UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

POSGRADOS

MAESTRÍA EN

ELECTRÓNICA Y AUTOMATIZACIÓN

RPC-SO-19-No.277-2018

OPCIÓN DE TITULACIÓN:

PROYECTOS DE DESARROLLO

TEMA:

DESARROLLO DE UN SISTEMA DE MONITOREO BASADO EN HERRAMIENTAS IOT, PARA ANALIZAR LOS PARÁMETROS DE UN TRANSFORMADOR DE POTENCIA

AUTOR:

ROBERT ANTONY COLOMA CLAVIJO

DIRECTOR:

MIGUEL ANGEL QUIROZ MARTÍNEZ

GUAYAQUIL - ECUADOR
2021

COHORTE
2018 - 2020

Autor:

Robert Antony Coloma Clavijo.
Ingeniero en Telemática.
Candidato a Magíster en Electrónica y Automatización, Mención en Informática Industrial por la Universidad Politécnica Salesiana - Sede Guayaquil.
rcoloma@est.ups.edu.ec

Dirigido por:

Miguel Angel Quiroz Martínez.
Ingeniero en Sistemas
Master Universitario en Ingeniería con especialidad en sistemas de calidad y productividad
mquiroz@ups.edu.ec

DESARROLLO DE UN SISTEMA DE MONITOREO BASADO EN HERRAMIENTAS IoT, PARA ANALIZAR LOS PARÁMETROS DE UN TRANSFORMADOR DE POTENCIA

Índice general

Índice de Figuras

Índice de Tablas

Resumen

En la actualidad las empresas eléctricas se ven en la necesidad de destinar recursos a la implementación de nuevas tecnologías que les permitan realizar el monitoreo y el control remoto de los elementos que componen una red eléctrica, como medidores, reconectadores, seccionadores, subestaciones, generadoras, con el objetivo de reducir al mínimo los cortes de energía a los usuarios, poder solventar problemas con una respuesta de tiempo eficaz cumpliendo con las normas de calidad del servicio y prevenir el daño total de la infraestructura eléctrica, con el fin de que el usuario se sienta complacido con un servicio de calidad. Es por eso que el presente proyecto tiene como finalidad aportar al monitoreo de uno de los complementos de las subestaciones eléctricas, desarrollando un sistema que permita monitorear los elementos de protección básica de un transformador de potencia en una subestación eléctrica, para lo cual se ha utilizado dispositivos que estén censando la máxima información que se pueda obtener de un transformador. Para la visualización y almacenamiento de la información se utilizan aplicaciones de software libre e IoT (Internet of thing) con la finalidad de tener un sistema redundante al sistema SCADA de la empresa y que funcione como una herramienta de monitoreo de parámetros de datos para el área de mantenimiento, las bondades con las que consta el aplicativo es el fácil acceso para realizar consultas de eventos pasados o comportamientos de alguna señal. Se observará las respuestas del sistema ante diferentes fallas simuladas, así como la interacción de los diferentes usuarios que tendrán acceso al aplicativo, se llevará un control acerca de los recursos de los diferentes elementos que componen el sistema. El sistema se pondrá a funcionar de manera continua durante un tiempo determinado, permitiendo evaluar el rendimiento del programa ante alguna posible inhibición de algunos de sus complementos

Abstract

At present, electric companies are in the need to allocate resources to the implementation of new technologies that allow them to monitor and remotely control the elements that make up an electrical network, such as meters, reclosers, disconnectors, substations, generators, in order to minimize power outages to users, to solve the problems with an effective response time while complying with service quality standards and prevent total damage to the electrical infrastructure, so that the user is pleased with a quality service. That is why this project aims to contribute to the monitoring of one of the complements of electrical substations, developing a system that allows monitoring the basic protection elements of a power transformer in an electrical substation, for which we have used devices that are censoring the maximum information that can be obtained from a transformer. For the visualization and storage of the information, free software applications and IoT (Internet of thing) are used in order to have a redundant system to the SCADA system of the company and to function as a tool for monitoring data parameters for the maintenance area, the advantages of the application are the easy access to consult past events or the behavior of any signal. The system responses to different simulated failures will be observed, as well as the interaction of the different users that will have access to the application, and the resources of the different elements that make up the system will be controlled. The system will be put into continuous operation for a certain period of time, allowing to evaluate the performance of the program in case of any possible inhibition of some of its complements.

Capítulo 1

Introducción

Las empresas eléctricas han tenido un gran cambio en la forma de la operación de las redes eléctricas, los avances tecnológicos en el campo de las telecomunicaciones han impactado en gran medida a las empresas motivando a la modernización de los elementos que componen una subestación eléctrica.

La implementación de las novedades tecnológicas cada vez le permiten a las distribuidoras acortar los tiempos de interrupción, localización mas exacta del lugar de las fallas, disminución de perdidas de energía y prevención de pérdidas totales de equipos.

En muchas ocasiones los aplicativos para el monitoreo y control son de muy altos costos y sin tener un servicio de respuesta inmediata, de esta manera puede provocar el desaprovechamiento de la funcionalidad de los equipos de campo modernos.

Por esta razón el presente trabajo propone la integración de los equipos que se ha adquirido dentro de la empresa eléctrica Guayas-Los Ríos con las aplicaciones IOT, que permitirán visualizar en Real-Time los parámetros de protecciones mecánicas, temperatura de aceite y temperatura de devanado de alta tensión consultando sus datos históricos, sin tener una inversión costosa para su implementación. Con la aplicación me permitirá planificar mantenimientos y prevenir daños en el funcionamiento del transformador y poder mitigar los tiempos de interrupciones eléctricas para los usuarios finales.

1.1. Justificación

La unidad de negocio de GUAYAS LOS RÍOS tienen un total de 33 subestaciones propias [CNEL, 2019].Cada una de ellas se controla a través del sistema SCADA en el cual los operadores verifican el comportamiento de los diferentes componentes. A pesar de tener un sistema SCADA donde se concentra la mayor cantidad información, en muchas ocasiones es un inconveniente debido a la gran cantidad de información que debe procesar el operador, en muchas ocasiones no se da la atención adecuada a las alarmas presentadas dentro del sistema. El internet de las cosas (IoT) tiene un desarrollo extremadamente rápido, IoT es una extensión del Internet y conecta equipos de trabajos independientes con acceso a internet, mediante equipos sensibles a la información para el tratamiento de los datos y las comunicaciones [Singh et al., 2017, Wang et al., 2017]. Hoy en día muchas aplicaciones se han venido desarrollando en el área del Internet de las cosas [Ramakrishnan and Gaur, 2016, Englert et al., 2015, Morello et al., 2017], con respecto al sector eléctrico de potencia [Kamlaesan et al., 2017, Kul, 2017, Hossain et al., 2018, Balamurugan and Saravanakamalam, 2017].

1.2. Descripción general del problema

La empresa eléctrica tiene implementado actualmente un sistema SCADA que permite monitorear y telecontrolar una gran cantidad de parámetros de campo de las subestaciones, dado a la gran cantidad de información se ha optado para que se visualicen las señales más primordiales de las subestaciones que me permitan no saturar al operador del centro de control. Sin embargo, hay una gran cantidad de señales que no se reportan y que para las otras áreas serian de gran utilidad como es el área de mantenimiento la encargada de trabajar directamente con los equipos de campo. Actualmente existe un aplicativo que se encuentra funcionando pero carece de un sistema redundante de fallo. El personal de mantenimiento son los encargados de reemplazar o reparar los equipos de campo que se encuentran en las subestaciones, uno de los equipos principales es el transformador de potencia, actualmente el área de mantenimiento no contaba con un aplicativo que les permita monitorear los diferentes parámetros de cada uno de los equipos, donde existe la oportunidad de optimizar los tiempos de mantenimiento, prioridad de atención de subestaciones y prevenir fallas. Los transformadores de potencia están divididos generalmente en dos partes principales que son:

- Sistema de regulación de voltaje (TAP's)

- Sistema de protecciones mecánicas

Del sistema de protecciones mecánicas se utilizan sus diferentes sensores para implementar una herramienta que sea capaz de extraer información en tiempo real para el monitoreo del transformador. El proyecto tiene la capacidad de almacenar la información extraída, para ser procesada y generar diferentes tipos de alarmas que sirven de gran ayuda para el área de mantenimiento. Estas alarmas servirán de ayuda para determinar un daño que se presente a mediano y largo plazo.

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar un sistema de monitoreo utilizando una plataforma IoT para la supervisión de la temperatura de aceite, bobinado y alarmas de protecciones mecánicas de un transformador de la Subestación Durán Norte.

1.3.2. Objetivos específicos

- Extraer la mayor cantidad de parámetros de protección de los transformadores de protección a través de un equipo de adquisición de datos para visualizarlo en una plataforma IoT.
- Establecer comunicación entre un servidor local y el dispositivo que recolecta la información de campo, mediante la investigación de los protocolos de comunicación disponibles en cada dispositivo, para la transmisión de las señales de protecciones del transformador de potencia.
- Adaptar la información obtenida de campo por medio de algoritmos informáticos aplicables en el servidor local, para presentar la información en una plataforma IoT.
- Realizar pruebas del servidor local con conexión a internet para la visualización de los datos, mediante pruebas de campo y simulaciones de fallas, para medir la eficiencia del sistema en tiempo real.

1.4. Contribuciones

El presente trabajo tiene como finalidad de cumplir con los siguientes puntos:

- Tener un sistema de acceso libre para el personal del área de mantenimiento,que les permita visualizar en tiempo real y realizar consulta de todos los parámetros que el dispositivo SEL 2411 se encuentren censando.
- Tener un sistema para integrar diferentes dispositivos para poder monitorizarlos en un futuro.
- Realizar análisis de mantenimientos preventivos, a partir de los datos almacenados en el aplicativo IoT
- Incentivar el desarrollo de soluciones supervisión y control con dispositivos de hardware y software libre en las empresas eléctricas del país.
- Obtener un sistema de respaldo para la supervisión y control de los equipos de campo de una subestación eléctrica.

1.5. Organización del manuscrito

De acuerdo a las investigaciones científicas y sistemáticas el presente proyecto se ha dividido en 5 capítulos.

En el primer capítulo se expone el problema, se analiza la situación problemática encontrada, el objetivo general y los objetivos específicos a cumplir.

En el segundo capítulo se realiza una explicación de las bases teóricas de los protocolos y herramientas que se emplean para la ejecución del proyecto, se realiza un estudio de los antecedentes, presentando diferentes soluciones a sistemas de monitoreo. Se realiza una breve explicación acerca de la estructura del protocolo de comunicación eléctrico DNP3 y sus componentes, el protocolo MQTT y sus funciones para el intercambio de información, la plataforma IoT Thingsboard y su estructura.

En el tercer capítulo se divide en varias secciones donde se detalla los diferentes parámetros de las librería opendnp3 y el funcionamiento de la misma para la extracción de los datos desde el equipo de campo al dispositivo que funcione como concentrador de datos, en otra sección se explica la instalación y configuración de la plataforma Thingsboard en un servidor local, el uso de las herramientas informática para poder transmitir los datos hacia la plataforma Thingsboard.

En el cuarto capítulo se expone las diferentes pruebas realizadas de todos los complementos que componen el proyecto, se realiza simulaciones de fallas

para monitorear el comportamiento del sistema, se somete el funcionamiento del sistema a un determinado tiempo continuo para observar su rendimiento.

Para finalizar en el último capítulo se presenta las recomendaciones y conclusiones del proyecto.

Capítulo 2

Antecedentes y Fundamentación teórica

En este capítulo se abordará la problemática del proyecto que se detallan a continuación:

- Se revisará los antecedentes relacionados con el proyecto.
- Se realizará una explicación breve acerca del protocolo utilizado para la comunicación entre el Equipo y el Gateway de comunicación.
- Se analiza el protocolo de comunicación entre el Gateway de comunicaciones y el aplicativo IoT a implementarse. En las construcción se revisará la estructura del aplicativo IoT y su forma de funcionamiento

2.1. Antecedentes

La finalidad de un sistema eléctrico es garantizar el suministro de energía en su área de aplicación [Pinela, 2011]. Los subestaciones de generación se instalan en sitios donde su inversión es económica y óptima. El sistema de transporte se utiliza para transmitir grandes cantidades de energía desde las principales zonas de generación hasta las zonas de demanda. Las distribuidoras transportan la energía a los consumidores finales, aplicando el nivel de tensión más adecuado [Bernal, 1998]. Una subestación es el núcleo central de un sistema eléctrico [Coronado and Palacio, 2008]. Las subestaciones eléctricas se componen de muchos elementos, los cuales son esenciales para el suministro eléctrico, como disyuntores, dispositivos inteligentes electrónicos , pararrayos, transformadores de corriente, aisladores, condensadores, cargadores de baterías, seccionadores [Ayala S et al., 2002, Sachan, 2012]. Uno de sus elementos importante son los transformadores de potencia, la falla en un transformador implicará asumir un elevado costo, no solo se debe contemplar el costo del transformador sino de la imposibilidad de abastecer del suministro de energía a los usuarios finales [Alvarez and Del Pozo, 2007].

El funcionamiento de un transformador de potencia siempre esta expuesto a condiciones que afectan su grado de aislamiento, en algunos casos se debe a temperaturas excesivas, presencias de humedad, fallas eléctricas o esfuerzos mecánicos, que puede repercutir en su tiempo de vida útil [Fernandez, 1999], según estudios realizados acerca de estadísticas en fallas en transformadores según las estadísticas CFE de Mexico [Neumann, 2006], en un gran porcentaje los elementos con mayor concurrencia en fallas son: los cambiadores de tomas, aceite, bushing, devanados, que de acuerdo al estudio realizado en conjunto representaban el 88 por ciento de fallas [Perez, 2012, Liñan, 2001].

La empresa eléctrica CNEL, en conjunto con sus Unidades de Negocio han implementado un aplicativo para el control y monitoreo de los componentes de las subestaciones Eléctricas CASTRO VAZQUEZ [2019]. En la Unidad de Negocio Guayas Los Ríos se lleva una estructura acerca de cómo es el funcionamiento del sistema SCADA como se lo observa en la figura 2.1.

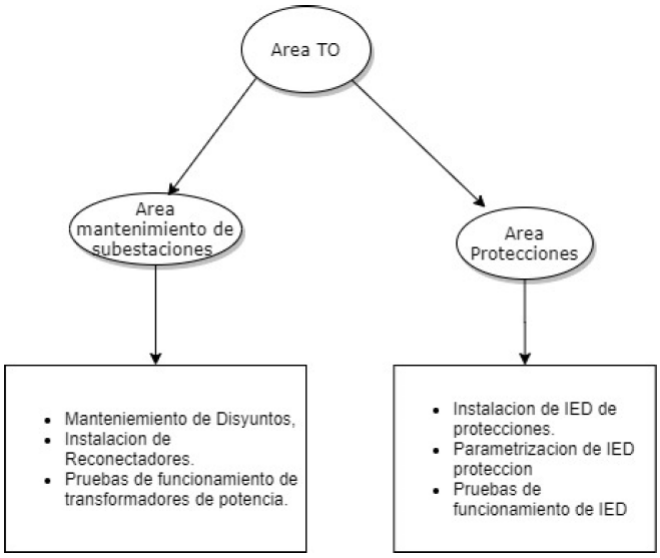


Figura 2.1: Estructura de funcionamiento de sistema SCADA

El área de protecciones es la encargada de parametrizar los dispositivos de protección con los correctos ajustes para proteger el transformador de potencia de las anormalidades presentadas en las redes eléctricas [Samuel, 2014]. Los funcionarios del área de Tecnologías de Operación (TO) son los encargadas de realizar las integraciones de las alarmas de protección y de los estados y controles de los equipos de nivel 1 hacia al sistema SCADA, para que puedan ser monitorizadas y controladas desde el centro de control. El área de mantenimiento como se observa en la Figura 1, se encarga de todo lo que corresponde a equipos primarios o de nivel 1, ellos llevan un control manual en el cual se destina unas fechas para realizar el mantenimiento en cada una de las subestaciones. Tanto el área de protecciones y TO tienen otros sistemas que permiten visualizar el comportamiento de los equipos de campo. Él área de mantenimiento actualmente no tiene una herramienta que permita verificar el estado de los equipos de nivel 1 o realizar consultas de históricos de los parámetros Censados de un transformador de potencia.

2.2. Fundamentación teórica

En esta sección se revisará la teoría acerca de los protocolos que se utilizarán para el desarrollo del proyecto con la finalidad de entender

términos técnicos que se emplean en este trabajo. Son muchas las funcionalidades tecnológicas que se emplea por lo que es recomendable entender su teoría en caso de replicar estas mismas funcionalidades en otros proyectos a futuro.

