Diseño de un sistema IoT para la monitorización de datos atmosféricos obtenidos por la estación de radioastronomía CASIRI

Juliam Andrés Díaz Barrera y Norbey Camilo Infante Villamil

Trabajo de grado para optar al título de Ingenieros Electrónicos

Director

Juan Manuel Rey López

Doctor en Ingeniería Electrónica

Codirector

Julian Gustavo Rodriguez Ferreira

Doctor en Astronomía y Astrofísica

Codirector

Andrés Felipe Rubio Toloza

Ingeniero Electrónico

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones

Ingenieria Electrónica

Bucaramanga

2023

Dedicatoria

Este proyecto se lo dedico de manera muy especial a mis padres por darme todo el apoyo durante toda la carrera, al director de este proyecto por darme las indicaciones y a la empresa Dautom que nos guió y orientó durante el desarrollo del proyecto.

Juliam Andrés Díaz Barrera

Este proyecto se lo dedico de manera muy especial a mi familia por brindarme apoyo durante mi carrera universitaria, a mis amigos y compañeros que me acompañaron durante cada semestre, al director y codirectores del proyecto y todos los que hicieron posible el proyecto.

Norbey Camilo infante Villamil

Agradecimientos

Quiero agradecer primeramente a Dios por permitirme culminar satisfactoriamente mi carrera profesional, a mis padres por siempre estar apoyándome y a todos los profesores que me enseñaron y me formaron durante mi carrera universitaria.

Juliam Andrés Díaz Barrera

Quiero dar gracias a Dios por haberme guiado en el camino de la educación y darme la fuerza para superar los desafíos que se me presentaron. A mis padres y hermano, les agradezco por su amor incondicional, su constante apoyo y por creer en mí en todo momento. Su ejemplo de dedicación y trabajo duro ha sido una gran fuente de motivación y una inspiración para mí.

Norbey Camilo Infante Villamil

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Resumen

Titulo: Diseño de un sistema IoT para la monitorización de datos atmosféricos obtenidos por la estación de radioastronomía CASIRI. [*]

Autores: Juliam Andrés Díaz Barrera y Norbey Camilo Infante Villamil. [**]

Palabras claves: Ignition, Interfaz, IoT, MQTT, Radioastronomía, Variables Atmosféricas.

Descripción: La estación de Radioastronomía CASIRI está ubicada actualmente en el laboratorio de investigación RadioGis, en el campus principal de la universidad. Está compuesta por diferentes dispositivos de sensado y recolección de datos que pretende ser instalada en lugares con condiciones ambientales adversas, lo cual dificulta la supervisión directa sobre la información obtenida. Por ello, es necesario diseñar un sistema IoT que permita la visualización de forma remota de las variables atmosféricas medidas. Para la transmisión y recepción de datos se deben integrar los sensores a la capa de aplicación del protocolo de comunicación más adecuado y el uso de una interfaz gráfica. Una posible solución implementa el protocolo MQTT utilizando el software Ignition 8.1. Los dispositivos empleados deben cumplir con requerimientos técnicos que permitan ser usados en lugares con condiciones poco favorables para tomar medidas de humedad, temperatura, presión, velocidad y dirección del viento visualizadas en una interfaz HMI.

---

[*] Trabajo de grado

[**] Facultad de Ingenierías Fisicomecánicas, Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director Juan Manuel Rey López. Codirector Julián Gustavo Rodríguez Ferreira. Codirector Andrés Felipe Rubio Toloza.

Abstract

Title: Design of an IoT system for the monitoring of atmospheric data obtained by the CASIRI radio astronomy station. [*]

Authors: Juliam Andrés Díaz Barrera, Norbey Camilo Infante Villamil. [**]

Keywords: Atmospheric Variables, Ignition, Interface, IoT, MQTT, Radioastronomía.

Description: The CASIRI Radio Astronomy station is located in the RadioGis research laboratory, on the UIS main campus. It is made up of different sensors and data storage devices that are intended to be used in places with adverse environmental conditions, making the real-time datasupervision a challenge. Therefore, it is necessary to design an IoT system that allows the remote visualization of the measured atmospheric variables. For the transmission and reception of data, the sensors must be integrated into the application layer of a communication protocol, using a graphical interface. The implementation of MQTT protocol using Ignition 8.1 software is explored in this work. The devices used must meet technical requirements considering the conditions of the expected application in order to measure variables as humidity, temperature, pressure, wind speed and direction, to be displayed on an HMI interface.

---

[*] Bachelor Thesis

[**] Facultad de Ingenierías Fisicomecánicas, Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Director Juan Manuel Rey López. Codirector Julian Gustavo Rodriguez Ferreira. Codirector Andrés Felipe Rubio Toloza.

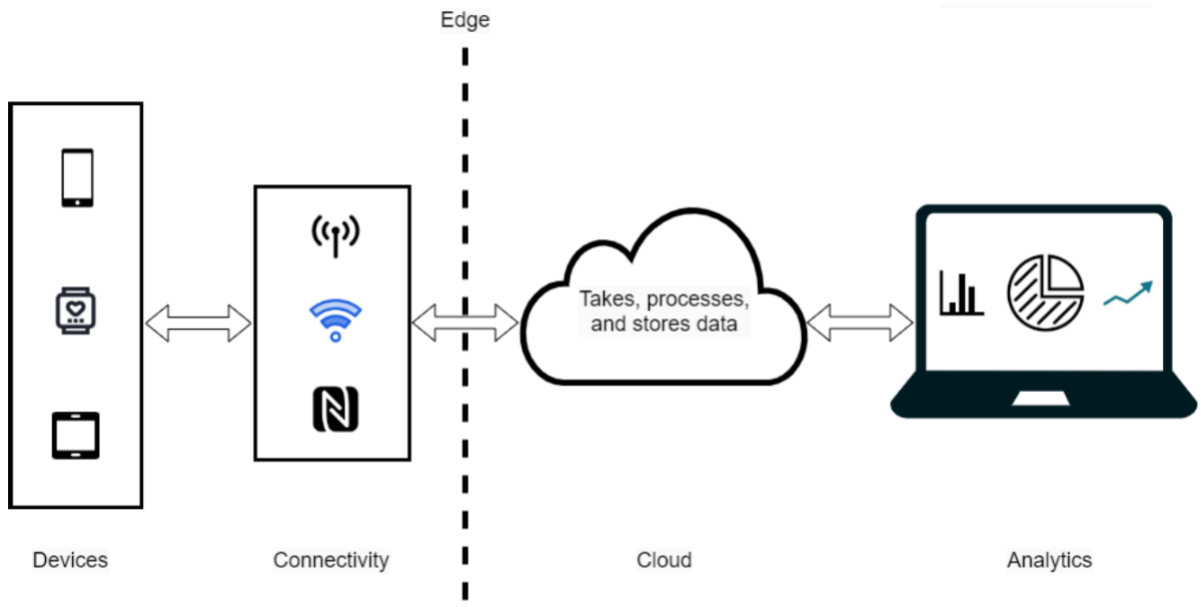
Introducción

La constante evolución de las tecnologías a nivel global ha obligado a que los diferentes usuarios progresen junto al desarrollo de la información digital. La cuarta revolución industrial ha sido la base de la transformación digital representada en nuevos modelos tecnológicos para ser integrados en diferentes entornos de aplicación. En los últimos años, el desarrollo tecnológico es cada vez más notable y exigente, al punto de convertirse en una necesidad para la humanidad, siendo utilizado en los objetos de uso cotidiano entrando a un modelo tecnológico desconocido llamado internet de las cosas, conocido en inglés como Internet of Things (IoT), que día a día nos sorprende siendo la base fundamental de necesidades básicas para un mundo moderno.

El IoT es la conexión a Internet de los objetos que influyen directamente en nuestra cotidianidad, convirtiendo objetos comunes en dispositivos conectados entre sí (Sisinni et al., 2018). La idea principal radica en tener la capacidad de desplegar un sin número de objetos inteligentes para detectar datos e información en algún entorno, procesar y retroalimentar la información en el mismo facilitando colectivamente un mundo inteligente y conectado.

Los IACS (Industrial Automation and Control Systems) son sistemas de control, como ordenadores o robots que manejan diferentes procesos y maquinarias en una industria para sustituir al ser humano. Estos utilizan tecnología operativa (OT), es decir, utilizan el software y el hardware para controlar los equipos industriales empleados en diversos entornos como la fabricación, el transporte, los servicios públicos, entre otros.

Figura 1
Arquitectura de un Sistema IoT



Nota. Imagen extraída de (Manahanm et al., 2020)

Los IACS suelen utilizar un modelo de funcionamiento con base en un sistema IoT como se muestra en la Figura 1 , iniciando con la toma de datos por medio de distintos dispositivos o sensores, luego se envía la información a la nube vía internet para el procesamiento de datos y finalmente son transmitidos a los receptores finales para el análisis de la información.

La cuarta revolución industrial ha sido una de las promotoras de los retos afrontados con IoT como parte de un "mundo inteligente y en red" (Liao et al., 2018) en los diferentes procesos domésticos y ahora industriales, donde encontramos un nuevo concepto derivado del mismo para integrar la electrónica inteligente en los sistemas de producción conocido como el IoT industrial (IIoT) (Manahanm et al., 2020).

La contaminación ambiental ha crecido a raíz de los desarrollos tecnológicos en esta re-

volución industrial. Los datos atmosféricos han empezado a alterarse según la contaminación generada, por lo que es necesario aumentar el control y monitoreo de variables como temperatura, humedad, presión, entre otras (Arízaga Silva, 2013). Para nuestro caso, se desea diseñar un sistema de monitoreo continuo por medio un prototipo de observatorio de radioastronomía en busca de una recolección fiable de datos para monitorear variables ambientales que afecten directamente nuestro entorno.

La estación de Radioastronomía CASIRI es un caso de estudio que busca la determinación de un espacio adecuado para la construcción de un radio-observatorio. Está compuesta por dispositivos de sensado que miden la temperatura, humedad, presión, velocidad y dirección del viento. La estación pretende ser instalada en lugares con condiciones ambientales adversas, lo cual dificulta la supervisión directa sobre la información obtenida. Por ello, es necesario diseñar un sistema IoT que permita la visualización de los datos atmosféricos de manera remota.

1. Definiciones y conceptos básicos

La finalidad de esta sección es exponer los conceptos, elementos y tecnologías que hacen parte de este proyecto con el propósito de dar contexto al lector y brindar una información mas clara del diseño planteado.

1.1. MQTT

Protocolo de mensajería basado en estándares, o un conjunto de reglas, que se utiliza para la comunicación de un equipo a otro. Los sensores inteligentes, los dispositivos portátiles y otros dispositivos de Internet de las cosas (IoT) generalmente tienen que transmitir y recibir datos a través de una red con recursos restringidos y un ancho de banda limitado. Estos dispositivos IoT

utilizan MQTT para la transmisión de datos, ya que resulta fácil de implementar y puede comunicar datos de manera eficiente. MQTT admite la mensajería entre dispositivos a la nube y la nube al dispositivo. (AWS Amazon, 2022)

1.2. Servidor

Es un software que almacena, distribuye y suministra información. Los servidores funcionan basándose en el modelo "cliente-servidor". El cliente puede ser tanto un ordenador como una aplicación que requiere información del servidor para funcionar. Por tanto, un servidor ofrecerá la información demandada por el cliente siempre y cuando el cliente esté autorizado. Los servidores pueden ser físicos o virtuales.(TicPortal, 2022)

1.3. Interfaz HMI (Human-Machine Interface)

Es una herramienta que interconecta el proceso y los trabajadores de una línea de manufactura, empresa o industria donde se requiera de una operación por parte de una persona. Su función es la traducir las diferentes variables del proceso industrial en información entendible y útil para el operario optimizando el proceso a través de la digitalización y centralización de datos.(TLT-IOT, 2020)

1.4. Mosquitto

Es un intermediario (Broker) de mensajes de código abierto que implementa el protocolo MQTT. Mosquitto es liviano y es adecuado para su uso en todos los dispositivos. El protocolo MQTT proporciona un método ligero para enviar mensajes utilizando un modelo de publicación/suscripción. Esto lo hace adecuado para la mensajería IoT por medio del software Mosquitto. (Mosquitto, 2022)

1.5. Cliente o Suscriptor

Es cualquier dispositivo que ejecuta una biblioteca MQTT. Si el cliente envía mensajes, actúa como editor, y si recibe mensajes, actúa como receptor. Básicamente, cualquier dispositivo que se comunique mediante MQTT a través de una red puede denominarse dispositivo cliente MQTT. (AWS Amazon, 2022)

1.6. SQLite

Es una biblioteca en lenguaje C que implementa un motor de base de datos SQL pequeño, rápido, autónomo, de alta confiabilidad y con todas las funciones. SQLite es el motor de base de datos más utilizado en el mundo. SQLite está integrado en todos los teléfonos móviles y en la mayoría de las computadoras y viene incluido dentro de innumerables aplicaciones que la gente usa todos los días. (SQLite.org, 2022)

1.7. Ignition

Es un software distribuido por la empresa Inductive Automation el cual puede implementarse como HMI (Human Machine Interface), SCADA (Supervisory Control and Data Acquisition) o MES (Manufactuting Execution System), independientemente de esto, Ignition es un software que permite conectar, integrar y escalar hardware en sistemas complejos de monitoreo, control y visualización de datos. (Inductive Automation, 2022c)

1.8. Tag

Son puntos de datos y pueden tener valores estáticos o valores dinámicos que provienen de una dirección OPC, una expresión o una consulta SQL. Los valores se pueden utilizar en pantallas

y en grupos de transacciones. Los Tags brindan un modelo de datos consistente en todo Ignition y ofrecen la manera más fácil de comenzar y ejecutar la creación de sistemas de control y estado en tiempo real. (Inductive Automation, 2022a)

1.9. Python

Es un lenguaje de programación de alto nivel interpretado, orientado a objetos y con semántica dinámica. Sus estructuras de datos integradas de alto nivel, combinadas con la escritura dinámica y el enlace dinámico, lo hacen muy atractivo para el desarrollo rápido de aplicaciones, así como para su uso como lenguaje de secuencias de comandos o pegamento para conectar componentes existentes entre sí. (Python, 2022)

1.10. IoT

El término IoT, o Internet de las cosas, se refiere a la red colectiva de dispositivos conectados y a la tecnología que facilita la comunicación entre los dispositivos y la nube, así como entre los propios dispositivos. El IoT integra las "cosas" de uso diario con Internet además incluye procesamiento de datos y análisis al mundo de los objetos físicos.(Sisinni et al., 2018)

1.11. Gateway

Es un dispositivo de hardware o software que se encuentra entre diferentes redes o aplicaciones. La puerta de enlace o Gateway convierte información, datos u otras comunicaciones de un protocolo o formato a otro. Una puerta de enlace a Internet puede transferir comunicaciones entre una red empresarial e Internet y actuar como un convertidor de protocolo para que los usuarios puedan enviar y recibir comunicaciones a través de Internet.(Gartner Glossary, 2022)

1.12. OpenVPN

Es un proyecto de red privada virtual (VPN) de código abierto. Crea conexiones seguras a través de Internet utilizando un protocolo de seguridad personalizado que utiliza SSL/TLS. Este proyecto OSS (software de código abierto) respaldado por la comunidad, cuenta con el respaldo de muchos desarrolladores y colaboradores de OpenVPN Inc., así como de la comunidad extendida de OpenVPN. (OPENVPN, 2022)

1.13. CASIRI

Estación de Radioastronomía ubicada en el laboratorio de investigación RadioGis en la Universidad Industrial de Santander. Consta de 3 subsistemas, una estación meteorológica que toma datos atmosféricos, una cámara que toma fotografías a cielo abierto y un USRP (Universal Software Radio Peripheral) para adquisición de señales de radio entre un ancho de banda determinados.

1.14. Consola DAVIS Vantage Pro2

Una estación meteorológica de calidad industrial diseñada para soportar los entornos más duros y proporcionar datos con precisión científica, año tras año. La Vantage Pro2 ofrece al observador meteorológico profesional o al entusiasta de la meteorología un rendimiento sólido con una amplia gama de opciones y sensores.(DAVIS, 2022)

2. Planteamiento del problema

El estudio de la radioastronomía exige la construcción de sistemas que integren radio-observatorios para investigar y explorar el universo por medio de señales interestelares de radio (Tantitharanukul et al., 2017). Sin embargo, estos sistemas requieren un lugar específico de instalación donde se cuente con cielos silenciosos, los cuales son lugares donde las señales de radio provenientes de la actividad humana son muy bajas o inexistentes, junto con cielos despejados y baja humedad (Wilson et al., 2013). Para localizar estos lugares se utilizan dispositivos que miden las señales de radio circundantes, además de un sistema que incluye estación meteorológica y cámara para monitorear el clima y densidad de las nubes.

Una tarea relevante es la caracterización del lugar donde se plantea la construcción de los radio-observatorios. Se debe proteger la veracidad de los datos obtenidos y evitar al máximo las interferencias producidas por entes externos a las señales sensadas. Es por eso que se buscan lugares aislados y apartados que permitan tomar los datos con la menor distorsión posible(Censier et al., 2021).

Sin embargo, la toma de datos de forma directa genera problemas en el personal, dificultando la permanencia del usuario en el lugar de instalación y limitando la expansión de los datos a los clientes interesados. Por ello se requiere desarrollar un sistema de transmisión que permita visualizar la información de forma remota y en tiempo real para ser compartida con los usuarios que la requieran.

El rendimiento del sistema de transmisión depende en un alto porcentaje de la elección del protocolo de comunicación a utilizar (Naik et al., 2016). Existen diferentes protocolos según su aplicación y requerimientos que el sistema solicita. Se tienen aquellos que necesitan mensajería instantánea como XMPP y SIP. AMQP y JMS son usados por aplicaciones que requieren transacciones comerciales rápidas y fiables (Meng et al., 2017). Otros fueron creados para recopilar datos en redes restringidas dando una solución sencilla y ligera como MQTT y CoAP (Naik, 2017).

