos no se abrirá el puerto	
Prioridad	Primordial	

Tabla 8.

RFL -02 – Cerrar puerto para el Transactor.

Id. Requerimiento	**RFL -02**	
Actores	Administrador	
Descripción	Cerrar una conexión serial existente entre un puerto del Raspberry y el controlador industrial (Transactor)	
Entradas		
Salidas		
Proceso		**Sistema**
	Actor 1. Solicita cerrar puerto	2. Despliega formulario con los campos: “Puerto”, “Baudios” que se encuentran conectados en ese momento.
	3. Presiona el botón de cerrar puerto	4. Verifica que el puerto esté abierto 5. Cierra el puerto
Precondiciones	Haber ingresado a la opción de configuración de puerto Tener una conexión serial en progreso en un puerto abierto	
Post condiciones	El usuario puede volver a establecer una nueva conexión en ese puerto	
Flujo alterno		
Prioridad	Primordial	

Tabla 9.

RFL -03 – Enviar comandos en la consola del Transactor.

Id. Requerimiento	**RFL -03**
Actores	Administrador
Descripción	Enviar un comando mediante una conexión serial entre el Raspberry y el controlador industrial (Transactor)
Entradas	• Comando

Salidas	• Respuesta del Transactor	
		Sistema
	Actor	
Proceso	1. Escribe un comando en la consola 2. Presiona el botón enviar	3. Envía el comando desde el dispositivo Raspberry hacía el Transactor 4. Obtiene recibe una respuesta en lenguaje de bajo nivel 5. Transforma la respuesta recibida a un lenguaje legible 6. Presenta la respuesta en la consola
Precondiciones	Haber ingresado a la opción de consola de Transactor Conocer un comando del dispositivo	
Post condiciones	El administrador puede obtener información del Transactor	
Flujo alterno	Línea 3.1: El Transactor no entiende el comando presentado y no envía ninguna información de respuesta	
Prioridad	Primordial	

Tabla 10.

RFL F-04 – Abrir puerto para el sistema de tanques.

Id. Requerimiento	**RFL -04**
Actores	Administrador
Descripción	Abrir una conexión serial entre un puerto del Raspberry y el sistema de tanques
Entradas	• Baudios • Nombre del puerto • Bit de paridad • Tiempo de conexión

Salidas		
Proceso	**Actor** 1. Solicita abrir puerto 3. Selecciona el puerto adecuado 4. Selecciona el número de baudios respectivo (9600) 5. Confirma los datos ingresados 6. Presiona el botón de abrir puerto	**Sistema** 2. Despliega formulario con los siguientes campos: “Puerto”, “Baudios”. 7. Verifica que el puerto esté disponible y baudios sean los adecuados. 8. Abre el puerto
Precondiciones	Haber ingresado a la opción de configuración de puerto	
Post condiciones	El administrador puede ingresar a la opción de comandos de la consola de tanques	
Flujo alterno	Línea 6.1: Si el puerto o los baudios son incorrectos no se abrirá el puerto	
Prioridad	Primordial	

Tabla 11.

RFL -05 – Cerrar puerto para el sistema de tanques.

Id. Requerimiento	**RFL -05**
Actores	Administrador
Descripción	Cerrar una conexión serial existente entre un puerto del Raspberry y sistema de tanques
Entradas	
Salidas	

		Sistema
Proceso	**Actor** 1. Solicita cerrar puerto	2. Despliega formulario con los campos: “Puerto”, “Baudios” que se encuentran conectados en ese momento.
	3. Presiona el botón de cerrar puerto	4. Verifica que el puerto esté abierto 5. Cierra el puerto
Precondiciones	Haber ingresado a la opción de configuración de puerto Tener una conexión serial en progreso en un puerto abierto	
Post condiciones	El usuario puede volver a establecer una nueva conexión en ese puerto	
Flujo alterno		
Prioridad	Primordial	

Tabla 12.

RFL -06 – Enviar comandos en la consola del sistema de tanques.

Id. Requerimiento	**RFL -06**	
Actores	Administrador	
Descripción	Enviar un comando mediante una conexión serial entre el Raspberry y el sistema de tanques	
Entradas	• Comando	
Salidas	• Respuesta del sistema de tanques	
	Actor	**Sistema**
Proceso	1. Escribe un comando en la consola 2. Presiona el botón enviar	3. Envía el comando desde el dispositivo Raspberry hacía el sistema de tanques 4. obtiene una respuesta en lenguaje de bajo nivel

		5. Transforma la respuesta recibida a un lenguaje legible 6. Presenta la respuesta en la consola
Precondiciones	Haber ingresado a la opción de consola del sistema de tanques Conocer un comando del dispositivo	
Post condiciones	El administrador puede obtener información del sistema de tanques	
Flujo alterno	Línea 3.1: El sistema de tanques no entiende el comando presentado y no envía ninguna información de respuesta	
Prioridad	Primordial	

Tabla 13.

RFL -07 – Enviar comandos de Alertas automáticamente, por periodos.

Id. Requerimiento	**RFL -07**	
Actores	Administrador	
Descripción	Enviar un comando de alerta mediante una conexión serial entre el Raspberry y el sistema de tanques	
Entradas	• Comando	
Salidas	• Respuesta del sistema de tanques hacia la Base de Datos	
	Actor	**Sistema**
Proceso	**1.** Enviar un comando de alerta automáticamente, en un periodo específico	2. Envía el comando desde el dispositivo Raspberry hacía el sistema de tanques 3. obtiene una respuesta en lenguaje de bajo nivel 4. Transforma la respuesta recibida a un lenguaje legible

	5. La envía hacia la Base de datos No SQL
Precondiciones	Haber abierto el puerto correctamente para el dispositivo de medición de tanques.
Post condiciones	El administrador puede obtener información del sistema de tanques
Flujo alterno	Línea 3.1: El sistema de tanques no entiende el comando presentado y no envía ninguna información de respuesta
Prioridad	Primordial

Tabla 14.

RFL -08 – Actualizar datos en tiempo real automáticamente, por periodos.

Id. Requerimiento	**RFL -08**	
Actores	Administrador	
Descripción	Enviar un comando que actualice los registros en una tabla en tiempo real mediante una conexión serial entre el Raspberry y el sistema de tanques	
Entradas	• Comando	
Salidas	• Respuesta del sistema de tanques en la tabla tiempo real en la Base de Datos	
	Actor	**Sistema**
Proceso	1. Enviar un comando automáticamente, en un periodo específico	2. Envía el comando desde el dispositivo Raspberry hacía el sistema de tanques 3. Obtiene una respuesta en lenguaje de bajo nivel

		4. Transforma la respuesta recibida a un lenguaje legible 5. La envía hacia la Base de datos No SQL 6. Guarda la información en la tabla “tiempo real”
Precondiciones	Haber abierto el puerto correctamente para el dispositivo de medición de tanques.	
Post condiciones	El administrador puede obtener información del sistema de tanques	
Flujo alterno	Línea 3.1: El sistema de tanques no entiende el comando presentado y no envía ninguna información de respuesta	
Prioridad	Primordial	

Tabla 15.

RFL -09 – Actualizar datos de las descargas automáticamente, por periodos establecidos.

Id. Requerimiento	**RFL -09**	
Actores	Administrador	
Descripción	Enviar un comando que actualice los registros en la tabla descargas mediante una conexión serial entre el Raspberry y el sistema de tanques	
Entradas	• Comando	
Salidas	• Respuesta del sistema de tanques en la tabla descargas en la Base de Datos	
	Actor	**Sistema**
Proceso	1. Enviar un comando automáticamente, en un periodo específico.	
		2. Envía el comando desde el dispositivo Raspberry hacía el sistema de tanques

		3. Obtiene una respuesta en lenguaje de bajo nivel 4. Transforma la respuesta recibida a un lenguaje legible 5. La envía hacia la Base de datos No SQL 6. Guarda la información en la tabla “descargas”
Precondiciones	Haber abierto el puerto correctamente para el dispositivo de medición de tanques.	
Post condiciones	El administrador puede obtener información acerca de las descargas en el sistema de tanques	
Flujo alterno	Línea 3.1: El sistema de tanques no entiende el comando presentado y no envía ninguna información de respuesta	
Prioridad	Primordial	

Requerimientos no funcionales

Para el sistema “AdvancedOne” que se encuentra localmente en la estación de servicio Eco El Oasis se han identificado los siguientes requerimientos no funcionales.

Tabla 16.

Rendimiento

Número	**RFLN 1**
Nombre	Tiempo de Respuesta
Tipo	**Requisito no funcional.**
Descripción	El sistema deberá ser capaz de enviar y recibir respuestas en tiempo real, dependiendo del dispositivo al cual se encuentre conectado el Raspberry.
Prioridad	**Alta.**

Tabla 17.

Seguridad

Número	**RFLN 2**
Nombre	Navegación
Tipo	**Requisito no funcional.**
Descripción	El sistema permitirá navegar entre las distintas opciones, configuraciones y consolas que se presenten únicamente en la aplicación, y se podrán hacer uso de estas solamente si la conexión serial y el dispositivo seleccionado es exitosa.
Prioridad	**Alta.**

Tabla 18.

Fiabilidad

Número	**RFLN 3**
Nombre	Información confiable
Tipo	**Requisito no funcional.**
Descripción	Cada vez que se ejecute un comando el aplicativo responderá con datos que el dispositivo envíe, con el objetivo final que no se pierda información durante el proceso.
Prioridad	**Alta.**

Tabla 19.

Disponibilidad

Número	**RFLN 4**
Nombre	Periodo de disponibilidad del sistema
Tipo	**Requisito no funcional.**
Descripción	La aplicación deberá poseer una disponibilidad las 24 horas del día y los 7 días de la semana
Prioridad	**Alta.**

Tabla 20.

Mantenibilidad

Número	**RFLN 5**
Nombre	Expedientes y diseño del sistema
Tipo	**Requisito no funcional.**
Descripción	La estructura de la aplicación debe ser intuitiva y fácil de manejar, por medio de manuales y documentos guía que ayuden con el manteamiento del sistema.
Prioridad	**Alta.**

Tabla 21.

Portabilidad

Número	**RFLN 6**
Nombre	Instalación y ejecución
Tipo	**Requisito no funcional.**
Descripción	La aplicación tendrá la capacidad de instalarse y ejecutarse en ambientes similares al de la estación actual y trabajar de la misma manera, y con la misma funcionalidad
Prioridad	**Alta.**

Especificación casos de uso aplicación web

Requerimientos Funcionales

Para el sistema “AdvancedOne Web” desarrollada en angular se han identificado los siguientes requerimientos funcionales.

Tabla 22.

RFW-01 – Conexión con la base de datos en tiempo real Firebase

Id. Requerimiento	**RFW-01**
Actores	Administrador
Descripción	El actor ejecutará el sistema en cualquier navegador web, y la aplicación deberá estar conectada correctamente con la base de datos en tiempo real.

Entradas	• Dirección URL en el navegador	
Salidas	• Respuesta del sistema con la pantalla de inicio de sesión	
Proceso	**Actor** 1. Ingresar la URL dotada por el servidor en el cual se almacena la aplicación web.	**Sistema** 2. Redirecciona a la página principal del sistema es decir al inicio de sesión
Precondiciones	Estar conectados a una red, para tener acceso a internet	
Post condiciones	Tener acceso al inicio de sesión del sistema.	
Flujo alterno	La conexión con la base de datos no está bien establecida, o no existe acceso a internet.	
Prioridad	Primordial	

Tabla 23.

RFW-02 – Iniciar Sesión como Administrador

Id. Requerimiento	**RFW-02**	
Actores	Administrador	
Descripción	El actor tendrá que ingresar sus credenciales para poder acceder al sistema.	
Entradas	• Credenciales	
Salidas	• Respuesta del sistema con la pantalla del Dashboard del sistema	
Proceso	**Actor** 1. Ingresar usuario en el campo establecido	**Sistema**

	2. Ingresar contraseña en el campo establecido	3. Validar que los campos estén llenados 4. Verificar que la información ingresada sea la correcta. 5. Dar acceso a las funciones del sistema 6. Direccionar al Dashboard de la aplicación
Precondiciones	Estar conectados a una red, además de estar registrados en la aplicación.	
Post condiciones	Tener acceso al sistema y por ende acceso a la información de la Base de datos.	
Flujo alterno	No está registrado el usuario. Las credenciales no fueron ingresadas correctamente. No existe conexión con la base de datos. No existe acceso a la red.	
Prioridad	Primordial	

Gestionar Tanques

Tabla 24.

RFW-03– Visualizar Inventario Tanques

Id. Requerimiento	**RFW-03**
Actores	Administrador
Descripción	El actor podrá visualizar la información en tiempo real de los tanques
Entradas	• Seleccionar la opción de visualizar la información de tanques.
Salidas	• Respuesta del sistema con el inventario de los tanques en tiempo real.

Proceso	**Actor** 1. Ingresar al sistema **2.** Seleccionar la opción de visualizar inventario tanques.	**Sistema** 3. Se muestran los datos de los tanques en tiempo real por medio de gráficos.
Precondiciones	Estar conectado a red El usuario debe estar registrado en la base de datos Seleccionar la opción de visualizar inventario tanques	
Post condiciones	Acceder al sistema	
Flujo alterno	No existe conexión con la base de datos. No existe acceso a la red.	
Prioridad	Primordial	

Tabla 25.

RFW-04 – Visualizar Descargas Tanques

Id. Requerimiento	**RFW-04**	
Actores	Administrador	
Descripción	El actor podrá visualizar las últimas descargas realizadas de los tanques.	
Entradas	• Seleccionar la opción de visualizar descargas de tanques.	
Salidas	• Respuesta del sistema con la información de descarga de los tanques	
Proceso	**Actor** **1.** Ingresar al sistema **2.** Seleccionar la opción de visualizar descargas tanques. 3. Seleccionar el tanque del cual se quiere ver las descargas.	**Sistema** 4. Se visualiza la información de descarga del tanque seleccionado.

Precondiciones	Estar conectado a red El usuario debe estar registrado en la base de datos Seleccionar la opción de visualizar descargas tanques
Post condiciones	Acceder al sistema
Flujo alterno	No existe conexión con la base de datos. No existe acceso a la red.
Prioridad	Primordial

Tabla 26.

RFW-05 – Visualizar Alarmas Tanques

Id. Requerimiento	**RFW-05**	
Actores	Administrador	
Descripción	El actor podrá visualizar las alarmas activas de los tanques.	
Entradas	• Seleccionar la opción de visualizar alarmas de tanques.	
Salidas	• Respuesta del sistema con la información de alarmas de los tanques	
Proceso	**Actor** **1.** Ingresar al sistema **2.** Seleccionar la opción de visualizar alarmas tanques.	**Sistema** 3. Se visualiza la información de alarmas de los tanques.
Precondiciones	Estar conectado a red El usuario debe estar registrado en la base de datos Seleccionar la opción de visualizar alarmas tanques	
Post condiciones	Acceder al sistema	
Flujo alterno	No existe conexión con la base de datos. No existe acceso a la red.	
Prioridad	Primordial	

Gestionar Transactor

Tabla 27.

RFW-06 – Visualizar Información Surtidor

Id. Requerimiento	**RFW-06**
Actores	Administrador
Descripción	El actor podrá visualizar la información del surtidor o dispensador.
Entradas	• Seleccionar la opción de visualizar la información del surtidor.
Salidas	• Respuesta del sistema con la información del surtidor.
Proceso	**Actor** / **Sistema** 1. Ingresar al sistema 2. Seleccionar la opción de visualizar información surtidora. 3. Se visualiza la información del surtidor.
Precondiciones	Estar conectado a red El usuario debe estar registrado en la base de datos Seleccionar la opción de visualizar información surtidora.
Post condiciones	Acceder al sistema
Flujo alterno	No existe conexión con la base de datos. No existe acceso a la red.
Prioridad	Primordial

Requerimientos no funcionales aplicación web

Para el sistema “AdvancedOne Web” desarrollada en angular se han identificado los siguientes requerimientos no funcionales.

Tabla 28.

Rendimiento

Número	**RFWN 1**

Nombre	Tiempo de réplica sobre las consultas
Tipo	**Requisito no funcional.**
Descripción	El sistema deberá ser capaz de mostrar los datos enviados desde el dispositivo Tanque o Transactor por medio de la aplicación local que se almacenan en la Base de datos.
Prioridad	**Alta.**

Número	**RFWN 2**
Nombre	Cantidad de usuarios conectados al mismo tiempo
Tipo	**Requisito no funcional.**
Descripción	La cantidad de usuarios sincrónicos, se opera al fraccionar las sesiones de usuario en varios entornos que se ejecutan al mismo tiempo; los tiempos de contestación se optimizan al excluir los cuellos de botella de rendimiento.
Prioridad	**Alta.**

Tabla 29.

Seguridad

Número	**RFWN 3**
Nombre	Conexión
Tipo	**Requisito no funcional.**
Descripción	El sistema establecerá niveles de ingreso hacia los datos, donde posee los siguientes elementos: El ingreso a los recursos de la red se ejecutará a través de un provisor de internet. Únicamente los administradores de las sucursales y el gerente de la matriz tendrán absoluto acceso al sistema con toda la información correspondiente a la aplicación.
Prioridad	**Alta.**

Número	**RFWN 4**
Nombre	Cifrado de Datos
Tipo	**Requisito no funcional.**
Descripción	Referente a la defensa en contra de las actividades de amenaza, se utilizará el Cifrado de Datos Transparente. En el momento en el que se encripta la base de datos, los registros de copias de seguridad y transacciones incorporados se cifran sin el requisito de efectuar cambios en los sistemas.
Prioridad	**Alta.**

Tabla 30.

Fiabilidad

Número	**RFWN 5**
Nombre	Almacenamiento de datos eficaz.
Tipo	**Requisito no funcional.**
Descripción	El sistema poseerá un diseño bien estructurado para el almacenamiento de datos en un entorno No SQL y de tiempo real.
Prioridad	**Alta.**

Tabla 31.

Disponibilidad

Número	**RFWN 6**
Nombre	Disponibilidad del sistema
Tipo	**Requisito no funcional.**
Descripción	El sistema deberá estar en funcionamiento y disponible con todas sus características para todos sus usuarios los 7 días de la semana por las 24 horas.
Prioridad	**Alta.**

Tabla 32.

Mantenibilidad

Número	**RFWN 7**
Nombre	Buenas prácticas
Tipo	**Requisito no funcional.**
Descripción	El sistema deberá estar diseñado y elaborado con técnicas de programación estructurada, una arquitectura y estándares.
Prioridad	**Alta.**

Número	RFWN 8
Nombre	Compresión del sistema
Tipo	**Requisito no funcional.**
Descripción	El sistema y toda su información referente debe ser comprensible, no debe existir complejidad en el aplicativo además de ser fácil de interpretar
Prioridad	**Alta.**

Tabla 33.

Portabilidad

Número	RFWN 9
Nombre	Red e interconexión
Tipo	**Requisito no funcional.**
Descripción	El sistema debe prestar todas sus características bajo cualquier red que tenga acceso a internet, sea LAN o Wifi.
Prioridad	**Alta.**

Número	RFWN 10
Nombre	Base de datos
Tipo	**Requisito no funcional.**
Descripción	El sistema debe usar una base de datos no relacional Firebase
Prioridad	**Alta.**

Especificaciones de protocolos de comunicación

Sistema de tanques – Veeder – Root

El sistema de tanques que se encuentra instalado en la estación de servicios “El Oasis” es de la marca Veeder-Root, el cual usa una interfaz serie RS-232 la misma que se utiliza para conectar el sistema a un ordenador de control, un terminal de visualización (CRT) o un terminal de impresión. Se tiene también la opción de conectar un módem directamente al sistema para proporcionar acceso a la línea telefónica. A continuación, se muestra en la Figura 30, el dispositivo Veeder – Root.

Figura 30.

Consola del sistema de tanques Veeder-Root

RS-232

El puerto 1 es un conector D hembra de 9 pines de montaje en panel, cableado en la configuración DCE. Un módem (DCE) puede conectarse a la interfaz utilizando un cable nulo que invierte los cables para las señales de transmisión/recepción. A un ordenador o una impresora en serie (DTE) pueden conectarse con un cable recto. El puerto 1 no requiere ni activa ninguna señal de handshaking por hardware a menos que se haya seleccionado esta opción en el menú de configuración. Las señales RS-232 para el puerto 1 se conectan al conector D hembra de 9 pines como se muestra en la Tabla 34.

Tabla 34.

Tabla de pines del puerto 1 de la interfaz RS-232

	PIN	
Port 1	1	(DCD) Detección del portador de datos
	2	(RXD) Datos recibidos por la consola

	3	(TXD) Datos transmitidos desde la consola
	4	(DTR) Terminal de datos listo
	5	(GND) Señal de tierra
	6	(DSR) Conjunto de datos listo
	7	(RTS) Solicitud de envío
	8	(CTS) Despejado para enviar
	9	(N/C) No conectado

Las señales de RS-232, así como en la Figura 31 para el puerto 2, que no incluye ninguna señal de handshaking por hardware, se conectan al conector interno J-9 de 5 pines interno J-9 como se muestra en la Tabla 35.