2.2.1. Distributed Network Protocol DNP

Es un protocolo abierto desarrollado por Harris, Distributed Automation Products empezó su producción alrededor de 1990 [Clarke et al., 2004], su distribución y continuo desarrollo fue asignado al grupo de Usuarios DNP3 [ORG], como tal DNP3 es un protocolo confiable y eficiente usado en infraestructuras criticas, como son las industrias petroleras, agua, electricidad y gas, el cual se lo emplea para reportar datos medidos desde una dispositivo de campo nivel 1 hacia un cliente tipo RTU o PLC o también puede funcionar como servidores locales RTU o concentradores de datos hacia centros de control como los sistemas SCADA. [Darwish et al., 2015, Normalizacyjnych, 2002]

El funcionamiento de DNP3 es de transmitir pequeñas cadenas de datos, llegando a una secuencia determinista como se observa en la figura 2.2 la secuencia de los mensajes [Clarke et al., 2004].Los datos en la capa de aplicación pueden variar de tamaños incluso pueden llegar a cero, como por ejemplo: los mensajes de comandos, los datos se dividen en varios PDU de la capa de aplicación sus tamaños son de 2048 bytes. Dividir los datos en pequeños datos ayudan a optimizar el control de errores, el paquete APDU se divide en pequeños paquetes PDU (TPDU), su tamaño máximo es de 250 bytes, estos se pueden ajustar a la capa de enlace de datos, el encabezado y los bytes de control CRC se añaden a los TPDU, el enlace resultante PDU se coloca en la capa física, en el que se transmite 8 bits de datos, 1 bit de inicio y fin para cada trama. [Mohagheghi et al., 2009]

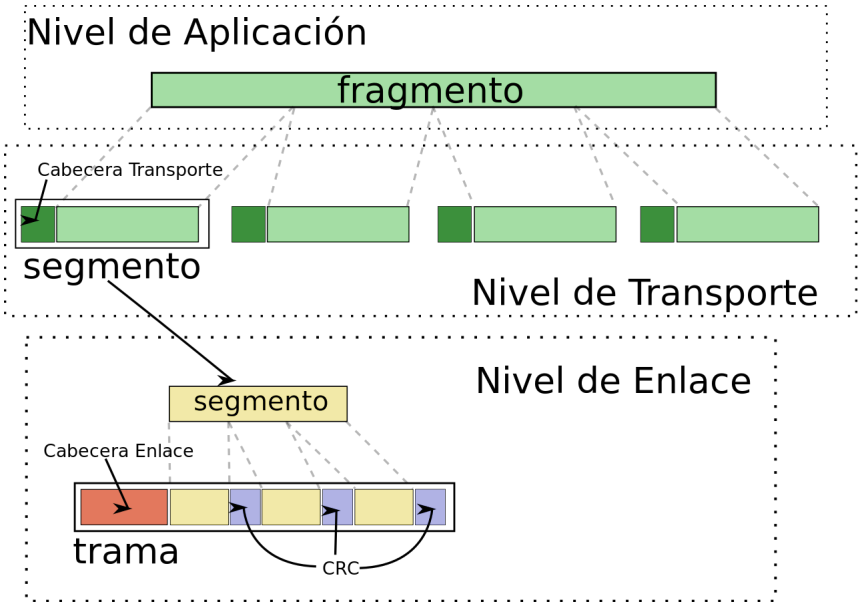


Figura 2.2: Estructura de datos DNP3, [ENSOTEST, 2021]

En las primeras versiones la capa física se trabajaba con estándares de comunicación como el RS232 para conexiones punto a punto, el R422 en conexiones bidireccionales y el RS485 para conexiones punto a multipunto. Con los avances de las redes de datos y el crecimiento de los centros de control, su demandan es el de telecontrolar grandes zonas geográficas [Mohagheghi et al., 2009]. El protocolo DNP3 ha tenido que adaptarse para funcionar por medio de los protocolos TCP/IP, esto lo ha logrado encapsulando su capa de aplicación, transporte y enlace de datos en la capa de aplicación del modelo TCP/IP como se muestra en la figura 2.3 [Clarke et al., 2004]

El protocolo se rige a un comportamiento similar al de otros protocolos de comunicación implementados en las industrias como es el modbus pero con más funcionalidades, los dispositivos pueden funcionar como maestro y esclavo. El protocolo DNP3 ha tenido un alto auge para la implementación de sistemas SCADA debido a que tiene muchas funcionalidades como son la transmisión de los datos con su estampa de tiempo, sincronizar fecha y hora a un dispositivo esclavo desde un dispositivo maestro, extraer información de oscilografía, eventos que tiene concurrencia en aplicaciones de sistemas eléctricos, los datos estáticos y eventos son agrupados en clases. Estos datos

estáticos siempre se encuentra asociado a la clase 0, y los datos de eventos se pueden asociar de acuerdo a la prioridad con la que se requieran los datos, se manejan 3 tipos de clases, la clase 1 prioridad alta, clase 2 prioridad media, clase 3 prioridad baja, todo esto con el fin de optimizar los recursos del ancho de banda en una red de datos.

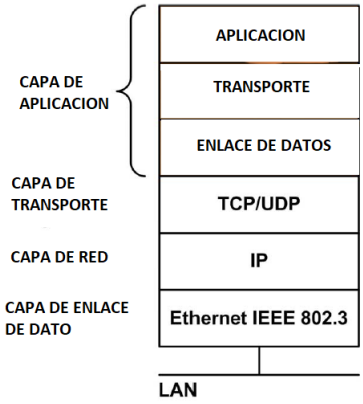


Figura 2.3: Protocolo DNP3 sobre redes ethernet, [PEREZ, 2011]

En el protocolo DNP3 la información se genera en la capa de aplicación en forma de objetos. Cada objeto tiene una estructura definida en la documentación realizada por el Grupo de Usuarios DNP3, con la finalidad de que los dispositivos sean interoperables entre si, al conjunto de objetos se la denomina como la librería de objetos, existen más de 90 objetos[Villalba Márquez, 2010], En la tabla 2.1 se presenta los diferentes grupos de objetos

Tabla 2.1: Rango de grupo de modelos de objetos

Rango de grupos	**Descripción de los objetos**
0-9	Entradas binarias
10-19	Salidas binarias
20-29	Contadores
30-39	Entradas Analógicas
40-49	Salidas Analógicas
50-59	Objetos de tiempo
60-69	Objetos de clase
70-79	Objetos de Archivos
80-89	Objetos de aplicación
90-95	Objetos Numéricos Alternativos

En cada objeto existe una variación que determina las características de esos datos tanto para las señales binarias y analógicas existen variaciones tal como se muestra en la tabla 2.2.

Tabla 2.2: Objetos de Entradas binarias

Grupo	**Variación**	**Tipo**	**Descripción**
01	1	Estático	Entradas binarias
01	2	Estático	Entradas binarias con estado
02	1	Evento	Cambio de entradas binarias con tiempo
02	2	Evento	Cambio de entradas binarias sin tiempo

Asi como existe las caracterizticas para los objeto binarios, también los objetos analógicos tienen sus variaciones a continuación en al tabla 2.3

Tabla 2.3: Objetos de Entradas Analógicas

Grupo	Variación	Tipo	Descripción
30	1	Estático	Entradas Analógicas de 32 bits
30	2	Estático	Entradas Analógicas de 16 bits
30	3	Estático	Entradas Analógicas de 32 bits sin bandera
30	4	Estático	Entradas Analógicas de 16 bits sin bandera
30	5	Estático	Entradas Analógicas de punto flotante corto
30	6	Estático	Entradas Analógicas de punto flotante largo

En el proyecto se va a utilizar el objeto 1 y objeto 30 para realizar las consultas de los datos binarios y analógicos que se extraerán del transformador de potencia. El protocolo DNP3 tiene muchas ventajas con respecto al tiempo de sincronización y la función de reportar datos sin que la estación maestra haya realizado un requerimiento de datos, los principios fundamentales que ayudan a mitigar los tiempos en caso de una falla y optimizar proceso de un sistema SCADA.

2.2.2. Message Queing Telemetry Transport MQTT

MQTT es un protocolo basado en publicación y subscripción, en el que los dispositivos envían los datos a un servidor central, este servidor es donde se va a publicar los datos o solicitar los datos.[MQTT, Ramia Tena, 2020], según[Vera Martín et al., 2019] el protocolo se ha desarrollado con los siguientes requisitos:

- Implementación simple.
- Entrega de datos de calidad de servicio.
- Ligero y ancho de banda eficiente.
- No es necesario que conozca los datos que maneja.
- Debe poder mantener una sesión continua.

El protocolo MQTT esta compuesto por los clientes y el Broker MQTT.

El cliente MQTT es el dispositivo que publica y/o se suscribe a un Topic y establece una conexión con un Broker MQTT, a través de la red usando librerías MQTT.

El Broker MQTT es el agente responsable de receptar todos los mensajes, filtrarlos y reenviarlos a todos los clientes suscritos a ese Topic. El Broker será el encargado de la autorización de los clientes dentro de la red de comunicación y garantizar las sesiones de los clientes persistentes. La forma en como funciona la conexión de los dispositivos es el siguiente:

- El dispositivo cliente envía un mensaje CONNECT y el broker responderá con un mensaje CONNACK.
- El broker mantendrá la sesión abierta mientras por parte de los dispositivos clientes no se envíe un mensaje disconnect.
- La forma del mensaje CONNECT debe ser la correcta para que el broker pueda responder con el mensaje CONNACK, todo esto con el fin de prevenir que dispositivos maliciosos traten de saturar el broker como se muestra en la figura 2.4.

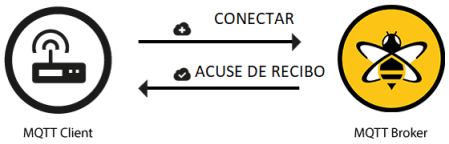


Figura 2.4: Ejemplo de Conexión MQTT, [GmbH, 2021]

El mensaje CONNECT esta compuesto por varios parámetros:

- ClientID: es la identificación de un cliente, esta identificación debe ser única.
- CleanSession: Es el que indica si la conexión va a ser persistente si es FALSE el broker mantiene todas las subscripciones y mensajes que se pierda la conexión se mantiene persistente, si es TRUE el broker elimina la información de una conexión persistente anterior, la conexión no es persistente.
- Usuario y Contraseña se lo emplea para una mayor seguridad es opcional, se emplea en los clientes para su autenticación y autorización.
- Will message: el broker notifica a los suscriptores cuando un cliente se desconecta sin haberlo deseado por algún problema de terceros.

- Keep alive: es el tiempo que el cliente notifica al broker el tiempo que va a estar activo sin haber intercambio de mensajes.

El mensaje CONNACK esta compuesto por los siguientes parámetros:

- Session Present: Notifica al cliente si existe una conexión persistente con el.
- Return Code:
  - 00: Conexión Establecida
  - 1: Error de conexión, versión de protocolo no admitida
  - 2: Error de conexión, identificador no aceptado
  - 3: Error de conexión: fuera de servicio el servidor
  - 44: Error de conexión: Nombre de usuario o contraseña incorrecta
  - 45: Error de conexión, no autorizada

Una vez establecida la conexión existen mensajes que realizan diferentes funciones, uno de los mensajes que intervienen entre los clientes y el broker y los suscriptores es el mensaje PUBLISH como se observa en la figura 2.5.

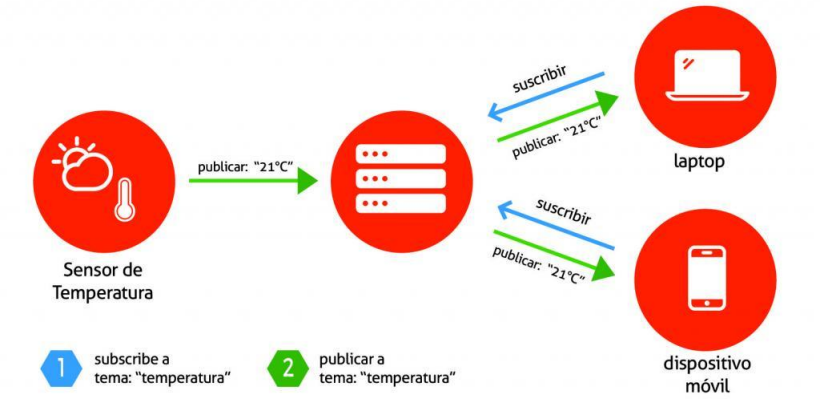


Figura 2.5: Funcionamiento protocolo MQTT, [Vargas, 2018]

Cada mensaje Publish debe contener el Topic que el intermediario que se emplea para reenviar el mensaje, el dato a enviar y la calidad del mensaje, el dato puede ser de cualquier tipo ya sea binario, texto o XML o JSON. El mensaje PUBLISH esta compuesto de los siguientes parámetros:

- TopicName: Es una ruta que se encuentra delimitada por diagonales
- QoS: Es el nivel de calidad de servicio del mensaje (0,1 ó 2).
- RetainFlag: es una bandera que indica si el mediador almacena el ultimo valor registrado para un especifico Topic. Si se encuentra activo, el broken envía el mensaje almacenada a los nuevos cliente suscritos.
- Payload: Es el contenido del mensaje. Se puede enviar casi cualquier tipo de dato.
- PacketID: Sirve para identificar los mensajes entre cliente y Broker. Se emplea en QoS mayores de cero.
- Dupflag: Sirve para identificar si es un mensaje duplicado. Se utiliza en QoS mayores a cero.

Los dispositivos que desean recibir los datos publicados por otros clientes deben suscribirse a un Topic. Para poder suscribirse a un Topic, el cliente debe enviar el mensaje SUSCRIBE al Broker Los parámetros del mensaje SUSCRIBE son los siguientes:

- PacketID: Sirve para identificar un mensaje entre un cliente y el broker.Solo se utiliza en broker mayores a cero
- Lista de Suscripciones: se forma por QoS y TopicName. Pueden existir un gran conjunto de datos como de suscripciones.

Para establecer la suscripción, el broker envía un mensaje SUBACK al cliente, el cual está compuesto por los siguientes parámetros:

- PacketID: es la identificación de un mensaje entre cliente y Broker. Solo se utiliza en QoS mayores a cero
- ReturnCode: Se envía un código por cada mensaje de suscripción enviada en el mensaje SUSCRIBE. Los códigos son los siguientes:
  - 00: Éxito: QoS 0
  - 1: Éxito: Qos1
  - 2: Éxito:Qos2
  - 128: Error

La programación MQTT es tan ligera que permite conexión entre máquinas donde el ancho de banda es bien limitado, su propósito es disminuir el requerimiento de recursos de los dispositivos, el protocolo presenta un cierto grado de seguridad con la entrega de la información.

Uno de los propósitos de MQTT es emplearse en una red conformada por una gran cantidad de dispositivos pequeños que se requieran monitorear y controlar desde servidores alojados en internet, la implementación es sencilla y con muy pocas opciones de control.

MQTT ha tenido grandes implementaciones debido a sus muchas funcionalidades, estas se han aplicado en prototipos de sistemas de control para el monitoreo y control del nivel de Agua [Salgado et al.].

2.2.3. Thingsboard

Es una plataforma IOT de código abierto que se puede descargar accediendo al siguiente repositorio (https://github.com/thingsboard/thingsboard) funciona bajo Apache2.0, la empresa tienes 2 tipos de ediciones una comunnity y una professional, esta última tiene soporte y funciones adicionales. En la figura 2.6 se observa su estructura, la comunicación con los dispositivos se efectúa mediantes diferentes protocolos de transporte, aparte de su plataforma que puede ser instalada en un servidor local, existe su puerta de enlace que permite integrar dispositivos IoT que funcionan con sistema de terceros en la plataforma Thingsboard. El aplicativo trabajo con gestores de base de datos PostgreSQL o Cassandra externa para almacenar los datos. Sus funciones principales son la administración de dispositivos, administración de usuarios y de paneles. El aplicativo cuenta con una API que funciona una puerta de enlace, que autoriza a los usuarios registrados interactuar con los dispositivo. Su arquitectura es Plug-ins con lo que puede acoplarse a aplicaciones externas, los plug-ins son para Apache Kafka y también se puede realizar el envío de correos electrónicos. Para el manejo de mensajes generados por eventos se utiliza internamente Akka [ThingsBoard].

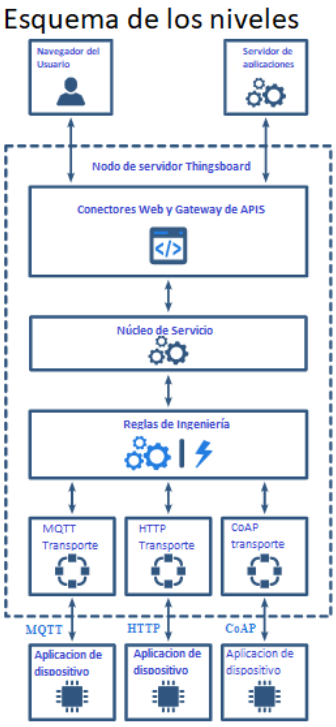


Figura 2.6: Estructura de plataforma Thingsboard, [ThingsBoard]

Capítulo 3

Estructura y Desarrollo del Proyecto

En este capítulo se detalla el paso a paso de como es el diseño del sistema propuesto, las herramientas que se implementaron para el funcionamiento del aplicativo y la interactúan con todos los componentes. Se detalla la configuración del gateway de comunicaciones(Raspberry pi 4), como configurar la librería que permite utilizarel protocolo DNP3, el script que se implementa con el protocolo MQTT, para el envío de la información hacia el aplicativo IoT, también se revisa la configuración en el aplicativo para el funcionamiento general del sistema.