La estación de radioastronomía CASIRI es un dispositivo apropiado para instalar en lugares aislados gracias a su facilidad de desplazamiento. Permite la caracterización de los sitios candidatos a radio-observatorios donde las condiciones no son favorables para la supervisión de su funcionamiento. El monitoreo de la toma de datos es un tema difícil a la hora de hablar de CASIRI debido a que las condiciones ambientales no son aptas para que un usuario esté recolectando la información en el lugar de instalación. Es necesario diseñar un enlace de comunicación entre el dispositivo y los usuarios interesados para procesar la información en alguna estación central.

3. Objetivos

3.1. Objetivo General

Diseñar un sistema IoT que permita la visualización de variables atmosféricas de forma remota obtenidas por la estación de radioastronomía CASIRI por medio de una interfaz gráfica HMI.

3.2. Objetivos Específicos

- Integrar los sensores usados por CASIRI a un sistema IoT por medio de un protocolo de comunicación ajustando los requerimientos a las necesidades de la aplicación.
- Diseñar una interfaz gráfica HMI para visualizar de forma remota las variables tomadas de CASIRI incorporando el sistema a una plataforma de automatización industrial.
- Verificar la correcta integración de la interfaz gráfica HMI con los dispositivos usados por CASIRI a partir de una serie de pruebas de funcionamiento en el laboratorio del grupo de investigación Radiogis, dentro del campus universitario, con el fin de realizar un primer ensayo y extender el sistema a otros lugares de interés.
- Redactar un manual de software y de integración para la estación presentando los esquemas de conexión y enlace para los dispositivos a utilizar.

4. Requerimientos de CASIRI

4.1. Identificación de los interesados

CASIRI es un sistema que puede ser pretendido por la Academia en general, los diferentes estudios e indagaciones acerca de los radio-observatorios pueden usar como base el sistema que se planea diseñar. A continuación se nombra los interesados específicos para el desarrollo del mismo.

- La Agencia Nacional del Espectro (ANE) brinda atención al Ministerio de Tecnologías de la Información y Comunicaciones en la planeación, vigilancia y control del espectro radioeléctrico en el país, la cual constantemente realiza estudios por medio de radio-observatorios.
- El Ministerio de Comunicaciones (MINTIC) y el Ministerio de Ciencia Tecnología e Innovación (Minciencias) son agentes interesados en la construcción del conocimiento e investigación de los fenómenos astronómicos.
- La comunidad de astrónomos de Colombia (AstroCO), nodo asociado a la Academia Colombiana de Ciencias Exactas, Físicas y Naturales (ACCEFYN) une en su actividad científica a los astrónomos, astrofísicos, cosmólogos y egresados de áreas afines para generar investigaciones por medio de los estudios y análisis en radio-observatorios.
- La Universidad de Antióquia (UdeA), la Universidad ECCI (Escuela Colombiana de Carreras Industriales) y la Universidad Católica de la Santísima Concepción (UCSC) de Chile son algunas universidades a nivel nacional e internacional interesados en el caso de estudio CASIRI.

- Consorcios internacionales como Myst conformado por Estados Unidos, Canadá y Australia estan interesados en el estudio de la radioastronomía y su desarrollo correspondiente.

4.2. Necesidades de los interesados

La toma de datos atmosféricos de forma remota es muy importante para la comunidad científica y personas en general, con esta información se pueden realizar investigaciones, estudios o simplemente ser usada para la divulgación del estado del clima. CASIRI, al ser un sistema con fines científicos, debe contar con una variedad de sensores y dispositivos precisos para la toma de datos, una cámara para la toma de imágenes del cielo y un instrumento de medición de ondas electromagnéticas teniendo en cuenta que es una estación de Radioastronomía.

Por esta razón, se ha propuesto un sistema de monitoreo de variables ambientales que permita medir variables como temperatura, humedad, presión, velocidad y dirección del viento, para realizar un monitoreo continuo en búsqueda de una recolección fiable de datos y ser usados en estudios posteriores, que busquen mejorar la calidad del medio ambiente y el entorno en general.

La contaminación electromagnética es un factor que ha tomado fuerza en los últimos tiempos, la cual puede entenderse como las emisiones electromagnéticas generadas por uno o varios focos, de una misma o de distintas frecuencias. Las ondas de televisión, radio, telefonía o las líneas eléctricas emiten campos electromagnéticos en distintas frecuencias y sus efectos se suman creando así puntos de riesgo que afectan los datos obtenidos por la estación, es decir, las emisiones de radiofrecuencia de origen humano interfieren directamente con la medición de las señales del espacio tomadas por la Radioastronomía.

En la actualidad, se ha hecho común el estudio de RFI (Interferencia de Radiofrecuencia) y la caracterización de la misma es una servicio que puede brindar CASIRI, el cual se enfoca en bajas frecuencias (100 - 200 MHz para Radioastronomía) utilizando USRP (Universal Software Radio Peripheral) para medir el espectro radioeléctrico y evitar que la interferencia sea lo suficientemente grave que genere un mal funcionamiento en los dispositivos electrónicos.

Adicionalmente, CASIRI es un sistema movible e inalámbrico, el cual puede ser ubicado en cualquier lugar donde se requiera algún estudio de RFI y caracterizar las variables necesarias a pesar de las condiciones climáticas. Gracias al protocolo de comunicación que se planea implementar, el sistema puede ser desplazado sin dificultad y emitir la información solicitada sin necesidad de acompañamiento humano continuo. Además, se proyecta condensar todo el sistema en una estructura mecánica sencilla que permita una movilidad fácil y accesible al personal encargado.

4.3. Recursos necesarios para la realización del proyecto

El estudio de la radioastronomía exige la construcción de radio-observatorios para investigar y explorar el universo por medio de señales interestelares de radio. Sin embargo, estos sistemas requieren de lugares específicos de instalación donde se cuente con cielos silenciosos, los cuales se encuentran en zonas alejadas y remotas en donde las señales de radio provenientes de la actividad humana sean muy bajas o inexistentes. Para la implementación de CASIRI se deben tener en cuenta los siguientes aspectos:

4.3.1. Dispositivos que Soportan Condiciones Ambientales Extremas. Al ser un sistema que se va a ubicar a la intemperie, CASIRI requiere dispositivos capaces de soportar cambios de temperatura y humedad, por esta razón, es fundamental una buena selección de los elementos y sensores.

4.3.2. Sistema de Cómputo Central. Los distintos dispositivos para la toma de datos que tiene CASIRI deben estar enlazados con un ordenador central que permita el tratamiento y almacenamiento de los datos que registran los sensores para su posterior transmisión a través de internet por medio del protocolo seleccionado.

4.3.3. Fuente de Energía. En los lugares disponibles para la implementación de CASIRI no se cuentan con suministro de energía por medio de la red eléctrica, por esto es importante tener una fuente de energía confiable y portátil para garantizar la toma de muestras de manera continua y eficiente. Una solución a esta problemática es contar con un banco de baterías para suministrar energía a todos los dispositivos.

4.3.4. Interfaz para la Muestra de Datos. La forma en que se visualizan los datos es muy importante para los usuarios que quieran acceder a la información recopilada, por esta razón se requiere que la interfaz gráfica sea sencilla de entender y amigable visualmente con los usuarios.

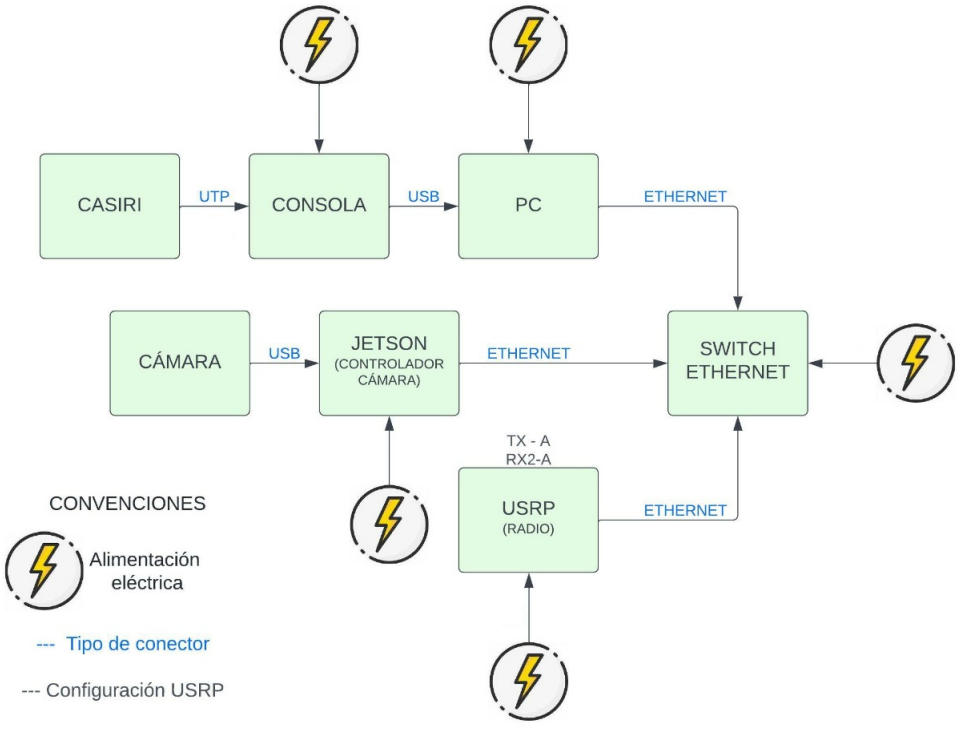
4.3.5. Personal de Montaje y Mantenimiento. Para hacer el montaje en campo y mantenimiento es necesario contar con personal que tenga los conocimientos previos de conexión de los dispositivos del sistema, así como conocimiento en el manejo de protocolos de comunicación

para la toma y transmisión de datos, reduciendo las posibilidades de errores operativos. Los datos no pueden ser transmitidos por internet en cercanía de la estación de Radioastronomía, ya que estas señales influyen directamente en la obtención de señales del USRP. Por eso es necesario contar con una subestación alejada para enviar los datos de manera segura y sin interferencias.

En el diagrama de la Figura 2 se muestra el esquema de conexión del sistema con los diferentes dispositivos de recolección de información y el tipo de cable utilizado para que los datos lleguen al servidor principal. Todos los dispositivos se unen por medio de un switch ethernet para luego enviar la información. De igual forma se observa que se necesita una alimentación constante y estable de energía debido a la cantidad de elementos que requieren de ella.

Figura 2

Esquema de Conexión CASIRI.



4.3.6. Descripción del escenario de implementación . CASIRI está diseñado para ser un sistema versátil, que por su portabilidad puede ser llevado a cualquier lugar donde se requiera hacer reconocimiento de sitios (datos atmosféricos y de interferencia por radiofrecuencia RFI). Actualmente se han hecho mediciones en el páramo de Berlín, en lugares apartados como se muestra en la Figura 3. Para el estudio de la radioastronomía se requieren “cielos silenciosos”, lugares donde hay poca actividad humana en el rango de 100 - 200 MHz de frecuencia, allí se alcanzan a ver algunos elementos del sistema y la practicidad para efectos de la movilidad e instalación del mismo.

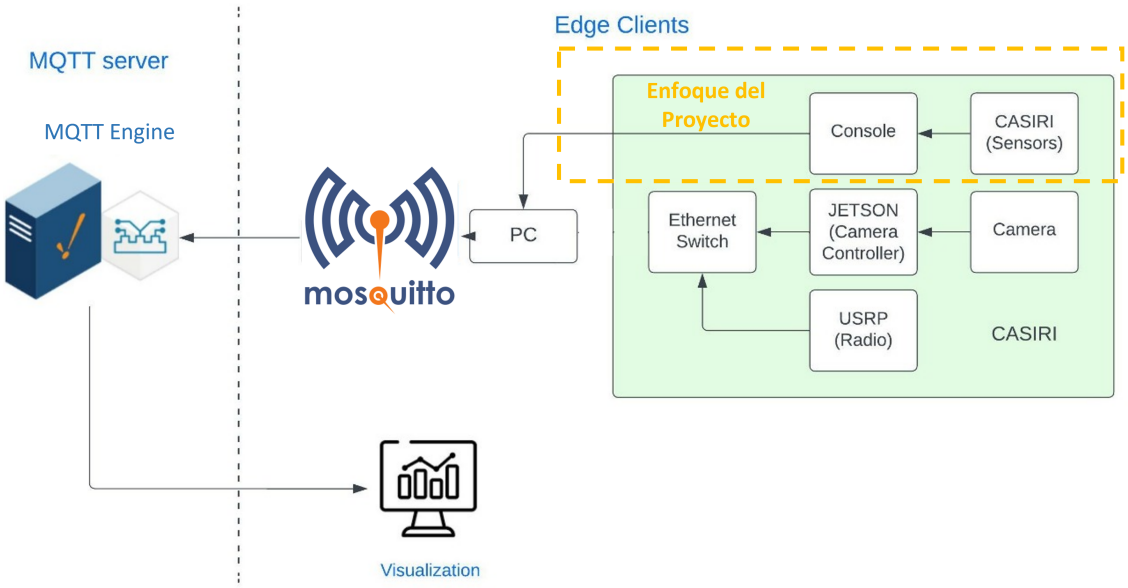
Figura 3

Estación de Radioastronomía en el Páramo de Santurbán

El sistema se compone de las conexiones a nivel de hardware mostrado en la Figura 2 y del protocolo de comunicación que se planea implementar como se muestra en la Figura 4 . La conexión de estas dos partes del sistema genera el enlace desde la recolección de la información hasta la interfaz mostrada a los “clientes” suscritos al servidor.

Figura 4

Diagrama de conexión del protocolo de comunicación con CASIRI



Es de aclarar que la estación de radioastronomía CASIRI se comprende de 3 subsistemas como se observa en el cuadro verde de la Figura 4, sin embargo el enfoque de este proyecto apunta a la estación meteorológica, la cual toma datos atmosféricos para ser monitoreados a través de un servidor remoto.

5. Integración del Sistema IoT

La integración del sistema IoT consiste en la interconexión de los sensores usados por CASIRI a una estación central de monitoreo por medio de una protocolo de comunicación que se ajuste a las limitaciones y requerimientos del sistema en cuestión. Por ello, es de gran importancia la correcta elección del protocolo a implementar. En la tabla 1 se encuentra la información clara y especifica para decidir cuál protocolo se acomoda mas a las necesidades del sistema IoT que se planea diseñar. Se debe tener en cuenta principalmente el trafico de datos, la arquitectura que maneja, el protocolo de transporte y los puertos predeterminados. Todos estos deben ser compatibles con el servidor que se planea implementar y evitar errores de compatibilidad.

Tabla 1
Comparación de Protocolos de Comunicación

CRITERIO	MQTT	CoAP	AMQP	HTTP
1. Año	1999	2010	2003	1997
2. Arquitectura	Cliente/Broker	Cliente/Servidor o Cliente/Broker	Cliente/Servidor or Cliente/Broker	Cliente/Servidor
3. Abstracción	Publicador/Suscripto	Solicitud/Respuesta o Publicador/Suscriptor	Publicador/Suscripto Solicitud/Respuesta	Solicitud/Respuesta
4. Capacidad del Encabezamiento	2 Byte	4 Byte	8 Byte	Indefinido
5. Capacidad del Mensaje	Pequeño e Indefinido (capacidad máxima de 256 MB)	Pequeño e Indefinido (normalmente pequeño para encajar en una sola datagrama IP)	Negociable e Indefinido	Grande e Indefinido (depende del servidor web o la tecnología de programación)

CRITERIO	MQTT	CoAP	AMQP	HTTP
6.Estándares	OASIS, Eclipe Foundations	IETF, Eclipse Foundations	OASIS, ISO/IEC	IEFT and W3C
7. Semánticas/-Métodos	Çonnect, Disconnect, Publish, Subscribe, Unsubscribe, Close"	"Get, Post, Put, Delete"	Çonsume, Deliver, Publish, Get, Select, Ack, Delete, Nack, Recover, Reject, Open, Close"	"Get, Post, Head, Put, Patch, Options, Connect, Delete"
8. Compatibilidad con Cache y Proxy	Parcial	Si	Si	Si
9. Protocolo de Transporte	TCP (MQTT-SN puede usar UDP)	UDP, SCTP	TCP, SCTP	TCP
10. Puerto por Defecto	1883/8883 (TLS/SSL)	5683 (Puerto UDP)/5684 (DLTS)	5671 (TLS/SSL), 5672	80/ 443 (TLS/SSL)
11. Seguridad	TLS/SSL	DTLS, IPSec	TLS/SSL, IPSec, SASL	TLS/SSL
12. Formato de Codificación	Binario	Binario	Binario	Texto
13. Modelo de Licencia	Código Abierto	Código Abierto	Código Abierto	Gratis
14. Apoyo Organizacional	IBM, Facebook, Eurotech, Cisco, Red Hat, Software AG, Tibco, ITSO, M2Mi, Amazon Web Services (AWS)	Large Web Community Support, Cisco, Contiki, Erika, IoTivity	Microsoft , JP Morgan, Bank of America, Barclays, Goldman Sachs, Credit Suisse	Global Web Protocol Standard

*Nota.*Tabla obtenida de (Naik, 2017)

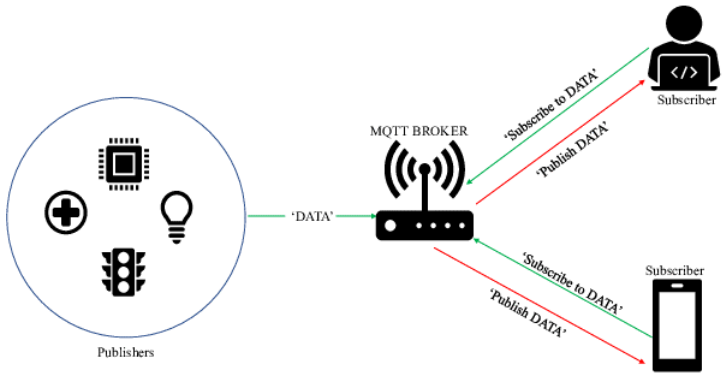
5.1. Protocolo MQTT

El protocolo MQTT es uno de los protocolos más populares para compartir información debido a su ligereza, especialmente para sistemas que necesitan compartir grandes cantidades de datos en tiempo real (Tantitharanukul et al., 2017). Como se observa en la tabla 1, MQTT es un sistema de publicación-suscripción que es ligero, abierto y fácil de implementar, tanto por los editores como por los suscriptores. Estas características lo hacen adecuado para su uso en diferentes situaciones y ambientes, este último de preferencia en entornos limitados, como comunicaciones Máquina a Máquina (M2M) o IoT Bryce et al. (2018).