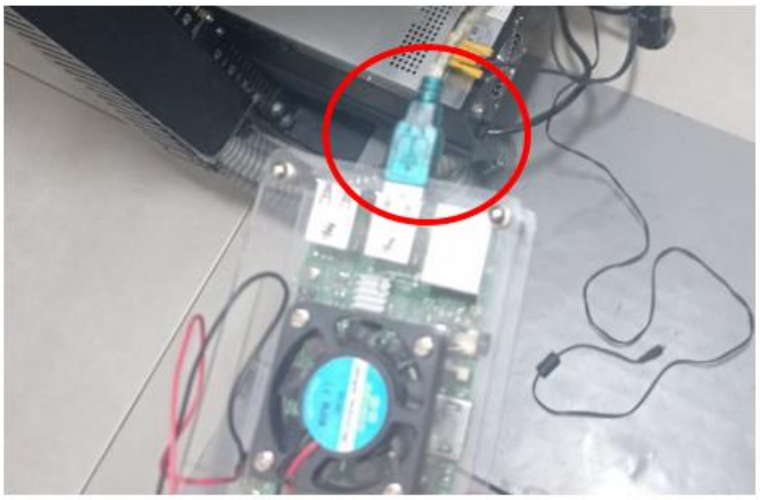
Tabla 35.

Tabla de pines del puerto 2 de la interfaz RS-232

	PIN	
Port 2	1	(TXD) Datos transmitidos desde la consola
	2	(RXD) Datos recibidos por la consola
	5	(GND) Señal de tierra

Figura 31.

Cable interfaz RS-232 conectado a un Raspberry



Formato de caracteres y tasa de baudios

En el caso del sistema de medición de tanques Veeder-Root, este envía y recibe caracteres a través de la interfaz RS-232 en un formato ASCII que se configura a través de los teclados del panel frontal del sistema. Las selecciones consisten en: 1 bit de inicio; 7 u 8 bits de datos; par, impar o sin paridad; y 1 o 2 bits de parada. La velocidad de comunicación es seleccionable: 300, 1200, 2400, 4800 o 9600 baudios. El sistema funciona en modo dúplex completo. Los caracteres no se repiten cuando se reciben, los caracteres transmitidos no deben ser devueltos al sistema. La transmisión y la recepción pueden producirse simultáneamente, y los comandos pueden apilarse en el búfer del sistema, hasta 128 caracteres.

Configuración del código de seguridad

Se delega un código de seguridad para cada puerto desde los menús de configuración del panel frontal, o utilizando los comandos seriales apropiados. Cada puerto tiene su propio código de seguridad, el cual, se habilita y configura independientemente. Cuando se habilita un código de seguridad para un puerto determinado, se utiliza en cualquier comando de acuerdo con el formato que se muestra en la sección posterior, o de lo contrario la consola no responde al comando.

Análisis y Diseño

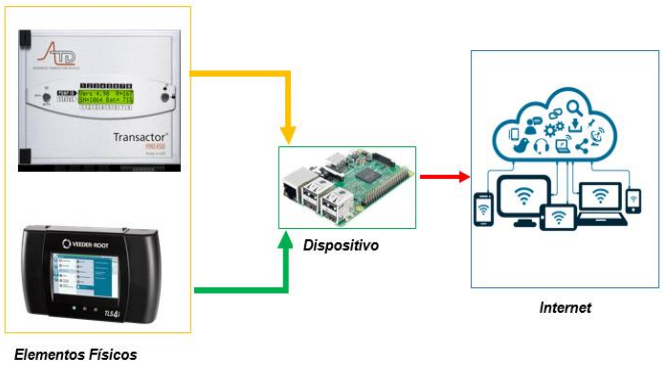
Arquitectura del Prototipo

Para entender la arquitectura de la cual se hizo uso a lo largo del proyecto de tesis, es importante mencionar que la base del prototipo es la tecnología IoT (Internet de las Cosas), la cual, es una infraestructura que engloba al colectivo de datos e información, con el objetivo de brindar servicios modernos, a través de la relación entre elementos materiales y virtuales, dicho concepto se lo puede representar mediante la siguiente ecuación, convirtiéndose en el fundamento del entorno para interactuar en diferentes ámbitos:

$$Internet + Elemento\ Físico + (Dispositivo + Sensor + Controlador)$$

Figura 32.

Arquitectura Prototipo



Dicho entrono se ejecuta, por medio de las funciones que poseen los elementos, con el fin de brindar toda la información que un determinado campo requiere, dependiendo sus actividades. Todo este proceso se rige a través de un modelo de arquitectura operable, manejable, funcional y con características técnicas. La arquitectura del prototipo es el Modelo IoTWF o arquitectura 7 capas, el proceso se basa en que los elementos físicos transmiten y receptan datos, creando una interacción con el internet, debido a que, por medio de este enlace, los datos pueden ser enviados, procesados y depurados usando Edge Computing, previo a ser archivados en una Data Base, a la cual se puede acceder a través de la aplicación, para la visualización e interacción con la información.

Con base a lo anterior, el flujo de la información tiene un proceso esencial, el cual se fundamenta en la recepción, transformación, procesamiento y almacenamiento de la misma, por medio de los elementos físicos que en este caso son el Transactor y el Sistema de Tanques Veeder-Root, los que se encargan de enviar los datos, para ser analizados y pulidos por medio de un software que se ejecuta en el Raspberry, y una vez filtrados poder almacenarlos, así como se visualiza en la Figura 32. Este procedimiento al final está estructurado, por un conjunto de programas, los cuales permiten la interacción con el dispositivo Raspberry, a continuación, se mencionan algunos:

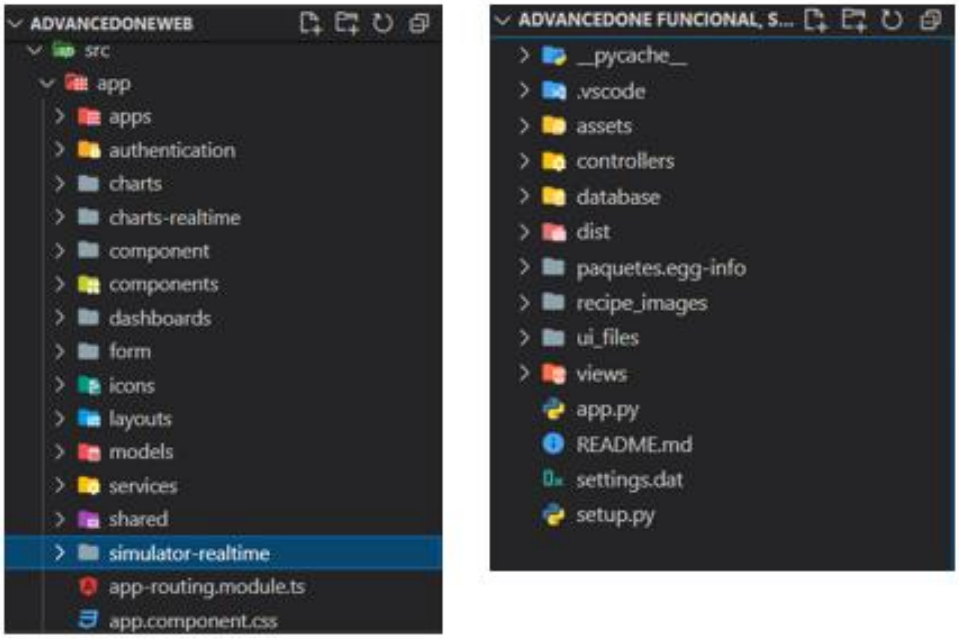
- Control de elementos físicos y dispositivos: Ajustes, configuración y estados.
- Interconexión entre objetos y dispositivos: Enlaces
- Listeners
- Datos e Información: Decodificación y alerta.

Patrón de arquitectura del prototipo

El patrón de arquitectura con el cual se trabajó a lo largo del proyecto de tesis fue el MVC, el cual posee una estructura que se fundamenta en tres componentes esenciales, como lo son, el modelo, la vista y el controlador, teniendo como eje, tener elementos específicos para cada paquete, es decir, que no puede existir una combinación de elementos dentro de una misma clase. Ofreciendo un orden y tareas definidas para cada componente. De esta manera es mucho más sencillo el desarrollo de un producto software, por el hecho de tener clasificado y dividido cada elemento correspondiente a cada función. A continuación, en la Figura 33, se muestra el patrón arquitectónico utilizado.

Figura 33.

Patrón de Arquitectura MVC

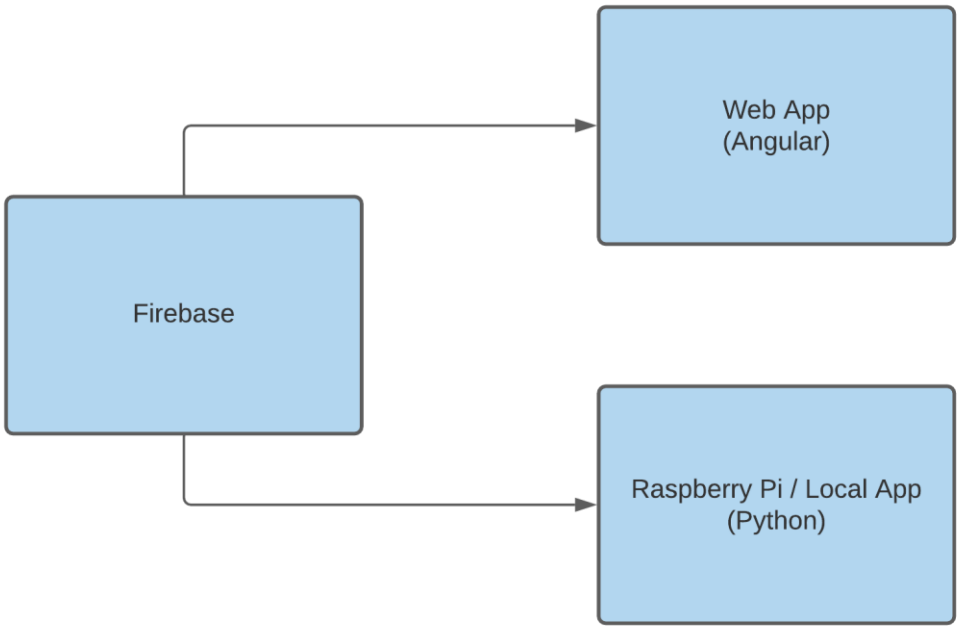


Módulos de la arquitectura

Para el desarrollo de los módulos tanto de aplicaciones local y web se seleccionó una base de datos no relacional como Firebase, debido a las ventajas que se obtiene para el manejo de información no estructurada, ya que, tanto el controlador industrial como el sistema de tanques envían información en grandes cantidades y para su rápido almacenamiento se definió un motor que pueda estar disponible desde cualquier sitio (la nube) y que soporte el manejo de información en tiempo real. A continuación, en la Figura 34, se indica la estructura general del módulo arquitectónico.

Figura 34.

Módulo de arquitectura



Motor de base de datos (Firebase)

Para el desarrollo del proyecto se toma una base de datos en tiempo real (realtime database) provista por Firebase, la misma que se encuentra alojada en la nube. Los datos se sincronizan continuamente con la aplicación web asociada. Además, el uso de este motor de base de datos al estar en la nube y ser NoSQL permite reducir el tráfico de información ya que se la ha diseñado como una estructura plana.

Diseño. Se usa colecciones y documentos para estructurar y consultar datos, con este modelo se puede realizar querys que no se ve afectado en el tiempo si el resultado es un conjunto de 100 o 100 000 000 resultados. Al ser una estructura en tiempo real todos los datos se almacenan como objetos JSON, de esta manera la base de datos puede entenderse como un árbol JSON, como se visualiza en la Figura 35, alojado en un servicio en la nube.

En comparación con una base de datos SQL en esta estructura no existen tablas ni registros, sin embargo, se consideraron varios nodos basados en el modelo de negocio para la aplicación y la persistencia de información, de esta manera se consideraron:

Empresa. Donde se guardan los datos de la empresa actual que vaya a usar la aplicación.

Inventario. Este nodo se usa exclusivamente para el sistema de tanques, y aquí se albergan las consultas por comandos enviados desde el aplicativo.

Menú consola. Nodo usado para configurar lo que puede ver el usuario según su perfil

Menú perfil. Nodo usado para guardar información que es visible de distintos perfiles que tiene la aplicación.

Movimiento surtidor. Nodo usado exclusivamente para el controlador industrial (Transactor) donde se alberga cada transacción de despacho de combustible con información enviada desde el simulador Tokheim.

Perfil. Perfiles que existen dentro del aplicativo.

Perfil Usuario. Nombres de los usuarios con distintos perfiles que tiene el aplicativo.

Sesión. Nodo donde se alberga momentáneamente las sesiones actuales que estén usando el sistema.

Sucursal. Nodo para albergar las sucursales que tenga disponible una empresa.

Tanque. Nodo usado para guardar información de configuración y modelo del sistema de tanques.

Usuario. Usuarios y tipos de usuarios existentes dentro del aplicativo.

Usuario Sucursal. Usuarios disponibles dentro de una sucursal.

Al momento de agregar datos al árbol JSON estos se convierten en un nodo de la estructura existente con una clave asociada, así como, se indica en la Figura 35.

Figura 35.

Consola de administrador de firebase

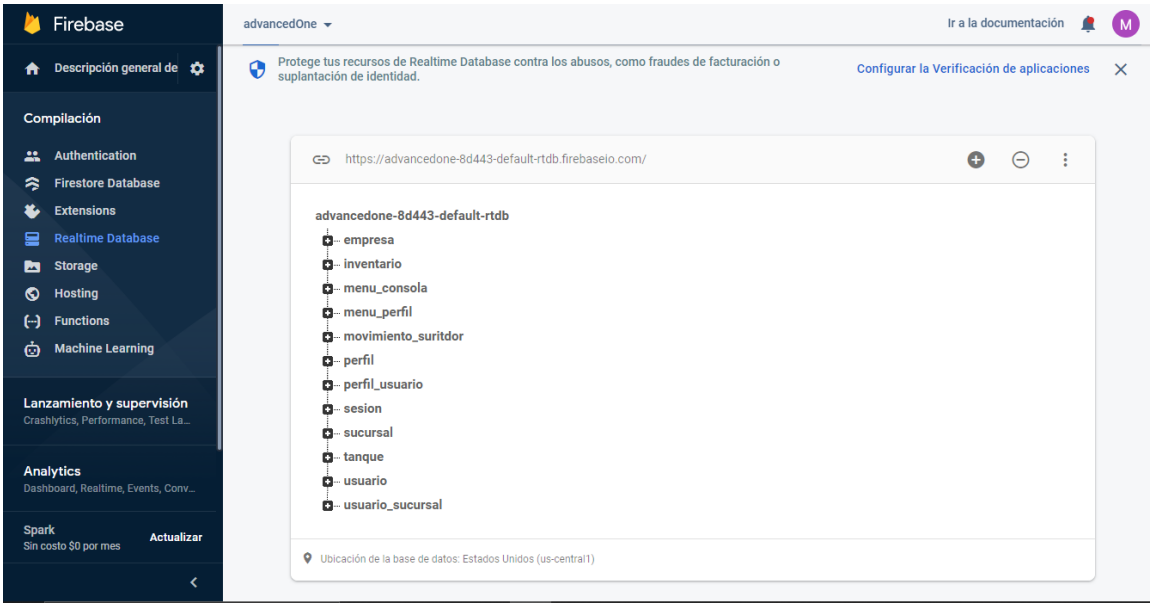
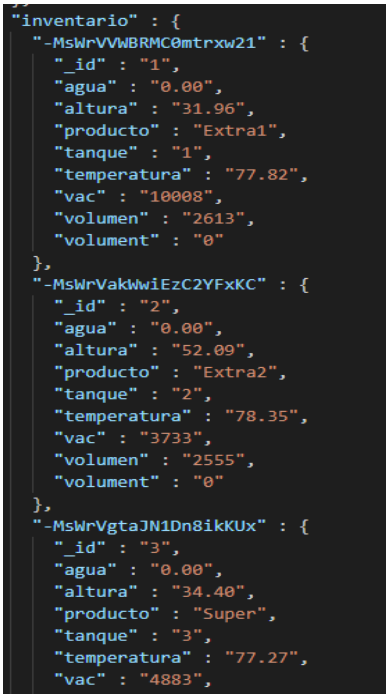


Figura 36.

Modelo de árbol JSON para AdvancedOne en la estación "El Oasis"



```
"inventario" : {
  "-MsWrVVWBRMC0mtrxw21" : {
    "_id" : "1",
    "agua" : "0.00",
    "altura" : "31.96",
    "producto" : "Extra1",
    "tanque" : "1",
    "temperatura" : "77.82",
    "vac" : "10008",
    "volumen" : "2613",
    "volument" : "0"
  },
  "-MsWrVakWwiEzC2YFxKC" : {
    "_id" : "2",
    "agua" : "0.00",
    "altura" : "52.09",
    "producto" : "Extra2",
    "tanque" : "2",
    "temperatura" : "78.35",
    "vac" : "3733",
    "volumen" : "2555",
    "volument" : "0"
  },
  "-MsWrVgtaJN1Dn8ikKUx" : {
    "_id" : "3",
    "agua" : "0.00",
    "altura" : "34.40",
    "producto" : "Super",
    "tanque" : "3",
    "temperatura" : "77.27",
    "vac" : "4883",
```


Módulo de conexión y enlace

Para el desarrollo del módulo de conexión entre un controlador industrial y un sistema de integración en tiempo real se planteó el uso de un convertidor serial RS232 a USB (Universal Serial Bus). Este se encarga de recibir datos de la interfaz en serie del controlador industrial y sistema de tanques para enviarlos a la interfaz USB de un PC. De igual este convertidor se encarga de enviar la información desde el PC hasta el dispositivo, por estas características el convertidor RS232 es el encargado de controlar el módulo de conexión, mas no de la aplicación, una vez receptada la información de bajo nivel se requiere de un módulo de comunicación y aplicación para poder manejar los datos según sea requerido por un usuario. A continuación, en la Figura 37, se muestra el convertidor serial utilizado.

Figura 37.

Convertidor Serial RS232 a USB

Módulo de comunicación

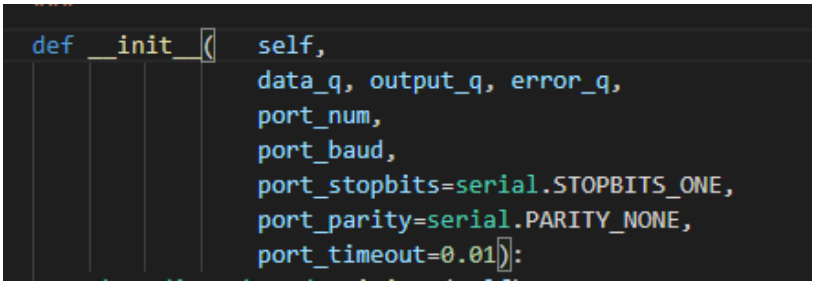
El módulo de comunicación es el encargado de establecer el intercambio de datos entre el Transactor, sistema de medición y el dispositivo IOT (Raspberry Pi), haciendo uso del convertidor serial RS232 y el lenguaje de programación Python, en el mismo que se usa la librería “serial” como base, para poder acceder a los parámetros de comunicación del sistema de control de tanques y Transactor entre los que se incluyen los datos, las salidas, los errores, el número de puerto, tasa de baudios, numero de bits, paridad, y tiempo de espera, como se puede ver en la Figura 38.

Una vez que se han interceptado todos los parámetros del dispositivo este módulo nos permite identificar si el dispositivo se encuentra listo para enviar información y de esta manera permite abrir el puerto o de igual manera cerrarlo según se necesite por el usuario, así como se visualiza en la Figura 39.

Es importante mencionar que en este módulo gracias a esta librería se pueden controlar excepciones al momento de realizar una escritura o lectura de un dispositivo con el Raspberry Pi y todo esto es posible mediante otra librería llamada “queue” la misma que se encarga de procesar información al mismo tiempo que se controla que el dispositivo este comunicándose exitosamente con la aplicación.

Figura 38.