3.1. Diseño del experimento

En la primera parte del proyecto se realiza un levantamiento de información de señales del transformador de potencia de la subestación Durán Norte, la empresa CNEL Guayas los Ríos adquirió un conjunto de dispositivos SEL 2414 que son dedicados al monitoreo en transformadores de potencia [Laboratories]. Su diseño está dedicado para funcionar en la intemperie contando con muchas aplicaciones en el campo [CASTRO VAZQUEZ, 2019]. Se realiza la instalación del equipo y la conexión de sus respectivas señales analógicas y digitales, para que pueda ser transmitidas al centro de control y a la plataforma IoT. De acuerdo a la información levantada con el área de mantenimiento, se obtuvo la siguiente información que puede reportar el transformador de potencia de la subestación Durán Norte.

La información se va a dividir en dos tipos:
Señales Digitales :

- Nivel mínimo de aceite .
- Nivel máximo de aceite.
- Alarma relé bucholz.
- Disparo de válvula de sobrepresión.
- Disparo de relé de presión súbita.

Señales Analógicas:

- Temperatura de Aceite.
- Temperatura de devanado de Alta.

3.2. Etapa de comunicación de Dispositivos

Se implementa un servidor para que funcione como un concentrador de información de señales de campo y se analice los diferentes algoritmos que permitan adquirir la información desde el dispositivo SEL 2414, el medio de comunicación entre el concentrador y el dispositivo de adquisición de datos es utilizando el servicio de CNT, el cual se encuentra instalado en todas las subestaciones y permite establecer la comunicación entre la subestación Durán Norte y la subestación Durán sur que es el lugar donde se aloja nuestro

concentrador de datos al cual lo llamaremos servidor tal como se muestra en la figura 3.1.

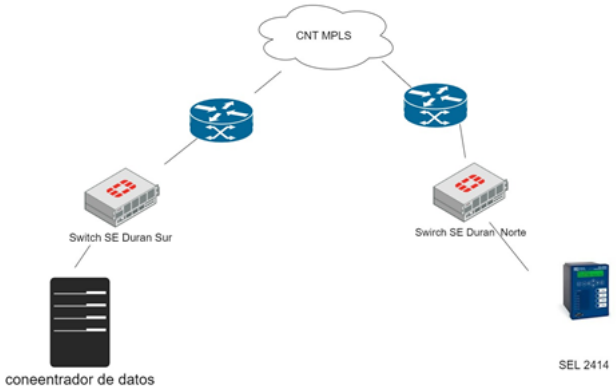


Figura 3.1: Esquema de comunicación

Con la comunicación entre el concentrador y el dispositivo SEL 2414 se procesará la información mediante subprogramas para poder establecer la comunicación entre el concentrador y la plataforma IoT, Aplicativo que permite conectar dispositivos a través de internet donde se visualiza los datos en tiempo real para posteriormente ser procesados.

3.3. Etapa de presentación y experimentación

Se parametriza los datos tanto en la plataforma IoT y el servidor, se diseñará una pantalla de visualización para monitorear los datos del transformador de potencia, así como realizar consultas de datos históricos de los parámetros censados. Para finalizar se realizó pruebas con diferentes dispositivos para comprobar el acceso remoto a la plataformaIoT para evaluar el comportamiento de los datos monitoreados en el sistema sus esquema de comunicación es tal como se visualiza en la figura 3.2.

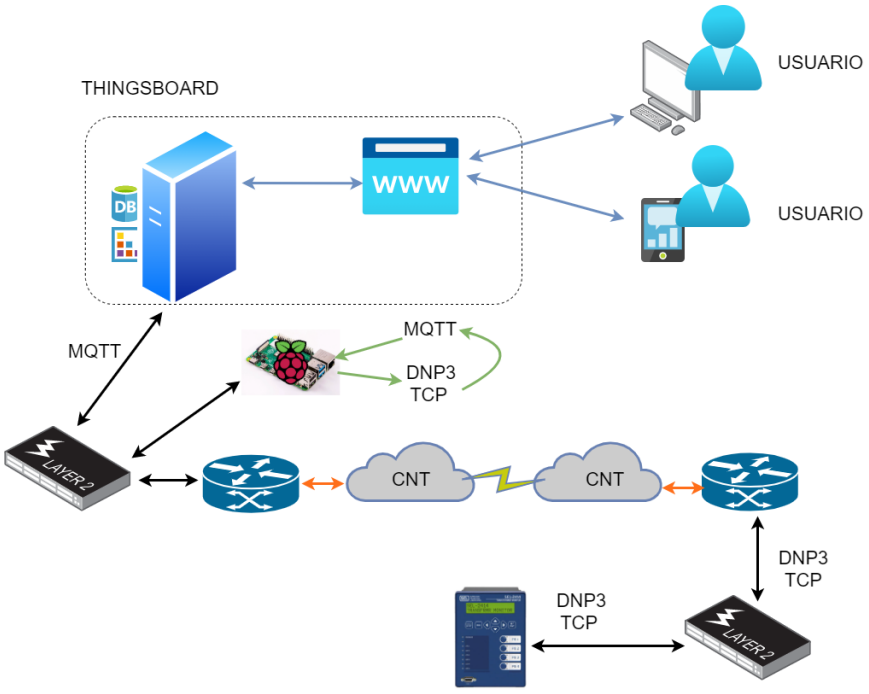


Figura 3.2: Arquitectura del proyecto

3.4. Desarrollo

En esta sección se describirá paso a paso las configuraciones realizadas en todos los dispositivos que se utilizaron para llevar a cabo el desarrollo del proyecto.

3.4.1. Configuración y adecuación de dispositivo SEL 2414

Se realiza la parametrización del IED 2414, dispositivo fabricado por la empresa Schweitzer Engineering Laboratories, Inc. El dispositivo tiene como finalidad ser un equipo de campo, con la capacidad de ser instalado en la intemperie. Puede enviar la información que se esté receptando de campo hacia una unidad remota a través de los protocolos de comunicación industrial por medio de los diferentes medios de comunicación, ya sea cable de cobre de red STP o de fibra óptica.

El IED SEL 2414 posee entradas y salidas tanto digitales como analógicas que pueden funcionar a diferentes niveles de voltaje y corriente que se puede

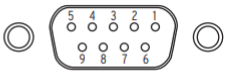
solicitar de fábrica. Los protocolos que se pueden utilizar son:

- DNP3 TCP/IP
- MODBUS TCP
- IEC61850
- MODBUS SERIAL
- DNP3 SERIAL

El dispositivo puede adquirirse con 1 o 2 puertos de cobre conector RJ45 o de fibra Óptica en la parte posterior y 1 puerto RS232 en la parte frontal de comunicación serial en la figura 3.3 se observa el modelo del dispositivo.

Figura 3.3: Dispositivo SEL 2414

La parametrización del equipo se puede realizar de 2 manera, la primera es a través del puerto frontal el cual es un puerto de tecnología serial véase figura 3.4, cuyo inconveniente es la lentitud para enviar y recibir datos.



EIA-232 DB-9 Connector Pin Numbers

Figura 3.4: Modelo de conectores SEL Serial

La segunda forma es a través de sus puertos de tecnología Ethernet, para el caso de este proyecto el dispositivo SEL 2414 cuenta con conectores RJ45 y se utiliza un Patch-Cord Cat 6a que sirve para establecer comunicación con el computador.

Para empezar, primero se configura la dirección IP del segmento de subred en el cual el dispositivo trabajará, por medio del panel frontal se puede acceder al menú donde se configuran los parámetros de red del equipo. Para acceder a estos parámetros se presiona la tecla enter, y a continuación se accede a un conjunto de opciones del equipo, a través de las teclas direcciones se navega hasta encontrar el submenú Set/Show, se presiona la tecla enter y se accede a otro submenú, se navega hasta encontrar la pestaña Port tal como se muestra en la figura 3.5.

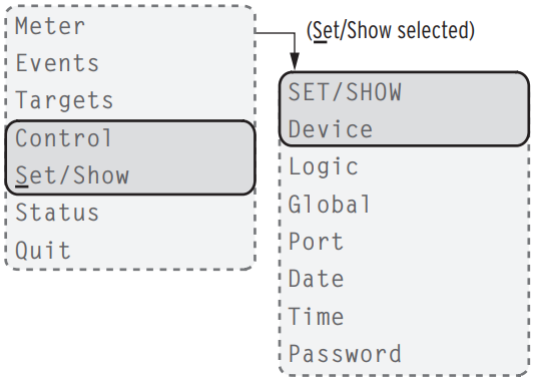


Figura 3.5: Menú de dispositivo SEL

En el submenú Port, se escoge el puerto que se va a configurar, para el proyecto es el puerto 1 que funciona con tecnología Ethernet, se presiona la tecla enter sobre el puerto y se selecciona la opción Port 1 Settings, dentro de ese submenú con las teclas direcciones se configuran los parámetros de red para el proyecto, se colocan los siguientes datos.

En la figura 3.6 se muestra la dirección IP que el equipo ocupará en la red funcional de la subestación, esta dirección debe ser única por dispositivo que se encuentren funcionando.

Figura 3.6: Dirección IP del sel 2414

La mascara de subred es la que indica en que subred trabajará el equipo, en la figura 3.7 la máscara 255.255.255.192 es el segmento de subred de los equipos de potencia.

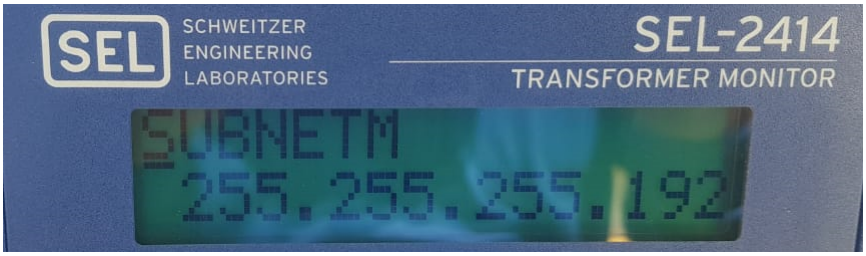


Figura 3.7: Mascara de subred del SEL 2414

El Gateway (puerta de enlace) por lo común es la dirección IP del enrutador cuya función es interconectar diferentes redes, es esencial colocar este parámetro ya que sin este dato, no se puede establecer una comunicación remota, para este caso la puerta de enlace es la dirección mostrada en la figura 3.8.

Figura 3.8: Puerta de enlace del SEL 2414

.

Los otros parámetros a configurar son los protocolos de comunicaciones, para el proyecto utiliza el protocolo DNP3, debajo de la opción de Port 1 Settings existe la opción Protocol Settings, se accede a esa opción y en la opción de EDNP se selecciona 3, esto significa que el dispositivo puede reportar a 3 estaciones maestras tal como se muestra la figura 3.9.

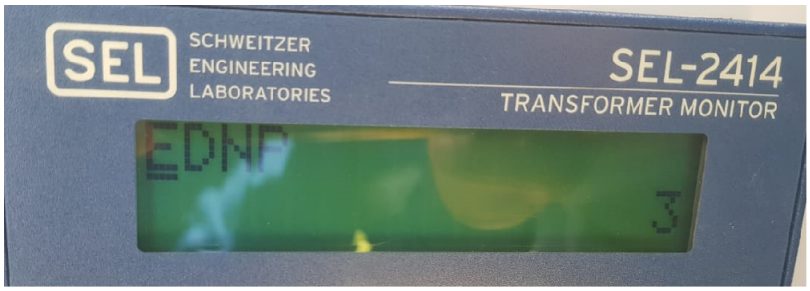


Figura 3.9: Numero de sessiones DNP3

De acuerdo a lo revisado en el capítulo 2 acerca de la teoría del protocolo DNP3, el parámetro que se debe configurar es la identificación DNP3 del equipo esclavo, en este caso es la identificación que va a utilizar el dispositivo SEL 2414 para el intercambio de información, su identificación es 18 y se configura en la siguiente opción tal como se muestra en la siguiente figura 3.10.

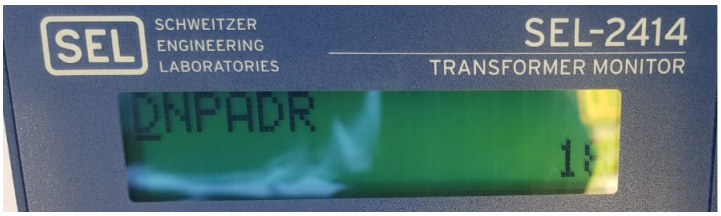


Figura 3.10: Identificación DNP3 equipo esclavo

Para concluir, en la figura 3.11 se observa el parámetro para establecer la conexión entre el dispositivo maestro y esclavo será la identificación del dispositivo maestro, que en este caso es 102, esta identificación se configura en el siguiente parámetro del dispositivo SEL 2414.

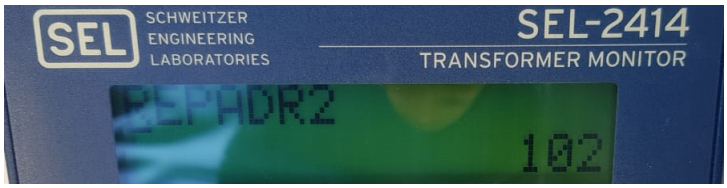


Figura 3.11: Identificación DNP3 equipo maestro

Con las diferentes configuraciones realizadas, el dispositivo ya se encuentra listo para transmitir su información a través del protocolo DNP3. A continuación se configura la información que se va a transmitir a la estación maestra. Para configurar las señales que se van a transmitir o las lógicas que se requieran, se va a utilizar el software AcSELerator Quickset, software que permite establecer la comunicación de forma remota por medio de los puertos Ethernet o los puertos seriales. El software se lo puede descargar desde la página www.selinc.com, para poder descargar el software se debe crear una cuenta en la página del fabricante, la descarga del programa es totalmente gratuito, una vez instalado el programa se inicia y se muestra una ventana con las siguientes características como en la figura 3.12.

Figura 3.12: Software de configuración de dispositivo SEL 2414

Se realiza un clic en el icono comunicación y se mostrarán los parámetros que se deben de ingresar para poder establecer la comunicación con el dispositivo tal como se muestra en la figura 3.13.

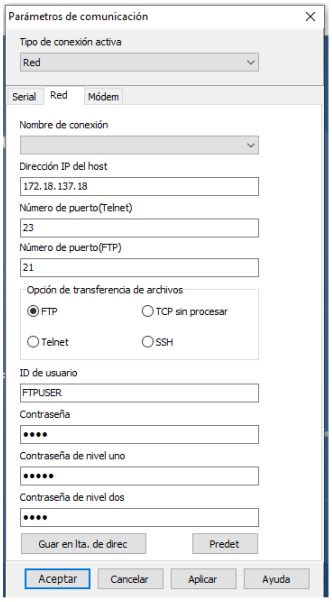


Figura 3.13: Ventana de parámetros de comunicación

La conexión se realiza por medio de los puertos Ethernet, en la opción de tipo de conexión se escoge red, en el campo dirección IP de host se coloca la dirección IP del equipo que fue configurado de forma manual, se escoge el tipo de conexión Telnet o FTP, debido a la versión del firmware se debe escoger que la transferencia de archivos sea por medio de protocolo FTP, luego en ID de usuario se escribe FTPUSER y en Contraseña TAIL, los demás parámetros quedan con sus configuraciones por defecto, se presiona el botón aplicar y en la barra inferior se observa el intercambio de información en los leds TX y RX y como cambia la palabra de desconectado ha conectado, el botón de lectura se lo presiona y el aplicativo empezará a adquirir la última configuración que se encuentra cargada en el dispositivo. Una vez que el programa termine de realizar la lectura, automáticamente se abrirá una ventana donde ser observará toda la configuración del equipo tal como se muestra en la figura 3.14.

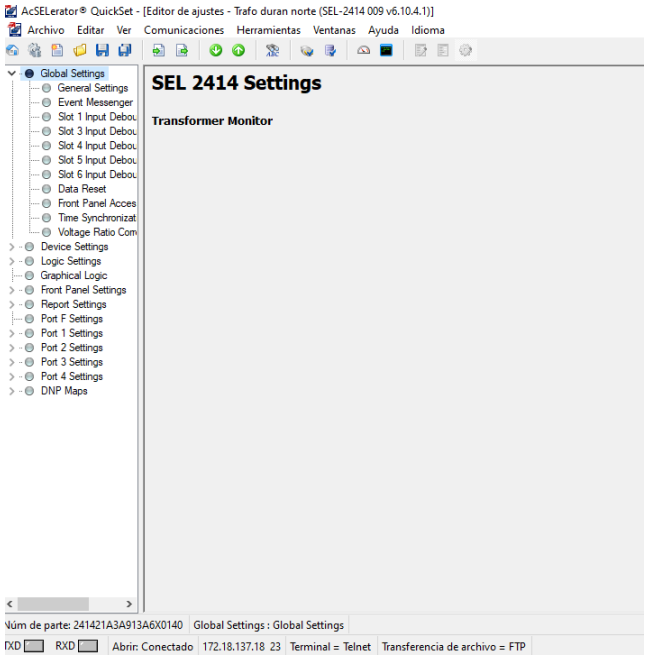


Figura 3.14: Ventana de de configuración de dispositivo SEL 2414

Como se puede observar se aprecia los diferentes parámetros que se pueden configurar, para el caso del proyecto se escoge a la opción de DNP map y desplegarán los mapas de configuración, para el caso del proyecto el mapa 2 es el que se requiere configurar, se presiona en DNP map 2 y se muestra las diferentes señales que se pueden transmitir en la figura 3.15.