En el protocolo MQTT existen dos elementos en la información que se envía, estos son el tema y el mensaje. Los datos se emiten desde un editor hacia los clientes que se hayan suscrito al tema compartido por el publicador, y el elemento mensaje es la información que el publicador desea compartir.Para enviar cualquier dato abierto a través del protocolo MQTT, los suscriptores o clientes necesitan saber cuáles son los temas de datos a los cuales deben suscribirse desde el broker (Tantitharanukul et al., 2017).

Figura 5

Comunicación de datos con el protocolo MQTT



En la Figura 5 se ilustra cómo es la comunicación del publicador con sus clientes a través de un servidor que funcionará como el broker MQTT, el cual enlaza la información recibida por el publicador y la envía a sus clientes. La comunicación es bidireccional, es decir, es posible que los suscriptores envíen información a los publicadores y viceversa. Dentro de los objetivos específicos que busca el protocolo MQTT se incluyen (Bryce et al., 2018):

- Simplicidad de implementación
- Entrega de datos con calidad de servicio
- Eficiencia en el ancho de banda
- Independencia de los datos
- Conocimiento continuo de la sesión

5.2. Servidor MQTT

Existen diferentes servidores MQTT disponibles en el mercado. Todos ellos soportan la versión 3.1.1 del protocolo MQTT como se muestra en la Tabla 2, según su lenguaje de programación y versión correspondiente. Mosquitto es uno de los más populares entre la comunidad de investigadores, debido a que es un servidor construido en lenguaje de programación C, ligero y con una interfaz sencilla que permite crear redes MQTT de manera rápida (Bender et al., 2021).

Tabla 2
Consolidado de los servidores MQTT disponibles en el mercado

Nombre	Lenguaje	Versión
RabbitMQ	Erlang	3.1.1
EMQ X Broker	Erlang	3.1, 3.1.1, 5
ejabberd	Erlang	3.1, 3.1.1, 5
Mosquitto	C/C++	3.1, 3.1.1, 5
VerneMQ	Erlang	3.1, 3.1.1, 5
MQTTnet	C	3.1, 3.1.1, 5
ActiveMQ	Java	3.1
HiveMQ CE	Java	3.1, 3.1.1, 5
MQTT.js	Javascript	3.1, 3.1.1
Paho MQTT	Various	3.1, 3.1.1
Aedes	Javascript	3.1, 3.1.1
esp-mqtt	C	3.1

*Nota.*Tabla obtenida de (Bender et al., 2021)

Cabe destacar que, basándonos en los estudios y hallazgos obtenidos en la evaluación de servidores MQTT de código abierto (Bender et al., 2021)., se escogió el servidor Mosquitto para el desarrollo de este proyecto debido a su fácil implementación, su arquitectura robusta en cuanto a la conexión con clientes y su eficiencia en el consumo de recursos de computación.

5.3. Conexión del Servidor MQTT-Ignition

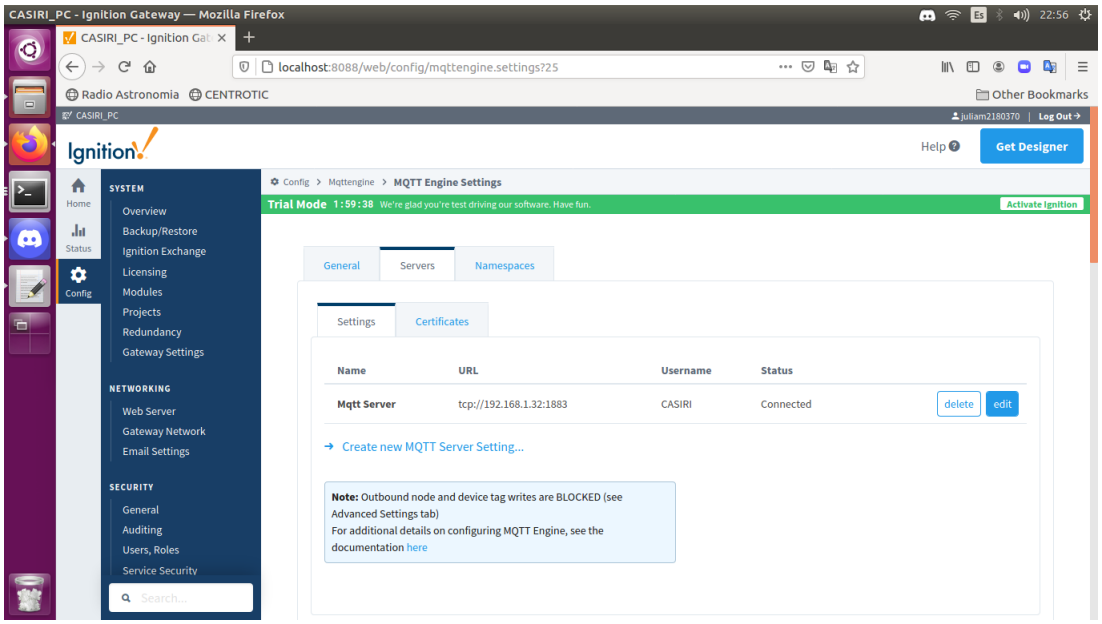
Para dar inicio de manera pertinente a este apartado, es de suma importancia resaltar que Ignition es una plataforma de automatización industrial que permite conectar dispositivos y sistemas a través de diferentes protocolos de comunicación. También, destacar que MQTT Engine es un módulo de Ignition que permite conectar dispositivos y sistemas que utilizan el protocolo MQTT (Message Queuing Telemetry Transport) con el sistema de automatización de Ignition. El

módulo proporciona una interfaz para configurar y gestionar conexiones MQTT y permite publicar y suscribirse a mensajes MQTT (Inductive Automation, 2022b).

Para conectar un servidor Mosquitto con Ignition a través del módulo “MQTT Engine”, primero se debe configurar una conexión MQTT en el módulo. Esto incluye especificar la URL o la dirección IP del servidor, así como el nombre de usuario y la contraseña si se requiere dentro de los detalles del servidor. La Figura 6 muestra cómo se configura la dirección IP del servidor, en este caso, es la IP del computador principal (192.168.1.32) seguido del puerto utilizado para la conexión con Ignition (8088).

Figura 6

Configuración del Módulo MQTT Engine en Ignition



Luego, se puede utilizar la funcionalidad de publicación y suscripción del módulo para enviar y recibir mensajes a través de la conexión MQTT. Una vez configurada la conexión, se utilizan las funcionalidades de publicación y suscripción del módulo MQTT Engine para enviar y recibir

datos entre el sistema de automatización de Ignition y los dispositivos o sistemas conectados al servidor Mosquitto. Esto permite una comunicación bidireccional entre los sistemas y dispositivos, lo que facilita la monitorización y control de procesos industriales.

Para este proyecto, se llevó a cabo la conexión entre un grupo de sensores y la plataforma de automatización industrial Ignition, esto con el fin de recopilar y presentar la información recolectada por los sensores en una interfaz gráfica donde la información tomada es fácil de entender y analizar. Utilizando el módulo, se estableció una conexión segura y confiable con los sensores, permitiendo la recopilación y visualización de los datos en tiempo real mientras se genera un historial de variables por medio de una base de datos.

5.4. Implementación

Para la implementación conjunta del sistema, la estación de radioastronomía CASIRI, el servidor MQTT de Mosquitto y el módulo MQTT Engine de Ignition se enlazaron por medio de un código de Python, donde se toman muestras de manera periódica y una vez obtenidas, se publican en el servidor Mosquitto para ser leídas en Ignition y ser mostradas en la interfaz gráfica.

El código de la Figura 7 está estructurado de la siguiente manera:

- Importación de librerías necesarias para obtener los datos de CASIRI y publicación en el servidor MQTT Mosquitto.
- Declaración de un objeto llamado “device” que corresponde a la consola de los sensores de CASIRI.
- Bucle principal donde se toman muestras de manera periódica.

- Obtención de datos para guardan en el diccionario “data”.
- Cambio de unidades para una mejor representación en la interfaz.
- Creación de un diccionario para después añadir los datos a un archivo “.json” para facilitar la publicación en el servidor MQTT de Mosquitto.

Figura 7

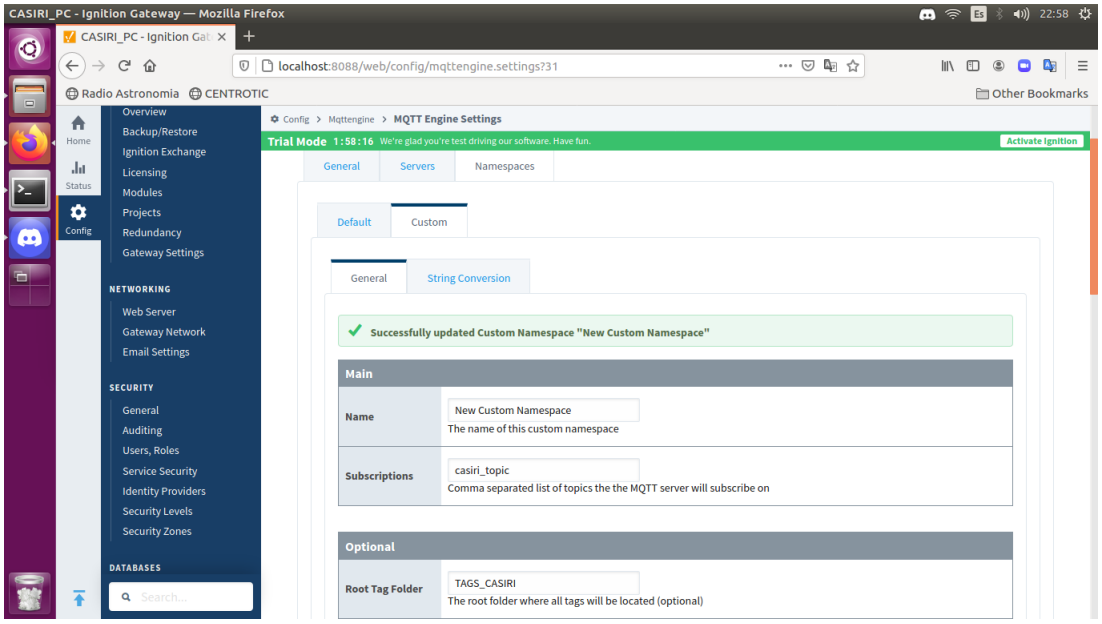
Código para interconectar CASIRI con el servidor MQTT Mosquitto

```
from pyvantagepro import VantagePro2
import json
import time
import paho.mqtt.client as mqtt

device = VantagePro2.from_url('serial:/dev/ttyUSB0:19200:8N1')


while True:

        data = device.get_current_data()
        dateNow=str(data['Datetime'])
        fecha=dateNow[:10]
        hora=dateNow[11:19]

        #Cambio de Unidades
        tempInC="{:.2f}".format((data['TempIn']-32)/1.8)
        tempOutC="{:.2f}".format((data['TempOut']-32)/1.8)
        Velms="{:.2f}".format(data['WindSpeed']*0.44704)
        BarmmHg="{:.2f}".format(data['Barometer']*25.4)

        datos ={'Date':fecha,'Hour':hora,'Temp_Out':tempOutC,'Temp_In':tempInC,'Wind_Speed':Velms,'Win_Dir':data['WindDir'],'Hum_In':data
['HumIn'],'Hum_Out':data['HumOut'],'Barometer':BarmmHg}

        with open("datos_casiri.json","w") as file:
                json.dump(datos,file)


        with open("datos_casiri.json") as json_file:
                data_mqtt = json.load(json_file)

        client = mqtt.Client()
        client.connect("10.14.44.177",1883)

        client.publish("casiri_topic",json.dumps(data_mqtt))

        client.disconnect()

        time.sleep(1)
```

Una vez ejecutado el código en una terminal, se recolectarán periódicamente los datos de los sensores. Estos datos serán automáticamente publicados en el servidor MQTT de Mosquito y el módulo MQTT Engine, al estar suscrito al tema “casiri_topic” mostrado en la Figura 8, reflejará automáticamente la actualización de los datos para su uso posterior como “tags” en la interfaz gráfica dentro de la carpeta TAGS_CASIRI.

Figura 8

Suscripción al tema del Broker MQTT

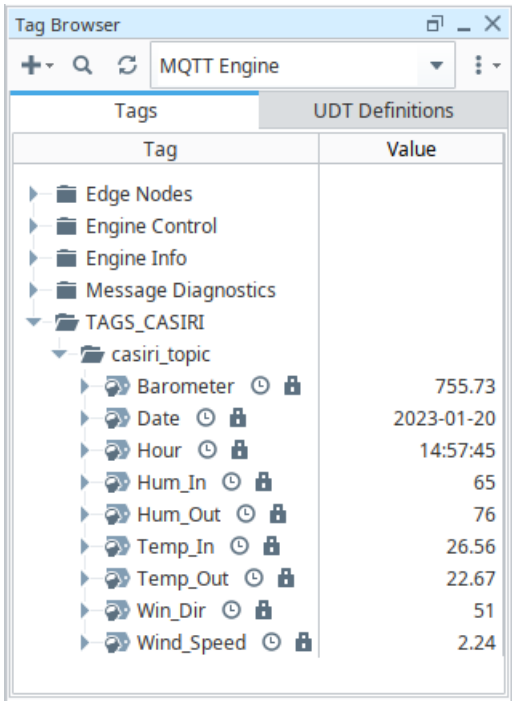


Es de suma importancia aclarar que el módulo “MQTT Engine” no está instalado en el “Gateway” de forma predeterminada o en las configuraciones de fábrica, por ello se debe buscar en la página oficial de Inductive Automation el módulo para descargar y luego instalarlo en la puerta de enlace.

Finalmente, se abre la aplicación de diseñador para recibir los tags enviados por el servidor de Mosquitto, y en su apartado, se pueden encontrar cada una de las variables atmosféricas sensadas por CASIRI con su correspondiente valor, estos proporcionados por los sensores de la estación como se observa en la Figura 9.

Figura 9

Recepción de Tags en el diseñador de Ignition



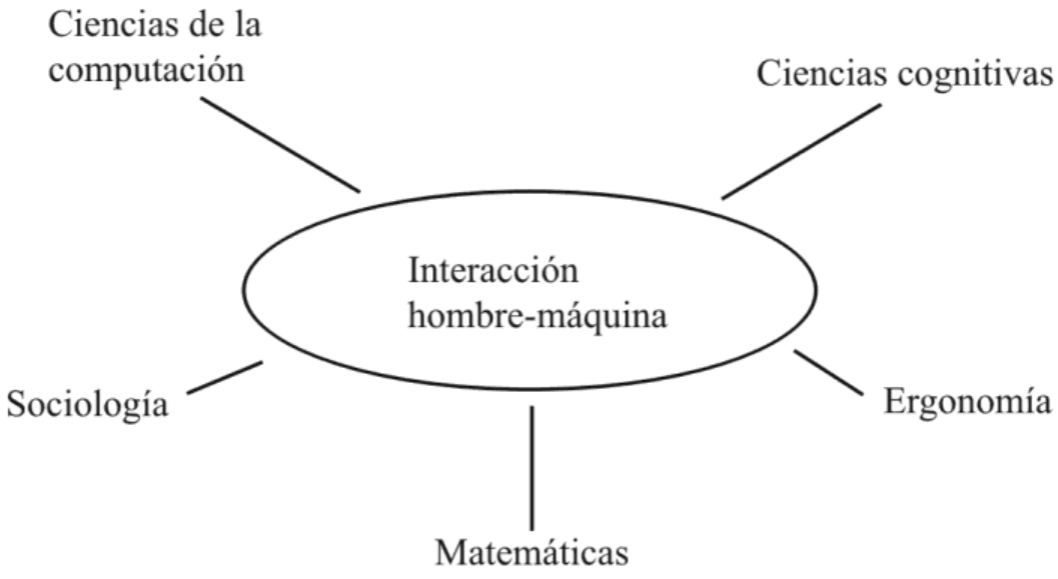
Así, se integraron los sensores usados por CASIRI a un sistema IoT que enlaza software con hardware por medio de un protocolo de comunicación MQTT ajustado a las necesidades de la aplicación y a sus condiciones de funcionamiento.

6. Diseño de la Interfaz Gráfica HMI

Hoy en día, las computadoras son utilizadas por las personas para todo tipo de aplicaciones y objetivos de la vida cotidiana y empresarial. Los sistemas informáticos pueden llegar a tener poca intervención humana o incluso nula, únicamente para control y supervisión dando accesibilidad a todo tipo de personas y haciendo más eficientes los procesos a realizar (Carlos et al., 2019). Teniendo en cuenta esta realidad, es necesario avanzar en el desarrollo tecnológico y en la relación de uso entre humano y máquina. Por ello se empiezan a construir diferentes herramientas para agilizar este proceso y encontrar soluciones que se adapten a las necesidades de las personas en su entorno.

Figura 10

Disciplinas que se utilizan para diseñar una interfaz HMI



Nota. Imagen extraída de (Quezada Quezada et al., 2014)

La Interfaz Gráfica HMI es una herramienta que interconecta la persona y la máquina de manera confiable e intuitiva para el uso cotidiano por medio de diferentes disciplinas del conocimiento, como se muestra en la Figura 10. Lo que permite la visualización de información, almacenamiento de datos e incluso manipulación de variables para realizar una conexión de lectura y escritura con la máquina en cuestión.

El desarrollo de una interfaz HMI se basa en los requerimientos del sistema, es decir, debe cumplir con el objetivo para el cual fue diseñada, así como las restricciones o condiciones de diseño con el fin de que el usuario se sienta satisfecho con la información recibida en la misma.

En su mayoría, las interfaces HMI ejecutadas en un computador se utilizan para representar de forma idéntica la realidad de los procesos, permitiendo a los operadores una interrelación de los equipos físicos de la planta con los equipos virtuales de las interfaces gráfica de usuario (Quezada Quezada et al., 2014). Además, como su nombre lo indica, la herramienta permite observar de manera gráfica la información para que sea más clara y entendible, poder analizar un historial de acciones y una proyección de lo que se espera en futuros eventos.