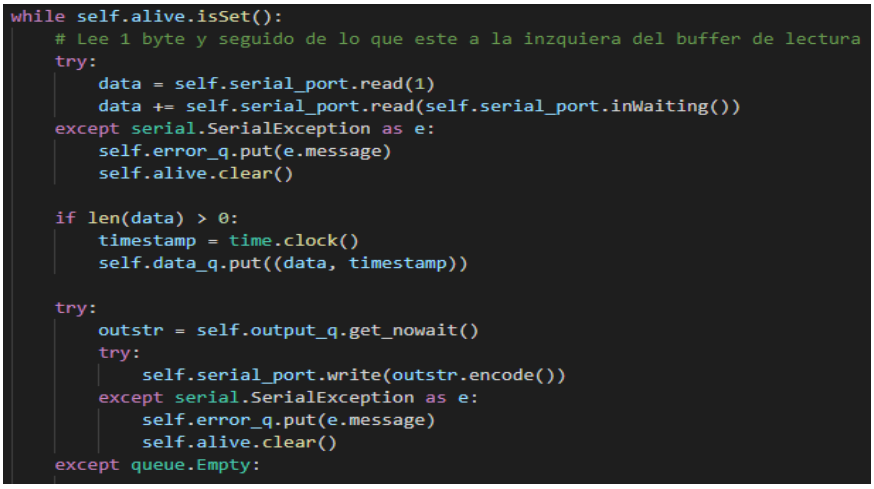
Código de inicialización para leer los parámetros de comunicación



```
def __init__(   self,
                data_q, output_q, error_q,
                port_num,
                port_baud,
                port_stopbits=serial.STOPBITS_ONE,
                port_parity=serial.PARITY_NONE,
                port_timeout=0.01):
```

Figura 39.

Modelo de código para abrir y cerrar un puerto de comunicación serial



```
while self.alive.isSet():
    # Lee 1 byte y seguido de lo que este a la inzquiera del buffer de lectura
    try:
        data = self.serial_port.read(1)
        data += self.serial_port.read(self.serial_port.inWaiting())
    except serial.SerialException as e:
        self.error_q.put(e.message)
        self.alive.clear()

    if len(data) > 0:
        timestamp = time.clock()
        self.data_q.put((data, timestamp))

    try:
        outstr = self.output_q.get_nowait()
        try:
            self.serial_port.write(outstr.encode())
        except serial.SerialException as e:
            self.error_q.put(e.message)
            self.alive.clear()
    except queue.Empty:
```

Módulo de transformación y gestión de tramas

El módulo de gestión de tramas dentro de la arquitectura en tiempo real es el encargado de tomar los datos de bajo nivel transmitidos desde el sistema de tanques y el Transactor para transformarlos en información que pueda ser entendida tanto por la aplicación como por el usuario, y todo esto está programado en lenguaje Python con la ayuda de librerías como serial, threading, queue y time.

Para que esta traducción de información sea posible se tiene que pasar por el módulo de comunicación y una vez que el aplicativo posea un enlace estable con el dispositivo, además de un puerto abierto exitosamente, está listo para poder recibir y enviar datos mutuamente. Todo este proceso se lo realiza dentro de una clase específica para cada dispositivo, se capta el buffer de salida, un buffer de errores, el número de puerto y la velocidad de baudios a los que trabaja como se ve en la Figura 40.

Figura 40.

*Código que inicializa la gestión de tramas desde un dispositivo y el Raspberry Pi***.**

```
def __init__(self,output_buffer=None,error_buffer=None,port=None,baud=None):
```

Una vez que se han captado los buffers del dispositivo, se obtiene la información y se procede a parsear los datos de manera que empiecen a ser transformados en caracteres que se son concatenando hasta formar una cadena que se alberga en una función para retornar hasta el módulo principal y así ser mostrado en la interfaz para el usuario.

Todo este proceso de parseo se hace con la ayuda de cláusulas if-else, y se agrupan las cadenas dependiendo del dispositivo:

Transactor. Dentro de su protocolo usa un tipo de comandos de dos caracteres separado de un espacio y el número del lado del surtidor con el que esté relacionado. En este dispositivo el comando más usado es RT (Real Time) el mismo que indica al dispositivo que se requiere la información de que se esté transaccionando en ese momento, a lo cual él entrega una respuesta y esta se muestra en la interfaz para el usuario.

Sistema de tanques. Dentro del protocolo del sistema de tanques se usan comandos de 6 caracteres, y su ejemplo más usado es el i20100, el mismo que entrega un inventario del sistema de tanques hasta el momento en que se realizó la consulta, pero toda esta información llega en formato hexadecimal, y el formato en que responde es el siguiente:

i201TTYYMMDDHHmmTTpssssNNFFFFFFFF&&CCCC

A continuación, en la Tabla 36, se observa la estructura en formato hexadecimal de un comando ejecutado.

Tabla 36.

Tabla de estructura de comando en formato hexadecimal del sistema de tanques

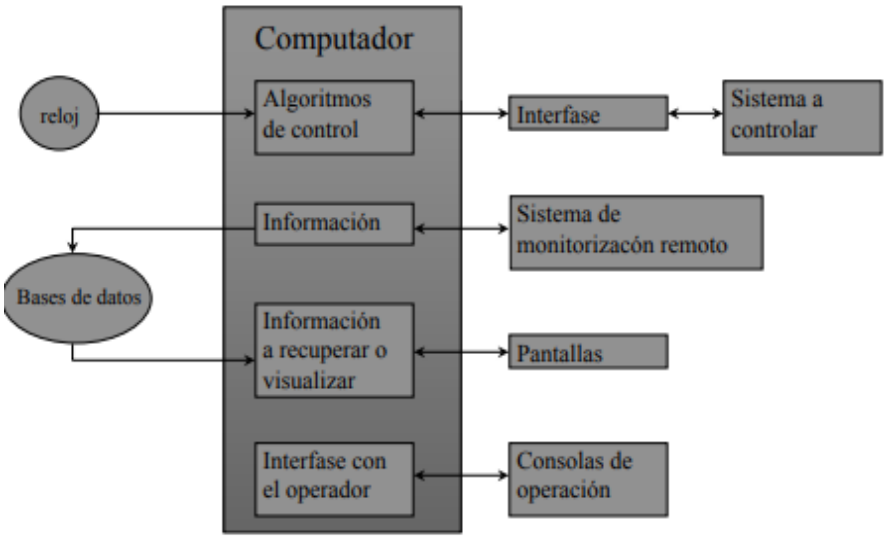
Parte del comando	Significado
i201	Número de comando
TT	Número de tanque
YYMMDDHHmm	Fecha y tiempo actual
p	Código del producto
SSSS	Estado del tanque en bits: • Bit 1: Venta en proceso • Bit 2: En proceso de prueba • Bit 3: Alarma inválida • Bit 4-16: No se usa
NN	Información de hasta ocho caracteres en hexadecimal
FFFFFFFF	Información en formato ASCII HEX IEEE
&&	Bandera para indicar dónde termina la información
CCCC	Checksum

Módulo de procesamiento de información

Para el módulo de procesamiento de información en tiempo real hay que entender cómo funciona la estructura de este tipo. Dichas aplicaciones o dichos entornos se basan en el flujo de datos, con el fin de captarlos al instante y procesarlos con un delay lo más bajo posible, para gestionar, visualizar o manipular la información. En el caso del procesamiento de información en tiempo real, para la estación de servicio "ECO EL OASIS" en el sistema de venta de combustible, y en el sistema de medición de tanques, el proceso se mide en segundos, debido a que, de esa manera, se decidió trabajar en el lenguaje PYTHON y con un formato estructurado de tipo JSON, que es lo que se envía y por medio del lenguaje angular se lo transforma, para que sea visible y entendible en la parte web. A continuación, en la Figura 41, se visualiza un esquema general sobre el procesamiento de información en tiempo real.

Figura 41.

Esquema de procesamiento de información en tiempo real



Nota. (Departamento de Tecnología de Computadores, 2019)

Este módulo está conformado por todos los módulos anteriores, ya que, es indispensable cada uno de ellos para su correcto funcionamiento, el procesamiento de información en tiempo real, se lo realizó estableciendo un enlace por sockets entre la aplicación de escritorio que fue diseñado en Python, y la aplicación Web que se desarrolló en Angular, la tecnología socket que se usó para esta interconexión fue la tecnología SocketIO, que está presente en los dos lenguajes ya mencionados anteriormente, y resulta mucho más ágil y fácil enlazarlos. A continuación, en la Figura 42, se muestra un ejemplo de la codificación, para la interconexión por medio de Socket IO, entre el lenguaje Python y Angulas respectivamente.

Figura 42.

Interacción por medio de SocketIo python - angular

```python
from flask import Flask, render_template
from flask_socketio import SocketIO

app = Flask(__name__)
app.config['SECRET_KEY'] = 'secret!'
socketio = SocketIO(app)

if __name__ == '__main__':
    socketio.run(app, host='0.0.0.0', debug=True)
```

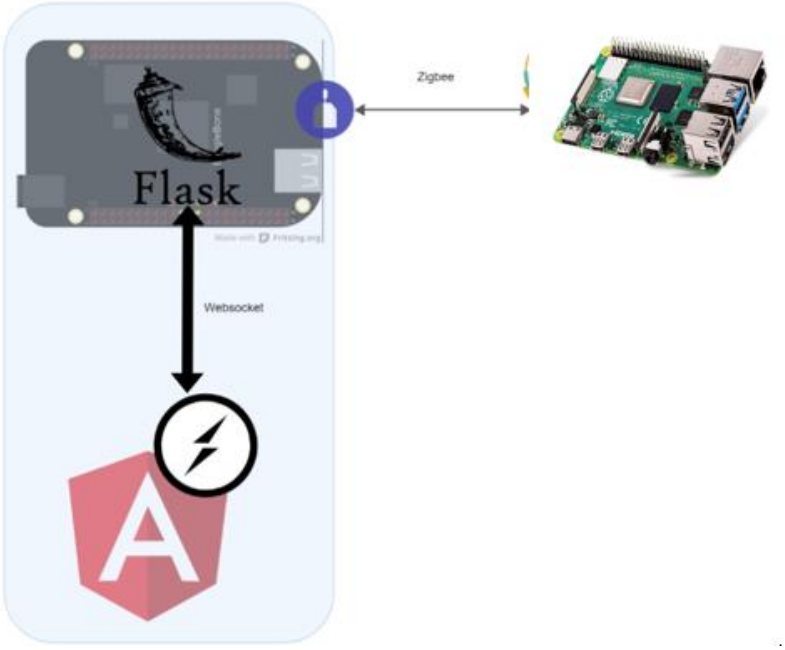
```typescript
private listenSockets() {
    this.io.on('connection', client => {

        // Listen Events

        socket.connectUser(client, this.io);

        socket.disconnected(client, this.io);

        socket.getUsers(client, this.io);

        socket.message(client, this.io);
    });
}
```

La información que se envía por los sockets, es la que se obtuvo por medio del envió de datos del sistema de tanques y del surtidor hacia el Transactor, y del Transactor hacia el Raspberry, todos los datos se procesan en la aplicación interna, por medio de la ejecución de una función que opera de forma constante, dicho método, se encuentran en un ciclo infinito, porque es la que se encarga del procesamiento de todos los datos que se reciben, por medio de los dispositivos. Este ciclo infinito, está

definido en este proyecto y en el campo de las estaciones de servicio como “POLEO”, toda la información procesada en este punto es enviada por los sockets hacia la parte web, y dependiendo del tipo de tarea que se quiera ejecutar, se inicializa un thread diferente. A continuación, en la Figura 43, se visualiza la forma de interacción de los dos lenguajes, con el dispositivo Raspberry.

Figura 43.

Tecnología python - angular



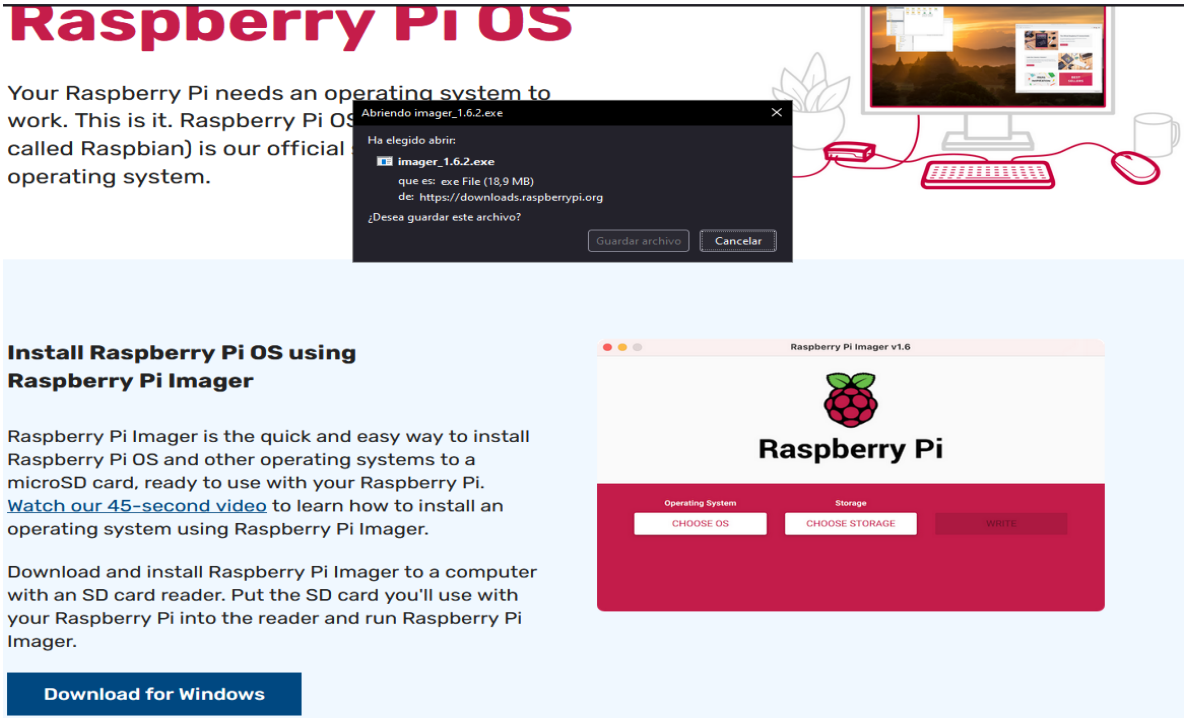
Construcción del Prototipo

Configuración Raspberry

Para la configuración del dispositivo Raspberry, existe un procedimiento que consta de varios pasos para la correcta y adecuada instalación del sistema operativos y sus componentes, el primer paso a seguir es descargar el componente llamado NOOBS (New Out of Box Software). A continuación, en la Figura 44, se observa la página oficial, a la cual se debe acceder para descargar el sistema operativo, con el cual se trabajó durante la ejecución del proyecto de tesis.

Figura 44.

Descarga componente NOOBS

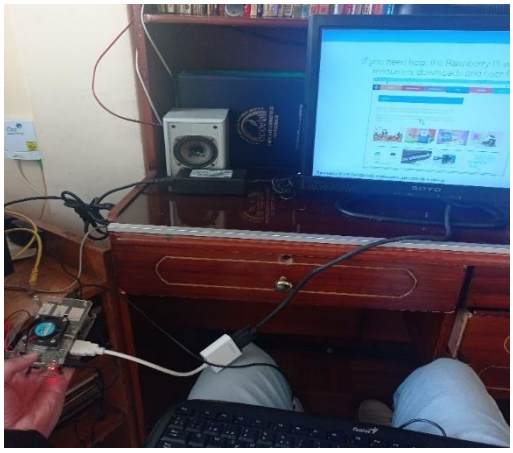


Noobs. Es un componente que ayuda en el proceso de instalación de distintos tipos de sistemas operativos, teniendo como factor fundamental la compatibilidad con el dispositivo Raspberry, el proceso de descarga de esta herramienta software es simple, se ingresa al link de descarga y posteriormente a la su finalización, se debe introducir todos los archivos obtenidos en una tarjeta micro SD previamente formateada, la cual debe contar con un tamaño de almacenamiento considerable, para el desarrollo de este proyecto se utiliza una micro SD de 64gb.

Seguido de la descarga del componente NOOBS (New Out of Box Software) y de almacenar los archivos en una tarjeta micro SD, se procede a conectar el Raspberry PI a los dispositivos de salida como: el monitor, el mouse, el teclado y establecer la conexión con la red, posteriormente, se debe conectar el cable de alimentación eléctrica y de esta manera se inicia el proceso de instalación del sistema operativo en el dispositivo. A continuación, en la Figura 45, se visualiza, la conexión de los dispositivos de salida con el Raspberry, para su correcta instalación.

Figura 45.

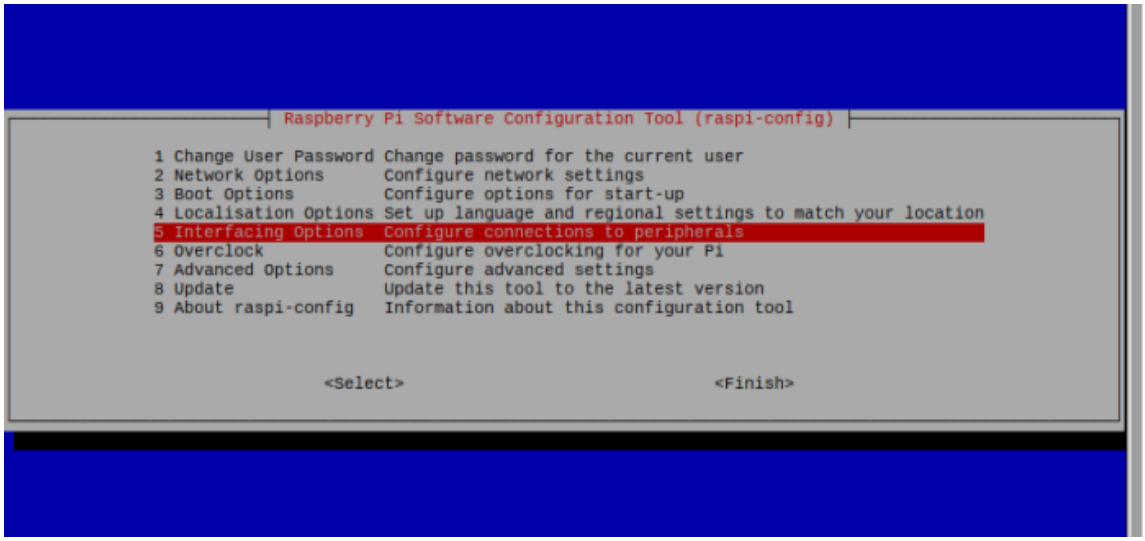
Instalación Sistema Operativo



Una vez que se tiene el sistema operativo instalado y funcionando correctamente, se estableció una conexión con el puerto serial, para realizar dicha conexión primero hay que ejecutar el componente de configuración del Raspberry con el siguiente comando **sudo raspi-config,** al introducir este código se despliega la Figura 46, en la cual, se selecciona la opción de **Interfacing Options**.

Figura 46.

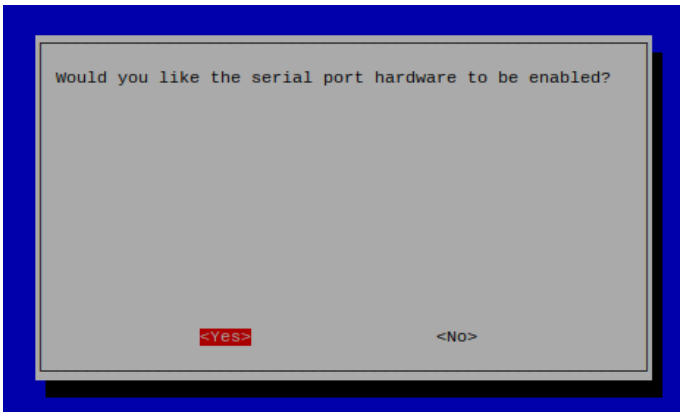
Configuración Puerto Serial



Dentro de la pantalla de **Interfacing Options**, se selecciona la opción P6 Serial, la cual, hace referencia a la habilitación del puerto serial, seguido de esto se despliega la pantalla que se muestra en la Figura 47, para aceptar la configuración del puerto, una vez aceptado se reinicia el dispositivo Raspberry, para que los cambios realizados se guarden y se ejecuten adecuadamente.

Figura 47.

P6 Serial



Para probar que el puerto serial del dispositivo Raspberry Pi, este abierto correctamente y que exista una conexión satisfactoria se crea una pequeña rutina de programación en cualquier lenguaje, en este caso Python para verificar que existe una respuesta exitosa. A continuación, en la Figura 48, se indica un ejemplo de codificación, para la verificación de que el puerto serial esté conectado correctamente.

Figura 48.

Código de verificación del puerto serial.