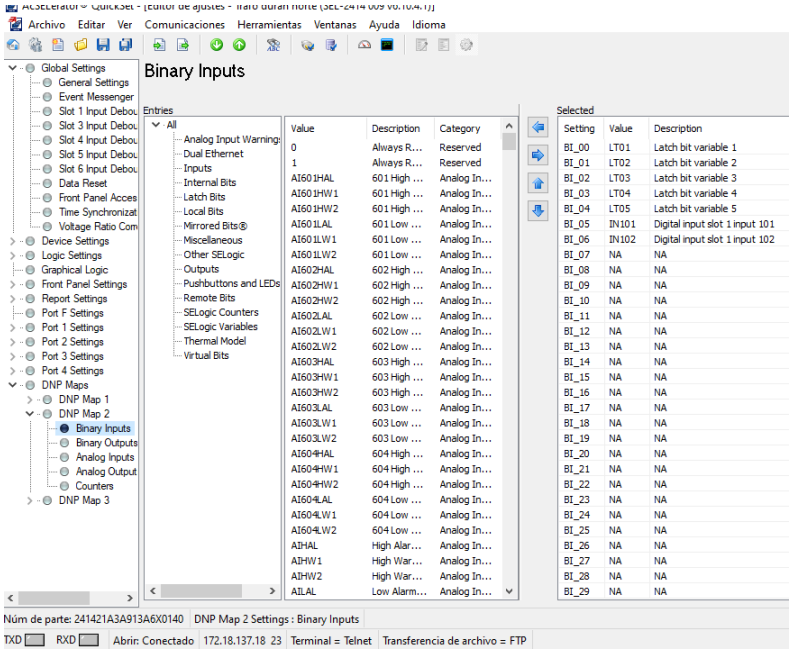


Figura 3.15: Ventana de de configuración de señales SEL 2414

Como se observa se encuentran los 5 tipos de señales el cual el dispositivo puede transmitir y recibir, estas son:

- Binary inputs
- Binary outputs
- Analog inputs
- Analog outputs
- Counter

En el presente proyecto solo se transmiten las señales que son de tipo Binary inputs y Analog inputs para la construcción del mapa de señales de DNP3, con los íconos de las flechas hacia la izquierda y derecha se eliminan o se añaden las señales que se transmiten a través del protocolo DNP3, a medida que se añaden señales, automáticamente se les asigna un índice que como ya se ha explicado es el registro que se debe apuntar para poder leer el valor de determinada señal.

3.4.2. Configuración de Raspberry pi

Primero se realiza la instalación del sistema operativo Raspbian, a través de las diferentes herramientas que existen en internet.

Las aplicaciones como Python ya vienen instaladas por defecto en el sistema operativo Raspbian. Para el proyecto se utiliza la librería opendnp3, librería que funciona para que la Raspberry establezca comunicación por medio de protocolo DNP3 con el equipo instalado en el transformador de potencia.

La librería se encuentra en el siguiente enlace https://github.com/dnp3/opendnp3 tal como se observa en la figura 3.16, al momento de acceder al link se encontrará un conjunto de carpetas y archivos, se debe descargar todo de la siguiente forma:

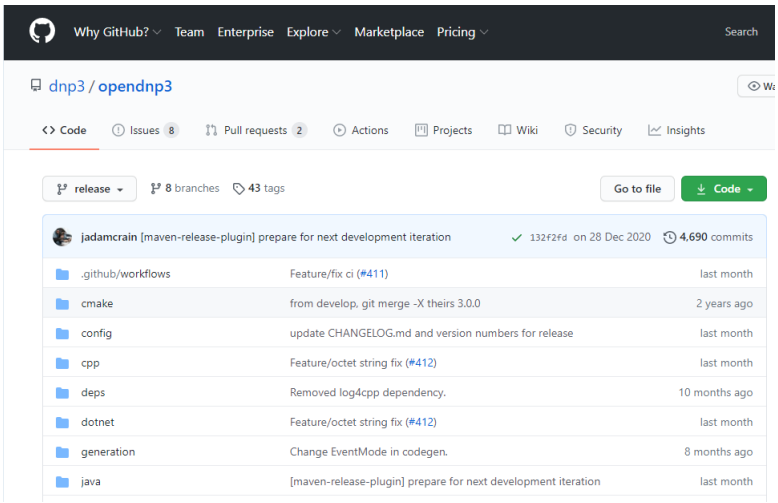


Figura 3.16: Pagina de librería Opendnp3.

Se descarga la librería opendnp3 desde el botón CODE en donde se escoge la opción descargar como archivo tipo .zip, tal como se muestra en la siguiente figura 3.17.

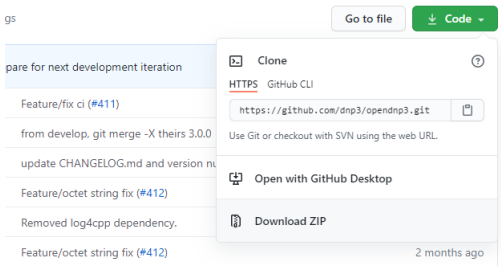


Figura 3.17: Forma de descarga de librería Opendnp3

Se procede a descomprimir el archivo zip y automáticamente se creará la carpeta openDNP3 con todos los archivos que se observa en la pagina fuente.

Para poder utilizar la librería, se debe instalar el programa CMAKE, para el proyecto se utilizó la versión 3.18.5, se descomprime el archivo CMAKE-3.18.5.tar.gz con el comando que se muestra en la figura 3.18

```
cmake-3.18.5/        cmake-3.18.5.tar.gz
pi@raspberrypi:~/Downloads $ tar -zxvf cmake-3.18.5.tar.gz
```

Figura 3.18: Comando para descomprimir archivo Tar de cmake

Se crea una carpeta con el nombre cmake-3.18.5 y se ingresa a la carpeta. En el interior de la carpeta se ejecuta las siguientes lineas de código tal como se observa en la figura 3.19.

```
pi@raspberrypi:~/Downloads $ cd cmake-3.18.5/
pi@raspberrypi:~/Downloads/cmake-3.18.5 $ ./bootstrap
```

Figura 3.19: Ejecución del comando bootstrap

Cuando finalice la ejecución de la instrucción bootstrap, se continua con el comando que se visualiza en la figura 3.20.

```
pi@raspberrypi:~/Downloads $ cd cmake-3.18.5/
pi@raspberrypi:~/Downloads/cmake-3.18.5 $ make
```

Figura 3.20: Ejecución del comando make

La instalación de todas las dependencias del programa tomará un buen tiempo, para concluir la instalación del programa cmake, en la figura 3.21 se ejecuta la siguiente línea.

```
pi@raspberrypi:~/Downloads $ cd cmake-3.18.5/
pi@raspberrypi:~/Downloads/cmake-3.18.5 $ sudo make install
```

Figura 3.21: Instalación de los paquetes make

Una vez finalizado todos los pasos para la instalación, En la figura 3.22 se realiza la verificación de la instalación del programa de la siguiente forma.

```
pi@raspberrypi:~ $ cmake --version
cmake version 3.18.5

CMake suite maintained and supported by Kitware (kitware.com/cmake).
pi@raspberrypi:~ $
```

Figura 3.22: Verificación de versión de cmake instalada

Como se observa en la figura 3.22 en el dispositivo se realizó la instalación de la versión 3.18.5 correctamente. La librería contiene varios ejemplos para que la raspberry funcione como maestro o esclavo DNP3, en este caso se necesita que la raspberry funcione como maestro DNP3, ya que va a estar solicitando datos cada cierto tiempo al dispositivo SEL 2414, para esto se debe modificar el ejemplo de la carpeta master para que la raspberry se comunique con el dispositivo SEL 2414, el archivo a modificar es el main.cpp tal como se muestra en la figura 3.23.

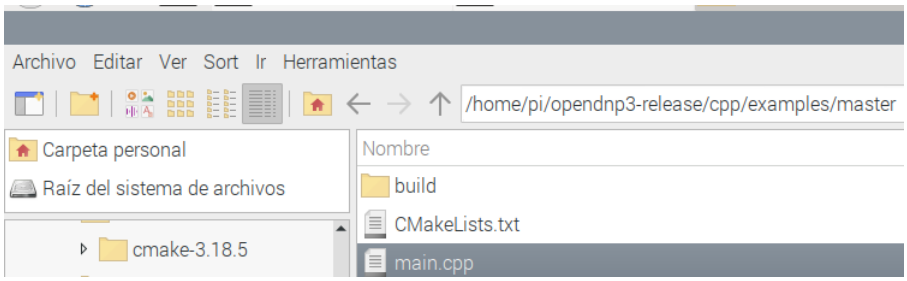


Figura 3.23: Ruta de archivo main

El archivo main a modificar es el de la carpeta que se descomprimió al descargar del repositorio github, se puede modificar el archivo con cualquier editor, para el proyecto se requiere editar las siguientes partes:

```
//Código donde se coloca la dirección IP del equipo esclavo

auto channel = manager.AddTCPClient("tcpclient", logLevels,
ChannelRetry::Default(), {IPEndpoint("192.168.0.4", 20000)},
                                          "0.0.0.0",
                                PrintingChannelListener::Create());
```

El campo IPEndpoint contiene dos parámetros, el primero es la IP del equipo al que se desea interrogar para el proyecto, que en este caso es la IP del dispositivo SEL 2414 el segundo parámetro es el puerto en el que esté funcionando el protocolo DNP3, por defecto el puerto es el 20000, los demás parámetros no se modifican.

Lo siguiente a configurar son las direcciones de identificación de DNP3, como es el maestro y el esclavo, eso se modifica en las siguientes líneas de código como lo muestra la figura 3.24.

```
// You can override the default link layer settings here
// in this example we've changed the default link layer addressing
stackConfig.link.LocalAddr = 102;
stackConfig.link.RemoteAddr = 18;
```

Figura 3.24: Identificación dnp3 de los dispositivos

Link.LocalAddr es un atributo que contiene la identificación DNP3 del equipo que funciona como maestro, en esta caso la raspberry, la dirección que se ha colocado es 102, Link.RemoteAddr es el atributo que va a contener la identificación de el equipo del que se va a solicitar la información, en este caso es la identificacion de esclavo del dispositivo SEL, se vuelve a recordar que estos campos deben estar configurados igual en el dispositivo SEL 2414. El archivo main.cpp tiene un conjunto de funciones, en dnp3 se puede realizar interrogaciones por clases 1, 2, 3 o clase 0 o todas las clases al mismo tiempo, también se puede realizar interrogación por objetos.

```
//Código para interrogar Datos Analógicas

arch_1.open("Analog.txt",std::ofstream::trunc);
        master->ScanRange(GroupVariationID(30,0), 0, 1,
        test_soe_handler);
```

Para este caso se utiliza la función de interrogación por objetos, dentro del parámetro GroupVariationID se coloca el tipo de objeto y su variación, en este caso se ha usado el objeto 30 que se refiere a todas las señales analógicas de lectura, y 0 lo cual significa cualquier variación de las señales analógicas, con esto se dice que se requiere leer cualquier señal analógica de cualquier variación, los siguientes dos parámetros son desde que índice a que índice se requiere leer, en este caso se va a leer todas las señales analógicas desde el índice 0 hasta el índice 1. Se ha modificado el script para que los datos que sean leídos se almacenen en un archivo temporal. La siguiente parte es generar los archivos de la librería opendnp3, para generar el proyecto se utiliza los siguientes comandos tal como se observa en la figura 3.25

```
pi@raspberrypi:~ $ cmake -S opendnp3-release -B PRUEBA -DDNP3_EXMAPLES=TRUE
```

Figura 3.25: Generar archivos de librería opendnp3 a través de cmake

El comando -S “ruta de Libería opendnp3” es para indicar cuál es el directorio que contiene el archivo cmakelist y es el que contiene todas las instrucciones, -B “ruta de destino” es la ruta donde se van a generar los archivos, para este proyecto se generan los proyectos en la carpeta llamaba proyectodnp3.

En la carpeta proyectodnp3 se encuentran algunos archivos y carpetas, se ingresa a la carpeta cpp, luego a la carpeta examples, en esta sección se encuentran los diferentes ejemplos de cómo puede funcionar la raspberry, para el funcionamiento del proyecto, la carpeta “master” es a la que se debe acceder.

Dentro de la carpeta master se ejecuta el comando “make” y se genera un programa llamado master-demo.

al como se observa en la figura 3.26 se lista los archivos y programas dentro de la carpeta master, en él se encuentra el programa master-demo generado, para que empiece a funcionar se ejecuta las siguientes líneas “./master-demo” y empieza el intercambio de información entre la raspberry pi y el dispositivo SEL2414.

```
pi@raspberrypi:~/proyectoDNP3/cpp $ cd examples/
pi@raspberrypi:~/proyectoDNP3/cpp/examples $ ls
decoder  master  master-gprs  master-udp  outstation  outstation-udp
pi@raspberrypi:~/proyectoDNP3/cpp/examples $ cd master
pi@raspberrypi:~/proyectoDNP3/cpp/examples/master $ ls
Analog.txt  CMakeFiles  cmake_install.cmake  digitales.txt  Makefile  master-demo  Prueba2.py
pi@raspberrypi:~/proyectoDNP3/cpp/examples/master $ ./master-demo
```

Figura 3.26: Ejecución de master-demo

En el proyecto se está adquiriendo datos analógicos y digitales, para que el dispositivo realice consultas por las señales digitales se añaden las siguientes líneas de código.

```
1 //Código para interrogar datos digitales
2
3 arch_2.open("digitales.txt",std::ofstream::trunc);
4         master->ScanRange(GroupVariationID(1,0), 0, 5,
5         test_soe_handler);
```

Como se observa se está realizando una interrogación de objetos tipo 1, variación 0 y el rango va desde el índice 0 hasta el índice 5.

3.4.3. Instalación y configuración de aplicativo Thingsboard

En la siguiente sección se revisará la instalación del aplicativo cuya función es almacenar los datos y ser la interfaz en tiempo real para consultar el estado de las señales que se están monitorizando del transformador de potencia de la subestación Duran norte.

El aplicativo Thingsboard tiene algunas versiones como son la comunity y profesional, en este proyecto se va a utilizar la versión comunity, ya que es el que contiene una gran cantidad de herramientas que para el proyecto es suficiente.

Para empezar se debe instalar la herramienta de desarrollo OpenJDK8 que se puede obtener desde el siguiente enlace https://adoptopenjdk.net/index.html, se recomienda la versión OPENJDK8 (LTS).

Al momento de la instalación se escoge la opción ADD TO PATH y Set JAVA HOME variable, una vez concluida la instalación por medio del intérprete de comandos, se escribe la siguiente instrucción y se muestra un resultado como el de la figura 3.27.

```
C:\Users\robert.coloma>java -version
openjdk version "1.8.0_275"
OpenJDK Runtime Environment (AdoptOpenJDK)(build 1.8.0_275-b01)
OpenJDK 64-Bit Server VM (AdoptOpenJDK)(build 25.275-b01, mixed mode)

C:\Users\robert.coloma>
```

Figura 3.27: Verificacion de openjdk instalada

El siguiente paso es descargar el paquete de instalación del aplicativo thingsboard, se puede descargar el paquete desde la página web del aplicativo o desde el siguiente enlace:

www.github.com/thingsboard/thingsboard/releases/download/v3.2.1/thingsboard-windows-3.2.1.zip

Se descomprime el paquete en el disco local C dentro de la carpeta Program File (x86) tal como en la figura 3.28.