En la Figura 11, se puede observar una Interfaz HMI de un PLC (Programmable Logic Controller) donde se observa la información de manera gráfica durante un intervalo de tiempo, además de las diferentes configuraciones que tiene la máquina.

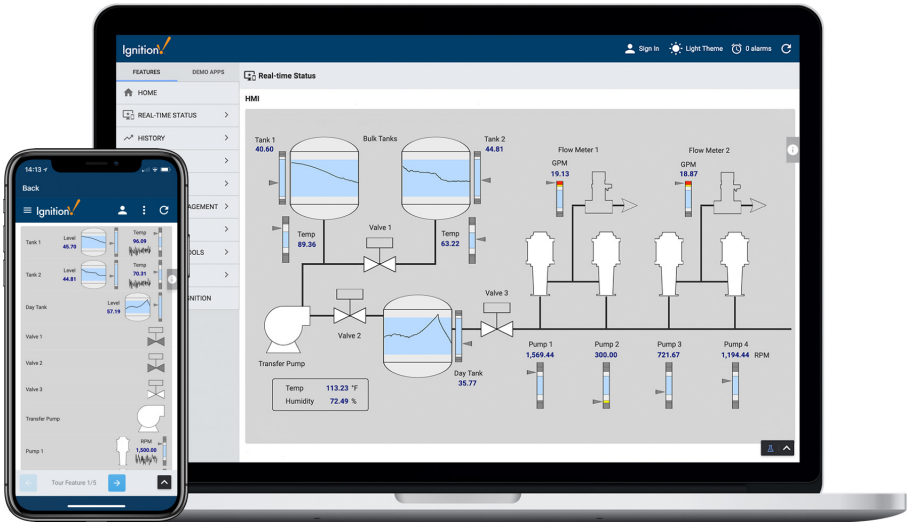
Figura 11
Interfaz gráfica HMI de un PLC

6.1. Interfaz Gráfica en el Gateway de Ignition

Para el desarrollo de este proyecto se utilizó un proveedor de software de automatización industrial basado en web llamado “Inductive Automation”, este permite diseñar, desarrollar e implementar una interfaz gráfica HMI por medio de una base de datos, servidores y diferentes módulos para visualizar, almacenar y procesar información de algún dispositivo de origen.

La plataforma Ignition es la principal herramienta de la compañía, utiliza el tipo de aplicación SCADA (Supervisory Control and Data Acquisition) común para emplear y diseñar un software que permita controlar, supervisar y tomar datos en procesos industriales de manera remota o local. En la Figura 12 se puede observar un ejemplo de una interfaz HMI desarrollada en Ignition tomando datos en tiempo real.

Figura 12
Interfaz Gráfica HMI en Ignition Designer

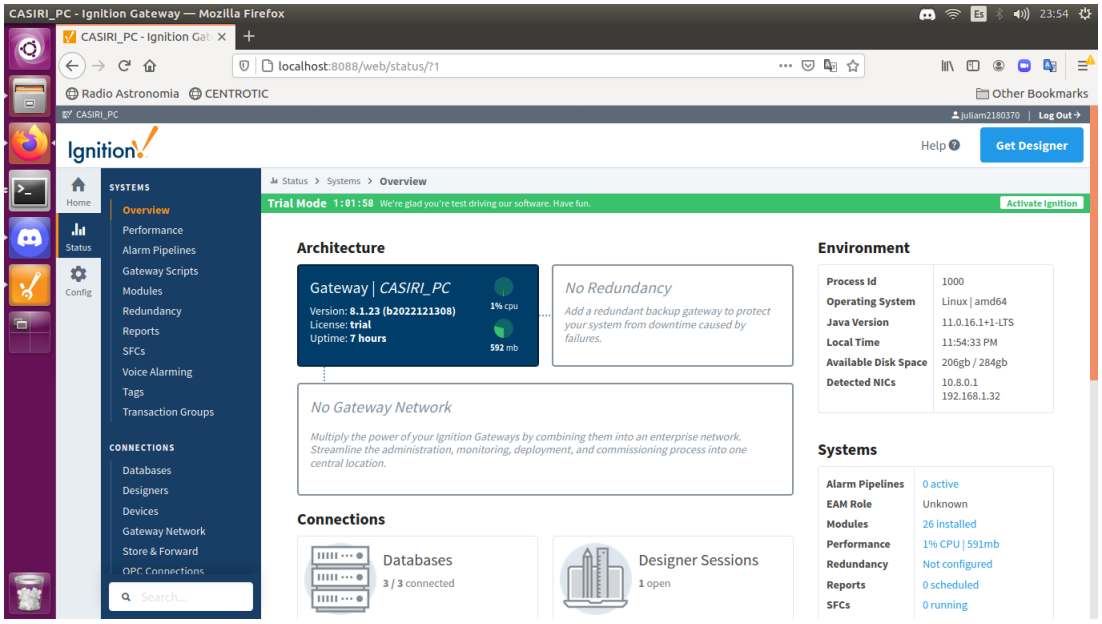


El diseño de la Interfaz HMI del sistema IoT que permita la visualización de las variables atmosféricas tomadas por la estación de radioastronomía CASIRI debe tener claros los requerimientos del sistema. La estación tiene la capacidad de tomar datos de temperatura, presión, humedad, velocidad y dirección del viento, a cada una de ellas se planea realizarle una sección de la interfaz, donde se visualice el dato en tiempo real e historial de datos para estudiar el comportamiento de la variable e incluso información precisa para predecir la proyección de la misma.

La creación y acceso a diferentes usuarios con sus roles respectivos es clave para la administración de esta, se planea asignar permisos de accesibilidad dependiendo de cada rol. Además, el proyecto tiene como finalidad visualizar la información de manera remota, es decir, la interfaz debe tener la capacidad de acceder desde otro computador externo al “localhost”, para ello se utilizará OpenVpn, el cual es un software que permite acceder a una red privada virtual desde una IP publica propia del servidor para conectarse a la plataforma diseñada.

Figura 13

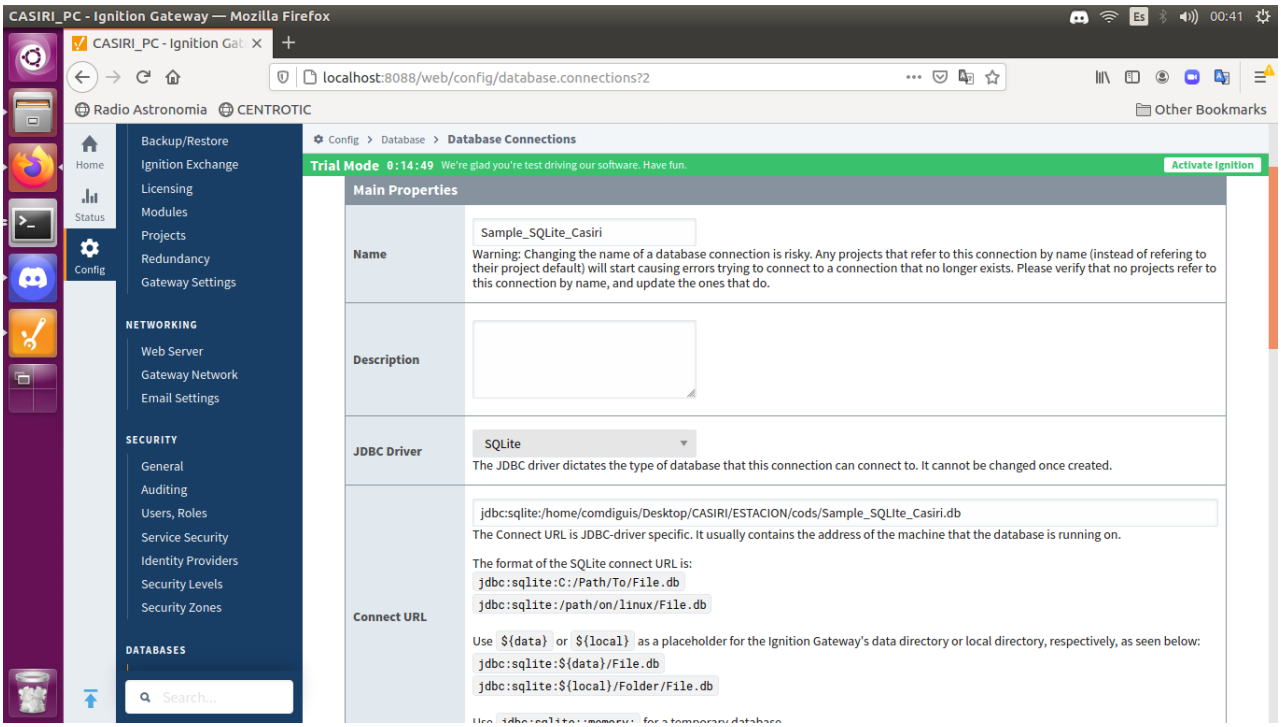
Pantalla Principal del Gateway de Ignition



En la Figura 13 se observa la puerta de enlace que conecta los dispositivos externos con el software de diseño. Allí se podrán crear base de datos, enlazar servidores, generar alarmas, configurar usuarios y roles, crear tags, entre otras herramientas que permitirán recibir y enviar información hacia los equipos conectados de manera eficiente e intuitiva de tal forma que cumpla con las necesidades del sistema.

De igual forma, se creó una base de datos desde Ignition en programación SQLite que permite una conexión estable con la versión de Ubuntu 16.04 con la que se cuenta. Allí se almacenará toda la información que le llegan a los tags de las variables tomadas por CASIRI, se nombró “Sample_SQLite_Casiri” como se muestra en la Figura 14 y se especificó la ruta donde se quería crear la base de datos.

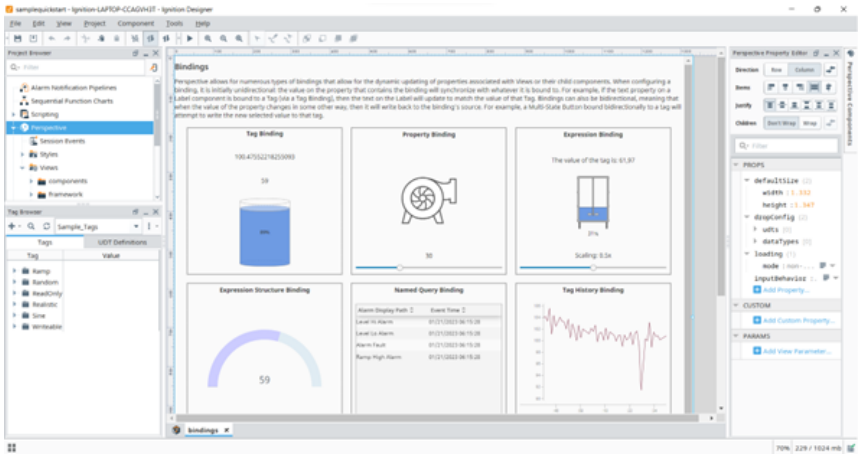
Figura 14
Creación de Base de datos SQLite



6.2. Interfaz Gráfica en el Diseñador de Ignition

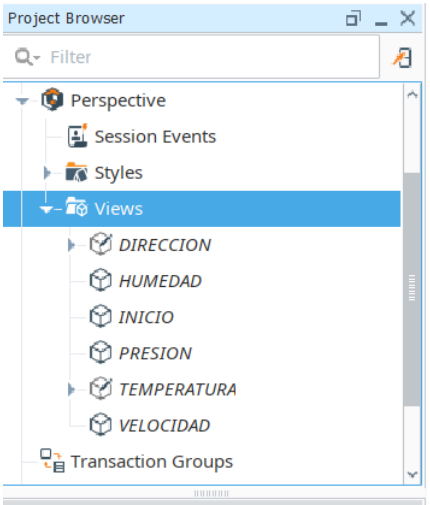
Para empezar a diseñar la interfaz gráfica, se debe crear el proyecto en el Gateway de Ignition mostrado en la Figura 13, allí se realizan todas las configuraciones de los tags, usuarios y conexiones en general provenientes del protocolo de comunicación. Luego, se abre el "Designer-Launcher" que es el aplicativo para diseñar gráficamente la interfaz y recibir toda la información que le envía el Gateway mostrada en la Figura 15. Se utilizó una plantilla "Perspective" debido a que cumple con la funcionalidad de publicar la página web en cualquier VPN (Virtual Private Network) y permite la visualización de la información de manera intuitiva y entendible para todo público.

Figura 15
Pantalla Principal del Designer de Ignition



Se tomó la decisión que para una mejor visualización de los datos, cada variable tuviera su propia vista o pestaña de la página web, y así poder restringir los accesos y mantener un control de los gráficos por separado. Allí fueron creadas en la Figura 16 para asignar las configuraciones y permisos correspondientes, además se añadió una ventana “INICIO” , la cual será la ventana principal y se encontrará una breve descripción de la Interfaz.

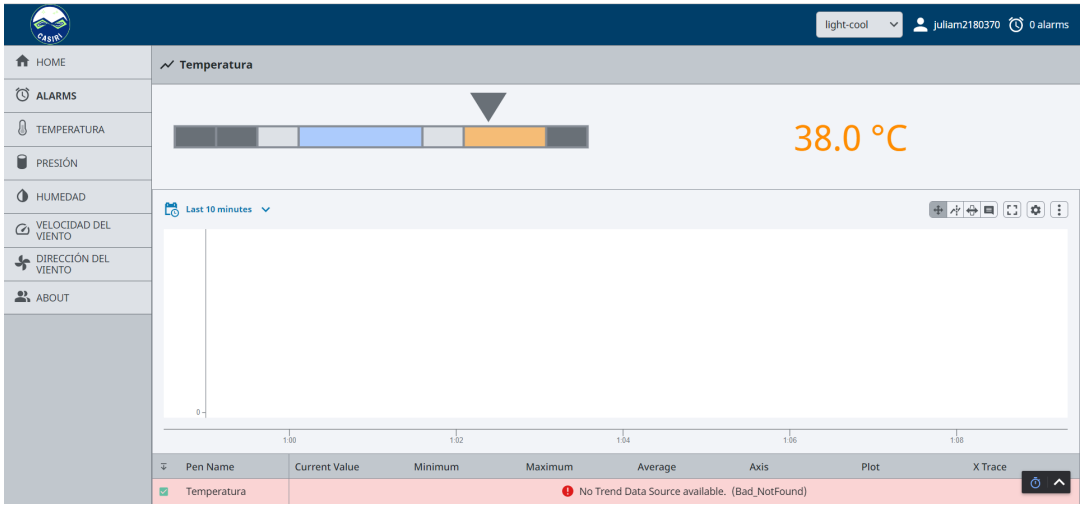
Figura 16
Pestañas diseñadas para la Interfaz



En general, las pestañas siguieron el diseño similar plasmado en la Figura 17, donde se tiene una barra de navegación superior construida por los botones que direccionan a las otras pestañas, un botón para cerrar la sesión del usuario y un botón para ir a “INICIO”. En la parte inferior izquierda se observa el valor en tiempo real por medio de un ícono que nos permite observar los datos de manera gráfica (widget), y en la parte derecha se tiene un historial de visualización, este se encarga de mostrar los valores que se han recolectado en cierto intervalo de tiempo, además de información de interés como el valor máximo, mínimo y promedio de los datos tomados.

Figura 17

Plantilla de diseño para la Interfaz



6.3. Configuración de Tags

Ahora se da paso a configurar la asignación de la información a los widgets para observar cómo cambia en tiempo real la variable obtenida de la transmisión por medio del protocolo de comunicación MQTT. En la Figura 18, se evidencia cómo se realiza la asignación del tag de Temperatura al widget del termómetro, se realizó de la misma forma para las demás variables.

Figura 18

Asignación de tags a los widgets.

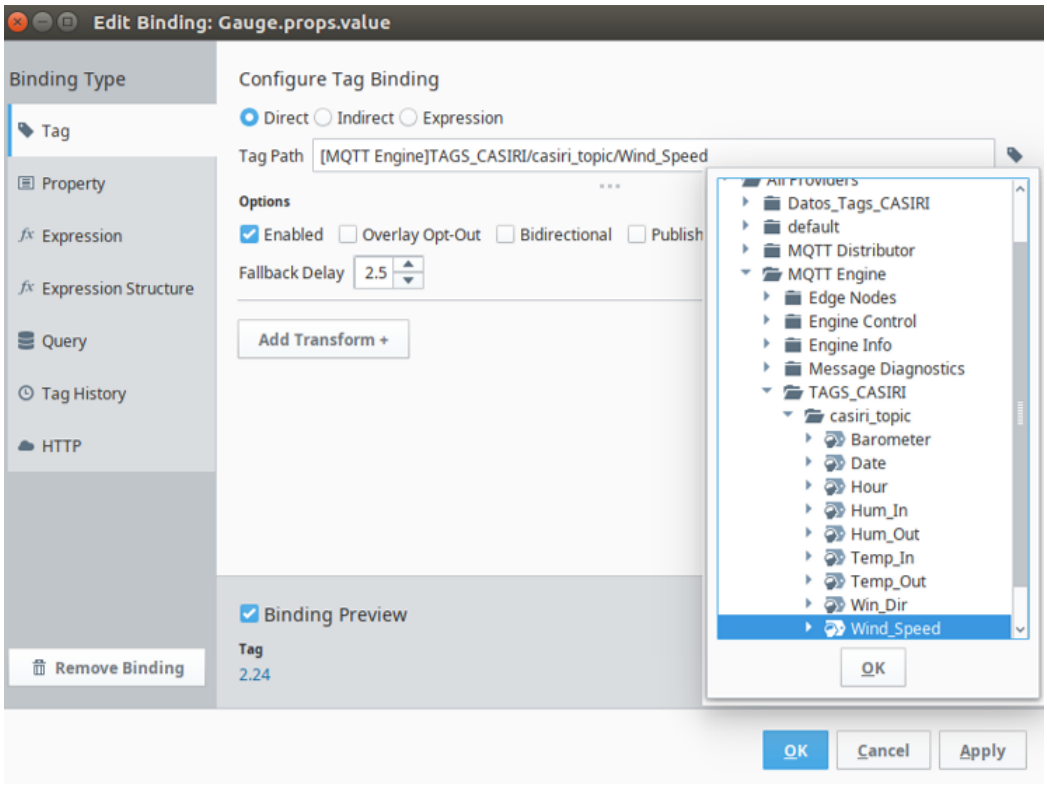
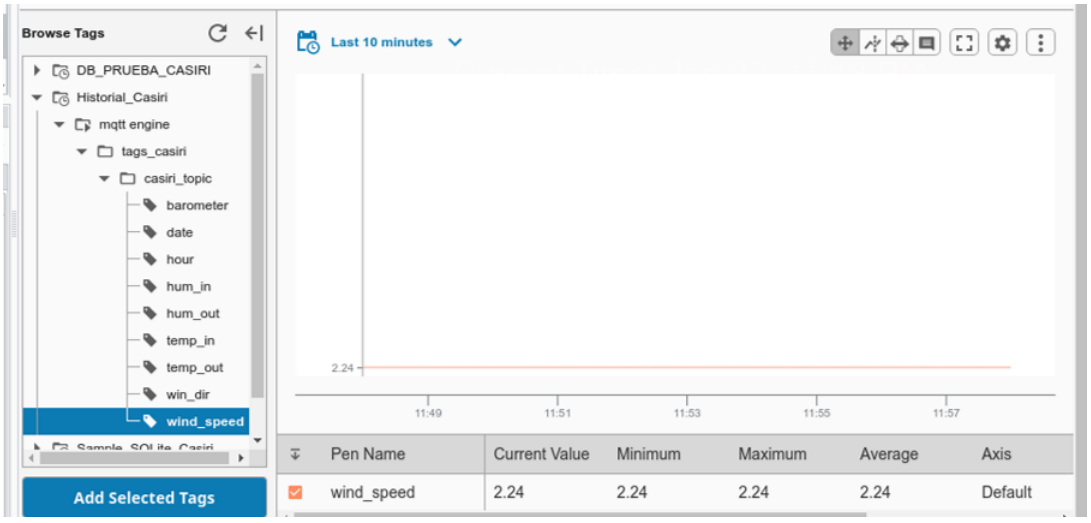


Figura 19

Asignación de tags al Historial de Visualización.