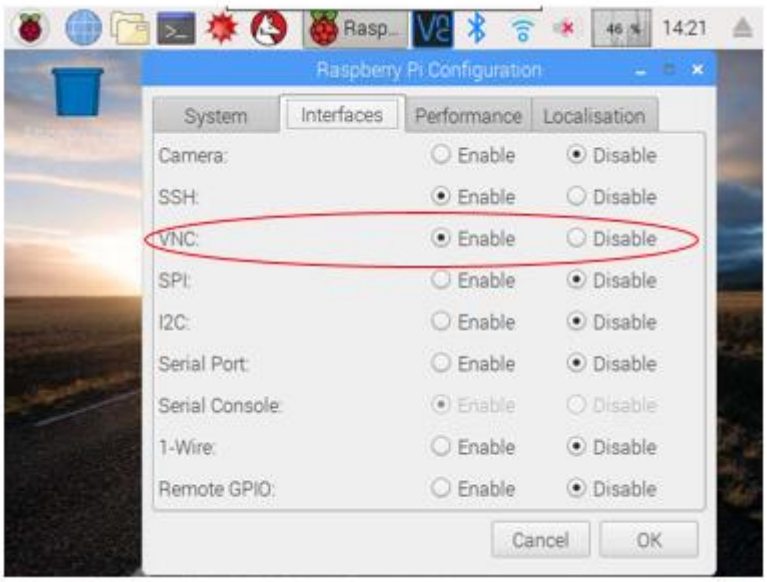
```
# -*- coding: utf-8 -*
import serial
import time
 # Abra el puerto serie
ser = serial.Serial("/dev/ttyAMA0", 9600)
def main():
    while True:
                # Recibe los caracteres del buffer de recepción
        count = ser.inWaiting()
        if count != 0:
                        # Leer contenido y eco
            recv = ser.read(count)
            print(recv)
            ser.write(recv)
                # Borrar el búfer de recepción
        ser.flushInput()
                # Retardo de software necesario
        time.sleep(0.1)

if __name__ == '__main__':
    try:
        main()
    except KeyboardInterrupt:
        if ser != None:
            ser.close()
```

Conexión remota al Raspberry por medio de la aplicación VNC

La conexión remota al Raspberry se la establece mediante un software que viene por defecto en el sistema operativo del mismo. El modelo que se usó para el proyecto es un Raspberry Pi 4 en el cual ya consta VNC, cabe mencionar que no en todos los modelos se encuentra presente dicho aplicativo y su instalación se la realiza de forma manual. Para activar esta función se elige la opción de configuración del dispositivo, seguido de seleccionar el recuadro de habilitar. A continuación, en la Figura 49, se muestra la pantalla en la cual se debe elegir la opción mencionada.

Figura 49.

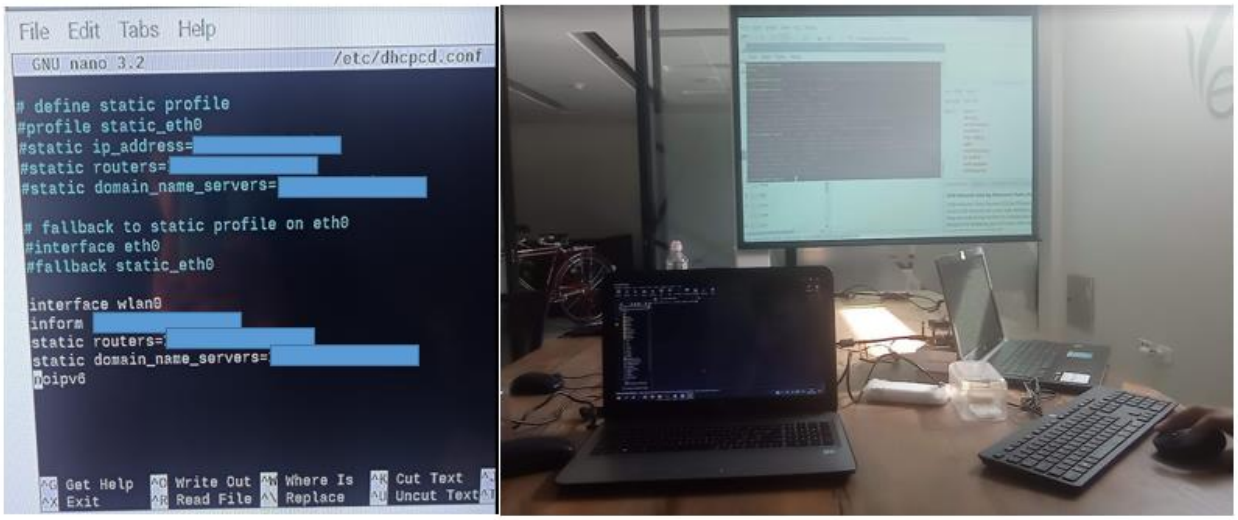
Habilitación VNC



Una vez habilitado el aplicativo se configura una IP estática en el Raspberry, con el objetivo de establecer una conexión de forma más rápida y sencilla, debido a que, para la ejecución de la primera parte del proyecto de tesis, los dos elementos que se usaron fueron el sistema de tanques y el Raspberry, el sistema de tanques se encuentra en la habitación de los servidores de la estación de servicio “ECO EL OASIS”, lo que dificultaba el acceso constate, por lo que se tomó la decisión de dejar el Raspberry encendido y conectado al sistema, y desde las computadoras establecer la conexión. A continuación, en la Figura 50, se observa la conexión remota desde las laptops hacia el dispositivo Raspberry.

Figura 50.

Conexión Raspberry por medio de VNC



Nota. Por motivos de seguridad y confidencialidad no se muestra en la Figura 50 las IPs. Autores, 2022

Configuración del Simulador Tokheim

Para la configuración del simulador Tokheim, lo primero que se realizó fue conectarlo con el dispositivo Transactor, ya que, este es el que envía la información del proceso de venta de combustible, una vez enlazados los dos dispositivos, el siguiente paso es poner en modo desarrollo al equipo Tokheim, lo cual se hizo de la siguiente manera. Primero se levanta el switch izquierdo que se encuentra debajo de las dos pantallas, en las que se visualiza el precio y los galones, una vez levantado, en el display superior se observa la letra F seguido de dos números, el dígito que se debe establecer es el número 19, para realizar esto, se procede a levantar el switch derecho y bajarlo cuando llegue a la cantidad ya mencionada. A continuación, en la Figura 51, se indica las palancas de operación, y las pantallas tanto de configuración, como de galones y de costo.

Figura 51.

Tokheim modo desarrollo

Consecutivo a esto, el switch derecho es levantado y bajado rápidamente para pasar al proceso en la pantalla inferior, en la cual, únicamente se encuentra un dígito, y a este hay que posicionarlo en el número 0, para establecer este valor, nuevamente se levanta el switch derecho hasta que llegue al 0 y bajarlo, a continuación, se baja el switch izquierdo y el simulador está listo para usarse y apto para la realización de pruebas mediante el software diseñado AdvancedOne. A continuación, en la Figura 52, se observa, como quedan las pantallas con los dígitos en 0, en caso de que la configuración haya sido realizada con éxito.

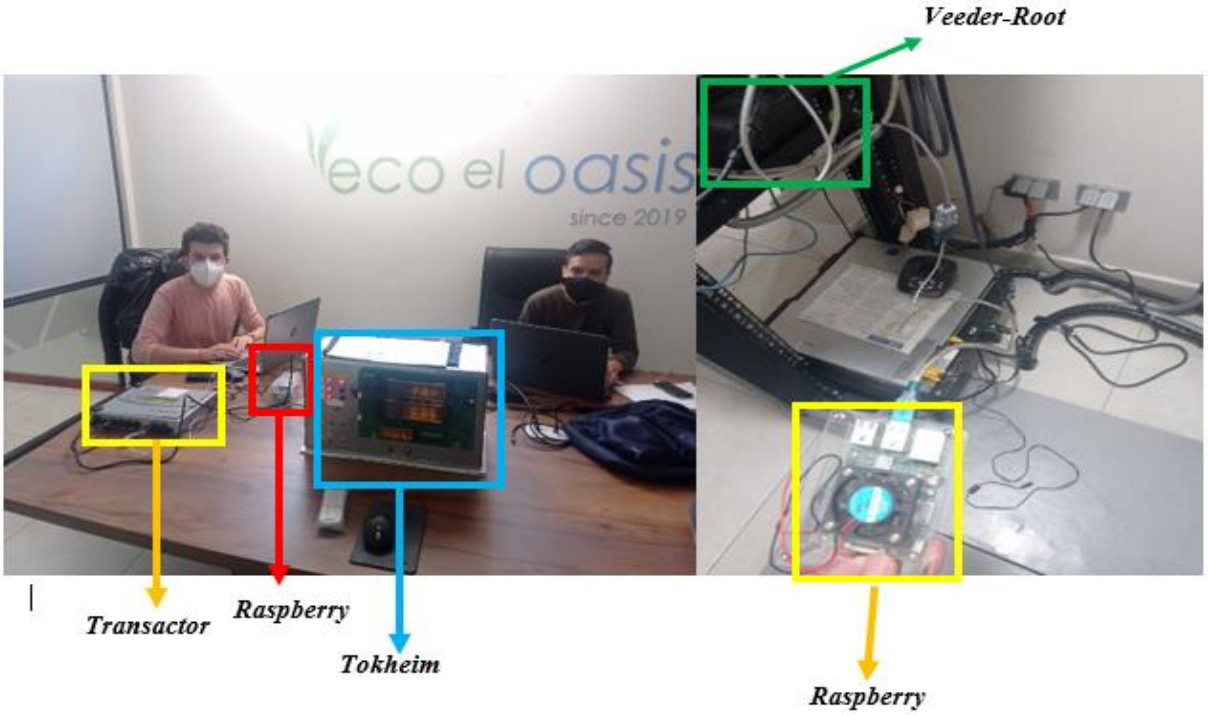
Figura 52.

Configuración completada

La configuración final del prototipo está constituida por todos los elementos mencionados anteriormente, es decir, el conjunto de elementos que conforman la primera parte del prototipo son, el Transactor, el simulador Tokheim y el Raspberry, la unión de todos estos dispositivos dan como resultado la funcionalidad para el proceso de ventas de la estación de servicio “ECO EL OASIS”, y en la segunda parte del proyecto, los elementos que forman parte de esta etapa son el sistema de medición de tanques controlado por el hardware Veeder-Root y por el Raspberry, dando como resultado las operaciones sobre los tanques, tanto de descarga, inventarios y alarmas, a continuación, se visualiza la Figura 53, que detalla el conjunto de elementos mencionados.

Figura 53.

Configuración del prototipo



Capítulo IV

Implementación y Pruebas

En capítulos previos se definió la mejor arquitectura y se explicó el uso de cada componente para que el aplicativo cumpla con todos los requerimientos de la estación de servicios, por lo que en este capítulo se puso en práctica el prototipo planteado y se exponen los resultados con los componentes elegidos para que estos puedan ser recabados de tal manera que en un futuro se pueda o no pensar en una implantación en un ambiente productivo.

Implementación

Implementación del prototipo

La implementación del prototipo constó de dos partes fundamentales, las cuales fueron la aplicación de escritorio y la aplicación web, la primera parte se desarrolló en el lenguaje Python debido a su facilidad de integración con las tecnologías y elementos IoT (Internet de las cosas). La aplicación de escritorio tiene la funcionalidad de conectarse con los dispositivos Transactor y con el sistema de Tanques, con la capacidad de recibir información enviada por dichos elementos, y transformar los datos a través de procedimientos diseñados, y establecidos para que la gestión de las tramas sea entendible y manejable. A continuación, en la Figura 54, se observa la pantalla inicial de la primera parte del proyecto, donde se visualizan los dispositivos ya mencionados.

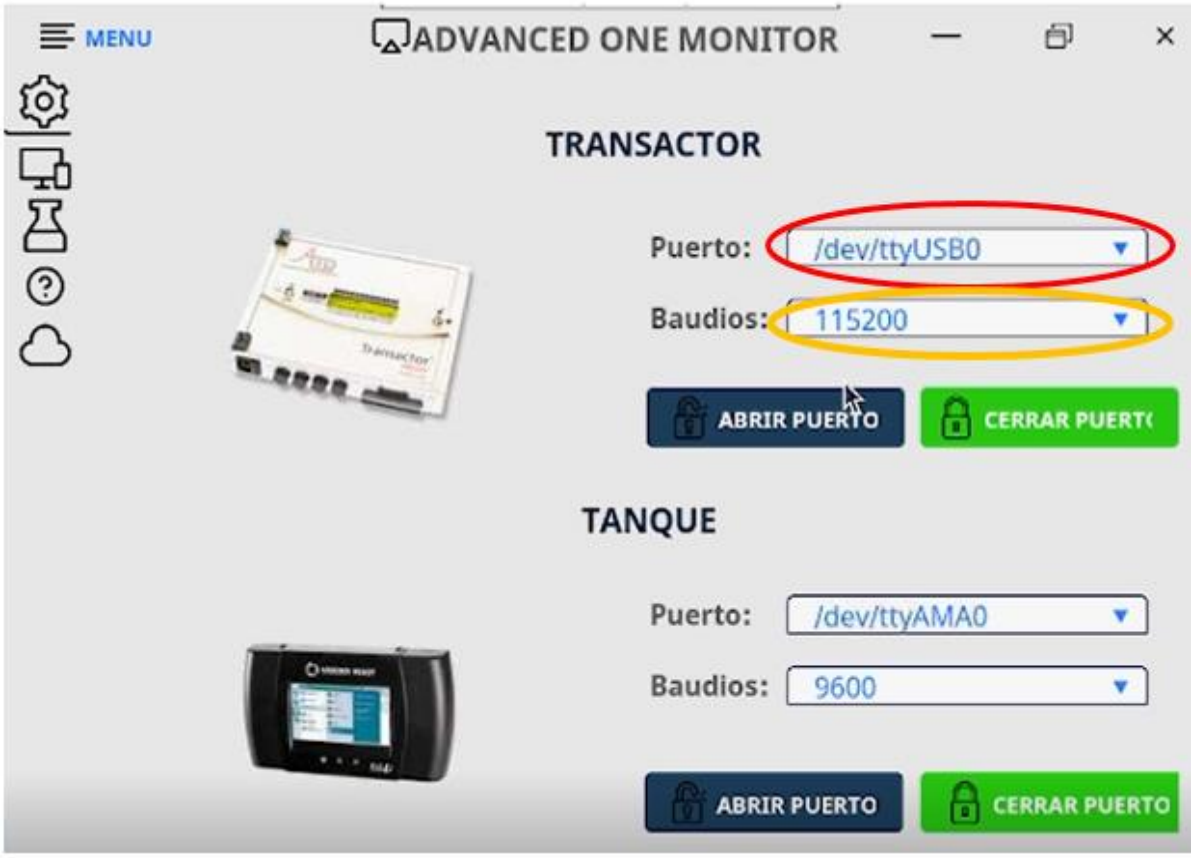
Figura 54.

Pantalla inicial Advanced One

Una vez ejecutada la aplicación de escritorio, el software detecta si alguno de los dos dispositivos se encuentra conectado mediante el puerto serial, e identifica los baudios establecidos, para su correcta comunicación, permitiendo un enlace tanto para la recepción de datos como para el envío de comandos, la conexión se establece mediante el puerto y los baudios seleccionados. A continuación, se observa la Figura 55.

Figura 55.

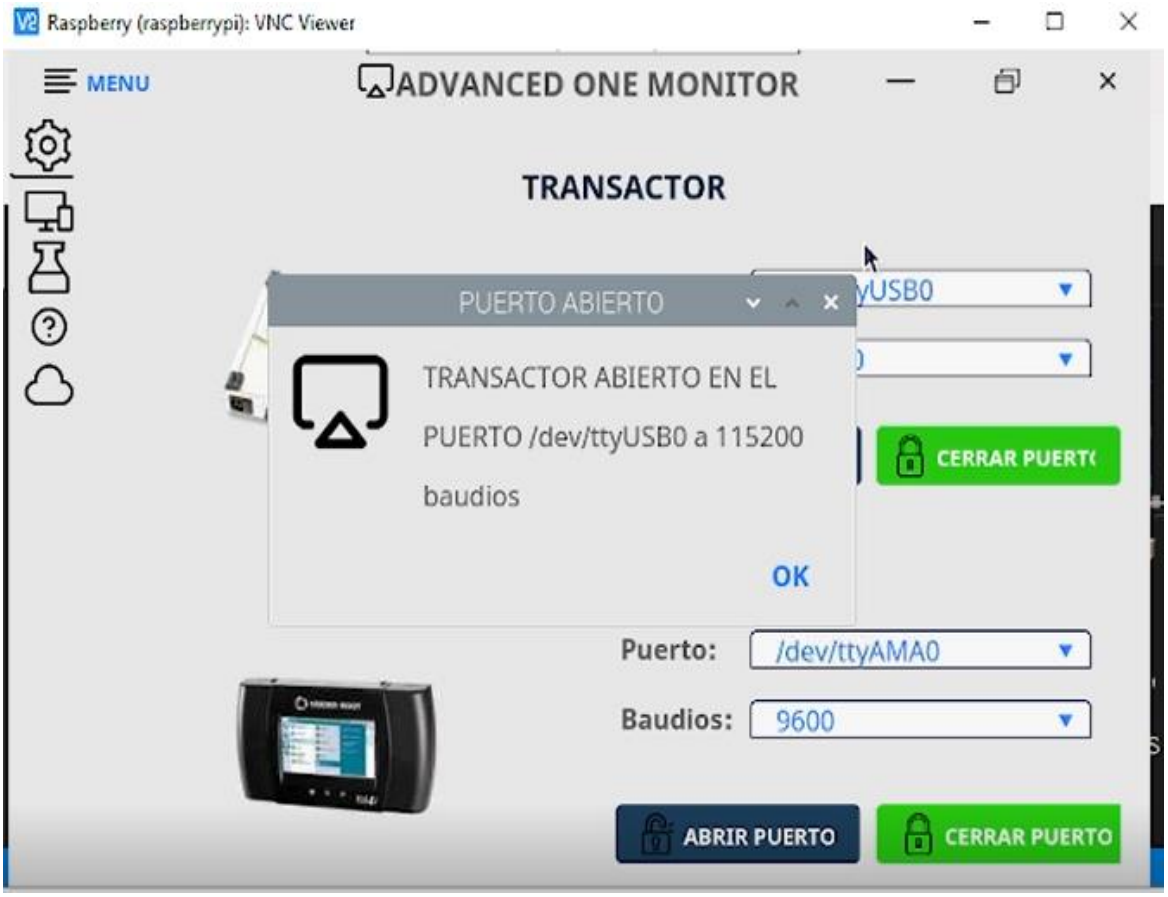
Lectura de puertos y baudios



El siguiente paso para establecer la comunicación entre el mecanismo, y el software alojado en el Raspberry es abrir el puerto del elemento el cual se requiera usar, si los datos identificados tanto del puerto serial como la de los baudios son correctos, la aplicación arroja un mensaje de éxito indicando la información la cual se está usando del dispositivo para comenzar con la interacción, en la Figura 56, se visualiza el proceso mencionado anteriormente.

Figura 56.

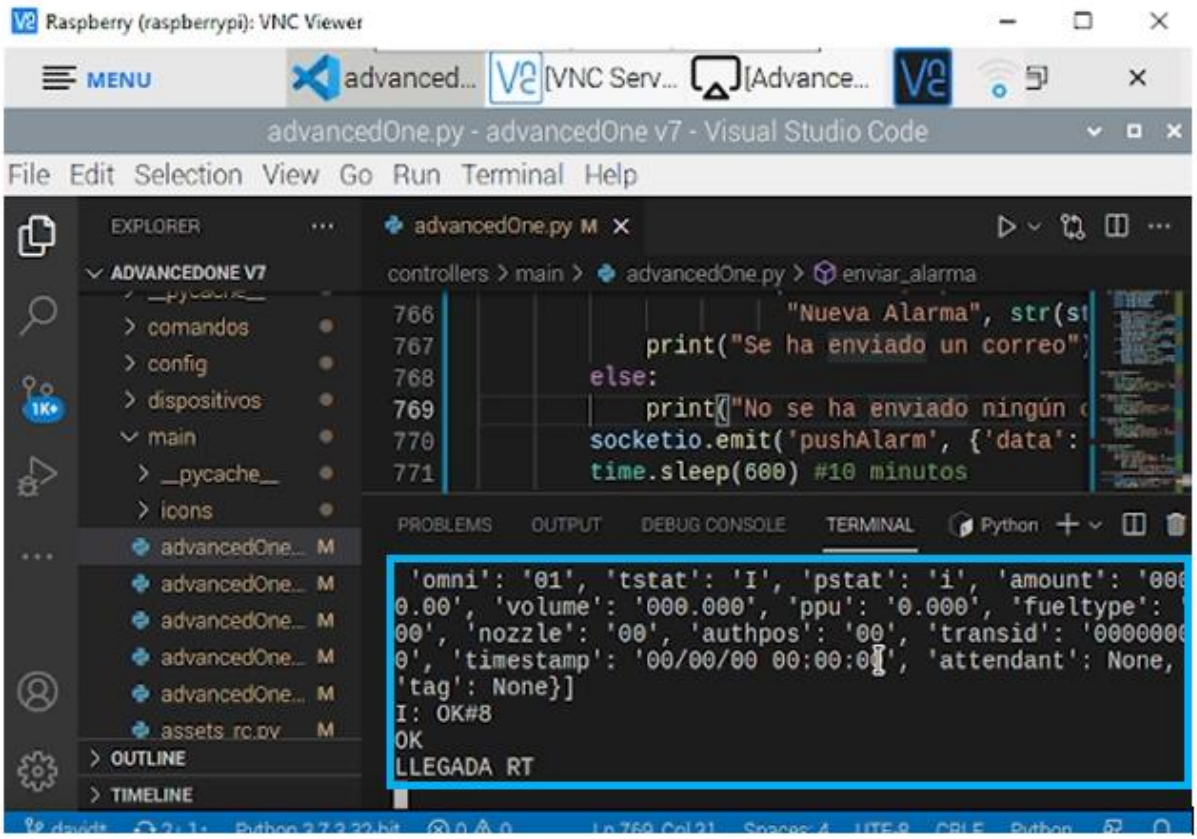
Conexión exitosa con dispositivo



Para comprobar el funcionamiento correcto de la aplicación, en la consola del IDE de desarrollo, se enviaron mensajes de la información que se está recibiendo del dispositivo en este caso del Transactor, los datos llegan de forma poco manejable y por medio del software se la procesa en formato JSON, para que sea más entendible y su gestión sea mucho más sencilla. A continuación, en la Figura 57, se visualiza la impresión de los datos en tiempo real.