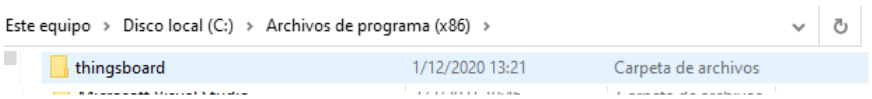


Figura 3.28: ruta de carpeta thingsboard

Previo a la instalación de la herramienta thingsboard, se debe instalar un gestor de base de datos, en este caso es Postgres Sql es con el cual el programa va a almacenar los datos que se estén reportando al aplicativo, se descarga de la página oficial el instalador de PostgreSQL, se recomienda descargar la versión 11.7 o versiones superiores, durante la instalación se solicita una clave para el superusuario acogiendo las instrucciones de instalación del aplicativo thingsboard, se coloca la clave “postgres”, una vez culminada la instalación se accede a el gestor de base de datos mediante el programa pgAdmin y se ingresa con las credenciales de superusuario, se ingresa al aplicativo PostgreSQL y se crea una base con nombre thingsboard, esta es la base donde se almacenan los datos. Se inicializa el intérprete de línea de comandos de Windows y se accede a la ruta donde se descomprimió la carpeta thinsgboard, una vez dentro de la carpeta se ejecuta las siguientes líneas de código, “install.bat –loadDemo” como en la figura 3.29, esta línea inicia la instalación del aplicativo.

```
C:\Program Files (x86)>cd thingsboard

C:\Program Files (x86)\thingsboard>install.bat --loadDemo
```

Figura 3.29: Linea de código para instalar thingsboard

Una vez culminada la instalación el paso siguiente es iniciar el servicio thingsboard se lo realiza de acuerdo como se observa en la figura 3.30.

```
C:\Program Files (x86)\thingsboard>net start thingsboard
El servicio de ThingsBoard Server Application está iniciándose.
El servicio de ThingsBoard Server Application se ha iniciado correctamente.


C:\Program Files (x86)\thingsboard>
```

Figura 3.30: Inicio de servicio thingsboard

Se accede al aplicativo a través de un navegador web con la siguiente dirección http://localhost:8080/, la página del programa solicita un usuario y contraseña, en la figura 3.31 se adjunta los diferentes roles de autenticación con los que se puede acceder.

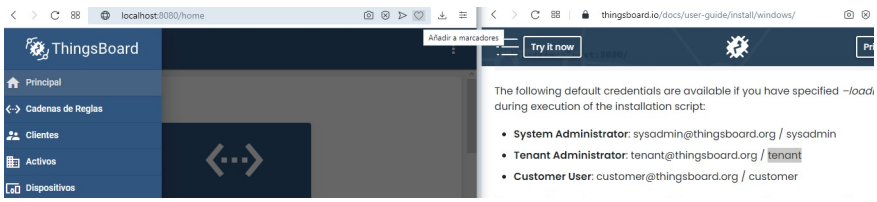


Figura 3.31: Formas de autenticación en aplicactivo thingsboard

3.4.4. Comunicación entre raspberry pi y aplicativo thingsboard

En esta sección se abordará la forma en como se ha establecido el flujo de información entre la raspberry pi y el aplicativo thingsboard, se implementa teorías y herramientas que se expusieron en el capítulo 2.

Los datos se almacenan en un archivo temporal, para acceder a estos datos del archivo temporal, se realizó un script en lenguaje de programación

Python con la finalidad de que los datos puedan ser publicados en el aplicativo thingsboard.

Para publicar los datos en la plataforma IoT, se debe crear un dispositivo en el aplicativo thingsboard igual a los de la figura 3.32, el dispositivo se creará con un parámetro denominado Token que es la identificación del dispositivo, este parámetro se lo emplea en el script de Python.

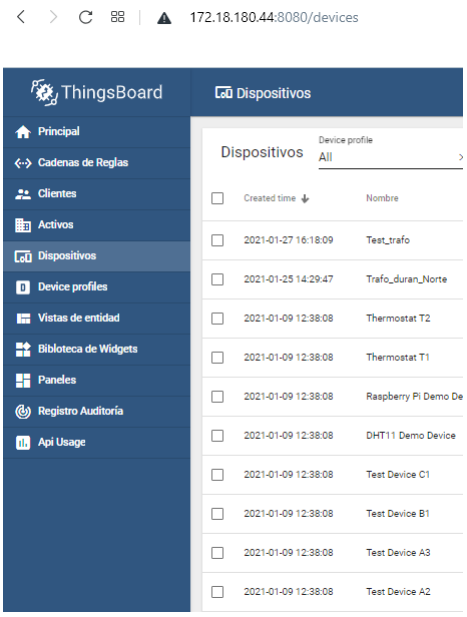


Figura 3.32: Dispositivos creados

Para crear o acceder a un dispositivo en el panel izquierdo del aplicativo web se encuentra una pestaña con nombre dispositivo, al momento de presionar sobre esta pestaña se desplegarán los dispositivos creados y la opción para crear un nuevo dispositivo tal como se observa en al figura 3.33.

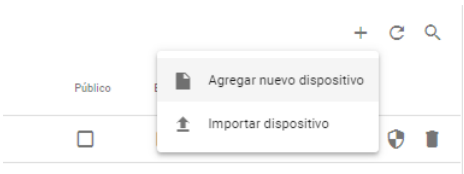


Figura 3.33: Agregar nuevo dispositivo

Como se observa en la figura 3.34 a través del icono + se crea un nuevo dispositivo, al momento de presionar se muestra una ventana donde se asigna un nombre, una etiqueta y como se va a transportar la información, el programa solicita si se desea asociar el dispositivo a un perfil o si se desea crear uno nuevo, para el proyecto se escoge uno nuevo. Una vez creado el dispositivo se presiona sobre el nombre del dispositivo y se muestra un conjunto de atributos, se copiará el access token que se lo va a emplear en el script de Python.

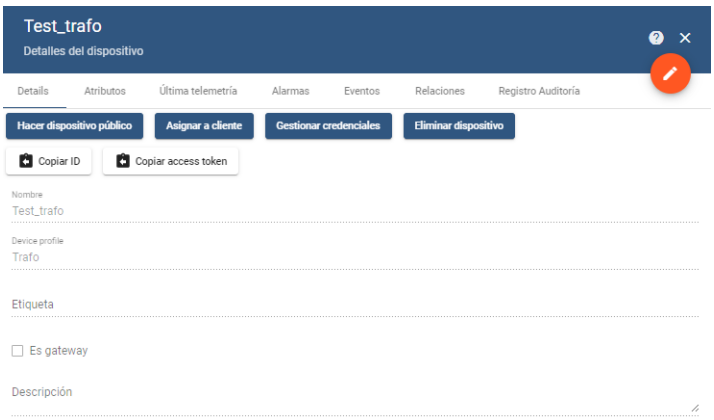


Figura 3.34: Atributos de dispositivo creado

El siguiente paso es crear el algoritmo de Python para esto se necesita primero instalar la librería Paho Mqtt la que tiene funciones para poder publicar datos a un dispositivo a través del de protocolo de MQTT, para instalar la librería se ejecuta en el shell del sistema operativo raspbian "pip install paho-mqtt". El script está compuesto por muchas partes, en la figura 3.35 se observa las librerías que se implementan y los atributos que se deben establecer.

En el atributo declarado access token es donde se coloca el código access token que se copió del dispositivo creado en el aplicativo thingsboard, en el atributo broker se coloca la IP del servidor o equipo de cómputo donde se encuentre funcionando el aplicativo thingsboard, el atributo port es el puerto por el cual funciona el protocolo MQTT en este caso es el 1883 asi como se visualiza en la figura 3.35.

```
import paho.mqtt.client as paho
import json
import random
import time
import os
from datetime import datetime

ACCESS_TOKEN='heb6bf9KCRnDNpeK7grr'
broker="172.18.180.44"
port=1883
```

Figura 3.35: Librerías y parámetros utilizados en script Python

En la figura 3.36 se declara un objeto de tipo de Paho se sobreescribe el método .ºn publish"para que se imprima por consola los datos que se van a publicar, con la función çonnect"se inicializa la conexión pasando como parámetros el broker y el puerto, con la función loop start se la implementa para mantener la conexión siempre activa.

```
def on_publish(client,userdata,result):
    print("data published to thingsboard \n")
pass
client1= paho.Client()
client1.on_publish = on_publish
client1.username_pw_set(ACCESS_TOKEN)
client1.connect(broker,port,keepalive=60)
client1.loop_start()
sensor_data={}
```

Figura 3.36: Creación y parametrización de objeto tipo Paho

Se especifica la forma de como publicar los datos al aplicativo thingsboard, se lee el archivo temporal de señales analógicas y se realiza la verificación si el índice es 0 corresponde al valor de la temperatura de Aceite, pero si el índice es 1 corresponde al valor de la temperatura de devanado, luego en la linea çlient.publish"se realiza la publicación del dato al aplicativo thingsboard, la función json.dumps realiza la conversión de datos tipo python a datos tipo JSON, ya que el aplicativo espera una estructura de este tipo para poder interpretar los datos.

```
1 #Programa que publica los datos Analógicos en la plataforma IOT
2
```

```
f=open('Analog.txt','r')
    for lista in f.readlines():
        caracter= ':'
        pos =lista.index(caracter)
        valor = lista[:pos]
        indice= lista[pos+1:]
        print(valor)
        if indice=='1\n':
           sensor_data['Tmp_Devanado']=valor
        if indice=='0\n':
           sensor_data['Tmp_Aceite']=valor
           print("mandando Valor")
        client1.publish("v1/devices/me/telemetry",
        json.dumps(sensor_data),1)
```

Para la publicación de señales digitales es el mismo procedimiento, se lee el archivo temporal y se valida que índice se está leyendo y dependiendo del valor del índice se enlaza el valor de la señal con la clave que le corresponde, tal como se lo realizó con las señales analógicas.

```
#Programa que publica los datos digitales en la plataforma IOT

arch_2=open('digitales.txt','r')
    for lista in arch_2.readlines():
        caracter = ':'
        pos=lista.index(caracter)
        valor=lista[:pos]
        indice=lista[pos+1:]
        print (valor)
        if indice=='0\n':
            sensor_data['Nivel_min_Aceite']=valor
        if indice=='1\n':
            sensor_data['Nivel_max_Aceite']=valor
        if indice=='2\n':
            sensor_data['Alarma_rele_Bucholz']=valor
        if indice=='3\n':
            sensor_data['Disp_Valvula_Sobrepresion']=valor
        if indice=='4\n':
            sensor_data['Disp_Presion_Subita']=valor
        client1.publish("v1/devices/me/telemetry",
        json.dumps(sensor_data),1)
```

Lo siguiente es crear una interfaz para que los usuarios visualicen los datos de una forma mas interactiva, el aplicativo IoT utiliza los paneles para visualizar los datos de cada dispositivo, para la creación de los paneles se presiona en la sección de paneles y se da un clic en el símbolo +, a continuación se muestra las opciones que se deben completar tal como se muestra en las siguiente figura 3.37.

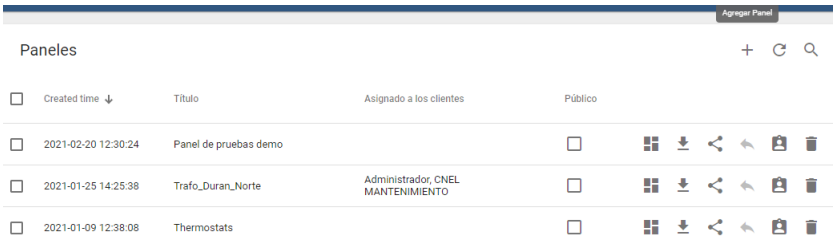


Figura 3.37: Pestaña de paneles

En la figura 3.38 se observa la ventana donde se crean los nuevos paneles.

Figura 3.38: Creación de nuevos paneles

Al momento de crear un nuevo panel el programa solicita un nombre al panel tal como en la figura 3.39

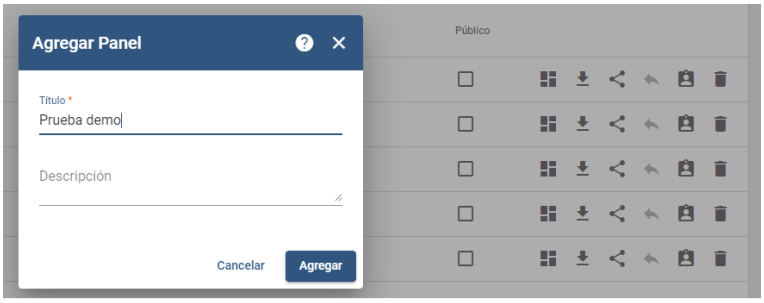


Figura 3.39: Parámetros de panel nuevo

Se completa el nombre y la descripción del panel, una vez se completen los datos, el nuevo panel se agrega al conjunto de paneles creados.

Una vez creado el panel se accede a el y se despliega una ventana donde

se muestran las diferentes opciones, se puede exportar el panel para tener un respaldo en caso de un problema con el servidor donde se tiene instalado la plataforma IoT, también se puede gestionar los usuarios que tengan acceso al panel, se selecciona la pestaña de abrir el panel para poder agregar los diferentes elementos.

En la figura 3.40 el panel es el lienzo donde se agregan los widgets que son los elementos que muestran los valores de las señales, reportes de alarmas o datos históricos, para agregar los widgets se debe entrar en modo edición del panel, el cual se lo realiza mediante el ícono en forma de lápiz, aquí se habilita las diferentes opciones como añadir un nuevo widget, editarlo o salir del modo edición.

Figura 3.40: Modo edición panel

Antes de agregar los widgets, se debe crear un alias de entidad, un alias de entidad es una identificación de un dispositivo o varios, para crearlo se debe presionar en el icono superior de la barra del panel.

El programa muestra una ventana donde se llenan los campos del alias de entidad, se debe colocar un nombre, en el filtro por entidad se escoge la opción única entidad, luego se despliega dos opciones más en la opción tipo se escoge dispositivo, en la pestaña dispositivo se muestra todos los dispositivos creados de la lista se escoge uno tal como se muestra en la figura 3.41, para finalizar se presiona en el botón agregar y se creará el alias de entidad.

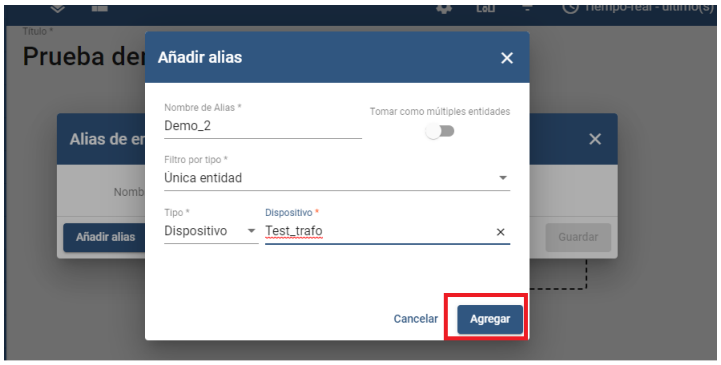


Figura 3.41: Parámetros de alias de entidad

Una vez creado el alias de entidad se presiona el botón guardar y se concluirá con la creación del alias de entidad tal como en la figura 3.42.

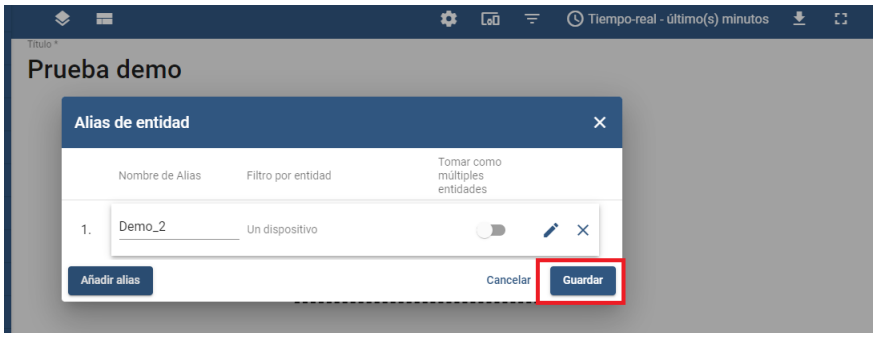


Figura 3.42: Alias de entidad agregado

Tal como se muestra en la figura 3.40 para agregar un widget se presiona en el icono de Agregar nuevo widget, el cual muestra una ventana con los diferentes widgets que tiene el aplicativo IoT, existen diferentes tipos de widgets ya depende del administrador como se desea que se visualizan los datos. Al momento de escoger un widget, se mostrará una ventana con las siguientes opciones tal como se muestra en la figura 3.43

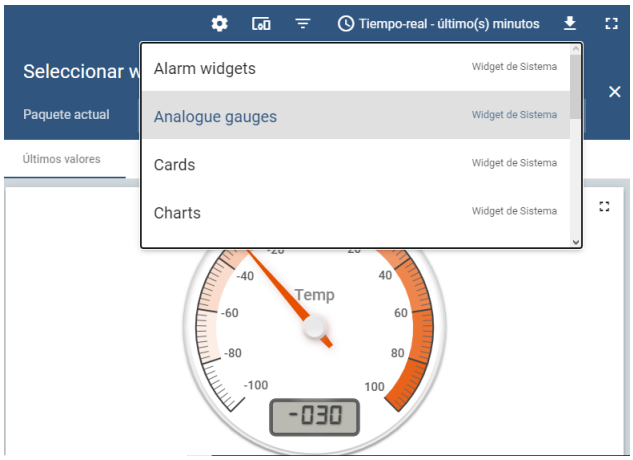


Figura 3.43: Diferentes opciones de widgets

Se escoge el tipo de dato, si es una respuesta de una función o un valor de una entidad, luego escogemos la entidad creada y de la entidad creada se mostrará los diferentes señales que se pueden mostrar, en este caso se necesita la señal de temperatura de aceite tal como se observa figura 3.44.