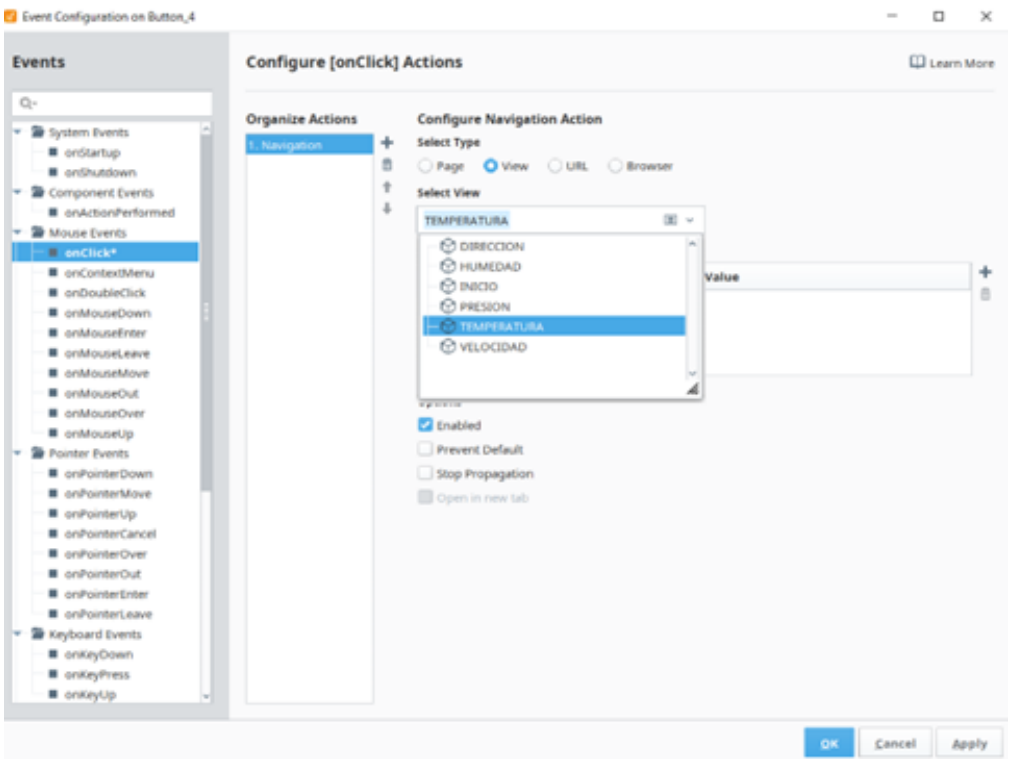


Así mismo, se realizó la asignación del tag al historial de visualización para observar el comportamiento de la variable a través del tiempo como se muestra en la Figura 19. Cabe aclarar que los tags fueron configurados para tener un historial a partir de la base de datos creada previamente en el Gateway, para así tener un respaldo de la información y que su visualización sea estable y confiable.

Luego, se configuraron los botones para navegar en las diferentes pestañas de la interfaz, esta acción se realiza mediante una configuración de un evento como muestra la Figura 20 , se da la orden para que al momento de dar un click en el botón, el usuario sea enviado a una pestaña predeterminada.

Figura 20

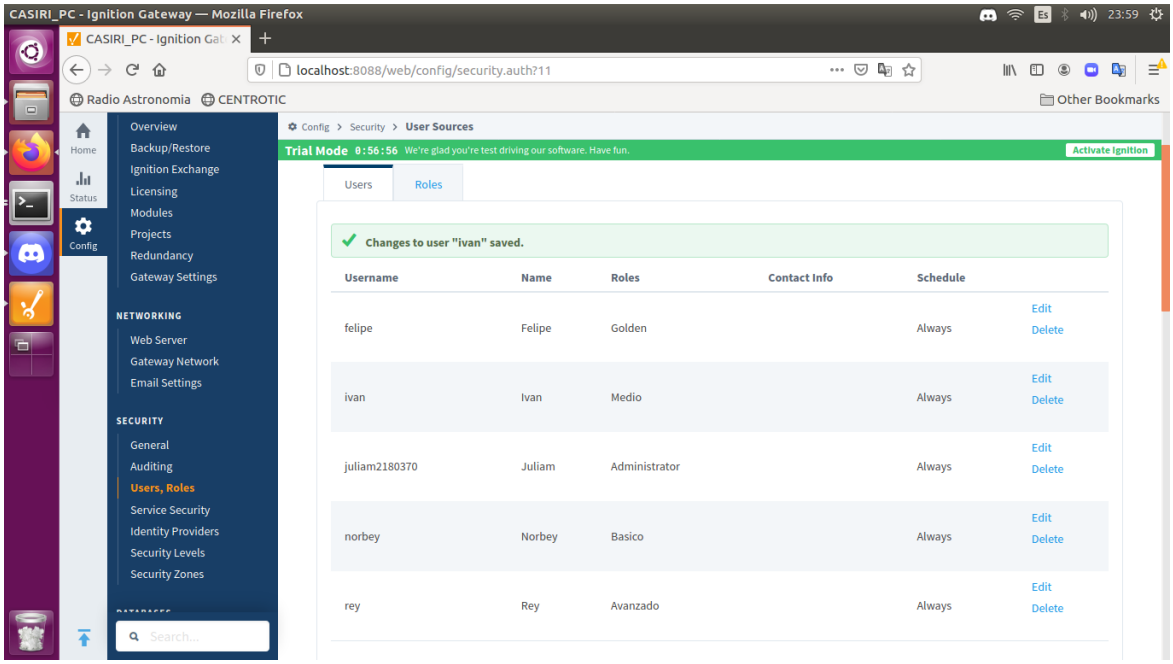
Configuración de Evento para la navegación de pestañas.



Ahora, se programa la privacidad de las pestañas en cuestión. Se planea que existan diferentes roles en los usuarios para poder tener acceso a ciertas pestañas según la subscripción que se haga a la plataforma. En la Figura 21, se muestran los roles que se planean ofrecer a los usuarios creados en el Gateway de Ignition.

Figura 21

Asignación de Roles a los Clientes.

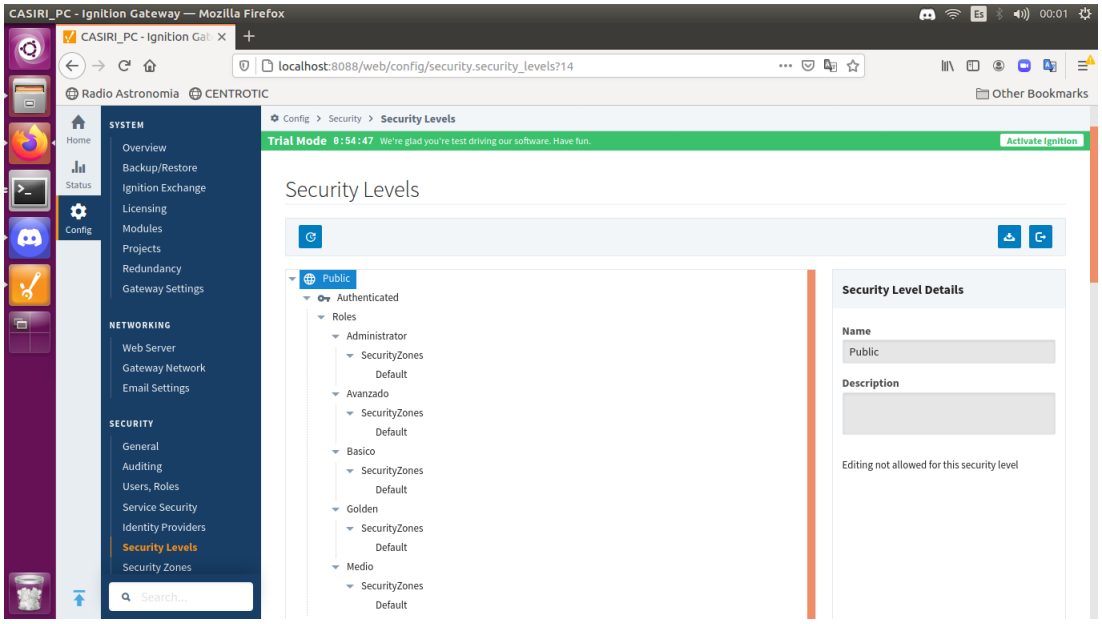


Luego de crear los roles, se les asignó un nivel de seguridad para configurar el botón con la privacidad correspondiente mostrada en la Figura 22 . Se dará acceso a las pestañas según el rol de la siguiente forma:

- **Administrador:** Temperatura, Presión, Humedad, Velocidad y Dirección del Viento
- **Golden:** Temperatura, Presión, Humedad, Velocidad y Dirección del Viento

- **Avanzado:** Temperatura, Presión, Humedad
- **Medio:** Temperatura, Presión
- **Básico:** Temperatura

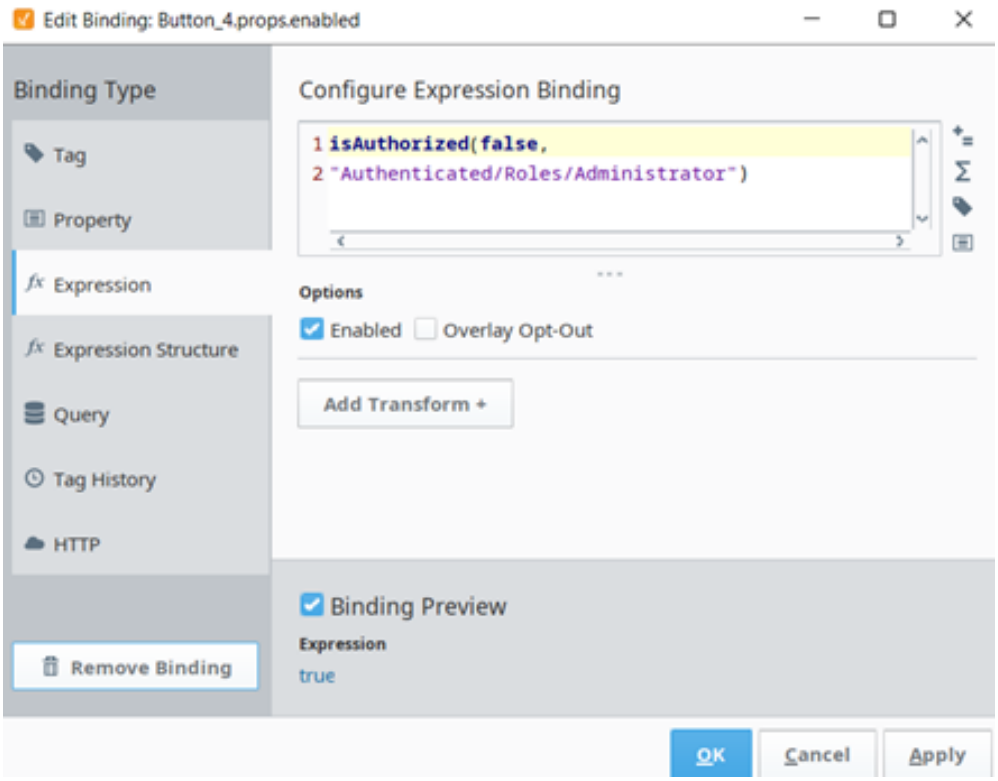
Figura 22
Configuración de niveles de seguridad.



Finalmente, en el DesignerLauncher se configuró el botón de cada pestaña editando el enlace por medio de un código como se muestra en la Figura 23, el cual habilita el botón solo para el rol del administrador, para este caso. Este procedimiento se realizó de la misma manera con los demás botones, tanto para la navegación como para la privacidad.

Figura 23

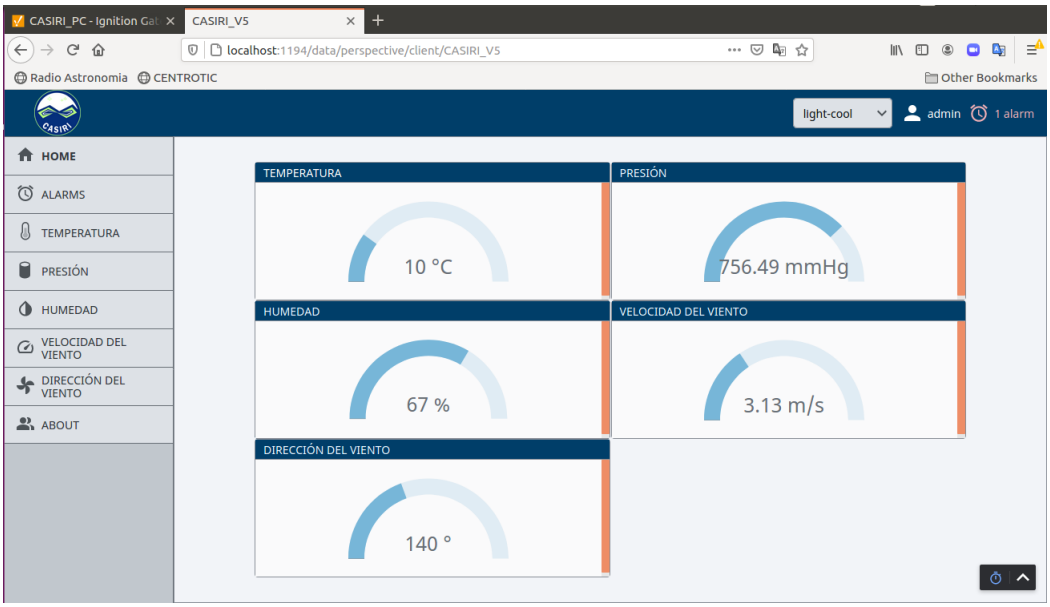
Asignación de Niveles de Seguridad a los botones



Luego de terminar toda la configuración de la interfaz gráfica HMI se lanza al servidor para poder visualizarla en la web y poder ser usada por los usuarios suscritos de manera local. Como se mencionó anteriormente, para visualizar la información de forma remota se debe conectar por medio del software OpenVPN y obtener todos los permisos de privacidad para acceder al servidor local. En la Figura 24 , se observa el diseño final de la interfaz gráfica HMI para visualizar de forma remota las variables tomadas de CASIRI incorporada a una plataforma de automatización industrial.

Figura 24

Diseño Final de Interfaz en Funcionamiento



Es de importancia resaltar que, las habilidades para diseñar la interfaz gráfica fueron desarrolladas a través de un curso de la página oficial de “Inductive Automation” y apoyado por la empresa DAUTOM (empresa con amplia experiencia en el campo de la automatización industrial e IoT). Además, se realizó un acompañamiento durante el desarrollo del proyecto por medio de una vinculación al grupo de investigación GISEL de la universidad.

Recordar que, esta configuración permite visualizar la interfaz de manera local, es decir en el servidor que está conectado directamente la estación de radioastronomía, sin embargo, la finalidad de este proyecto se enfoca en monitorear la información de manera remota. Para ello, se hará uso de un servidor VPN llamado OpenVPN, el cual nos permitirá conectarnos a la red local del servidor principal y acceder a la interfaz diseñada. En el anexo 1 se encuentra detalladamente este proceso para conectar un cliente externo al servidor local.

7. Verificación de Funcionamiento del Sistema

La implementación del sistema CASIRI requiere tener en cuenta la composición de cada subsistema (adquisición de imágenes, adquisición de RFI y estación meteorológica) a nivel de software y hardware. Las conexiones entre los elementos, la descripción y ejecución de los archivos python se debe tener claro antes de iniciar alguna verificación e implementación. Sin embargo, el enfoque de este proyecto está en la recolección de variables atmosféricas por parte de la estación meteorológica.

Luego de integrar los sensores a la interfaz gráfica HMI por medio del protocolo de comunicación MQTT para visualizar las variables atmosféricas, se procede a realizar una verificación del sistema y corrección de errores que se puedan presentar al momento de la implementación del sistema en cuestión.

La estación de radioastronomía CASIRI está ubicada actualmente en el laboratorio de investigación RadioGis, allí se realizó la verificación del funcionamiento a partir de una serie de pruebas experimentales.

Para iniciar la toma de datos, se necesita que todos los periféricos se encuentren conectados de manera correcta, dado que, de lo contrario, el software desarrollado no podrá ejecutarse y se verá afectada la información que se desea recopilar.