Figura 57.

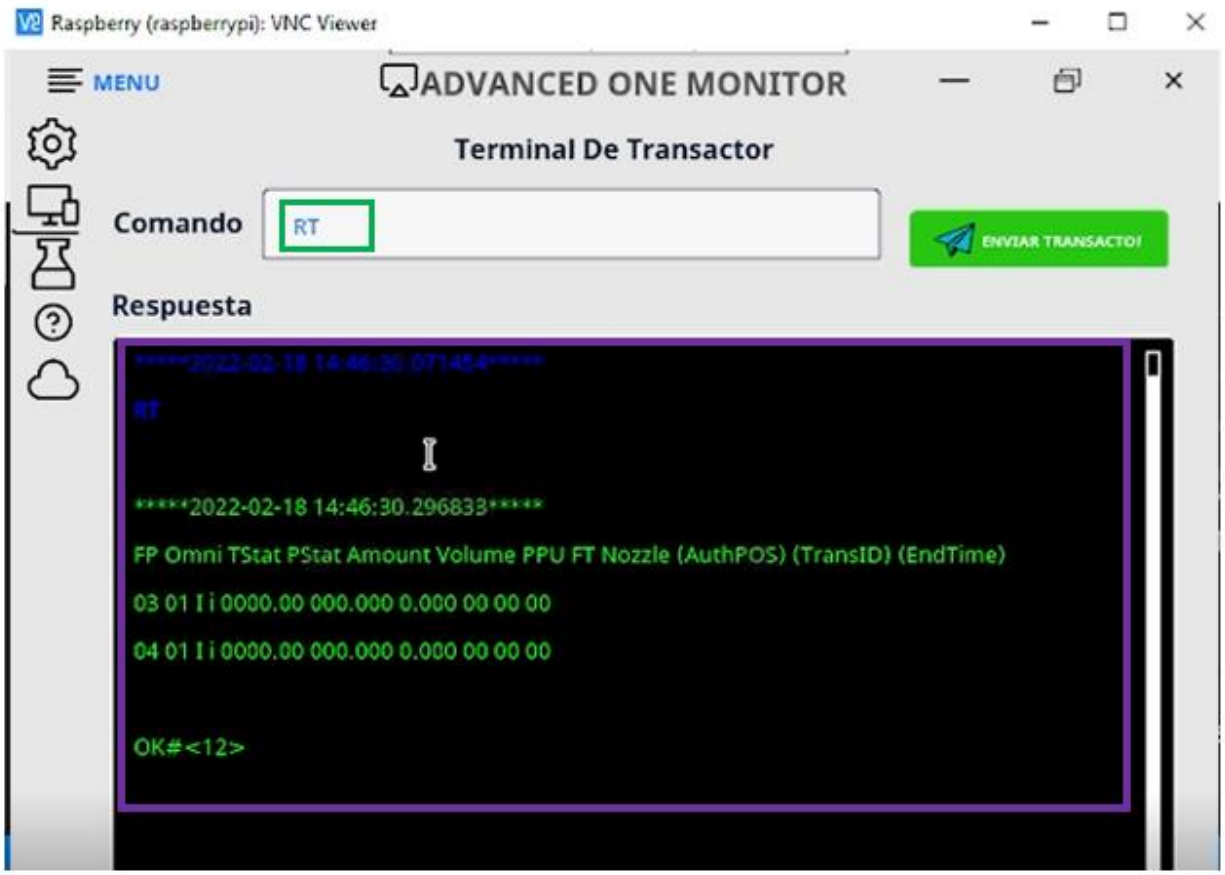
Pruebas en consola del poleo



A continuación, en la Figura 58, se muestra como la interacción tiene que integrar ambos lados, es decir, tanto del Transactor o del Tanque hacia el Raspberry, como del Raspberry hacia cualquiera de los dos dispositivos, se diseñó una ventana de comandos en la cual se puede escribir un comando y el elemento físico responde con información ya procesada en el software, dependiendo del tipo de orden que se haya enviado, en este caso, se escribió el comando RT el cual muestra las ventas que se están ejecutando en ese momento.

Figura 58.

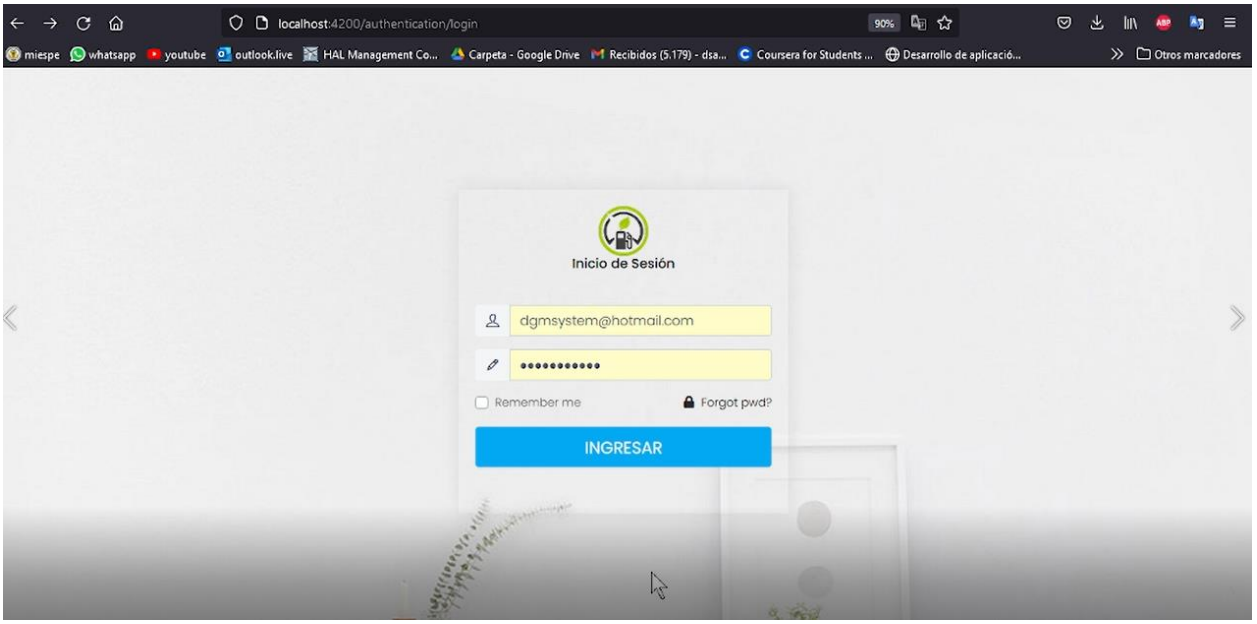
Respuesta exitosa del dispositivo



En la segunda parte del prototipo, se encuentra la parte web, como se visualiza en la Figura 59, la cual, fue desarrollada en el lenguaje Angular JS y es la encargada de transformar y mostrar toda la información enviada por sockets a través de la aplicación de escritorio, en tiempo real. Para el ingreso al aplicativo web se requiere de un correo y una contraseña la cual posee únicamente el administrador del sistema, en este caso el presidente ejecutivo de la estación de servicio “ECO EL OASIS”.

Figura 59.

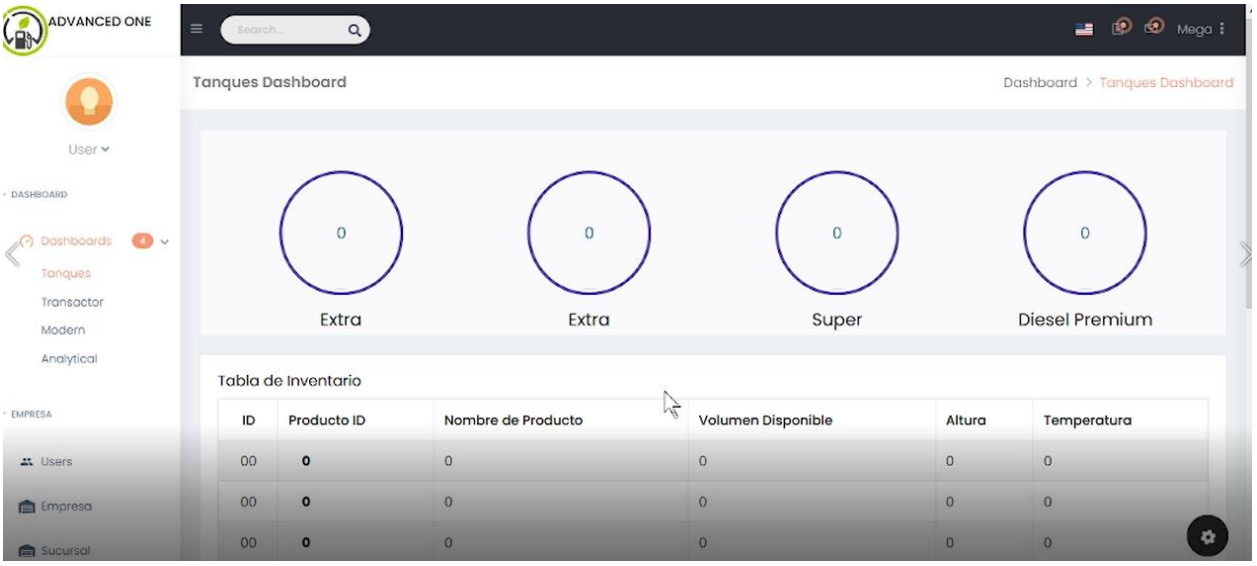
Aplicativo web Advaced One



Una vez ingresado en el software web, se muestra la pantalla para visualización de información de los tanques, en dicha pantalla, por requerimientos de la estación de servicio, se colocaron tres paneles, uno en el cual, se observa la cantidad de combustible de cada tanque que posee la gasolinera, en el segundo panel, una tabla con toda la información detallada respecto a cada tanque, en el tercer panel, las alarmas que pueden saltar dependiendo de la situación de los tanques, y obviamente todo lo mencionado anteriormente en tiempo real. A continuación, en la Figura 60, se muestra la pantalla para la gestión del sistema de Tanques.

Figura 60.

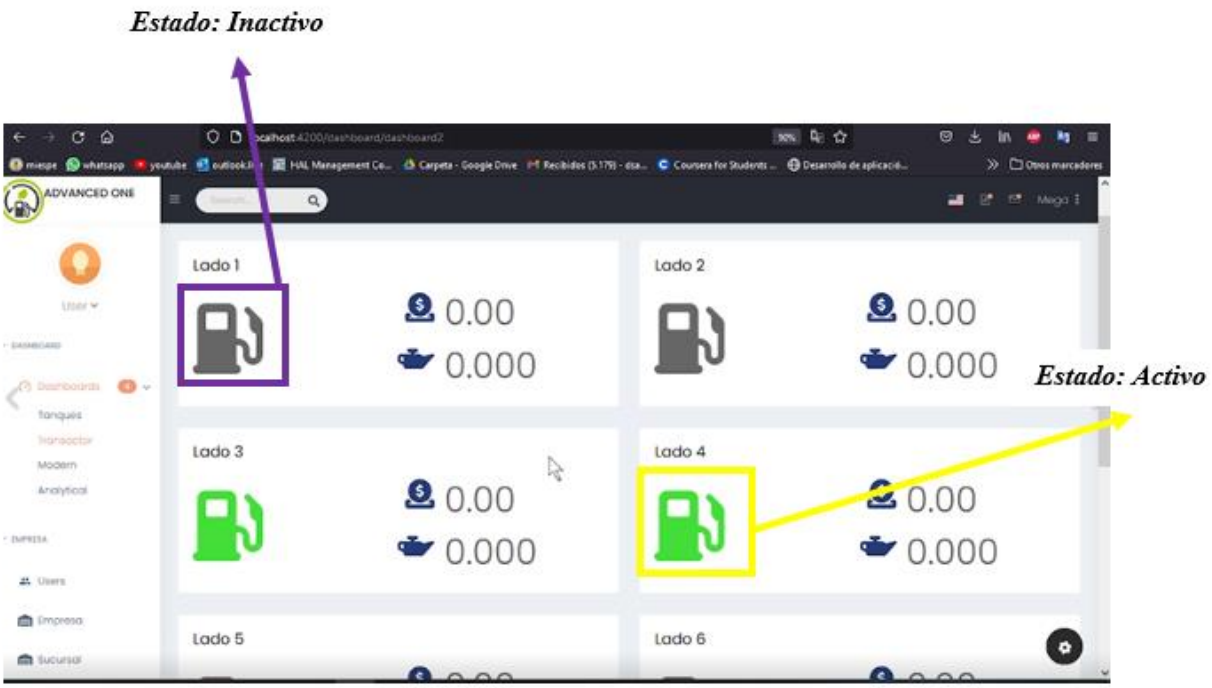
Procesamiento de información en tiempo real sistemas de tanques



Ahora en cuestión al dispositivo Transactor, existe otra pantalla en la que se observan todos los lados a los cuales el mecanismo puede estar conectado, dicho esto, este elemento puede poseer hasta 10 lados, pero en este caso por ser un prototipo y estar conectado a un simulador de pruebas de marca TOKHEIM, solo se encuentran activos dos lados, que es lo que el simulador permite. Los colores utilizados para representar el estado de los surtidores se detallaron de la siguiente manera: verde activo, azul vendiendo, grises inactivos, y el color café que se ha concluido la venta, a continuación, la Figura 61 representativa.

Figura 61.

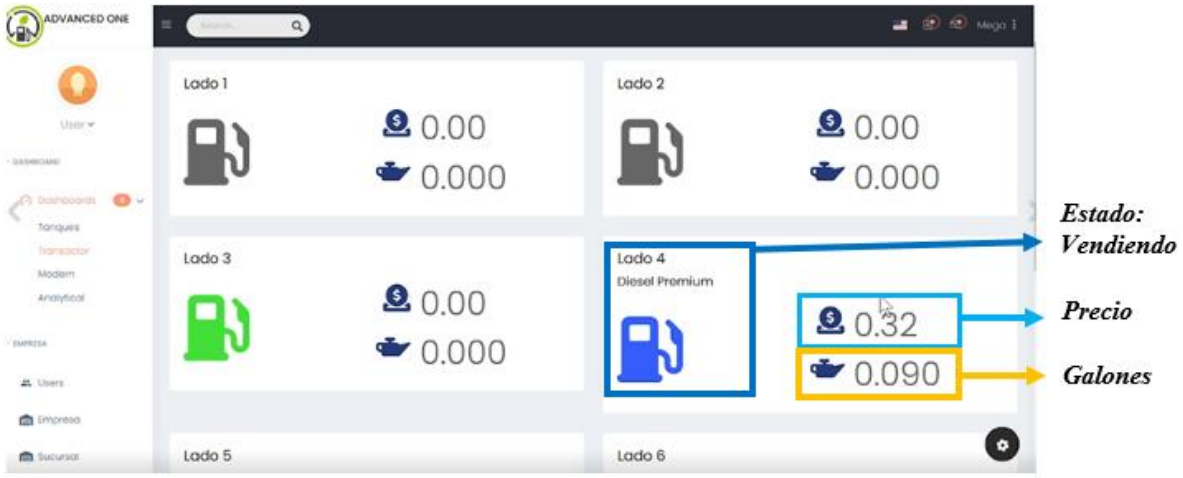
Procesamiento de información en tiempo real venta de combustible



En la Figura 62, se puede observar cómo cambia el color del surtidor dependiendo el estado del simulador, en este caso ya se encuentra vendiendo, por lo cual, se puede observar los galones y el precio de la venta en tiempo real. Cabe recalcar que todo el proceso que se visualiza en la aplicación web está determinado por la aplicación de escritorio que es la que está procesando al momento la información enviada por el dispositivo.

Figura 62.

Surtidor vendiendo



Para verificar que los datos estaban llegando por medio de los sockets, se mandó a imprimir en la consola web el arribo de la información, así como se observa en la Figura 63, allí si observa los datos en formato JSON, y se visualiza el lado del surtidor que se encuentra conectado, en caso de que se esté produciendo una venta, esta varía en la cantidad de galones y el precio, esta información recibida en el formato mencionado, es captada y transformada, por funciones desarrolladas en el lenguaje angular, para que se puedan visualizar y puedan ser entendibles por el usuario.

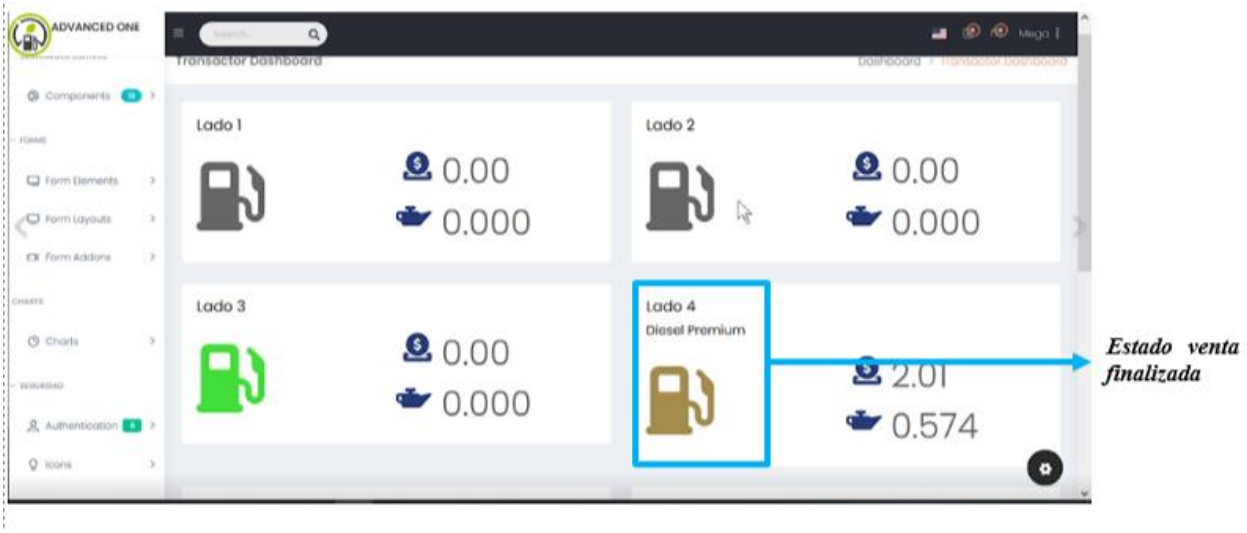
Figura 63.

Recepción de la información en tiempo real

Como se mencionó en los párrafos anteriores, cuando una venta ha sido finalizada, el color del lado del surtidor cambia a café, indicando el nuevo estado, por medio de los colores establecidos es fácil para el usuario identificar la acción que el dispositivo está realizando, los controles establecidos se diseñaron basados en los requerimientos de la estación de servicio “ECO EL OASIS”, a continuación, se observa la Figura 64, en la cual, se detalla lo mencionado en el párrafo.

Figura 64.