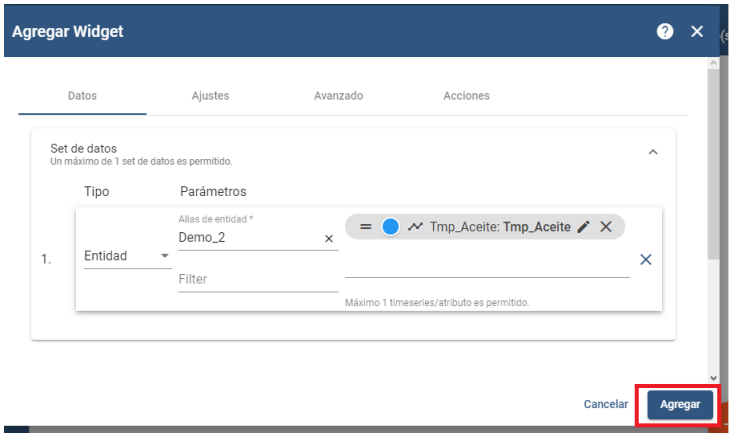


Figura 3.44: Vinculación de señal con los wigets

Para finalizar se presiona en el botón agregar y se crea el widget con la señal vinculada, para finalizar la edición en el panel se presiona en el icono visto y se observarán los datos tal como en la figura 3.45 .

Figura 3.45: Widget agregado al panel

Otro tipo de widget es el de tendencia, en el cual se puede observar el comportamiento de la señal a través del tiempo, este tipo de widget se encuentra en el paquete chart tal como se muestra en la figura 3.46

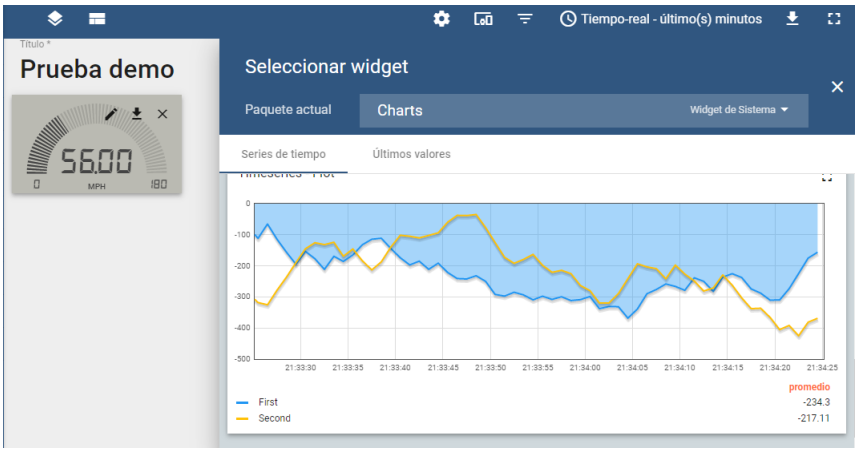


Figura 3.46: Widget de tendencia

Este tipo de widget son los que se agregaron en el proyecto para mostrar los datos de temperatura de aceite y devanado. Para las señales digitales se utiliza el paquete de widget control widgets como se observa en la figura 3.47, el que se emplea es el led indicator el cual cambia de color cuando varía

el valor de la señal que se vincule, el widget se agrega de la misma forma de como se agregaron los widgets para las señales analógicas.

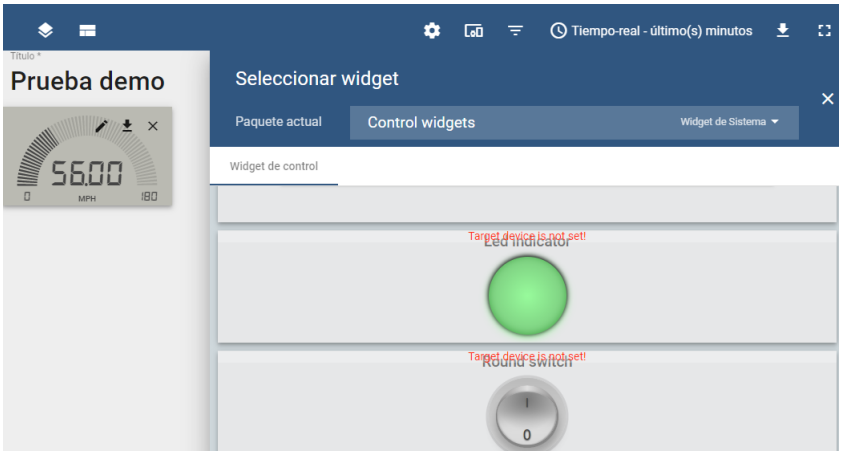


Figura 3.47: Widget para señales digitales

Para el funcionamiento del widget digital se realiza un clic en la pestaña avanzada, luego se desactiva la opción perform rpc device status check, se configura el resto de parámetros tal como se muestra en la figura 3.48, en el atributo del dispositivo se coloca el nombre de la señal, así el widget queda asociado a la señal que se escogió.

Figura 3.48: Parámetros de configuración de widget

El widget que se agregó para completar el panel del proyecto es el widget de alarmas, los parámetros de configuración del widget de alarmas para el funcionamiento son los que se muestran en la figura 3.49.

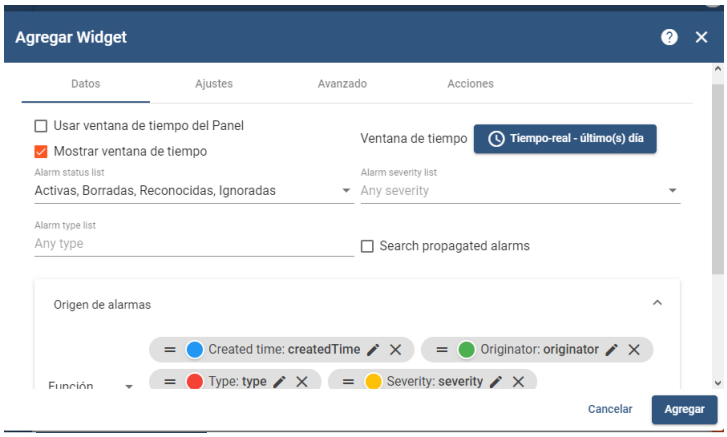


Figura 3.49: Primera parte de configuración de widget alarma

En la primera parte que se muestra en la figura 3.49 se puede seleccionar que tipo de señales se requiere que se muestren en el widget de alarmas, también se puede filtrar por la severidad de la señal si se desea mostrar en tiempo real o por históricos, en la segunda parte se asocia el alias de entidad, en la columna de origen de alarmas se escoge entidad y luego en el campo alias de entidad se escoge la entidad que ya se había creado tal como se muestra en la figura 3.50.

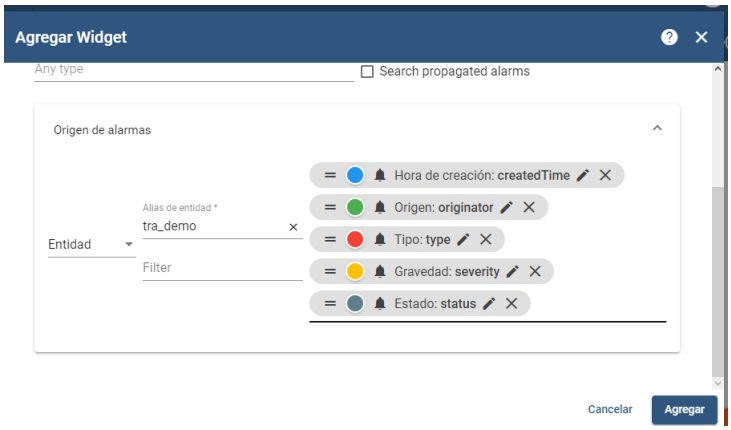


Figura 3.50: Segunda parte de configuración de widget alarma

Para el funcionamiento del widget de alarmas se debe realizar la configuración de las cadenas de reglas que es donde se ajustará los parámetros para que se generen las alarmas. En la figura 3.51 en la parte izquierda del panel escogemos la pestaña de cadena reglas en el icono +, se completa los datos como el nombre y la descripción de la cadena, una vez completado los datos se presiona agregar.

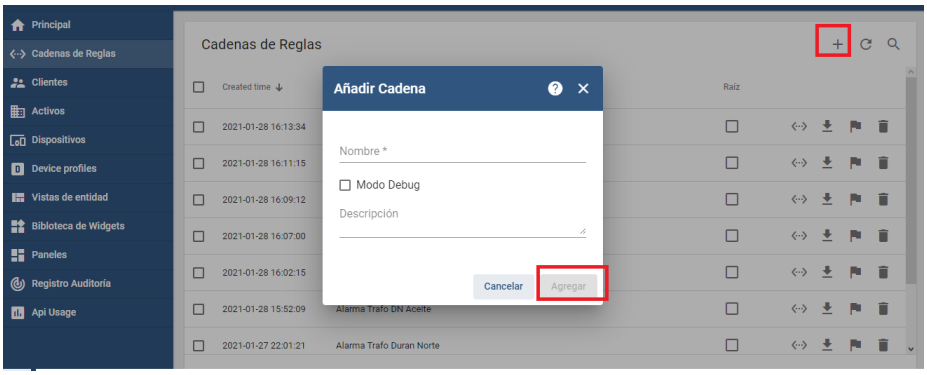


Figura 3.51: Ventanas de cadena de reglas

Luego de crear la cadena de regla se realiza un clic sobre el nombre de la cadena el cual abrirá una ventana y se presiona en la pestaña open rule chain, se abrirá una ventana que es donde se realiza la agregación y programación de cada bloque tal como se muestra la figura 3.52, en el panel izquierdo

se encuentran los diferentes bloques que se pueden programar para que se generen los diferentes eventos.

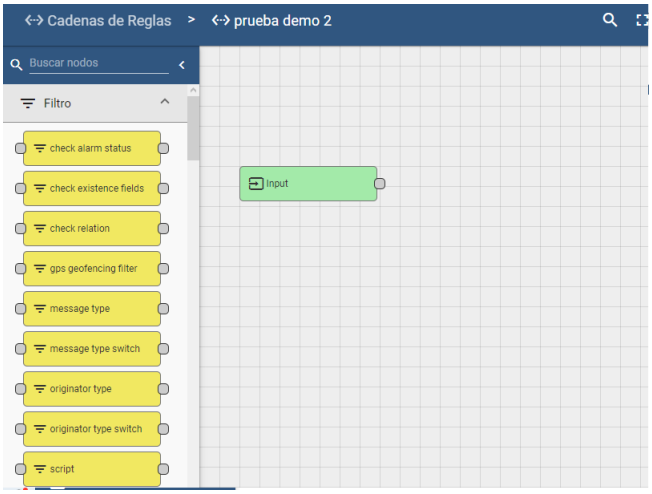


Figura 3.52: Ventana de bloques y programación

La programación de bloques que se va a utilizar para generar una alarma, son las que se muestran en la figura 3.53

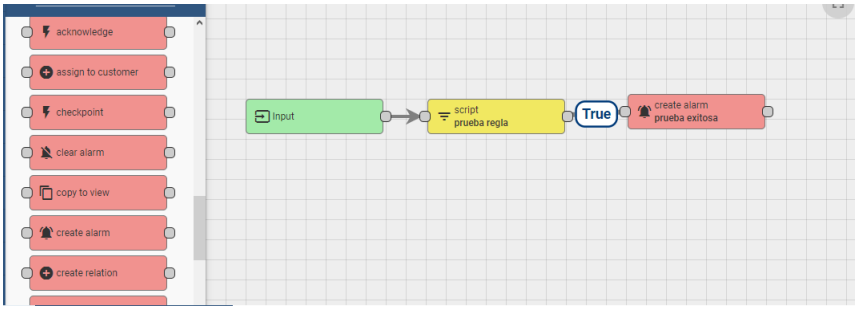


Figura 3.53: Configuración de bloques

El bloque script es donde se realiza la programación, tal como se observa en la figura 3.54 aquí se coloca la condición para que se genere un valor de true o false.

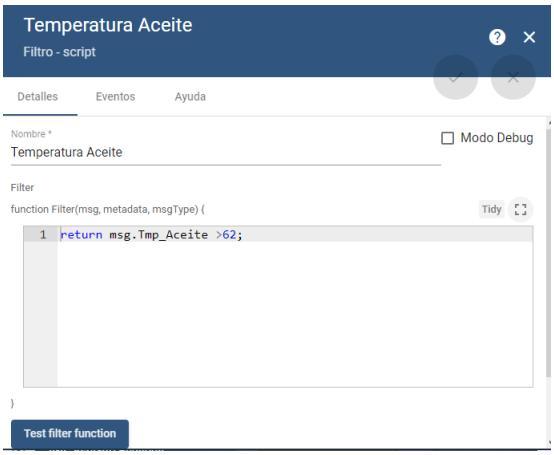


Figura 3.54: Programación bloque script

El bloque que realiza que se publique las alarmas en el aplicativo IoT y por lo tanto en el widget de alarmas es el bloque create alarm, en este bloque se especifica el tipo de alarma que se va a generar y el mensaje que se va a mostrar en el panel de alarmas como en la figura 3.55, entre el bloque script y el bloque create alarm debe estar enlazado mediante el mensaje true.

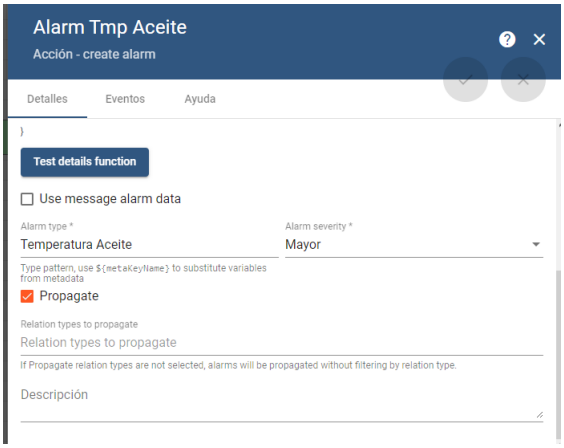


Figura 3.55: Bloque de creación de alarmas

Otro bloque que se debe agregar es el bloque de clear alarma que funciona cada vez que se reconoce las alarmas generadas, automáticamente se limpian

del panel de alarmas, el bloque se activa cuando le llega un mensaje false, por lo que la programación final de la cadena de regla es como se muestra en la figura 3.56

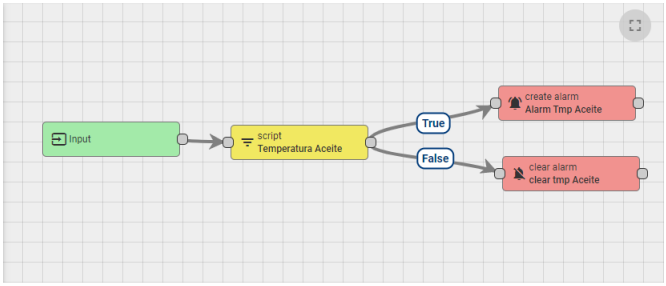


Figura 3.56: Configuración final de bloques

Para que funcione la cadena de regla, se accede a la pestaña root rule chain a continuación se abrirá la ventana para añadir bloques, buscamos la categoría cadena de regla y se agrega un bloque de tipo cadena de regla, al agregar el bloque se debe asociar una de las cadenas de reglas creadas y se enlaza con el bloque save Timeseries como se observa en la figura 3.57

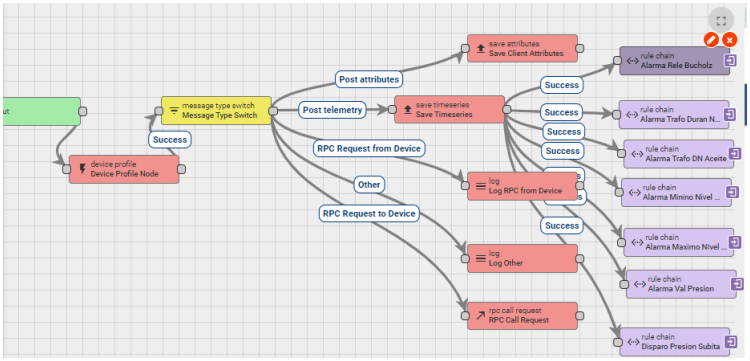


Figura 3.57: Bloque de cadena de regla raíz

Con todas las configuraciones revisadas ya el widget de alarmas mostrará los mensajes correspondientes cuando algunas de las señales cambie de sus límites tal como fueron configuradas para el proyecto, en total se crearon 7 cadenas de reglas, ya que se configuraron una cadena por cada señal.

Capítulo 4

Resultados

En este capítulo se revisará las diferentes pruebas a las que fue sometido el sistema, forzando cada una de las señales ya que someter a falla los elementos censados implica la desconexión eléctrica del transformador de la subestación. Se observará su desempeño y sus tiempos de respuesta con respecto a las señales digitales y señales analógicas, se verificará el acceso al sistema con los diferentes usuarios que fueron creados y así como la generación de alarmas con cada uno de ellos.