Figura 25
Dispositivos de sensado conectados

Como primera medida, se comprobó la conexión entre CASIRI y la consola DAVIS Vantage Pro2 para verificar la recepción de datos. Luego se procedió a calibrar los sensores, especialmente el sensor de dirección para ubicar el norte de referencia y obtener los datos de manera correcta. A pesar de que las unidades no son las convencionales, en el script de lectura de los datos fueron modificadas para trabajar con las unidades más utilizadas frecuentemente.

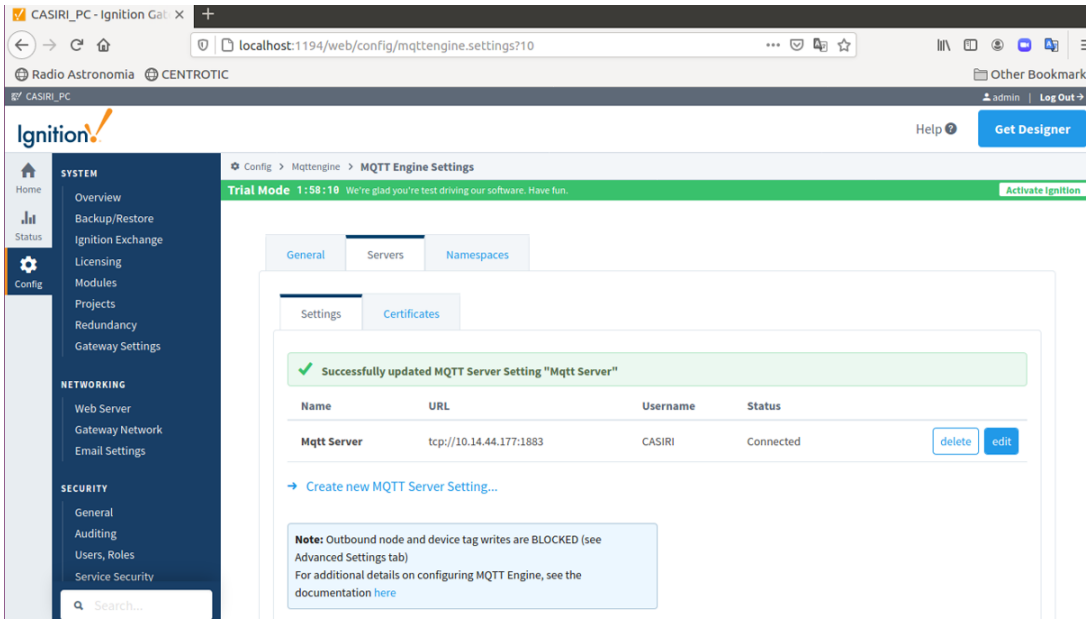
Figura 26
Consola DAVIS Vantage Pro2 en Funcionamiento

Para obtener las variables meteorológicas suministradas por la consola de la estación ambiental se utiliza la librería diseñada por el laboratorio ISTerre Grenoble para establecer comunicación, y cuatro archivos de tipo python ejecutados en el entorno virtual estacion_ambiental en la versión 2.7 de python. El archivo necesario para la adquisición, la organización y la centralización de la información al protocolo de comunicación se encuentra en la ruta: "/home/comdiguis/Desktop/CASIRI/ESTACION/cods/estacion.py"

Para mas información, en el anexo 1 se encuentra el manual de configuración para tomar los datos de la consola y enviarlos al servidor MQTT y el diagrama de conexiones hardware desde los sensores hasta el equipo de cómputo; así obtener una conexión exitosa desde la toma de datos hasta el módulo MQTT de Ignition como se muestra en la figura 27.

Figura 27

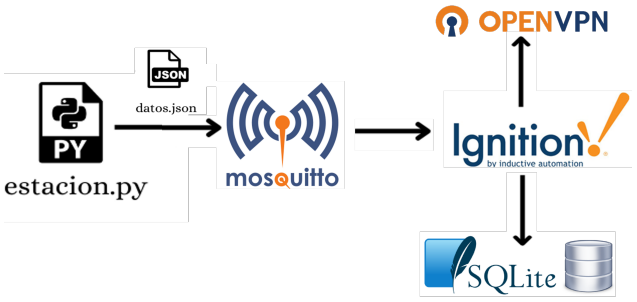
Conexión exitosa con el módulo MQTT Engine



El esquema representado en la Figura 28 muestra el orden de generación de archivos y programas utilizados por donde es enviada la información, comenzando desde la toma de datos en el script de Python hasta su publicación en la plataforma de Ignition, allí se guarda la información leída en una base de datos y simultáneamente es enviada al cliente remoto por medio de una VPN.

Figura 28

Diagrama del envío de la información



Para implementar este subsistema, es necesario conectar cada sensor individualmente al panel que se encuentra fijado al pluviómetro mediante cables de cuatro hilos (UTP). Luego, para ver los datos, se conecta el panel a la consola. Finalmente, se establece la comunicación entre la consola y el computador principal mediante el puerto serial ttyUSB0 y una tasa de baudios de 19200. La Tabla 3 proporciona una lista de los elementos y las conexiones necesarias para que el subsistema funcione correctamente desde el computador principal.

Tabla 3
Elementos subsistema estación meteorológica.

Elemento	Inventario	Conexión
Pluviómetro	M05	-
Veleta	M06	-
Consola estación	M07	-
Cable utp 4 hilos	M01	Panel de pluviómetro a consola
Cable usb	M02	Consola al computador

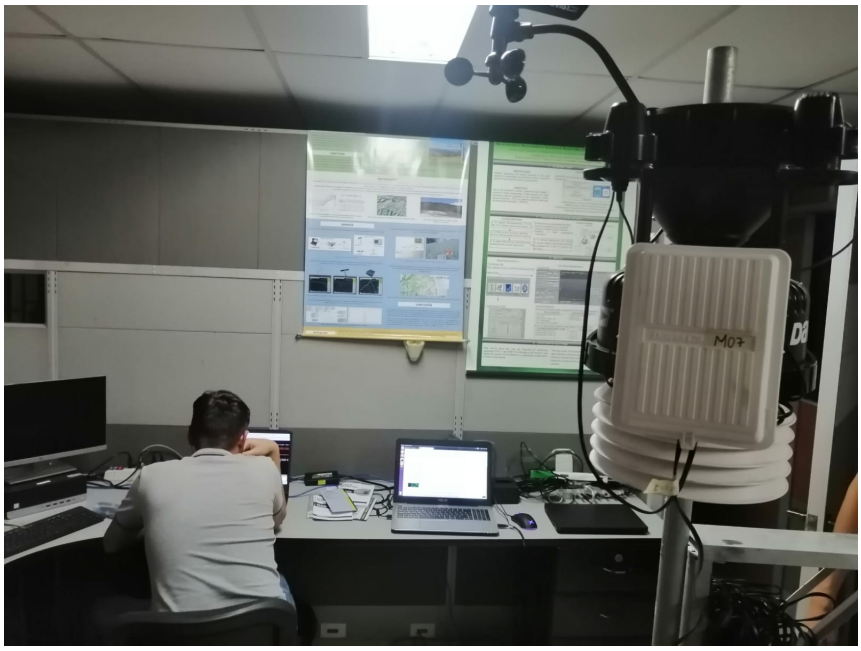
Figura 29
Conexión Hardware de la estación meteorológica

La Figura 29 muestra un esquema de conexión de hardware existente entre los dispositivos del subsistema de la estación meteorológica, donde la última se conecta al computador principal para recibir la información y dar inicio a todo el proceso de transmisión de datos.

Finalmente, se corre el script de python para leer los datos de la estación y enviarlos a los tags de Ignition por medio del protocolo MQTT como se muestra en la Figura 30, donde se hizo una prueba de la velocidad del viento tomada en tiempo real y vista en la interfaz HMI diseñada en Ignition.

Figura 30

Verificación del Sistema IoT Implementado de forma local



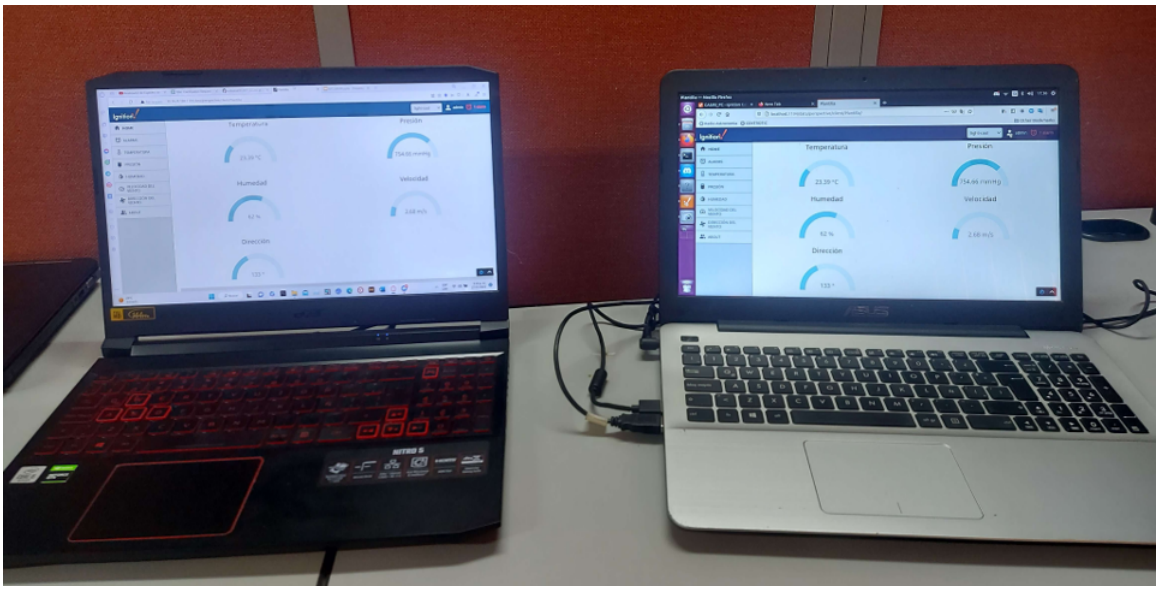
A continuación se encuentra un link de Youtube, donde se muestra la verificación del sistema IoT y la implementación en el laboratorio de la universidad donde se encuentra actualmente CASIRI.

Link Video: `https://youtu.be/NiJd2XLQeCE`

A demás se hizo una verificación del sistema diseñado de manera remota mostrado en el siguiente video y en la Figura 31.

Link Video: `https://youtu.be/J_ns-zQhMw8`

Figura 31
Verificación del Sistema IoT Implementado de forma remota



Así, se verificó la correcta integración de la interfaz gráfica HMI con los dispositivos usados por CASIRI con una serie de pruebas de funcionamiento en el laboratorio del grupo de investigación Radiogis, dentro del campus universitario. Es de suma importancia resaltar que estas pruebas se realizaron con el fin de brindar un primer ensayo y extender el sistema a otros lugares de interés para su implementación.

8. Conclusiones y Recomendaciones

8.1. Conclusiones

El sistema IoT diseñado para la obtención de variables atmosféricas tomadas por la estación de radioastronomía CASIRI fue desarrollado con el fin de brindar un servicio de información y monitoreo a usuarios que se suscriban a la plataforma por medio de una interfaz gráfica HMI de forma remota y en tiempo real dirigido a la academia e investigación en general, lo cual lo hace inmensamente maleable para cualquier proyecto macro que se planee implementar.

Al principio, CASIRI estaba configurado para recolectar datos en una frecuencia e intervalos de tiempo específicos utilizando terminales en una computadora principal, lo que restringía el acceso a la información de manera automática y remota. Los usuarios tenían que ingresar manualmente la cantidad de muestras que querían recolectar y el tiempo de muestreo entre cada una de ellas. Con la implementación de este proyecto, se ha mejorado la obtención y visualización de la información. Los usuarios pueden ver los datos de forma remota y en tiempo real y ver su historial en el transcurrir del tiempo, lo cual facilita el análisis y proyección de los datos para investigaciones futuras.

El uso del protocolo MQTT, la plataforma de Ignition y el software Mosquitto fueron fundamentales para el desarrollo del proyecto. La integración de los sensores con el protocolo MQTT dió un gran impulso al proyecto al permitir el envío de datos de manera automatizada y en tiempo real, lo que mejoró significativamente el método de obtención de muestras y el envío de las mismas. Además, el protocolo MQTT implementado por medio del software Mosquitto permite conectar-

se y transmitir datos desde una gran cantidad de sensores de manera simultánea aumentando la escalabilidad del sistema y su rango de aplicación.

Por otro lado, la plataforma de Ignition actuó como un intermediario entre los datos y el usuario final o cliente, permitiendo una mejor interacción y comprensión de las variables a través de herramientas de análisis y gráficos interactivos de acceso remoto para facilitar el monitoreo continuo de la información.

El sistema IoT implementado ha agregado nuevas funciones para ayudar a los usuarios o clientes a obtener una comprensión más completa de la información. Se han añadido gráficos interactivos y herramientas de análisis para ayudar a los usuarios a visualizar y entender mejor los datos. También se ha agregado la posibilidad de exportar las variables en una base de datos por medio del historial de tags que genera Ignition para que la información sea guardada de manera confiable y estable. Adicionalmente, se ha implementado un sistema de alertas para que los usuarios puedan recibir notificaciones en caso de cambios significativos en los datos y estar siempre al tanto de las condiciones atmosféricas donde se encuentra la estación.

La interacción publicador-cliente del protocolo de comunicación MQTT permite recibir y enviar información al servidor principal, sin embargo este proyecto tuvo como finalidad únicamente monitorear de las variables atmosféricas, es decir, el usuario solo tendrá la posibilidad de leer información. Por ello, se desarrollo una interfaz donde el componente gráfico sea lo mas importante y así obtener una lectura de datos amigable e intuitiva con el público. De igual forma, se resalta la asignación de roles a los usuarios para el desarrollo de la interfaz, lo cual le permitirá ciertos permisos de acceso según sea la suscripción que haya realizado.

Después de desarrollar el proyecto se realizó una fase de pruebas en el laboratorio de la universidad, por medio de la implementación del sistema IoT con la estación meteorológica. Para este subsistema se obtuvieron las variables (temperatura, presión, humedad, velocidad y dirección del viento) y se organizaron en un archivo *"json"*. A través de un script de python fueron enviados a un servidor MQTT para generar la comunicación publicador-cliente en base a la plataforma Ignition y simultáneamente guardar las variables en una base de datos. De este modo, a través de estas pruebas, se garantiza el funcionamiento y correcta implementación del sistema.

La elección del lugar de implementación de CASIRI no debe considerar alguna infraestructura determinada ni condiciones específicas para cumplir con el funcionamiento del sistema. La movilidad y practicidad son las mayores ventajas de la estación. Idealmente se debería instalar en algún lugar viable para tomar señales sin interferencia en los cielos silenciosos. Sin embargo, se planea que el sistema se pueda implementar en lugares sin importar las condiciones para servir de base en cualquier caso de estudio o investigación que se requiera.

CASIRI es un proyecto que persigue metas ambiciosas. El agregado de un nuevo componente que permita comunicaciones remotas, aumenta su potencial de investigación, logrando que funcione prácticamente en cualquier lugar sin ningún tipo de limitación o restricción. Actualmente, se tiene un prototipo un poco mecánico y tedioso que se encuentra en pruebas de funcionamiento. Sin embargo, se planea transformarlo en un sistema mecánico robusto para cumplir con las necesidades de los interesados, utilizando los recursos ya disponibles en el prototipo y mejorando su funcionamiento con la implementación del sistema IoT y la interfaz gráfica HMI desarrollada.

8.2. Recomendaciones

El proyecto garantiza que el sistema es operativo para monitorear datos atmosféricos compartiendo la información de forma remota y automatizada cumpliendo con los objetivos planteados en este proyecto. Sin embargo, este proyecto posee retos para futuras investigaciones, entre ellas se encuentra implementar el diseño planteado a la estación de radioastronomía CASIRI en su totalidad, incluyendo todos los susbsistemas de la estación para transmitir imágenes obtenidas de la cámara y dataset de captura del espectro por medio de la interferencia de radiofrecuencia (RFI) en tiempo real.

Así mismo, el proyecto puede incluir la especificación Sparkplug B al protocolo de comunicación MQTT para definir cómo se envían y reciben los datos. Los dispositivos y sensores en el borde de una red pueden usar Sparkplug B para comunicarse con aplicaciones como sistemas SCADA, historiadores y programas de análisis.

La implementación de los scripts de Python puede explorarse internamente en Ignition, así integrar el desarrollo del proyecto en la misma plataforma para aumentar la escalabilidad, eficiencia y facilidad de implementación.

Referencias Bibliográficas

Arízaga Silva, J. A. (2013). Sistema de monitoreo de variables ambientales utilizando de ethernet.

AWS Amazon (2022). `https://aws.amazon.com/es/what-is/mqtt/`, consultado 22 de enero de 2023.

Bender, M., Kirdan, E., Pahl, M.-O., and Carle, G. (2021). Open-source mqtt evaluation. *IEEE 18th Annual Consumer Communications Networking Conference (CCNC)*, 1-4. https://doi.org/10.1109/CCNC49032.2021.9369499.

Bryce, R., Shaw, T., and Srivastava, G. (2018). Mqtt-g: A publish/subscribe protocol with geolocation. *41st International Conference on Telecommunications and Signal Processing (TSP)*, 1-4. https://doi.org/10.1109/TSP.2018.8441479.

Carlos, D. C., Adrian, J., Pablo, L., Benjamín, C., Eduardo, A., Javier, C., Ezequiel, G., and Matías, B. (2019). Interfaz humano-mÁquina (diseÑo ux). XXI Workshop de Investigadores en Ciencias de la Computación.

Censier, B., Thomas, I., Auxepaules, G., Flouret, B., and Renaud, P. (2021). Sky modeling for correction and calibration of a single vhf antenna system for interferences monitoring. *XXXIVth General Assembly and Scientific Symposium of the International Union of Radio Science (URSI GASS)*, 1-3. https://doi.org/10.23919/URSIGASS51995.2021.9560246.

DAVIS (2022).

Gartner Glossary (2022). https://www.gartner.com/en/information-technology/glossary/gateway, consultado 22 de enero de 2023.