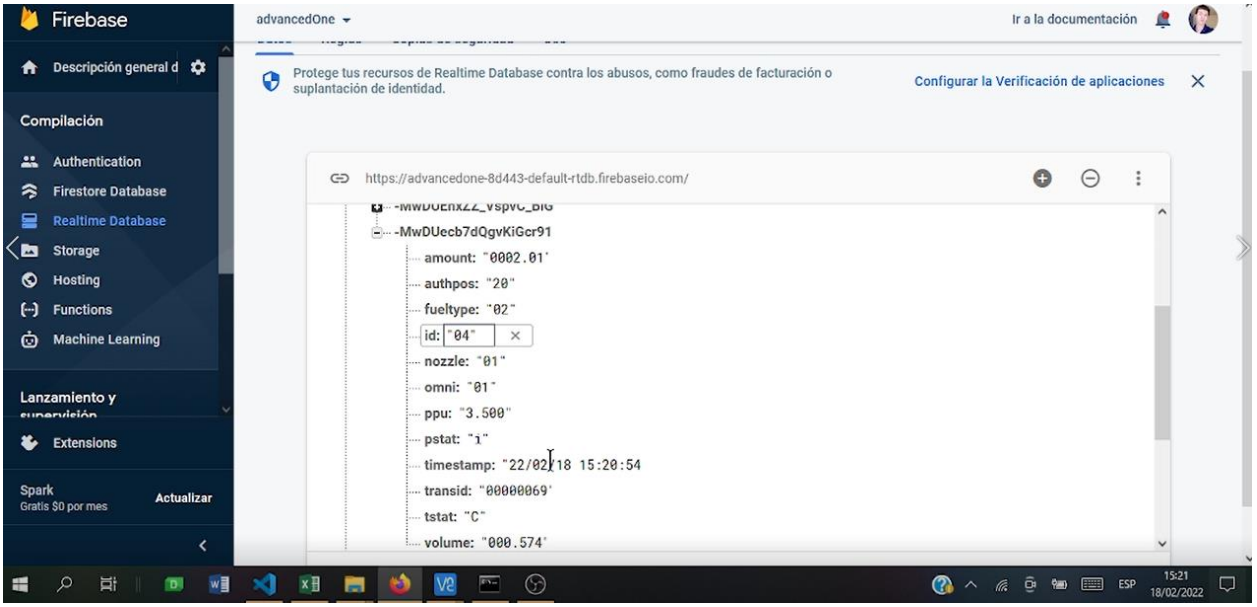
Surtidor venta terminada



En la base de datos no relacional, en la tabla movimiento surtidor, inmediatamente una venta haya cambiado el estado a venta finalizada, los datos se guardan instantáneamente, para tener un control de las ventas realizadas, esto para un proyecto futuro podría servir para llevar un inventario y generar reportes con la información almacenada en la base de datos, el control para que la información se guarde es que el Transactor en los datos que envía debe contener la letra “C”. A continuación, en la Figura 65, se muestra cómo se almacenan los datos en tiempo real.

Figura 65.

Almacenamiento de la información en firebase

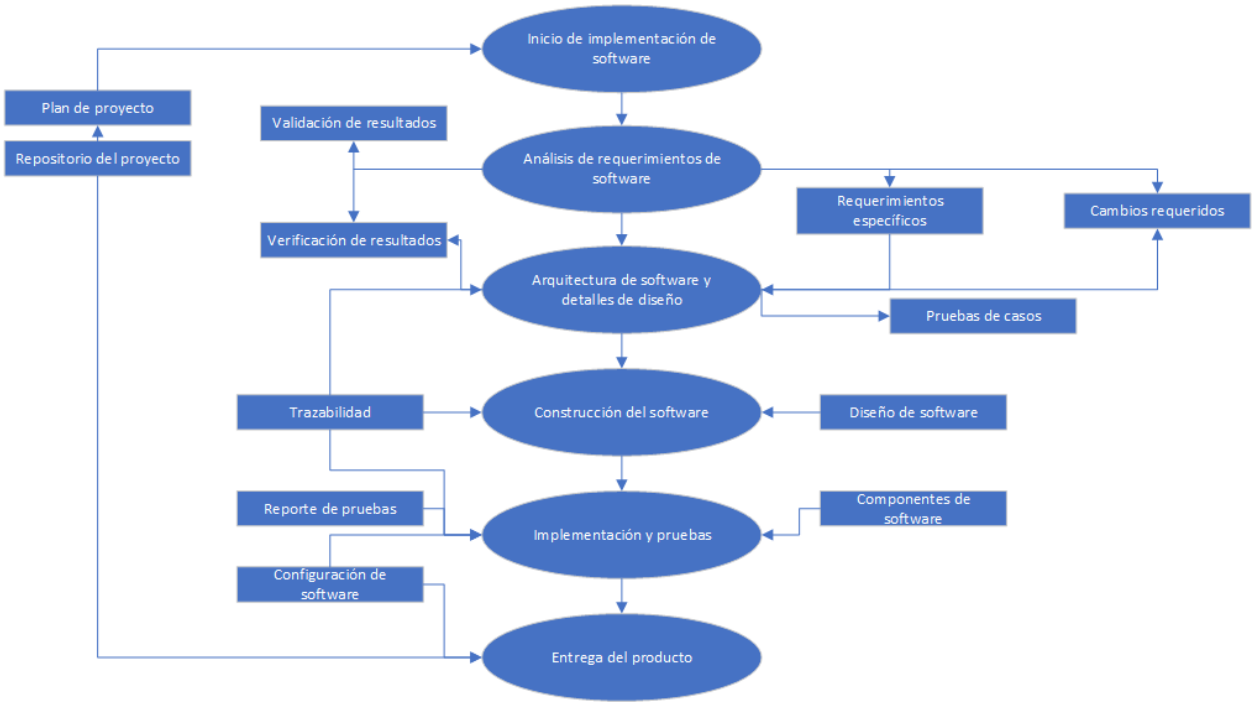


Estándares de la implementación

La estación ECO EL OASIS en la actualidad cuenta con un sistema de gestión el cual no maneja una arquitectura en tiempo real, para poder implementar la nueva aplicación planteada en tiempo real y una arquitectura IOT se usa el estándar ISO/IEC 29110, en el cual se especifica que el desarrollo de software para empresas pequeñas la calidad de los aplicativos que se diseña es el pilar dentro del campo competitivo, para lo cual el uso de las normas de ingeniería de software es bajo. Una de las principales razones por la cual se plantea usar este estándar es por la falta de adopción de normas de calidad, ya que en grandes empresas que se enfocan en el desarrollo de software o multinacionales la percepción de calidad se concibe a una escala mayor.

Figura 66.

Diagrama de implementación del estándar ISO/IEC 29110



Nota. (Sandoval, 2019)

Siguiendo el diagrama de la Figura 66 en este apartado se procede, a detallar el proceso de implementación, ya que, las etapas previas han sido detalladas en capítulos anteriores.

Diagnóstico. Para poder implementar el nuevo sistema, se realizaron pruebas de compatibilidad, funcionalidad y velocidad a la actual infraestructura con la que cuenta la estación, en la cual se encontró que cuenta con un:

- Terminal de despacho y tanques con 10 lados para 5 surtidores de combustible
- Servidor con Windows 7 Profesional Edition
- Base de datos SQLServer Express 2012
- Servicio Web albergado en un servidor Apache
- Aplicativo web diseñado en java para la parte de facturación

- Aplicativo de escritorio diseñado en Visual Basic para la parte contable, de gestión de tanques y despacho de combustible (sin tiempo real)

Al realizar una prueba de despacho mediante este sistema se registró, que el tiempo de diferencia entre llegar la información desde un dispensador es de 4 segundos hasta que se muestra en la aplicación, sin embargo, cumple con la funcionalidad que fue asignado este sistema, ya que nunca se pensó para aplicar conceptos de una arquitectura en tiempo real.

Planificación. Antes de implementar el nuevo software, con base en las conclusiones obtenidas en el diagnóstico, se plantea:

- Implementar la aplicación tanto web (Angular) como local (Python) en un entorno de pruebas con toda la infraestructura que esta conlleva, ya que depende de varios dispositivos interconectados entre sí para que su arquitectura pueda empezar a operar.
- Realizar un soporte correctivo para la estabilización del sistema.
- Implementar una solución para un entorno de producción.

Capacitación. Una vez se tenga clara la planificación de lo que se requiere implementar:

- Entregar un plan de capacitación para administradores y usuarios, según el perfil que requiera para el manejo del software.
- Realizar una capacitación para todos los usuarios.

Soporte. Cuando la aplicación se logre implementar se plantea brindar un soporte a fallas, únicamente de funcionalidad durante un tiempo de 6 meses según se ha acordado con la estación de servicio.

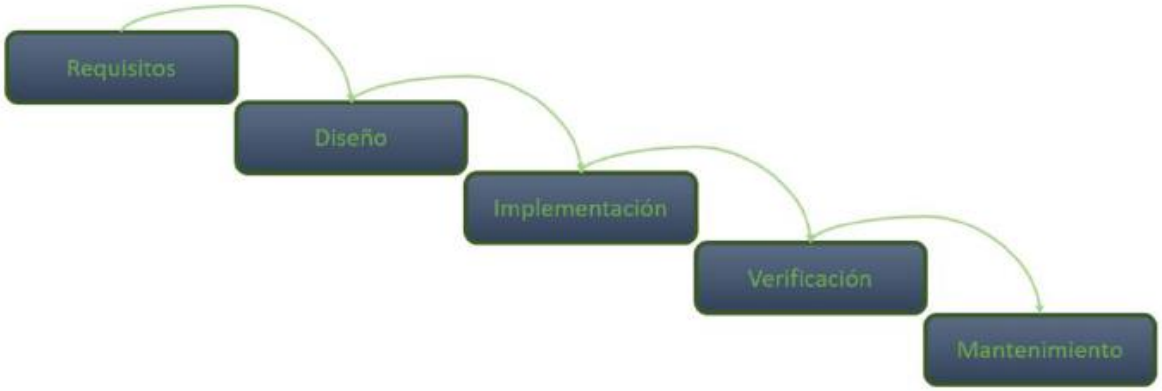
Modelo para el desarrollo del prototipo

Para el desarrollo de este proyecto de tesis, se tomó la decisión de aplicar un modelo en cascada, ya que, acorde al estudio realizado y a los factores como: los requerimientos por parte del presidente ejecutivo, los tiempos establecidos, los dispositivos usados y las necesidades que fueron

surgiendo a lo largo del proceso, resulta ser la más adecuada, eficaz y óptima para el diseño del prototipo propuesto. Las fases fueron establecidas desde un inicio, partiendo de la problemática y los requisitos establecidos por la estación de servicio “ECO EL OAISIS”, a continuación, se observa la Figura 67, la cual, representa el modelo aplicado.

Figura 67.

Metodología cascada



Nota. (Lopez, 2021)

La fase de requisitos nace posteriormente a la problemática establecida inicialmente, la cual, es el alto costo de la infraestructura tecnológica, el mantenimiento de la misma, y la dificultad para el control de las funciones que realiza la estación de servicio “ECO EL OASIS” como la medición de los tanques, sus descargas, alarmas e inventarios y el proceso de venta de combustible. Esta fase se realizó en conjunto con el presidente ejecutivo de la estación, partiendo de los puntos establecidos anteriormente y fundamentándose esencialmente en la infraestructura tecnológica, por lo cual el primer requisito fue reemplazar un servidor por un dispositivo de menor costo, como lo es el Raspberry,

seguido de los requerimientos de los módulos mencionados anteriormente que deben estar presentes en el software nombrado AdvancedOne.

En la fase de diseño, como primer punto se realizó un análisis acerca de los modelos de arquitectura tecnológica IoT(Internet de las cosas), para poder seleccionar el más adecuado acorde al prototipo propuesto, una vez elegido el modelo arquitectónico, se procedió a estructurar una base de datos no relacional, la cual está basada en el modelo de negocio de la estación de servicio “ECO EL OASIS”, con el fin de almacenar y gestionar las tramas enviadas por los dispositivos Transactor y Veeder-Root, de acuerdo a los requisitos y a las necesidades establecidas por la estación, se establecieron módulos los cuales se diseñaron para la conexión y comunicación, transformación y procesamiento de la información y para el almacenamiento de la misma.

Para la fase de implementación, con un modelo de arquitectura ya definido, con una base de datos no relacional ya estructurada, con los dispositivos y elementos que forman parte del prototipo y con los módulos establecidos para la ejecución de los procedimientos requeridos por la estación, se inició está fase colocando el software AdvancedOne en el Raspberry, el software se lo desarrolló por medio del lenguaje Angular y Python, uno para la parte web y el otro para la ejecución de las funciones de medición de tanques y la venta de combustible. La implementación consto de la conexión de los dispositivos tanto Transactor como Veeder-Root, hacia el Raspberry, para que haya una correcta interacción, por medio de los protocolos de comunicación concerniente a cada elemento.

Para la fase de verificación, se constató con el Presidente Ejecutivo la funcionalidad del prototipo desarrollado, validando los requisitos con cada uno de los procesos y los módulos que forman parte del software AdvancedOne, para la realización de las pruebas y las verificaciones, se usó el simulador Tokheim, el cual, realizaba la función de surtidor con dos lados en la estación de servicio, en cada lado colocando un producto diferente, ya sea extra, súper o diésel, ejecutando pruebas en tiempo

real y verificando los valores obtenidos por el software, con los valores recibidos por el simulador, y en el caso de los Tanques, verificando la información de las descargas e inventarios que posee el Veeder-Root, a la cual se puede acceder por comandos.

La fase de mantenimiento es una etapa, la cual se la puede realizar para un proyecto y prototipos futuros hablando en profundidad, en cuanto al prototipo propuesto, se la fue depurando acorde a las necesidades que fueron surgiendo a lo largo del desarrollo, manteniendo reuniones 3 días a la semana, y verificando los avances los días viernes, el mantenimiento se lo realizó en cada módulo debido a que era esencial, que cada proceso se ejecute de manera eficiente y adecuada, ya que, todas las funciones y métodos desarrollados se enlazan formando un solo elemento.

Estándares de comunicación basado en protocolos

Dentro de la arquitectura planteada para el software AdvancedOne se usó el protocolo TCP/IP como principal transmisor y gestor de datos entre todos los equipos interconectados de la arquitectura del software, pero además de ello se hizo uso de varios otros protocolos que son los encargados de ayudar a llevar a cabo operaciones de comunicación exitosamente, entre los cuales son:

HTTP. También conocido como Hypertext Transfer Protocol. Este protocolo es el encargado de la transferencia de documentos de hipertexto (HTML). Al estar basado en texto plano permite que sea legible e interpretable al momento de depurar el software, esto se vió reflejado al momento en que el software usa la parte web diseñada en AngularJS donde para poder consumir la información que llega desde el aplicativo en Python mediante sockets, hace uso del protocolo HTTP con sus principales operaciones GET y POST, de tal manera que los datos lleguen de manera fiable y estos posteriormente fueron graficados donde el usuario final puede interpretarlos.

I2C. Este protocolo es el encargado de establecer una comunicación serial sincrónico entre los principales componentes del simulador Tokheim, al estar compuesto de sensores digitales necesita que

dentro de su arquitectura pueda tener una confirmación de los datos recibidos del exterior dentro de la misma trama de información.

UART. Después del protocolo TCP/IP como el principal gestor de comunicación el receptor-transmisor asíncrono universal (UART) es el segundo protocolo más importante dentro de la arquitectura, ya que, este es el encargado de que la comunicación entre el dispositivo IOT (Raspberry Pi) y tanto el Transactor como el sistema de tanques Veeder-Root sea de manera asincrónica, lo que quiere decir que no existe una señal clock para sincronizar, sino que esto se realiza mediante una secuencia de bits para definir el inicio y el fin de un paquete de datos transferidos, asi como se visualiza en la Figura 68.

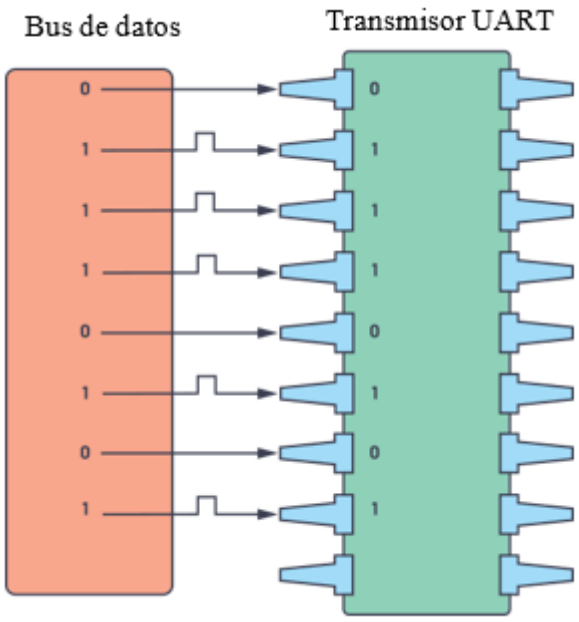
Figura 68.

Paquete de transmisión de datos dentro del protocolo UART

El proceso de transmisión UART empieza cuando el transmisor recibe los datos en paralelo desde el bus de datos, como se visualiza en la Figura 69, a continuación, la unidad transmisora UART agrega un bit de inicio, un bit de paridad, y un bit final a el marco de datos, así como se muestra en la Figura 70, una vez completa la trama se envía en serie empezando por el bit de inicio hasta el bit final desde la UART transmisora hasta la UART receptora, todo esto mientras la UART receptora muestrea los datos a la velocidad de transmisión (baudios) pre configurada por el dispositivo, una vez que la UART receptora tiene la trama descarta los bits de inicio, paridad, fin y convierte los datos en serie en datos en paralelo y los transfiere al bus de datos de parte de un dispositivo receptor, así como se muestra en la Figura 71.

Figura 69.

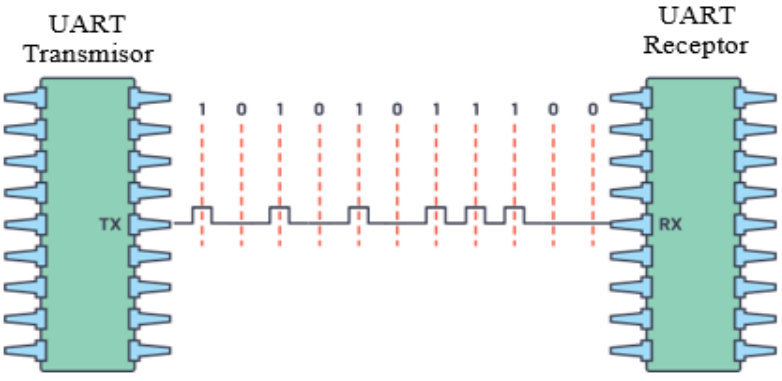
Bus de datos hacia el transmisor UART



Nota. (León, 2021)

Figura 70.

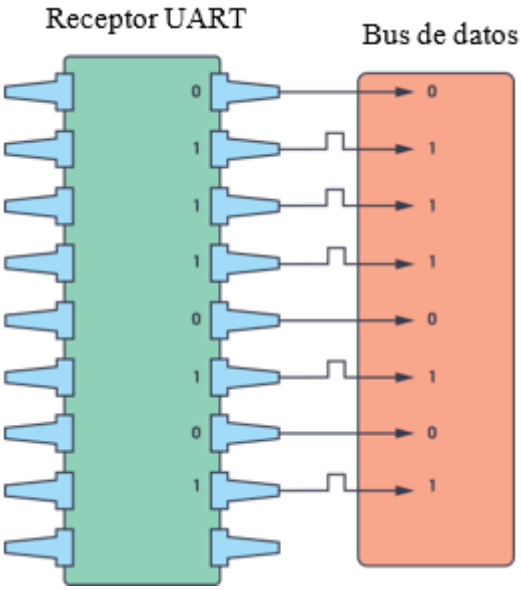
Transmisión UART



Nota. (León, 2021)

Figura 71.

UART receptor al bus de datos del dispositivo receptor



Nota. (León, 2021)

Pruebas

Plan de pruebas

Las pruebas establecidas para verificar la funcionalidad y eficacia del prototipo se fundamentaron básicamente en los tiempos de respuesta al momento del procesamiento de la información, en la validación acerca de los datos que eran enviados por los dispositivos tanto Transactor como Veeder-Root, constatando que habían sido transformados y mostrados correctamente y en el monitoreo de los métodos programados, a través del mismo software AdvancedOne.

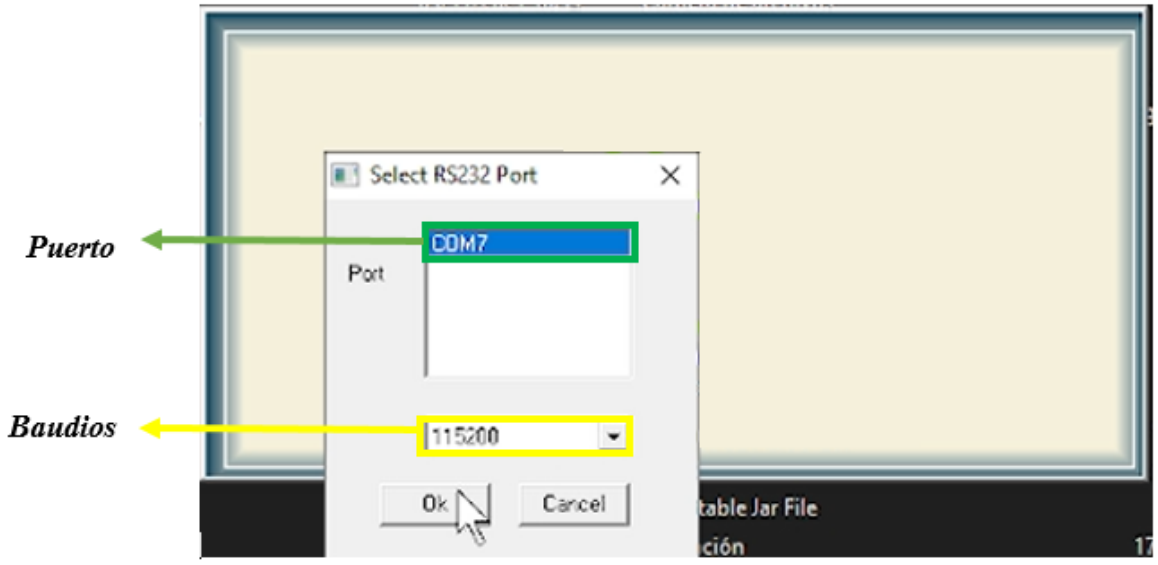
Inicialmente, una de las pruebas que se tenían previstas realizar, conforme se avanzaba con el proyecto y los módulos se iban desarrollando, era acerca de la conexión y lectura acerca del puerto y de los baudios del dispositivo; en el caso específico del Transactor, este elemento posee su propio software

de configuración en el cual se puede gestionar los ajustes que posee, así como consultas, envío y recepción de comandos, verificación y establecimiento de los lados, el formato de los precios etc.