También se presentará el panel definitivo del sistema después de las diferentes configuraciones que se explicaron en el capítulo 3 para visualizar los datos y el que el personal del área de mantenimiento utilizará para monitorear los elementos.

4.1. Resultados

Para comprobar que el IED este adquiriendo los datos se realizo simulaciones en cada uno de las entradas digitales para visualizar el correcto funcionamiento, para verificar la información se utilizo el software Quickset Ascelerator, este software aparte de configurar local y remotamente el dispositivo, también contiene una herramienta para visualizar los componentes del equipo.

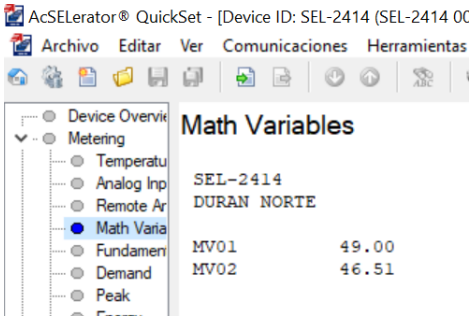


Figura 4.1: Valores censados de campo

Como se observa en la figura 4.1 el equipo esta censando los valores analógicos como son la temperatura de aceite y de Devanado 1.

Con el mismo software Ascelerator a través de su consola de comandos se verificó los eventos ocurridos donde se evidencia el estado de las entradas digitales.

```
114  02/18/2021  08:43:24.5306  IN301                Asserted
113  02/18/2021  08:43:25.8356  IN301                Deasserted
112  02/18/2021  08:47:01.5911  IN301                Asserted
111  02/18/2021  08:47:01.6576  IN301                Deasserted
110  02/18/2021  08:47:01.7071  IN301                Asserted
109  02/18/2021  08:47:38.8016  IN301                Deasserted
108  02/18/2021  08:48:35.8821  IN302                Asserted
107  02/18/2021  08:48:49.6881  IN302                Deasserted
106  02/18/2021  08:51:30.7896  IN303                Asserted
105  02/18/2021  08:51:30.9261  IN303                Deasserted
104  02/18/2021  08:51:30.9566  IN303                Asserted
103  02/18/2021  08:51:43.4726  IN303                Deasserted
102  02/18/2021  08:53:10.1171  IN304                Asserted
101  02/18/2021  08:53:18.1866  IN304                Deasserted
100  02/18/2021  08:56:46.1866  IN301                Asserted
99   02/18/2021  08:57:09.7801  IN301                Deasserted
98   02/18/2021  11:52:30.9036  IN301                Asserted
97   02/18/2021  11:52:39.7302  IN301                Deasserted
96   02/18/2021  11:59:19.8976  IN302                Asserted
95   02/18/2021  11:59:51.7681  IN302                Deasserted
94   02/18/2021  12:01:06.5561  IN301                Asserted
93   02/18/2021  12:01:24.9196  IN301                Deasserted
92   02/18/2021  12:02:58.5312  IN301                Asserted
91   02/18/2021  12:03:36.2346  IN301                Deasserted
90   02/18/2021  12:10:25.3821  IN102                Asserted
```

Figura 4.2: Registro de eventos

En la figura 4.2 se observa como cambia el estado de cada una de las

entradas digitales cuando se realizaron las simulaciones de falla.

La comunicación entre el Dispositivo SEL 2414 Y la raspberry Pi se implemento el protocolo DNP3 en la figura 4.3 se observa las tramas del protocolo.

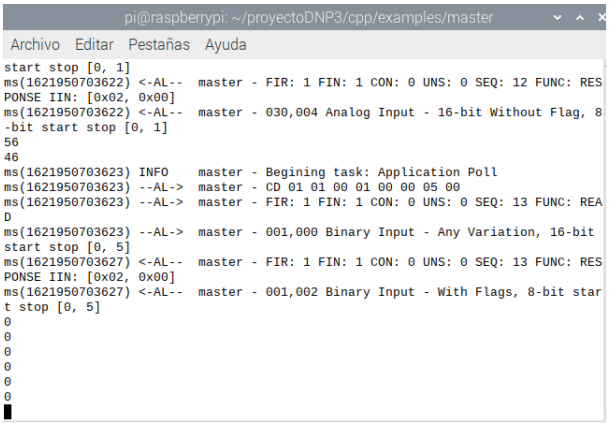


Figura 4.3: Tramas de comunicación DNP3

En la figura 4.3 se observa que la raspberry interroga por los objetos analógicos y el dispositivo SEL 2414 responde con los valores 56 y 46, en lineas mas abajo se observa que ahora la raspberry pregunta por los objetos digitales y el dispositivo SEL 2414 responde con los valores de las señales digitales sus valores son 0 ya que en ese momento no hay un evento.

Una vez se esten guardando los valores obtenidos por medio del protocolo DNP3 en pequeños archivos temporales se ejecuto el algoritmo de python.

```
51.000000
mandando Valor
46.000000
data published to thingsboard

0
 data published to thingsboard

0
0
 data published to thingsboard

0
0
 data published to thingsboard

0
data published to thingsboard
```

Figura 4.4: Publicación de datos en aplicativo thingsboard

En la figura 4.4 se observa que los datos se están publicando en la plataforma IoT, se evidencia la publicación de los valores analógicos estos son leídos desde el archivo temporal, así como las señales digitales.

Con el dispositivo creado en el aplicativo thingsboard, el algoritmo de python y la librería opendnp3 interactuando, se procede a verificar que los datos se estén publicando correctamente en el aplicativo.

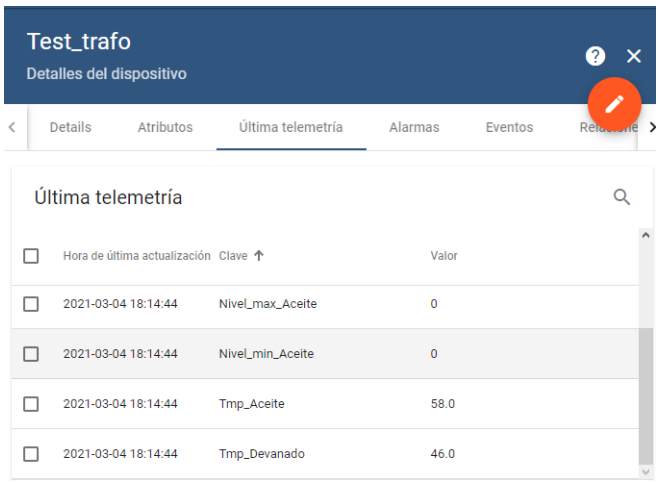


Figura 4.5: Publicación de datos en aplicativo thingsboard

Como se puede observar en la figura 4.5, los datos se están publicando y se ve cada cuanto tiempo se están actualizando los valores, la actualización de los datos depende cada cuanto se haya programado el tiempo de publicación, en el script de Python.

Una vez realizada las diferentes configuraciones en el aplicativo Thingsboard, en la figura 4.6 se presenta el panel definitivo con el cual los usuarios visualizaron los datos que la Raspberry está reportando a la plataforma Web.

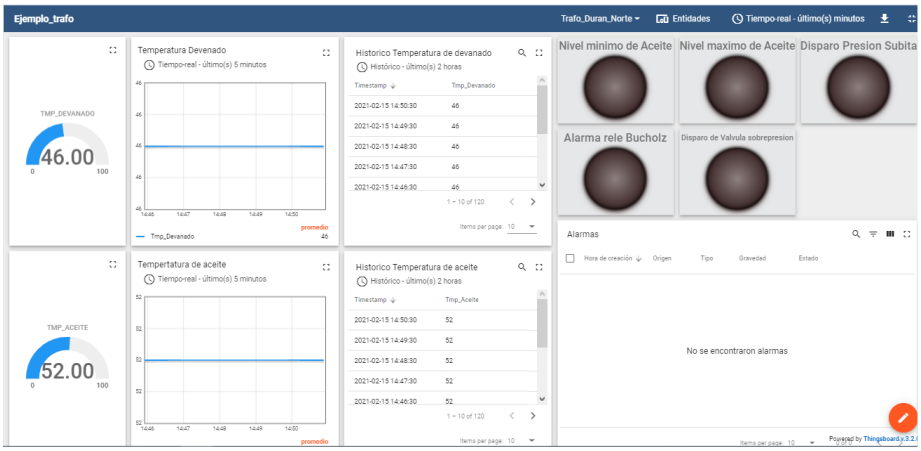


Figura 4.6: Panel de monitoreo de parámetros de transformador de potencia

En el panel se observa que se colocaron a cada una de las alarmas digitales un indicador led el cual se enciende cuando la señal cambia su estado de normal ha alarmado, en la parte inferior derecha se observa también las alarmas que se generan en el sistema

4.1.1. Prueba de señal digital mínimo nivel de aceite

Las primeras pruebas realizadas al sistema fueron las señales digitales, el dispositivo SEL 2414 tiene unos módulos de tarjetas, cada tarjeta se representan en orden del alfabeto A,B,C,..Etc. Para el proyecto se utiliza las entradas digitales de la tarjeta C, todas sus entradas tiene la codificación de IN301, IN302, IN303...Etc. La primera prueba fue polarizar la entrada digital IN301 tal como se revisó en la sección 3.4.1, la bornera corresponde a la señal de mínimo nivel de aceite, para realizar la simulación de falla de la señal, como se tiene polarizada la bornera C02 con negativo habría que

colocar en C01 un positivo de la fuente de voltaje de 125 DC tal como se muestra en la figura 4.7.

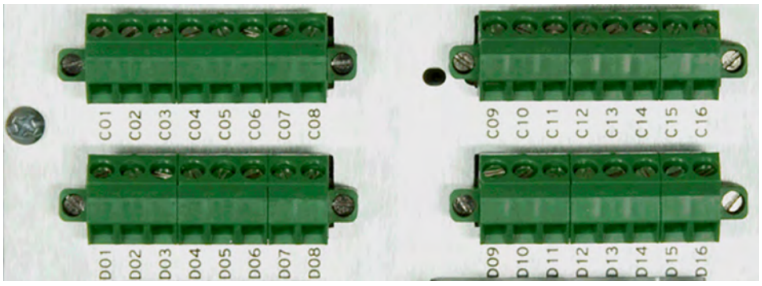


Figura 4.7: Bloque de borneras de la tarjeta C

Una vez polarizada la entrada digital IN301 en la figura 4.8 se observa que se enciende el led que corresponde al nivel mínimo de aceite, el color con el que se enciende será rojo, esto es configurable, ya que depende de los requerimientos de la empresa o de los usuarios que vayan a utilizar el aplicativo, también se genera un mensaje en el panel de alarmas, con el aplicativo Quickset también se verificó que la entrada IN301 se active.

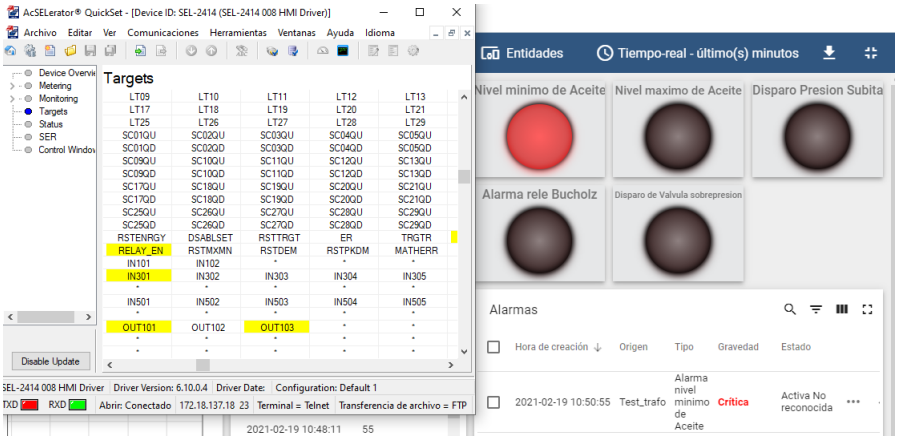


Figura 4.8: Señal de nivel mínimo de aceite alarmada

Se realizaron las pruebas con las demás señales digitales que se estén reportando al aplicativo Web.

4.1.2. Prueba de señal digital máximo nivel de aceite

En la figura 4.9 se repetirá los procesos que se mencionó en la sección anterior pero ahora con la entrada digital IN302 que corresponde a la señal de máximo nivel de aceite.

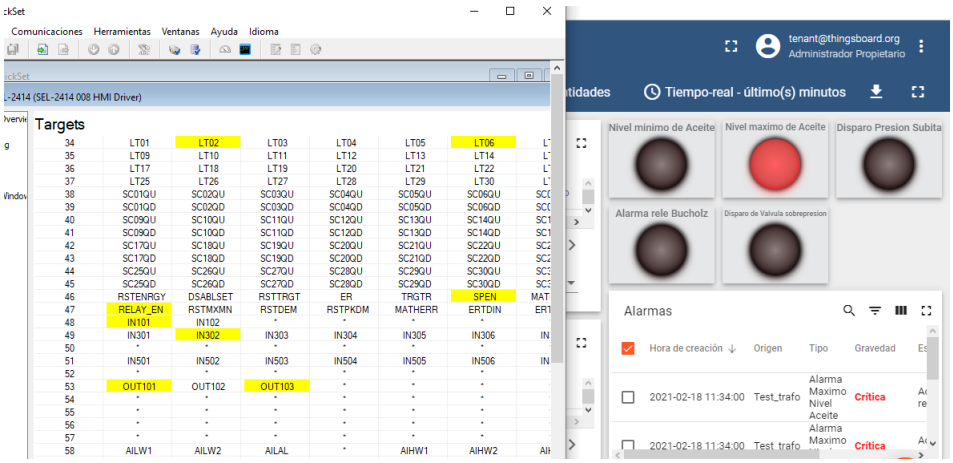


Figura 4.9: Señal de máximo nivel de aceite alarmada

4.1.3. Prueba de señal alarma de relé Buchholz

Se sigue el mismo proceso mencionado ahora con la entrada digital IN303 tal como se muestra en la figura 4.10, que corresponde a la alarma de relé Buchholz.

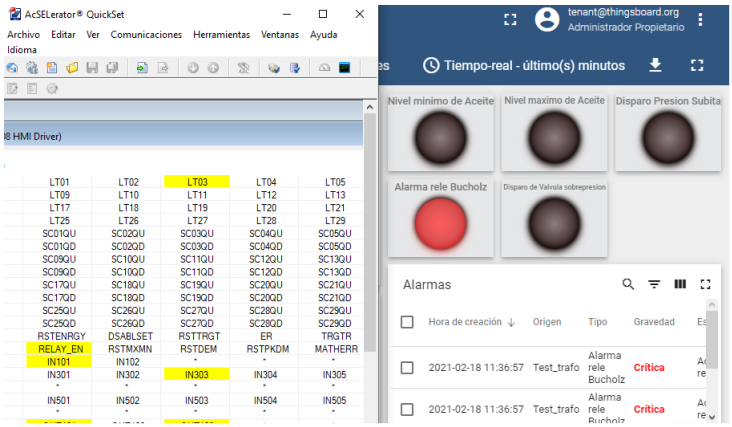


Figura 4.10: Señal de de relé buchholz alarmada

4.1.4. Prueba de señal Disparo de válvula sobrepresión

Para esta caso la entrada IN304 es la que se va a probar tal como se muestra en la figura 4.11 .

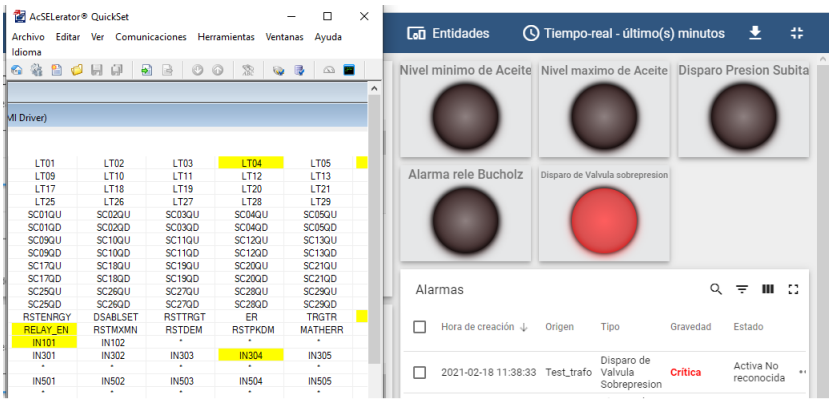


Figura 4.11: Señal de disparo de válvula de sobre presión alarmada

4.1.5. Prueba de señal Alarma de presión súbita

La última prueba que se realizó en las señales digitales es la de presión súbita tal como se observa en la figura 4.12, la señal se encuentra conectada en la entrada IN305.