Inductive Automation (2022a). https://docs.inductiveautomation.com/display/DOC80/Tags, consultado 22 de enero de 2023.

Inductive Automation (2022b). https://docs.inductiveautomation.com/display/DOC81, consultado 22 de enero de 2023.

Inductive Automation (2022c). https://inductiveautomation.com/ignition/, consultado 22 de enero de 2023.

Liao, Y., de Freitas Rocha Loures, E., and Deschamps, F. (2018). Industrial internet of things: A systematic literature review and insights. *IEEE Internet of Things*, 5(6) 4515–4525. https://doi.org/10.1109/JIOT.2018.2834151.

Manahanm, J., Templet, R., Estevez, C., and Grinberg, D. (2020). How to id internet of things solutions without feeling like an idiot: An application perspective. *IEEE Industry Applications Magazine*, 26(2) 28-38. https://doi.org/10.1109/MIAS.2019.2943648.

Meng, Z., Z, W., Muvianto, C., and Gray, J. (2017). A data-oriented m2m messaging mechanism for industrial iot applications. *IEEE Internet of Things Journal*, 4(1) 236–246. https://doi.org/10.1109/JIOT.2016.2646375.

Mosquitto (2022). https://mosquitto.org/, consultado 22 de enero de 2023.

Naik, N. (2017). Choice of effective messaging protocols for iot systems: Mqtt, coap, amqp and http. *2017 IEEE International Systems Engineering Symposium (ISSE)*, 1-7. https://doi.org/10.1109/SysEng.2017.8088251.

Naik, N., Jenkins, P., Davies, P., and Newell, D. (2016). Native web communication protocols and their effects on the performance of web services and systems. *2016 IEEE International Conference on Computer and Information Technology (CIT)*, 219-225. https://doi.org/10.1109/CIT.2016.100.

OPENVPN (2022). `https://public.nrao.edu/telescopes/radio-telescopes/`, consultado 22 de enero de 2023.

Python (2022). `https://www.python.org/doc/essays/blurb/`, consultado 22 de enero de 2023.

Quezada Quezada, J. C., Bautista López, J., Flores García, E., and Quezada Aguilar, V. (2014). Diseño e implementación de un sistema de control y monitoreo basado en hmi-plc para un pozo de agua potable. *Ingeniería Investigación y Tecnología*, XV:41–50.

Sisinni, E., Saifullah, A., Han, S., Jennehag, U., and Gidlund, M. (2018). Industrial internet of things: Challenges, opportunities, and directions. *IEEE Transactions on Industrial Informatics*, 14(11) 4724-4734. https://doi.org/10.1109/TII.2018.2852491.

SQLite.org (2022). `https://www.sqlite.org/index.html`, consultado 22 de enero de 2023.

Tantitharanukul, N., Osathanunkul, K., Hantrakul, K., Pramokchon, P., and Khoenkaw, P. (2017). Mqtt-topics management system for sharing of open data. *International Conference on Digital Arts, Media and Technology (ICDAMT)*, 62-65. https://doi.org/10.1109/ICDAMT.2017.7904935.

TicPortal (2022). `https://www.ticportal.es/glosario-tic/servidores`, consultado 22 de enero de 2023.

TLT-IOT (2020). `https://tlt-iot.com/que-es-un-sistema-hmi/`, consultado 22 de enero de 2023.

Wilson, C., Storel, M., and Tzioumis, T. (2013). Measures for control of emi and rfi at the murchison radioastronomy observatory. *2013 Asia-Pacific Symposium on Electromagnetic Compatibility (APEMC)*, 1-4. https://doi.org/10.1109/APEMC.2013.7360651.

Apéndices

Apéndice A. Manual de Conexión para Hardware y Software

En el presente anexo se muestra un manual de software y de integración para la estación presentando los esquemas de conexión y enlace para los dispositivos a utilizar. Inicialmente incluye un paso a paso de la instalación de una imagen clonada en un disco, donde se encuentra todos los archivo y programas necesarios para la implementación del sistema y que la conexión sea exitosa. Luego se encuentra un esquema sencillo de conexión de los dispositivos a nivel de hardware para conectar los equipos necesarios de manera correcta. Finalmente, se tiene una guía de como encender el broker de Mosquitto y enlazarlo con el módulo MQTT Engine de Ignition para poder ejecutar los scripts sin ningún tipo de error.

Universidad industrial de Santander

Guía para la toma y visualización de datos ambientales de CASIRI

Enero de 2023

Norbey Camilo Infante Villamil
Juliam Andrés Díaz Barrera

Índice general

Introducción

El sistema CASIRI cuenta con un subsistema que recolecta datos del medio ambiente, como la temperatura, humedad, presión, velocidad y dirección del viento. Por esta razón, se requiere una interfaz gráfica que permita visualizar las variables de manera clara y sencilla. Esta guía detalla cómo configurar la toma de datos y cómo visualizarlos en la interfaz gráfica. Además, incluye un paso a paso para la instalación de una imagen clonada de la computadora principal donde se desarrolló todo el proyecto, con el objetivo de ahorrar tiempo en la instalación de los programas necesarios si se desea utilizar en otra computadora distinta.

Capítulo 1

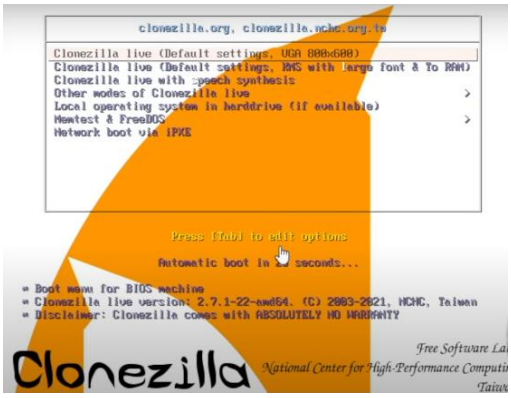
1. Instalación de la imagen clonada con el proyecto

1.1. Preparación de la computadora y demás requerimientos

- La computadora en la que se desea instalar la imagen del proyecto debe tener un disco duro no inferior a 500GB, esto debido a que la imagen fue tomada de un disco de ese tamaño.
- Se debe contar con una USB de mínimo 8GB para instalar el software “Clonezilla 2.7”. Este software se instala como boot en la memoria para después iniciar la computadora desde la USB.

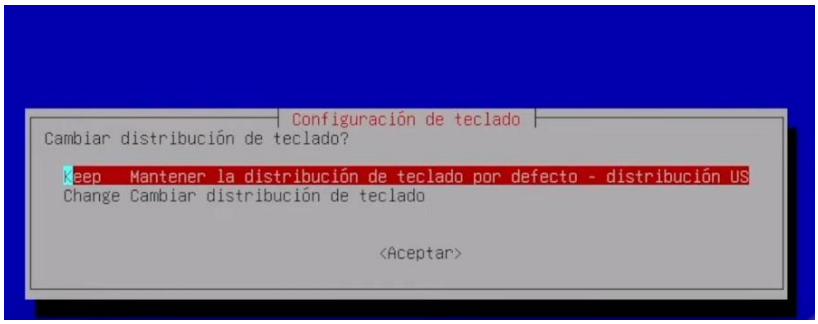
1.2. Instalación de la imagen

- Una vez se inicia la computadora desde la USB se tendrá la siguiente interfaz:

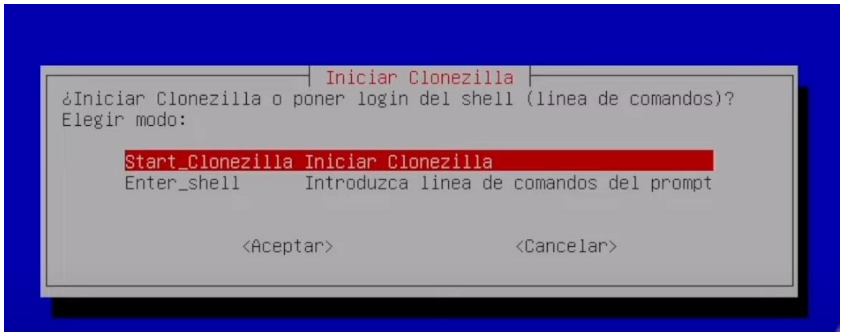


Se selecciona la opción por defecto y continua con el siguiente paso.

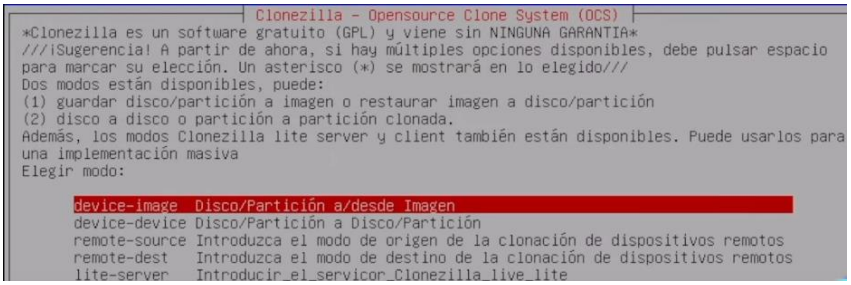
- Se espera a que cargue todo el proceso y la siguiente opción será seleccionar el idioma del software, en este caso seleccionar español.
- En el siguiente pantallazo del programa se debe seleccionar “Mantener la distribución del teclado por defecto”.



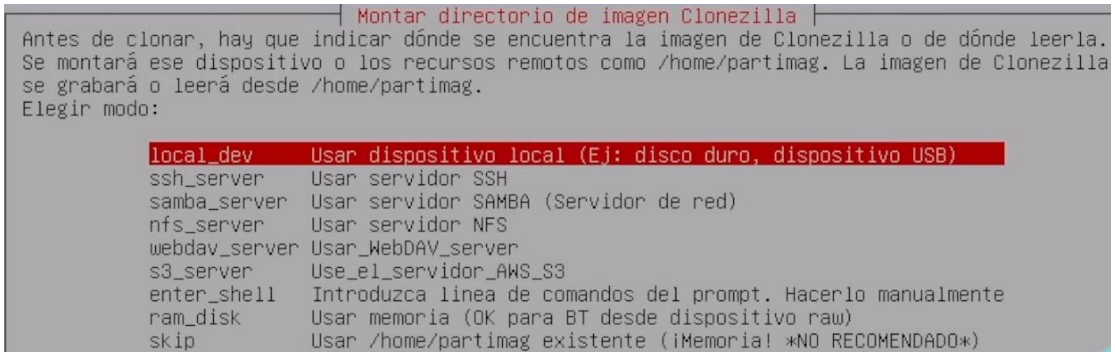
- Una vez realizado el paso anterior se selecciona iniciar Clonezilla.



- Se selecciona la opción que aparece en la siguiente imagen para continuar con la clonación del disco.

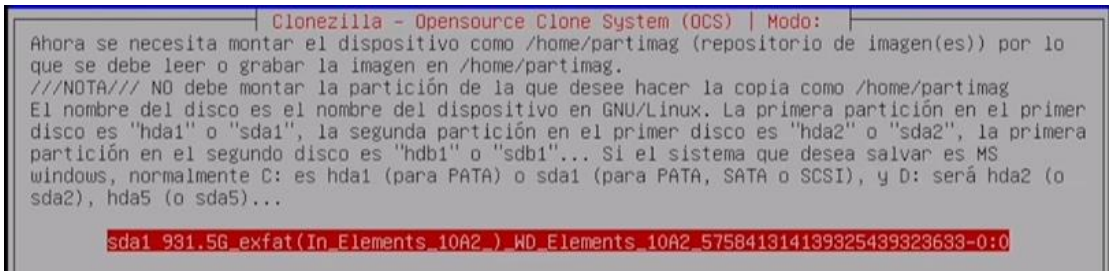


- Una vez que se termine el paso anterior se debe conectar el disco que contiene la imagen de la computadora principal y se selecciona la siguiente opción.

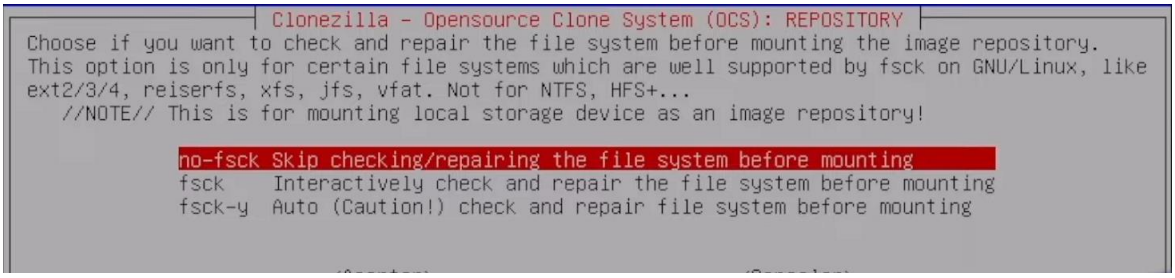


- Cuando se conecta el disco y se selecciona la opción, Clonezilla hará una búsqueda de los dispositivos de almacenamiento conectados, ahí debe aparecer el disco de la copia. Para continuar con el proceso y terminar la búsqueda presionar “Ctrl+C”.

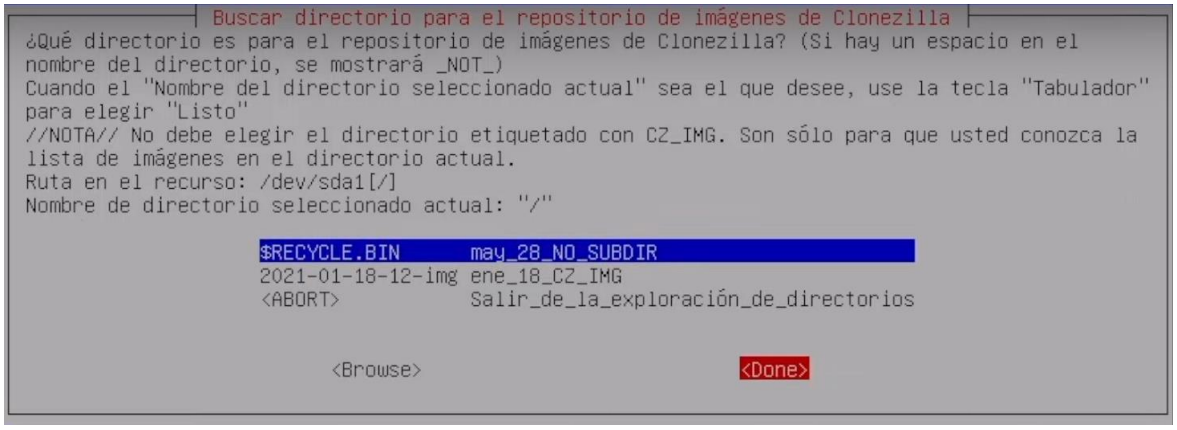
- En el siguiente paso se debe seleccionar el disco en donde está la imagen para clonar, en este caso es el disco proporcionado en los entregables.



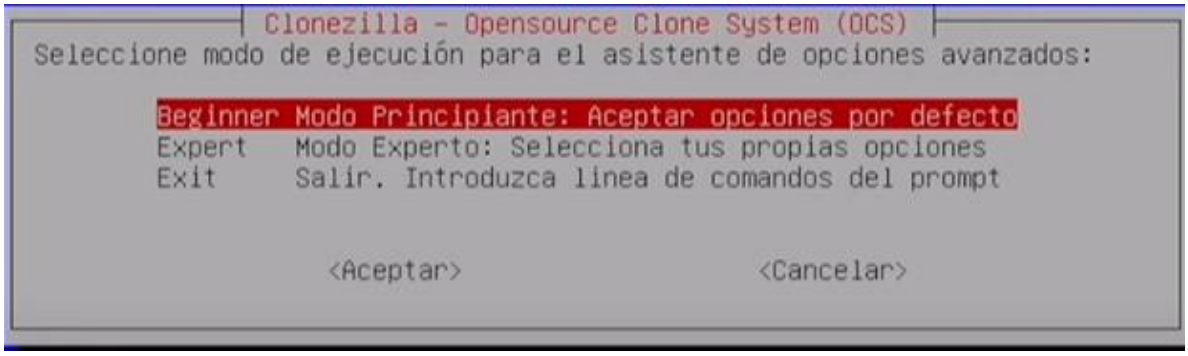
- En el siguiente paso se selecciona la opción de la siguiente imagen.



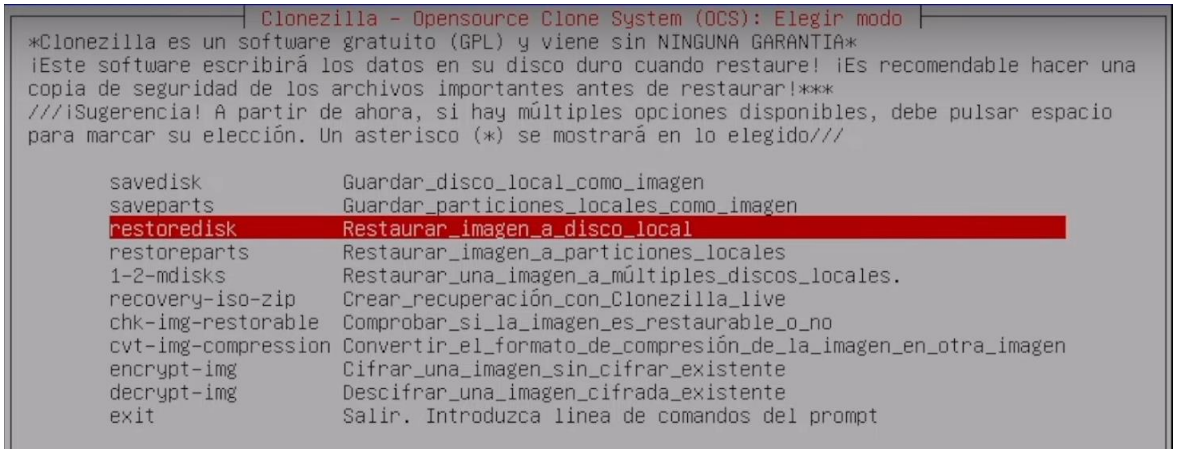
- Continuando con el proceso en la siguiente opción se debe presionar Tabulador y presionar Done.