En la Figura 72, se puede observar cómo al ejecutar la aplicación propia del dispositivo Transactor, lee de manera inmediata el puerto y los baudios que el mismo posee. Así fue, como se probó y se diseñó el software AdvancedOne, basado en sus características para que el funcionamiento y los procedimientos estén acorde a la operatividad del elemento, constatando que la lectura de los datos que estén preestablecidos sea correcta y exista una interacción adecuada.

Figura 72.

Prueba de conexión



Las pruebas se ejecutaron con los dispositivos conectados tales como el Transactor y el Tokheim, ya que es indispensable para validaciones y verificaciones, que exista un surtidor del cual se obtenga datos de los galones y del precio, ese papel está estructurado para el simulador de la marca ya mencionada, para constatar que la información de los lados, la información de las ventas, el tipo

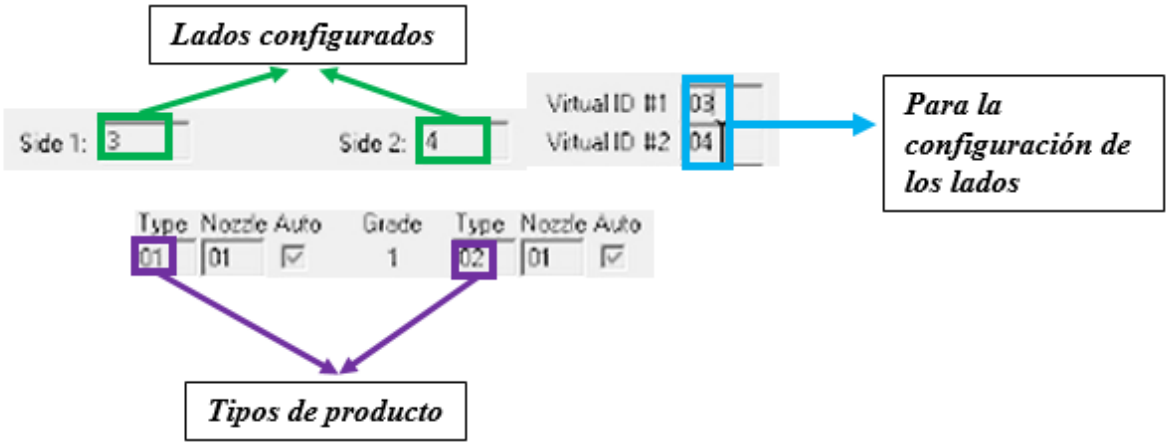
surtidor, y el formato del precio sean válidos y verdaderos. A continuación, en la Figura 73, se muestra como el software reconoce inmediatamente el simulador.

Figura 73.

Detección del modelo del surtidor

Figura 74.

Prueba de detección de lado y de producto

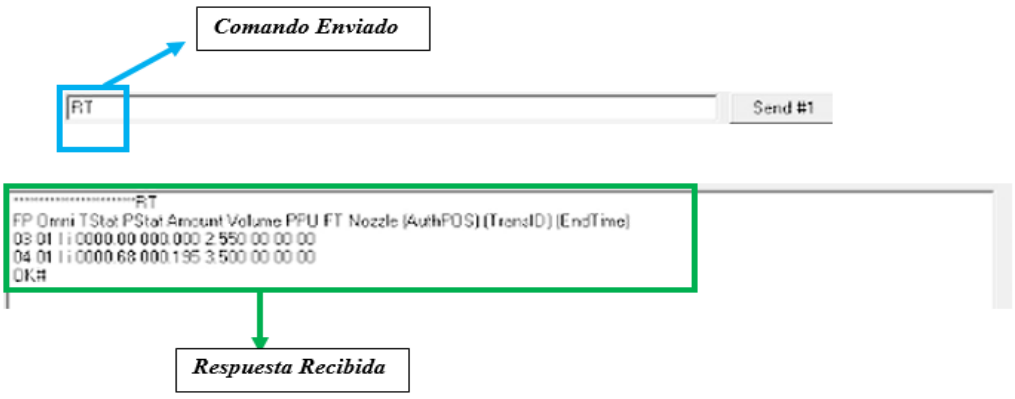


Así como el software del propio elemento Transactor detecta los lados que se encuentran preestablecidos y hacen comunicación con el simulador, de la misma manera opera el software AdvancedOne, ya que en la parte web, muestra cuales son los lados que se encuentran activos y cuáles no, de la misma forma, ocurre con los productos que ya se encuentran configurados, para validar que la configuración se visualiza tanto el sistema nativo, como en el proyecto realizado, es que, en la visualización del sistema desarrollado se indica que tipo de producto posee cada lado del surtidor, en la Figura 74, se puede apreciar los lados conectados, y el tipo de producto.

Para la validación de la información que emite el dispositivo Transactor, también se utilizó el mismo aplicativo, en el cual, existe un recuadro para escribir un comando y enviarlo, y posteriormente se muestra una pantalla con la trama procesada, fue así como, se logró constatar que la transformación que se realizó mediante las funciones diseñadas en el software AdvancedOne eran correctas. En la Figura 75, se observa un ejemplo de lo mencionado anteriormente.

Figura 75.

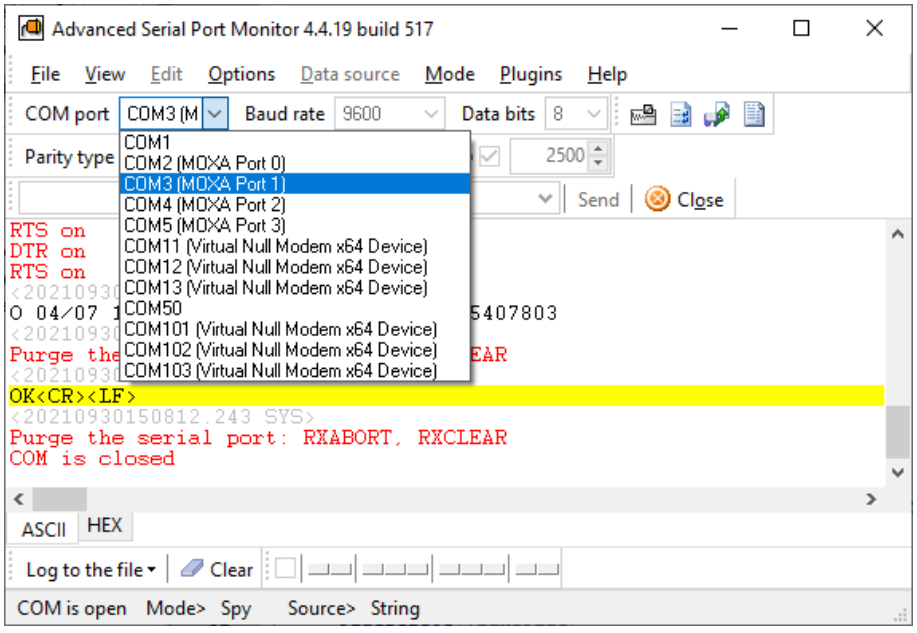
Prueba de envío y respuesta



Y por último para la validación tanto del dispositivo Transactor como del dispositivo de medición de tanques Veeder-Root, se utilizó otro sistema, el cual, actúa de forma similar al mencionado anteriormente, con la funcionalidad de detectar el puerto y los baudios que se encuentra conectado mediante el cable serial, ofreciendo la capacidad de enviar y recibir comandos. Este aplicativo se usó más para el caso de los tanques, ya que el anterior sistema es específicamente para las funciones del Transactor. A continuación, en la Figura 76, se muestra la consola del aplicativo Advanced Serial Port Terminal.

Figura 76.

Prueba sistema de tanques



Casos de Prueba

Para establecer la efectividad del software implementado, se ha realizado un esquema de casos de prueba (Tabla 37.) en el cual se evalúan distintos escenarios en el cual el sistema fue puesto a prueba donde se esperó distintos resultados dependiendo de acciones o entradas que el usuario estableció, de esta manera llegando a concluir si determinado caso de prueba ha sido o no exitoso y con ello, determinar el grado de efectividad al momento en que el proyecto está siendo ejecutado en la estación "ECO EL OASIS", y de esta manera establecer si el sistema se adapta con las necesidades, requiere mejoras o se encuentra listo para una implantación.

Tabla 37.

Tabla de casos de prueba implementados para el software AdvancedOne

Descripción	Fecha	Área Funcional	Funcionalidad	Entrada	Resultado Esperado	Dependencia de otros casos de prueba	Resultado obtenido	Estado
Despliegue de aplicación	17/01/2022	Módulo web	Ingreso a la aplicación mediante un browser	http://localhost:4200	Página del login del sistema AdvancedOne	-	Visualización del login del sistema AdvancedOne	Exitoso
Ingreso de usuario y password correcto	17/01/2022	Módulo web	Ingreso al dashboard del Transactor	User: dgmsystem@hotmail.com Pass: ******	Página del dashboard del Transactor	1	Visualización del dashboard del Transactor	Exitoso
Recepción de datos (Transactor)	21/02/2022	Módulo web y comunicación	Despliegue de gráficas con datos recibidos desde el Transactor mediante aplicativo Python (sockets)	-	Gráfica de al menos un dispensador de combustible en color azul, variación en los valores de cantidad y dinero si se está despachando combustible en el simulador	1, 2	Gráfica de color azul de los dispensadores 3 y 4 (no existe variación en la cantidad y valor porque no se estaba despachando al momento de la prueba)	Exitoso

Recepción de datos (sistema de tanques)	21/02/2022	Módulo web y comunicación	Despliegue de gráficas con datos recibidos desde el sistema de tanques mediante aplicativo Python (sockets)	-	Gráfica del inventario de tanques dependiendo del volumen de combustible, tabla de inventario y descargas actualizadas	1, 2	Gráfica del inventario de tanques dependiendo del volumen de combustible, tabla de inventario y descargas actualizadas	Exitoso
Envío y respuesta mediante un comando (Transactor)	21/02/2022	Módulo de comunicación	Envío de un comando propio del Transactor mediante el aplicativo Python (comunicación serial)	RT	Comando en formato: 00 00 A a 000000000000 00 00 00 00 00	-	04 01 I i 000000680000 195 3500 00 00 00	Exitoso
Envío y respuesta mediante un comando (sistema de tanques)	21/02/2022	Módulo de comunicación	Envío de un comando propio del sistema de tanques mediante el aplicativo Python (comunicación serial)	i20100	Comando en formato: i201 TT YY MM DD HHmm TT p ssss NN FFFFFFFF && CCCC	-	i201 01 22 02 21 1633 01 1 0000 01 429C3436458A8B && 80F7	Exitoso

Evaluación de Resultados

A través del proceso de desarrollo del prototipo AdvancedOne, para la integración de información en tiempo real, con el fin de gestionar y controlar el combustible, por medio de controladores industriales y el sistema de medición de tanques, se obtuvieron los siguientes resultados, que detallan de manera puntual los alcances, los fallos y las posibles mejoras que se pueden realizar en futuros desarrollos, teniendo en mente, la construcción de un sistema y un entorno mucho más completo acorde a las necesidades que puedan ir surgiendo en el camino.

Integración del Raspberry con los dispositivos Transactor y Veeder-Root

Como punto esencial y primera parte del proyecto, la interacción que hubo entre el Transactor y el Raspberry se dio de manera exitosa, debido a que, los dos dispositivos respondían de manera adecuada tanto en la recepción como en el envío de datos, por medio del Software AdvancedOne. No se pudo constatar una diferencia considerable entre el procesamiento de información entre el servidor de las estación de servicio "ECO EL OASIS" con el Raspberry PI, debido a que, el simulador Tokheim que se utilizó para el proceso de venta de combustible, únicamente permite la interacción con dos lados, y en el caso de un servidor que procesa datos reales por lo general interactúa con al menos 4 lados, en este caso de estudio específico la estación trabaja con 8 lados es decir 4 surtidores, por lo que la magnitud es totalmente diferente a la que se empleó para el trabajo con el Raspberry.

Y en el caso de la integración con el sistema de medición de Tanques, Veeder-Root, el Raspberry se enlazó de manera perfecta, ya que interactúa de forma similar a la del dispositivo ya mencionado con el servidor, los datos que se visualizaban tanto en el servidor como en el mismo dispositivo era totalmente iguales a las que se observan en el software AdvancedOne colocado en nuestro elemento IoT(Internet de las cosas), el procesamiento de daba en tiempo real, pero por requisitos del presidente ejecutivo, se establecieron tiempos para los procesos desarrollados, como por ejemplo en el caso de las descargas, que se pidió que se actualicen después de 10 minutos.

Raspberry Pi como solución de bajo costo

El Raspberry Pi, puede ser considerado como una alternativa a soluciones de infraestructura tecnológica de bajo costo. Dicho esto, en el proyecto de tesis realizado para le estación de servicio "ECO EL OASIS" se obtuvieron los siguientes resultados partiendo de la premisa mencionada. El rendimiento que ofrece el dispositivo Raspberry Pi 4, el cual, se usó para la realización del prototipo, es un hardware posicionado en el mercado como un competidor estrella, con las capacidades para simular un servidor

de características básicas, una de las características que se consideró importante el inició del desarrollo, fue la velocidad de conexión con internet, debido a que, el software que se realizó consta de una parte web, en la cual, se visualizan los procesos relacionados con la gestión del combustible, debido a estos motivos, se tomó en cuenta el puerto Gigabit Ehternet que posee el dispositivo, el cual, proporcionó velocidades de 105 MB/s sin problemas.

Es importante destacar que el modelo usado es poderoso y su comportamiento fue eficiente para el desarrollo del prototipo, teniendo en cuenta el precio del mismo, a pesar de que, existieron momentos en los cuales, el Raspberry sufrió de desconexiones o de estancamientos cuando se intentaba modificar código dentro del mismo dispositivo, ya que, a que diferencia de los servidores, no posee un disipador que ayude controlando la temperatura, sin embargo, su eficiencia escalo de manera considerable, cuando se le instaló un ventilador. Otra característica que jugó un papel importante para considerar al Raspberry como el servidor oficial de la estación de servicio “ECO EL OASIS”, fue el rendimiento concerniente a la escritura y lectura de datos que se realizó a través de los puertos que posee el elemento, ya que, su velocidad fue de máximo 3.5 Gbps, que, comparándolo con el servidor actual, el cual, posee una velocidad de 5.6 GBS, crea una brecha bastante grande.

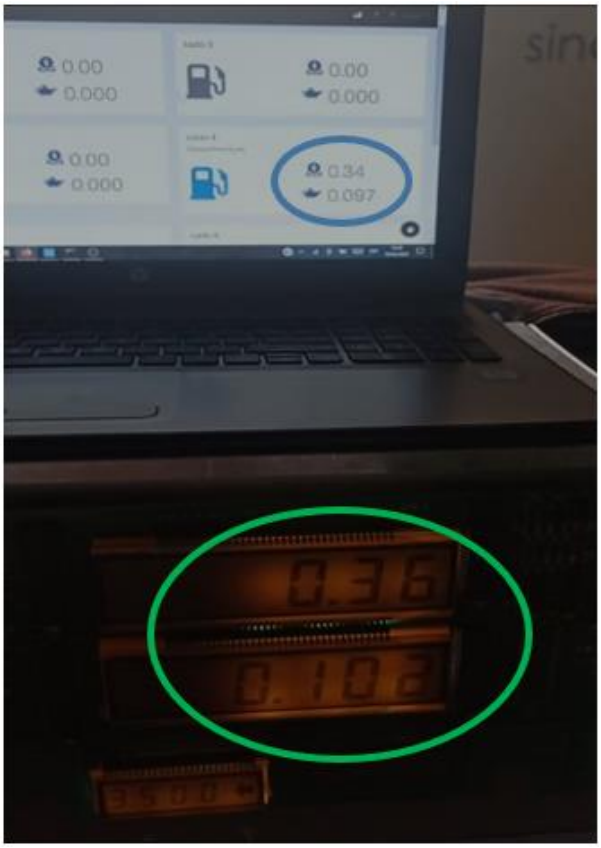
Resultado del uso de la Arquitectura IOT de 7 capas

La arquitectura implementada en el software ahora denominado AdvancedOne el cual está regido bajo un modelo de 7 capas basado en un componente IOT que es el Raspberry Pi ha demostrado un rendimiento bastante satisfactorio según el gerente de la estación ECO EL OASIS, sin embargo al momento de encaminar el proceso de diseño e implementación de esta capa se han visto distintas debilidades, especialmente en el Raspberry, ya que, el dispositivo es bastante versátil y cumple con su función de reemplazar a un costoso servidor, sin embargo, al ser un ordenador de placa reducida tiene limitaciones como una sobrecarga de operaciones en su procesador que va como máximo a 1.5Ghz, lo

que es bastante poco en comparación con el servidor actual de la estación que cuenta con un procesador Intel Atom D2700 con 2.13Ghz de capacidad de procesamiento, como resultado de esto al momento de querer desplegar la aplicación Python en el Raspberry esta se ve retrasada por segundos hasta finalmente lanzarse y preparar los sockets, pero esto se ve compensando al momento que la aplicación ya está en ejecución no existe ningún tipo de retraso de envío o recepción de información cumpliendo así con los conceptos de aplicativo en tiempo real como se puede evidenciar en la Figura 77.

Figura 77.

Transacción en tiempo real desde el simulador hasta el software AdvancedOne



Se puede evidenciar más a profundidad los resultados que provee la nueva arquitectura del software en el sistema de tanques Veeder-Root, donde desde la capa 1 y 2 se evidenciado el desenvolvimiento de este modelo donde mediante el protocolo UART se puede interconectar el sistema de tanques al Raspberry y gracias al aplicativo Python se abre un puerto serial para intercambiar datos en doble vía y en tiempo real, donde el máximo retraso que existe es de 2 centavos a comparación del simulador lo que en términos de "procesamiento en tiempo real" es bastante aceptable ya que es un desfase imperceptible para el ojo humano.

Si se requiere un enfoque en cosas que sean totalmente visibles para el usuario final sencillamente se puede comparar las interfaces del dashboard para el sistema, ya que el software AdvancedOne tecnologías de frontend de la actualidad como es AngularJS y el sistema actual de la estación usa tecnología basada en Visual Basic, la misma que se encuentra descontinuada a comparación con los nuevos entornos que existen en la actualidad, todo esto en la capa 6.

Finalmente, el uso de sockets para la transmisión de tramas de información, esta tecnología permite que la información sea confiable y al usar específicamente Socket.IO la comunicación que existe es bidireccional, ya que, este componente selecciona automáticamente el mejor protocolo de comunicación entre la aplicación servidor y el cliente que recibe la información para graficarla en el sistema de tanques como se puede ver en la Figura 78 y en la Figura 79.

Figura 78.