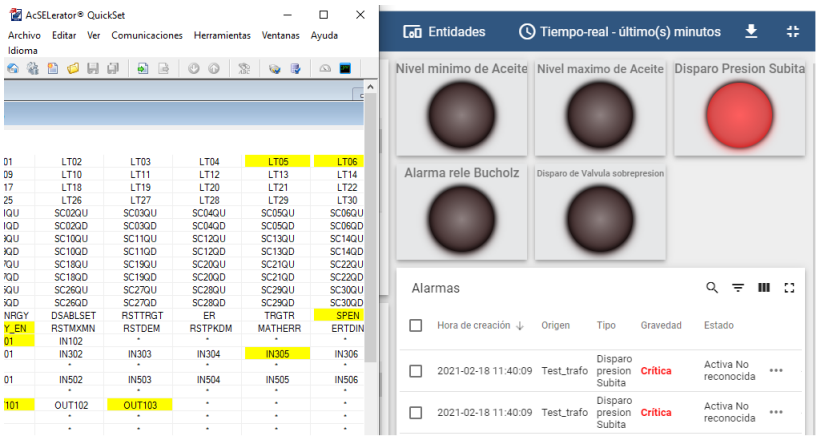


Figura 4.12: Señal de Disparo de presión súbita alarmada

4.1.6. Pruebas de generación de alarma de señales analógicas

Para poder generar la alarma de una señal analógica lo que se realizó es bajar el límite en el cual se considera como alarma el valor que se esté censando, el sistema se encontraba configurado en que se genere una alarma cuando la temperatura del aceite sobrepase los 70 grados, para esto se simulará que la señal tiene un valor de 75 grados con un pequeño script que se programó, en el panel de alarma se mostrará un mensaje en la columna de gravedad de nivel Mayor tal como se muestra en la figura 4.13

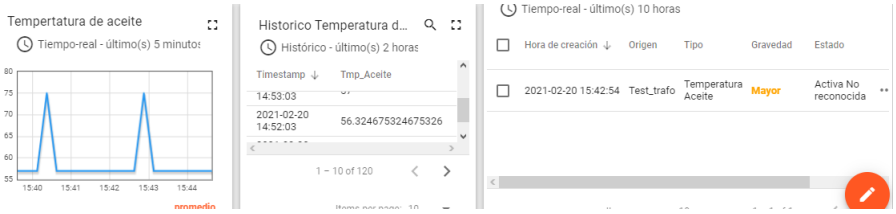


Figura 4.13: Generación de alarma de temperatura de aceite

4.1.7. Cantidad de usuarios en el aplicativo

Se crearon 5 cuentas para que puedan acceder al aplicativo sin tener ningún inconveniente, cada usuario puede visualizar los datos pero no pueden realizar alguna modificación sobre el panel, solo el usuario con el perfil de

administrador modificador puede modificar el sistema. En la figura 4.14 se observa la creación de los 5 usuarios que pueden acceder al sistema.

Figura 4.14: usuarios con acceso al panel del transformador

En la figura 4.15 se observa que el personal de mantenimiento accede al aplicativo y observa el panel con todas sus características.

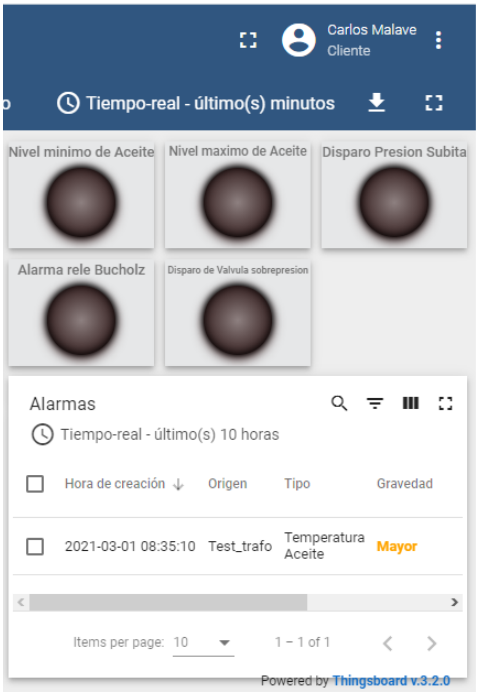


Figura 4.15: Personal de mantenimiento usando el aplicativo Thingsboard

Capítulo 5

Conclusiones y Recomendaciones

5.1. Conclusiones

Los transformadores contienen una gran cantidad de señales que se pueden extraer, sin embargo, para este proyecto se están monitoreando solo 7 parámetros que son los que el área de mantenimiento tiene identificados, esto se debe a que son equipos que tienen muchos años instalado y la información de los planos es escasa.

En la etapa de comunicación entre el dispositivo SEL 2414 y la raspberry PI se aplicaron herramientas de ciertas versiones anteriores como es el caso de CMAKE y la librería opendnp3, debido a que las versiones más actualizadas no eran compatibles con la raspberry PI. Por defecto se modificó el archivo principal de la librería para que funcione las interrogaciones por objeto ya que por defecto vienen activadas la interrogación por clases.

Para preparar la información que se iba a publicar en la plataforma IOT, el algoritmo de la librería opendnp3 al momento de obtener los datos se debía guardar en un archivo temporal, ese mismo archivo era leído por el algoritmo de pyhton, en esta parte del proyecto se tuvo que experimentar con los tiempos entre la lectura de los datos y la publicación de los datos. Se elevó un poco los tiempos en el algoritmo de Python para que no se publicaran datos repetidos. Al transmitir la información a la plataforma se debe tener mucha precaución con las etiquetas ya que estas son las que contienen los valores de las señales si la etiqueta es diferente la plataforma lo toma como si fuese una variable diferente y comienza a guardar los valores publicados en esa nueva variable.

En el instante de realizar las pruebas en conjunto de todos los elementos, como las señales de protecciones mecánicas del transformador son pulsos muy rápido, en el dispositivo SEL se tuvo que alargar el tiempo de activación de las señales, con la finalidad que las raspberry al momento de realizar la consulta de los valores, estos se encuentren aun presentes y por lo tanto se puedan transmitir a la plataforma, como el equipo se encuentra operativo se realizaron puentes en cada uno de las borneras de las entradas digitales para comprobar que se estén publicando las alarmas en la plataforma.

De las pruebas realizadas con los usuarios se evidenció que ellos tienen acceso a los paneles a los cuales ellos han sido asignados, así como a las alarmas que se generan o se generaron, los usuarios pueden visualizar si fueron reconocidas o no y poder reconocerlas.

Después de las diferentes pruebas y los tiempos de funcionamiento del sistema se puede concluir que el proyecto cumple con el objetivo general, se tiene un sistema que supervisa en tiempo real los parámetros de un transformador.

5.2. Recomendaciones

Como se mencionó en capítulos anteriores se crearon algunos usuarios para que interactúen con el aplicativo web, de ellos se recogió sus diferentes experiencias y recomendaciones que se podrían emplear en la mejora a futuro del sistema.

- El proyecto es novedoso e intuitivo, el costo de la implementación es bien económico, también se recibió sugerencias que se adquieran más señales que formen parte del transformador como las corrientes en los devanados y voltajes por cada fase.
- Aunque el script de Python que se utilizó cumple con el objetivo del proyecto se recomienda realizar mantenimiento trimestrales para las actualizaciones de parches y seguridad, así como mejorar los recursos consumidos en la red de datos.
- Al utilizar equipos de hardware libre es recomendable que el sistema operativo sea compatible con versiones de Python 3.2 y CMAKE, asi como la versiones de C++ y en este caso la libreria opendnp3.

Capítulo 6

Glosario

CMAKE herramienta multiplataforma de generación o automatización de código.

CNEL Corporación Nacional de Electricidad.

DNP3 Protocolo de comunicación para las redes de distribución.

FTP Protocolo de red para transferencia de Archivos.

gateway Puerta de enlace para establecer comunicación con otra red de datos.

IED Dispositivo electrónico Inteligente.

IoT Internet Of Everything (Internet de las cosas).

JSON formato de texto sencillo para el intercambio de datos.

MQTT Protocolo de mensajeria para telemetria máquina a máquina.

OpenJDK8 Plataforma de desarrollo de java de versión libre.

PLC Programador logico Controlable.

Python Lenguaje de programacion orientado a objetos.

Raspberry Mini Ordenador de Hardware libre.

Raspbian Sistema operativo del dispositivo Raspberry Pi.

RTU UNidad Terminal Remota.

SCADA Supervisory Control And Data Acquisition (Supervisión, Control y Adquisición de Datos).

SEL Empresa de desarrollo de equipos tecnológicos para la automatización y protección de sistemas de control.

telnet Protocolo de red que permite la conexion de una maquina remotamente.

Windows Sistema operativo de la empresa microsoft.

XML Lenguaje de Marcas Extensible.

Bibliografía

R. E. Alvarez and M. Del Pozo. MANTENIMIENTO DE TRANSFORMADORES DE POTENCIA. page 9, 2007. URL http://sedici.unlp.edu.ar/bitstream/handle/10915/36793/Documento{_}completo.04{_}Alvarez.pdf?sequence=1.

M. Ayala S, G. Botura, and O. A. Maldonado A. AI automates substation control. *IEEE Computer Applications in Power*, 15(1):41–46, 2002. ISSN 08950156. doi: 10.1109/67.976991.

S. Balamurugan and D. Saravanakamalam. Energy monitoring and management using internet of things. In *International Conference on Power and Embedded Drive Control, ICPEDC 2017*, pages 208–212. Institute of Electrical and Electronics Engineers Inc., oct 2017. ISBN 9781509046775. doi: 10.1109/ICPEDC.2017.8081088.

J. Bernal. *APLICACIÓN DE ALGORITMOS GENÉTICOS AL DISEÑO ÓPTIMO DE SISTEMAS DE DISTRIBUCIÓN DE ENERGÍA ELÉCTRICA*. PhD thesis, 1998.

J. C. CASTRO VAZQUEZ. No INTEGRACION DE SUBESTACIONES AL SISTEMA AVANZADO PARA EL MENAJO DE LA DISTRIBUCION DEL ECUADOR Title. Technical report, 2019.

G. Clarke, D. Reynders, and E. Wright. *Practical modern SCADA protocols: DNP3, 60870.5 and related systems*. Newnes, 2004.

CNEL. CNEL EN CIFRAS UNIDAD DE NEGOCIO GUAYAS LOS RIOS, 2019. URL https://www.cnelep.gob.ec/cnel-en-cifras/.

K. Coronado and I. Palacio. *ELEMENTOS DE DISEÑO DE SUBESTACIONES DE ALTA Y EXTRA ALTA TENSIÓ*. PhD thesis, 2008.

I. Darwish, O. Igbe, O. Celebi, T. Saadawi, and J. Soryal. Smart grid dnp3 vulnerability analysis and experimentation. In *2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing*, pages 141–147. IEEE, 2015.

F. Englert, P. Lieser, A. Alhamoud, D. Boehnstedt, and R. Steinmetz. Electricity-metering in a connected World: Virtual sensors for estimating the electricity consumption of IoT appliances. In *Proceedings - 2015 International Conference on Future Internet of Things and Cloud, FiCloud 2015 and 2015 International Conference on Open and Big Data, OBD 2015*, pages 317–324. Institute of Electrical and Electronics Engineers Inc., oct 2015. ISBN 9781467381031. doi: 10.1109/FiCloud.2015.38.

ENSOTEST. Introduccion la norma IEEE 1815 - DNP3, 2021. URL https://www.ensotest.com/es/dnp3/introduccion-a-la-norma-ieee-1815-dnp3/.

S. Fernandez. "Diagnostico táctica y estrategia. *Revista Ingeniería Energética del ISPJAE. XX*, pages 7–18, 1999.

H. GmbH. Reliable Data Movement for Connected Devices, 2021. URL https://www.hivemq.com/.

M. M. Hossain, M. R. Islam, A. S. R. Sarkar, M. M. Ali Khan, and S. Taneepanichskul. Prevalence and determinants risk factors of underweight and overweight among women in Bangladesh. *Obesity Medicine*, 11:1–5, sep 2018. ISSN 24518476. doi: 10.1016/j.obmed.2018.05.002.

B. Kamlaesan, K. A. Kumar, and S. A. David. Analysis of transformer faults using IOT. In *2017 IEEE International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials, ICSTM 2017 - Proceedings*, pages 239–241. Institute of Electrical and Electronics Engineers Inc., oct 2017. ISBN 9781509059058. doi: 10.1109/ICSTM.2017.8089159.

B. Kul. IoT-GSM-based high-efficiency LED street light control system (IoT-SLCS). In *2017 26th International Scientific Conference Electronics, ET 2017 - Proceedings*, volume 2017-January, pages 1–5. Institute of Electrical and Electronics Engineers Inc., nov 2017. ISBN 9781538617533. doi: 10.1109/ET.2017.8124361.

S. E. Laboratories. EQUIPO SEL 2414. URL https://selinc.com/es/products/2414/.

R. Liñan. Experiencias en el desarrollo de sistemas de monitoreo y diagnóstico paratransformadores de potencia. *Bienal CIGRÉ-MÉXICO*, pages 12–02, 2001.

S. Mohagheghi, J. Stoupis, and Z. Wang. Communication protocols and networks for power systems-current status and future trends. In *2009 IEEE/PES Power Systems Conference and Exposition*, pages 1–9. IEEE, 2009.

R. Morello, C. De Capua, G. Fulco, and S. C. Mukhopadhyay. A smart power meter to monitor energy flow in smart grids: The role of advanced sensing and iot in the electric grid of the future. *IEEE Sensors Journal*, 17(23): 7828–7837, dec 2017. ISSN 1530437X. doi: 10.1109/JSEN.2017.2760014.

O. MQTT. MQ Telemetry Transport. URL https://mqtt.org.

C. Neumann. The impact of insulation monitoring and diagnostics on reliability and exploitation of service life. *Memorias Congreso de la CIGRE 2006*, 2006.

P. P. K. N. W. W. Normalizacyjnych. *Telecontrol Equipment and Systems - Part 5-104: Transmission Protocols - Network Access for IEC 60870-5-101 Using Standard Transport Profiles (IEC 60870-5-104:2001)*. Polski Komitet Normalizacyjny, 2002. ISBN 9788324311545. URL https://books.google.com.ec/books?id=BavfjgEACAAJ.

D. ORG. No Title. URL https://www.dnp.org/About/Overview-of-DNP3-Protocol.

D. PEREZ. *ESPECIFICACIN DEL PROTOCOLO DNP3*. PhD thesis, UNIVERSIDAD TECNOLOGICA DE LA MIXTECA, 2011.

H. Perez. *"Aplicación de la matriz de falla como herramienta del diagnóstico integral en la detección y localización de fallas en un autotransformador de 125 MVA"*. PhD thesis, CIPEL-CUJAE, Cuba, 2012.

L. S. B. Pinela. Protección Y Control De. 1:246, 2011.

R. Ramakrishnan and L. Gaur. Smart electricity distribution in residential areas: Internet of Things (IoT) based advanced metering infrastructure and cloud analytics. In *2016 International Conference on Internet of*

Things and Applications, IOTA 2016, pages 46–51. Institute of Electrical and Electronics Engineers Inc., sep 2016. ISBN 9781509000449. doi: 10.1109/IOTA.2016.7562693.

F. Ramia Tena. Desarrollo de aplicación para la gestión de dispositivos remotos mediante protocolo mqtt. 2020.

A. Sachan. Microcontroller Based Substation Monitoring and Control System with Gsm Modem. *IOSR Journal of Electrical and Electronics Engineering*, 1(6):13–21, 2012. doi: 10.9790/1676-0161321.

H. R. Salgado, M. E. C. Mendoza, M. J. M. H. Bravo, and J. A. M. Valverde. Desarrollo de prototipo scada para control de nivel de agua en dos contenedores de la comisión de agua potable y alcantarillado del municipio de acapulco (capama).

R. Samuel. Eléctricos Protección de Sistemas Eléctricos. *Universidad Nacional de Colombia Manizales*, Primera Ed:664, 2014. doi: DOI10.1002/masy.201000052.

S. Singh, P. K. Sharma, S. Y. Moon, and J. H. Park. Advanced lightweight encryption algorithms for IoT devices: survey, challenges and solutions. *Journal of Ambient Intelligence and Humanized Computing*, 0(0):1–18, 2017. ISSN 18685145. doi: 10.1007/s12652-017-0494-4.

I. ThingsBoard. ThingsBoard - Open-source IoT Platform. URL https://thingsboard.io/docs/reference/architecture/{#}clustering-mode.

H. Vargas. No Title, 2018. URL https://www.codigoiot.com/base-de-conocimiento/mqtt/.

A. M. Vera Martín et al. Entorno de control de sistemas iiot basado en mqtt implementado en labview. 2019.

J. A. Villalba Márquez. Estudio y pruebas del protocolo de comunicación dnp3. 0 sobre tcp/ip para la comunicación entre la central de generación cumbayá de la empresa eléctrica quito sa y el cenace. B.S. thesis, QUITO/EPN/2010, 2010.

Y. P. Wang, X. Lin, A. Adhikary, A. Grövlen, Y. Sui, Y. Blankenship, J. Bergman, and H. S. Razaghi. A Primer on 3GPP Narrowband Internet of Things. *IEEE Communications Magazine*, 55(3):117–123, 2017. ISSN 01636804. doi: 10.1109/MCOM.2017.1600510CM.