Software del sistema de tanques actual de la estación ECO EL OASIS

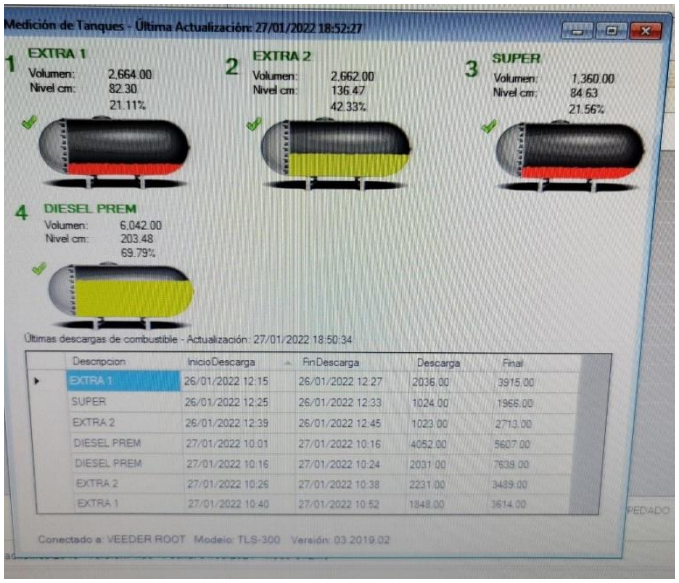
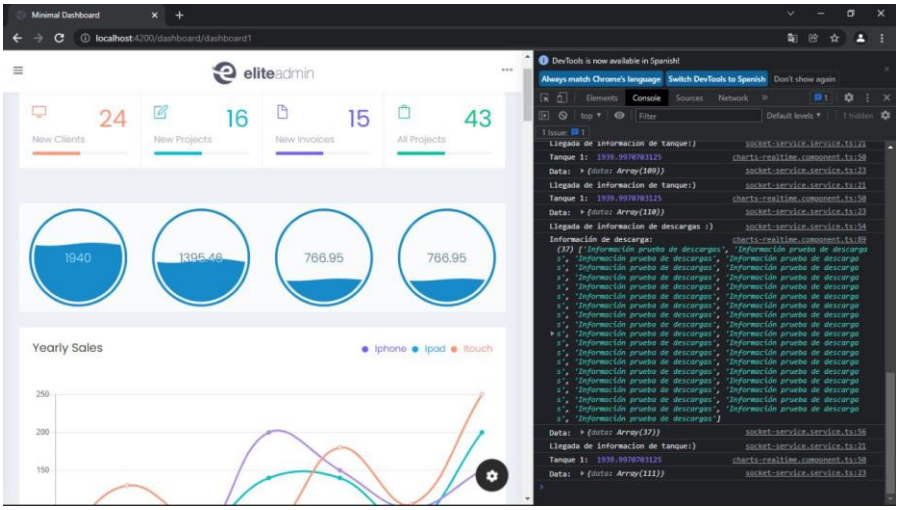


Figura 79.

Software del sistema de tanques con arquitectura de 7 capas en tiempo real



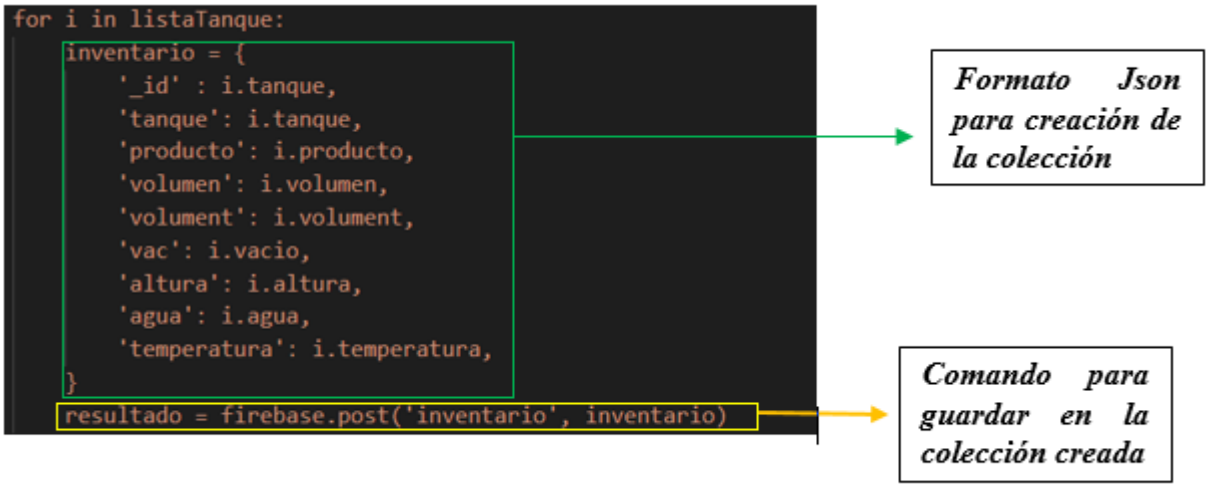
Integración del software AdvancedOne con una base de datos No Relacional

El gestor de base de datos que se usó para el almacenamiento de la información enviada por el prototipo desarrollado fue Firebase, que gracias a sus funcionalidades se acopló de manera adecuada, a las necesidades y requisitos establecidos por la estación de servicio “ECO EL OASIS”. La interacción que posee Python y las librerías existentes del lenguaje mencionado, fueron claves tanto para la conexión con la base, como para la creación y almacenamiento de los datos. El tipo de base que se uso fue una de tipo Real Time, una de las herramientas más novedosas que posee Firebase, ya que, gracias a esta funcionalidad, no es necesario actualizar la página para ver algún cambio realizado en cualquier colección, más bien, cualquier cambio ejecutado se visualizaba de manera inmediata en la consola de la base, ayudando de esta manera a la realización de pruebas para capturar los datos de forma instantánea. Una de las ventajas que se pudo obtener de esta herramienta era la facilidad para crear colecciones, y en cuestión a la codificación no fue nada complicado adaptarse al formato el cual requiere la base.

En la Figura 80, se puede observar un ejemplo de cómo se realizó el almacenamiento de la información, este ejemplo fue únicamente realizado para ejecutar una prueba tanto para la creación de una colección de forma inmediata como para guardar los datos enviados, ya en la codificación adecuada, se efectuó de manera distinta y óptima, pero sirve para entender, que la rapidez y la sencillez de esta base relacional que se usó para el desarrollo del prototipo funciono de manera adecuada.

Figura 80.

Almacenamiento y creación de colección firebase

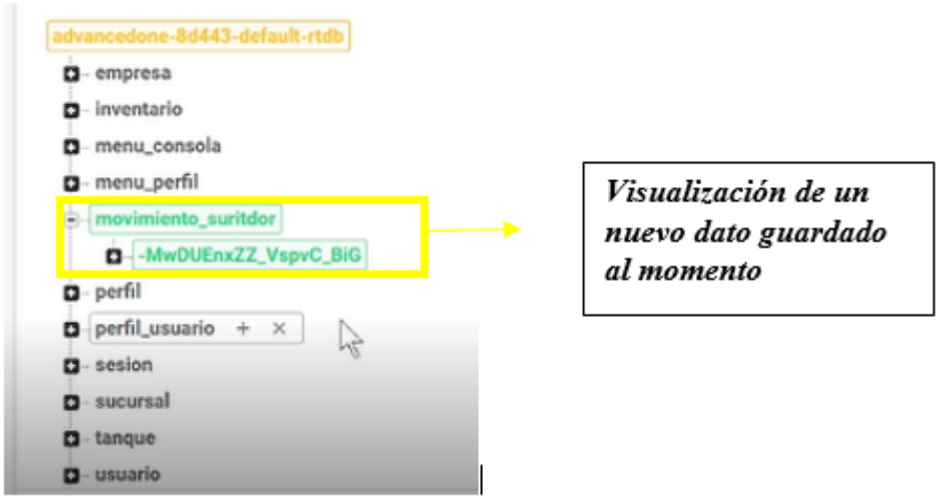


Como ya se mencionó anteriormente el tipo de base Real Time, fue de gran ayuda, ya que, en el desarrollo de la funcionalidad para el sistema de medición de tanques, realizado en conjunto con el dispositivo Veeder-Root y el Raspberry, el poder visualizar que los datos efectivamente se están guardando correctamente en la base, disminuyó el tiempo en la corrección de errores y también ayuda a ejecutar pruebas para validar que el funcionamiento era el correcto, en el caso de los tanques, la función diseñada es que mientras por medio del software se envía información y el tanque responde, esos datos que son receptados se guardan de manera inmediata, y en cuestión de las descargas y las alarmas se programó para que se guarde dependiendo de la fecha y si existió algún cambio o alguna variación entre los datos. Y en el caso del Transactor para el proceso de ventas, mientras se termina una venta esta guarda automáticamente en la base no relacional y eso se observaba al instante, tanto en el software web que cambiaba el estado del surtidor como en la consola que indicaba que un nuevo

elemento fue almacenado. A continuación, en la Figura 81, se observa como interactúa la base Firebase, con los datos que se envían para ser almacenados al instante.

Figura 81.

Visualización en tiempo real en firebase



Capítulo V

En este Capítulo se detallan las conclusiones y recomendaciones del presente proyecto, esto se lo realizó partiendo de los objetivos establecidos al inicio del mismo. Las recomendaciones se centraron en los procesos que se podrían mejorar en el prototipo desarrollado, además de que puede servir como ayuda para futuros proyectos relacionados al tema.

Conclusiones

- El sistema integrador de información en tiempo real que se diseñó, entre los dispositivos Transactor y Veeder-Root, con el Raspberry Pi, fue desarrollado con éxito, debido a que, se logró gestionar y controlar el combustible, además de, ofrecer una alternativa de bajo costo a la infraestructura tecnológica existente en la estación de servicio “ECO EL OASIS”, gracias a la ayuda de las tecnologías IoT y al desarrollo del software basado en la interacción de Python y Angular.
- En el desarrollo del módulo de comunicación, se pudo evidenciar que la codificación y el procesamiento de datos, tanto en el Transactor, como en el sistema de medición de tanques Veeder-Root, es bastante similar, ya que, su funcionamiento depende del envió de comandos para que dichos elementos emitan una respuesta, con la diferencia que, el Transactor responde con caracteres tradicionales, exceptuando su validador checksum, el cual, se encuentra en código ASCII, mientras que, el sistema de tanques, actúa dependiendo del formato del comando, ya que, este puede enviar la respuesta en formato hexadecimal, ocasionado un enlace con un módulo de transformación distinto al del Transactor.
- El módulo para la gestión y procesamiento de las tramas, referentes a las operaciones de control y venta de combustible, es el punto esencial del proyecto, y debe ser llevado a cabo con suma precaución, debido a que, todos los datos deben ser procesados correctamente;

tanto el emisor, como el receptor son determinantes, y deben actuar de manera sincronizada, de tal forma que, cada vez que exista el envío de un comando este responda inmediatamente, con el fin de que no exista perdida de información al momento de ejecutar el poleo y de visualizar en la aplicación web.

- Para el procesamiento de la información en tiempo real, la tecnología Socket IO jugó un papel fundamental, en conjunto con los Threads, ya que, ayudaron para el enlace y división de tareas, obteniendo una comunicación exitosa entre la aplicación de escritorio y la aplicación web, permitiendo que el control de combustible y el proceso de ventas, se visualice de manera inmediata por el usuario, con una reducción de delay de 1 segundo con el sistema actual.
- El modelo de 7 capas para una arquitectura IOT en tiempo real demostró cubrir las necesidades del software, desde la capa uno al interactuar con los dispositivos físicos, en la capa dos sincronizándose mediante el protocolo UART a la capa tres con el Raspberry Pi y de esta manera secuencialmente hasta la capa siete, donde la información llega a presentarse para el usuario en menos de un segundo, cumpliendo con el principio de sistema en tiempo real.
- El análisis comparativo realizado entre los tres modelos de arquitectura IOT, ayudó a la construcción de un software estable, confiable y versátil para la estación de servicio “ECO EL OASIS”, ya que, dadas las características del modelo de 7 capas seleccionado permite albergar y encajar todos los componentes planteados desde el inicio del proyecto, además de poder remplazar cualquier componente en cualquier capa siempre que cumpla con las necesidades de la arquitectura.
    - Es importante tener en cuenta el entorno y el modelo de negocio en el cual se va a ejecutar o implantar un proyecto software de este estilo; en el caso específico de la estación de servicio “ECO EL OASIS”, para la realización del sistema integrador fue adecuado hacer uso de una base

de datos no relacional, debido a su facilidad y agilidad en cuanto al ingreso y manejo de datos, pero en el caso que se quiera desarrollar un proyecto que abarque todas las funcionalidades de la estación, es necesario hacer uso de una base de datos relacional, ya que, es fundamental enlazar las diferentes actividades y tareas que se ejecutan dentro de la misma, actividad que es más complicada de realizar en una base NoSQL.

- El Raspberry Pi, versió 4, demostró que se puede solventar las funciones de un servidor para procesos de despacho de combustible y gestión de tanques, sin embargo, existen deficiencias presentadas por el dispositivo dada su capacidad de procesamiento, que a comparación del servidor actual de la estación de servicio se encuentra en una relación de 1.5Ghz a 2.13GHz respectivamente.

Recomendaciones

- Reemplazar el Raspberry Pi 4 por un dispositivo IOT con mayor capacidad de procesamiento que permita que se ejecuten los mismos procesos de manera más rápida y con la misma fluidez de comunicación entre los demás componentes de la arquitectura, pero sin dejar de tomar en cuenta que sea un dispositivo de bajo costo al momento de su implementación.
- Alojar el aplicativo web en un servidor con IP pública, de tal manera que se puedan realizar pruebas de rendimiento, haciendo énfasis en la manera que se comporta el envío de tramas desde el aplicativo local (Python) y si se logra cumplir con el principio de arquitectura de tiempo real al momento de recibir estas tramas en la parte web.
- Continuar con el desarrollo del software para la estación de servicio a fin de que a futuro pueda reemplazar a su aplicativo actual, el cual se encuentra diseñado con tecnologías en muchos casos descontinuadas, de esta manera trayendo innovación y procesos más exactos al momento de realizar sus operaciones dentro del área hidrocarburífera.

- Realizar un análisis comparativo basado en tiempos de respuesta en cada capa del sistema en tiempo real y contrastándola con su similar en el actual aplicativo en producción de la estación de servicio “ECO EL OASIS”, dando como resultado varios indicadores que permitan pensar en un cambio de arquitectura a futuro.
- Actualizar el firmware y el software de los dispositivos IoT, es indispensable, ya que evita que sean vulnerados y puedan ser víctimas de amenazas, estos elementos son potencialmente propensos a verse afectados por este tipo ataques, la seguridad en IoT requiere incluir conocimientos de privacidad por diseño**,** además de que, sus elementos puedan hacer frente a las vulnerabilidades ya conocidas.

Trabajos Futuros

La estación de servicio “ECO EL OASIS” a futuro, planea implementar en el sistema integrador para la gestión y control de combustible las siguientes actividades:

- Estructurar un módulo con reportes para los administradores puedan consultar información de ventas, despacho, ingresos o egresos.
- Deberá contar con un sistema de auto gestión de copias de seguridad, de tal manera que cada determinado tiempo se genere una copia de la base de datos tratando de evitar que se produzcan perdidas de información por fallos humanos o externos
- Se diseñará una versión específica para dispositivos móviles Android y Apple con la intención de permitir una portabilidad total al administrador y de esta manera pueda acceder a gestionar su sistema desde un dispositivo más rápidamente.
- Diseñar un módulo de ayuda auto guiada para nuevos usuarios o administradores que no conozcan todas las opciones y aplicaciones del sistema, en afán de sacar el máximo de provecho y prever un correcto uso de este.

Referencias Bibliográficas

Alarcon, C. (2014). *ANÁLISIS DEL PROCESO DE DESPACHO DE COMBUSTIBLES Y SU INFLUENCIA EN LA PRODUCTIVIDAD DE LA EMPRESA EP PETROECUADOR TERMINAL FUEL OIL DEL CANTÓN GUAYAQUIL*.

Bridgestone Company. (2015). *Reducción de costes con gestión de flotas*.

Camones Justiniano, A. H. (2020). Implementación de un sistema de gestión automatizado para el terminal de despacho de combustibles en refineria Conchán. In *Ucv*.

Challenger, I., Díaz, Y., & García, R. B. (2014). El lenguaje de programación Python/The programming language Python. *Redalyc*, *XX*(2), 1–13. https://www.redalyc.org/pdf/1815/181531232001.pdf

D. Evans. (2011). *The internet of things: how the next evolution of the internet is changing everything, CISCO white paper*. 1–11.

Departamento de Tecnología de Computadores. (2019). *Sistemas en Tiempo Real Introducción al Tiempo Real.* 1–53.

DOF. (2017). *PROYECTO de Norma Oficial Mexicana PROY-NOM-004-ASEA-2017, Sistemas de recuperación de vapores de gasolinas para el control de emisiones en estaciones de servicio para expendio al público de gasolinas-Métodos de prueba para determinar la eficiencia, mante*.

Espín Ramos, P. R. (2016). *Estudio de factibilidad para la implementación de biodiésel en las estaciones de servicio de la ciudad de Ibarra.*

Gareth, H. (2018). *Raspberry Pi ® User Guide*.

Huamancayo, C. ., & Saavedra, F. . (2020). *Propuesta de un sistema de control de combustible para reducir*

el consumo en los vehículos de una empresa de telecomunicaciones. *September*, 10. https://www.sbs.gob.pe/Portals/0/jer/opinion_proy_leg/Informe-N-053-2020-SAAJ.pdf

Instituto Ecuatoriano de Normalización. (2006). *NTE 2346*. 21.

Khan, R., Khan, S. U., Zaheer, R., & Khan, S. (2012). Future internet: The internet of things architecture, possible applications and key challenges. *Proceedings - 10th International Conference on Frontiers of Information Technology, FIT 2012*, 257–260. https://doi.org/10.1109/FIT.2012.53

Kumar, N. M., & Mallick, P. K. (2018). The Internet of Things: Insights into the building blocks, component interactions, and architecture layers. *Procedia Computer Science*, *132*, 109–117. https://doi.org/10.1016/j.procs.2018.05.170

Laudon, K., & Laudon, J. (2012). Sistemas de informacion gerencial. In *Pearson Educación*. https://juanantonioleonlopez.files.wordpress.com/2017/08/sistemas-de-informacic3b3n-gerencial-12va-edicic3b3n-kenneth-c-laudon.pdf

León, F. (2021). *Buses de datos en electrónica Paralelo, UART, SPI, I2C*. 1–8.

Lifelong Learning. (2011). Ingenieria de los sistemas embebidos. *Industrial Systems Enginering*, 1–19.

Loarte Cajamarca, B. G., & Maldonado Soliz, I. F. (2019). Desarrollo de una aplicación web y móvil en tiempo real, una evolución de las aplicaciones actuales. *Ciencia Digital*, *3*(1), 189–198. https://doi.org/10.33262/cienciadigital.v3i1.282

Lopez, S. (2021). *Modelo en cascada en el desarrollo de software*. 2021.

Martín Santana, D., & Martínez Santana, L. (2007). *Comunicación entre procesos sockets*.

Martínez, P. (2010). *Desarrollo de sistemas de tiempo real basados en componentes utilizando modelos de comportamiento reactivos*. http://www.tesisenred.net/TESIS_UC/AVAILABLE/TDR-0928110-111910/TesisPLM.pdf

Merriam Webster. (2021). Visual Dictionary Online. *Reference Reviews*, *28*(5), 30–31. https://doi.org/10.1108/rr-03-2014-0063

Microcontroladores. (2022). *Qué es un Hilo de Ejecución (Thread) y su uso Un*.

Montgomery, B., & Nakroshis, P. (2019). labjack-controller: Robust and Easy Data Collection with Labjack T-Series DAQs in Python. *Journal of Open Source Software*, *4*(38), 1448. https://doi.org/10.21105/joss.01448

Naumovich, G., Avrunin, G. S., Clarke, L. A., & Osterweil, L. J. (1997). Applying static analysis to software architectures. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *1301*, 77–93. https://doi.org/10.1007/3-540-63531-9_8

Norma Argentina GE-N1-118. (2016). *Diseño para la playa de carga, islas y boca de expendio de GNV*.

Osborne, M. A., Roberts, S. J., Rogers, A., & Jennings, N. R. (2012). Real-time information processing of environmental sensor network data using Bayesian Gaussian processes. *ACM Transactions on Sensor Networks*, *9*(1), 1–4. https://doi.org/10.1145/2379799.2379800

Programación Web. (2022). *Chat con socket . io Back-end*. 1–12.

RasterSoft. (2022). *Concepto de thread*.

Salazar, J., & Silvestre, S. (2017). Internet de las cosas (IoT). *TechPedia*, 34.

https://core.ac.uk/download/pdf/81581111.pdf

Sandoval, P. (2019). *Diagrama de implementación del estándar ISO/IEC 29110*. 13–15.

Sarabia, D. F. (2020). Arquitectura de análisis de datos generados por el internet de las cosas IoT en tiempo ... *Universidad Politécnica de Valencia*.

Thoen, F., Cornero, M., Goossens, G., Man, H. De, & Info, A. (2022). *Real-time multi-tasking in software synthesis for information processing systems Get full access to this Publication Purchase , subscribe or recommend this publication to your librarian* . 1–5.

UPV. (2022). *Sistemas Empotrados Introducción y Evolución*.

Villalta, J. (2010). *Sistema de control de rendimiento de combustible, en cabezales cañeros, para la generación de indicadores en la toma de decisiones*.

Wai Zhao, C., Jegatheesan, J., & Chee Loon, S. (2015). Exploring IOT Application Using Raspberry Pi. *International Journal of Computer Networks and Applications*, *2*(1), 27–34. http://www.digi.com

World Energy Trade. (2019). *Total inaugura su milésima estación de servicio con energía solar*. 1–8.