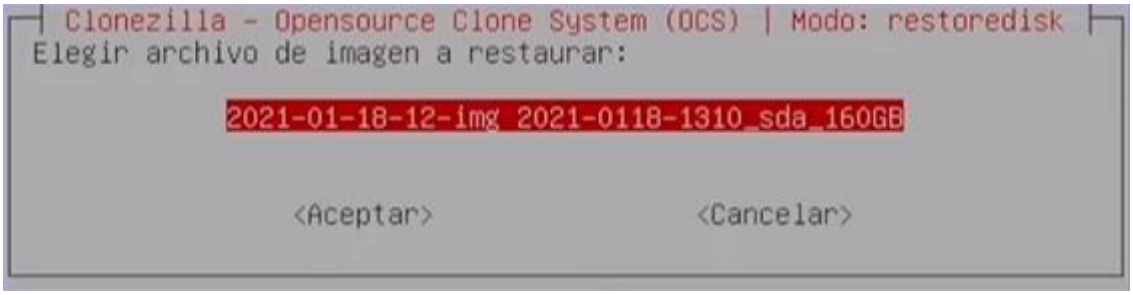
- Se selecciona la opción de principiante



- Para el siguiente paso se seleccionar restaurar imagen en el disco local.

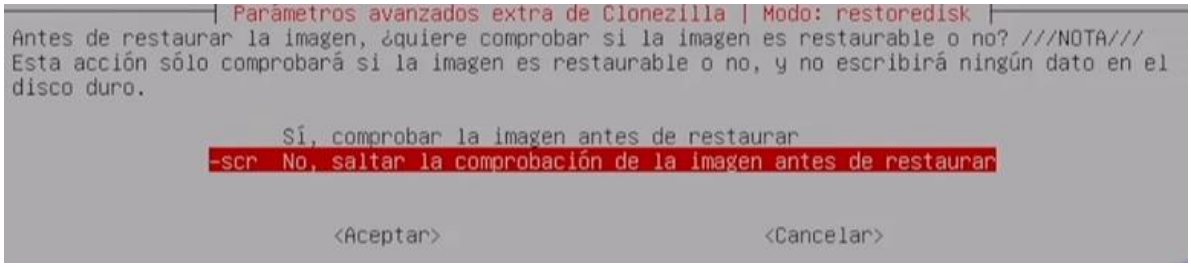


- Ahora terminado el paso anterior se debe seleccionar la clonación que esta guardada en el disco.

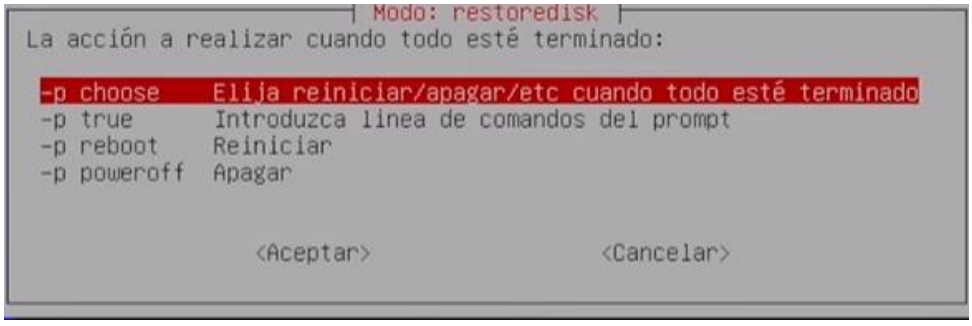


- Ahora se debe seleccionar donde se quiere restaurar la clonación en este caso es el disco interno de la nueva computadora que es donde se desea instalar.

- Terminado el proceso anterior se debe seleccionar la siguiente opción (Recomendada).



- Ahora preguntará que hacer cuando termine la clonación, en este caso seleccionar la de interés



- Al pulsar “Enter” el programa va a avisar que el disco duro interno se va a formatear, para continuar vuelve a presionar “Enter”.
- Por ultimó el programa va a dar aviso que los datos se van a perder de manera permanente, para continuar se presiona la techa “y” seguido de “Enter”.
- El proceso anterior se repite dos veces y inicia con la restauración.

Cuando se termina la restauración se selecciona apagar equipo, una vez apagado se retira la memoria USB y el disco externo, esto para que la computadora inicie con el disco interno completamente idéntico al original.

NOTA: La computadora debe estar en el usuario comdiguis, la contraseña es ieeeieee.

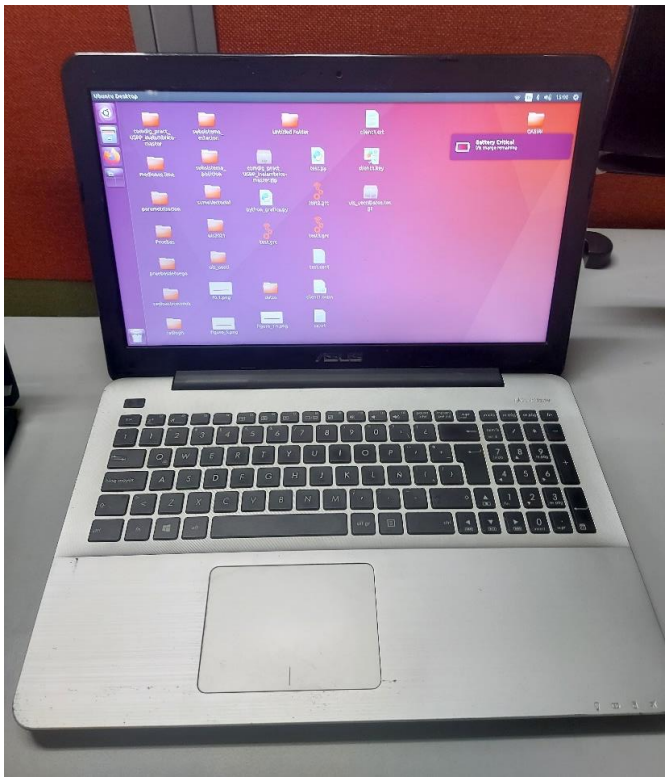
Capítulo 2

2. Diagramas y conexión física de CASIRI

2.1. Componentes para usar en la conexión

Para realizar la conexión de la estación ambientar de CASIRI se necesitan los siguientes elementos:

- Computadora donde está la imagen ya ejecutándose del proyecto.

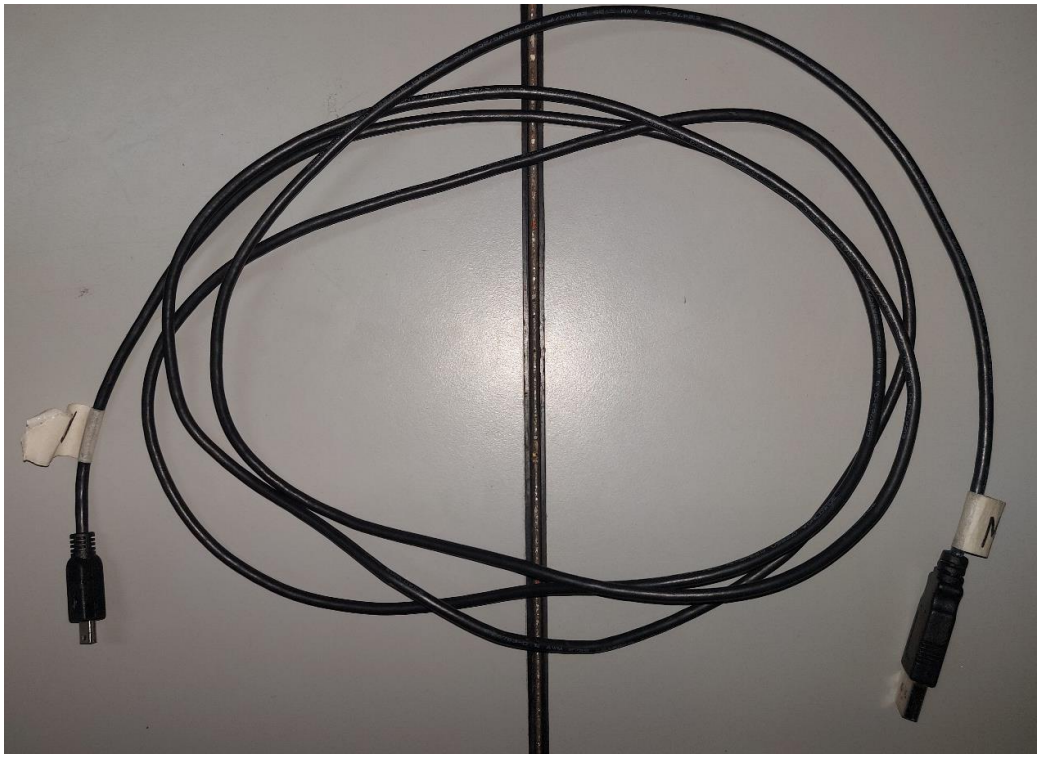


- Consola de datos de la estación ambiental.

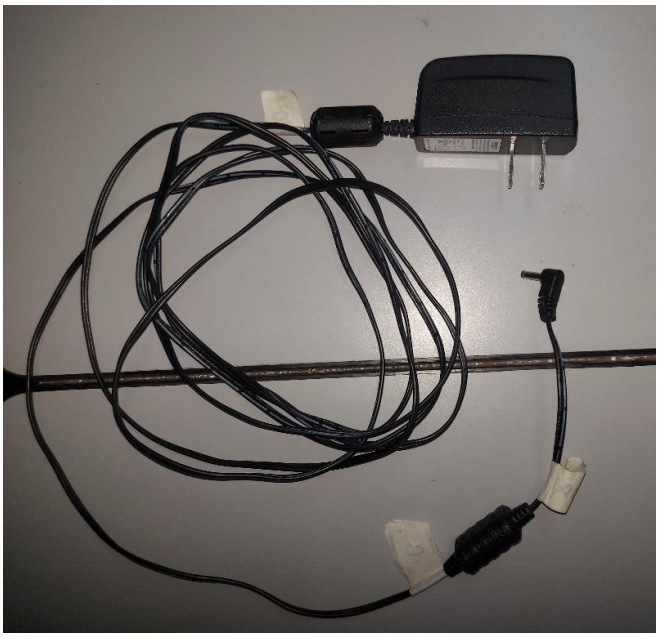
- Estación ambiental DAVIS.

- Cable UTP de cuatro hilos para conectar la consola con la estación ambiental DAVIS.

- Cable de conexión USB entre la consola y la computadora.



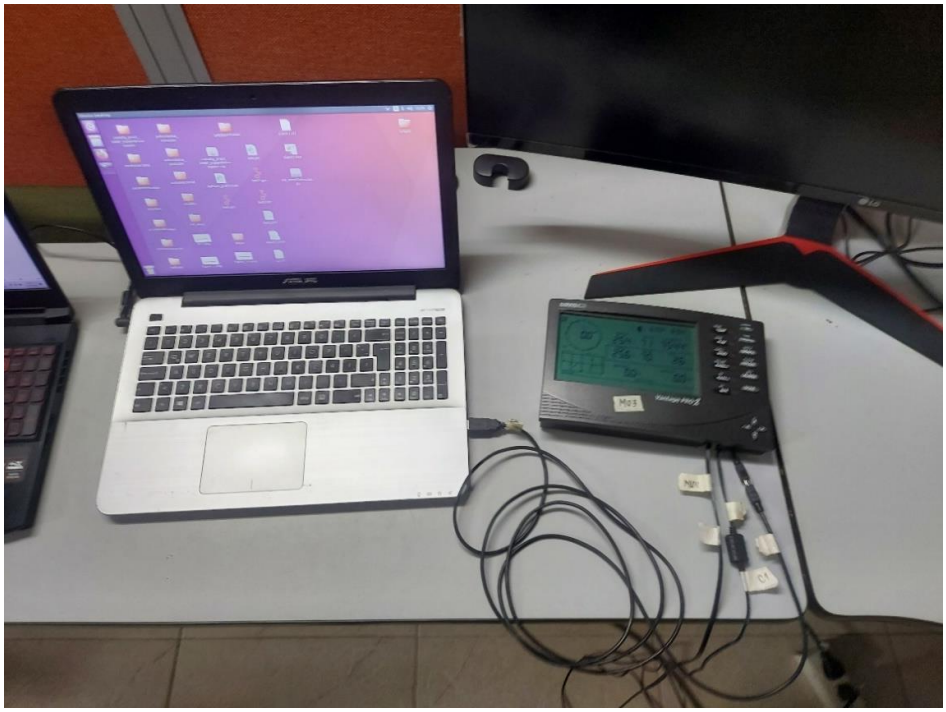
- Cable de alimentación de la consola.



2.2. Diagrama de conexión

2.3. Conexión física

La consola cuenta con un cable de alimentación, un cable para la comunicación con la estación y otro para comunicación USB con la computadora. La conexión física está representada en las siguientes imágenes:



Capítulo 3

3. Toma y envío de datos

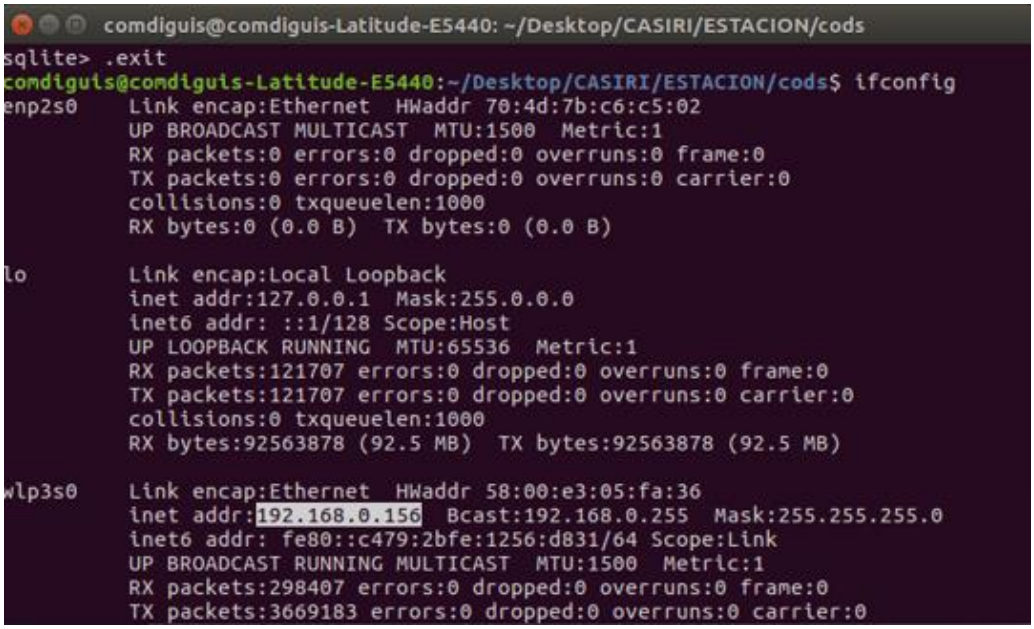
3.1. Preparación del servidor Mosquitto

Los siguientes pasos correspondo al encendido del servidor mosquito y comprobación del estado

- Abrir una terminal del sistema presionando las teclas “Ctrl+Alt+T”.
- Ejecutar la instrucción “sudo systemctl status mosquitto”, para verificar que mosquitto este corriendo.
- Si en la terminal aparece inactivo ejecutar la siguiente instrucción, “sudo systemctl start mosquitto”.
- Se vuelve a ejecutar la instrucción ,“sudo systemctl status mosquitto” para comprobar de nuevo el estado.
- Abrir otra terminal presionando las teclas “Ctrl+Alt+T” y ejecutar las siguientes instrucciones.
- Ejecutar el comando “cd /home/comdiguis/Desktop/CASIRI/ESTACION/cods/”.
- Ejecutar la instrucción “mosquitto -c server.conf -v”.
- Comprobar que el puerto 1883 de ipv4 este escuchando.
- Cerrar todas las terminales.

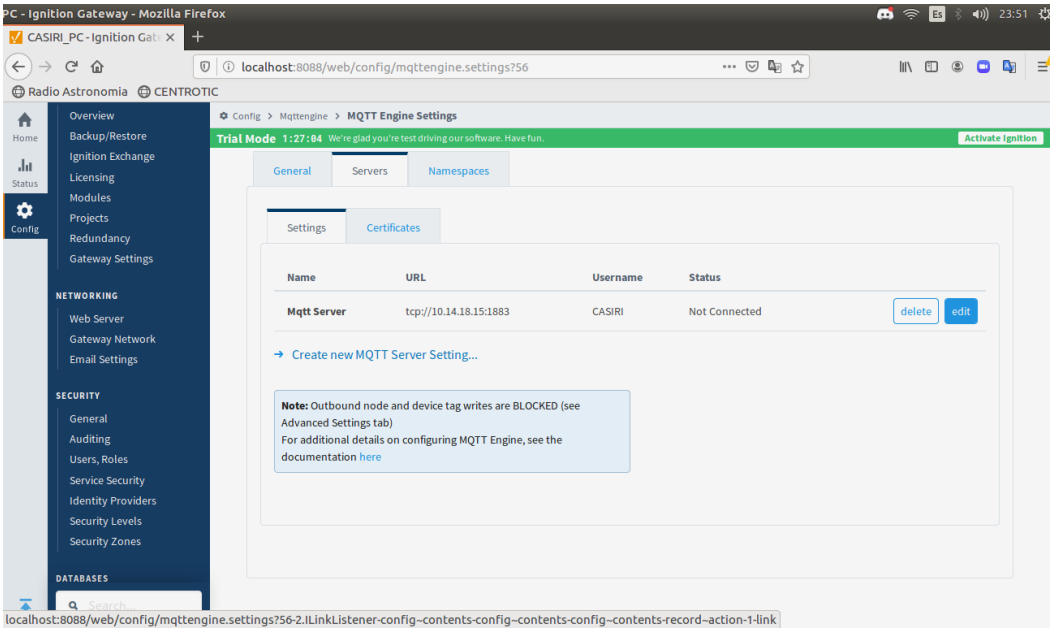
3.2. Enlace del módulo MQTT Engine y el servidor Mosquitto

- Abrir una terminar del sistema presionando las teclas “Ctrl+Alt+T”.
- Ejecutar la instrucción “ifconfig” para obtener la ip de la computadora.

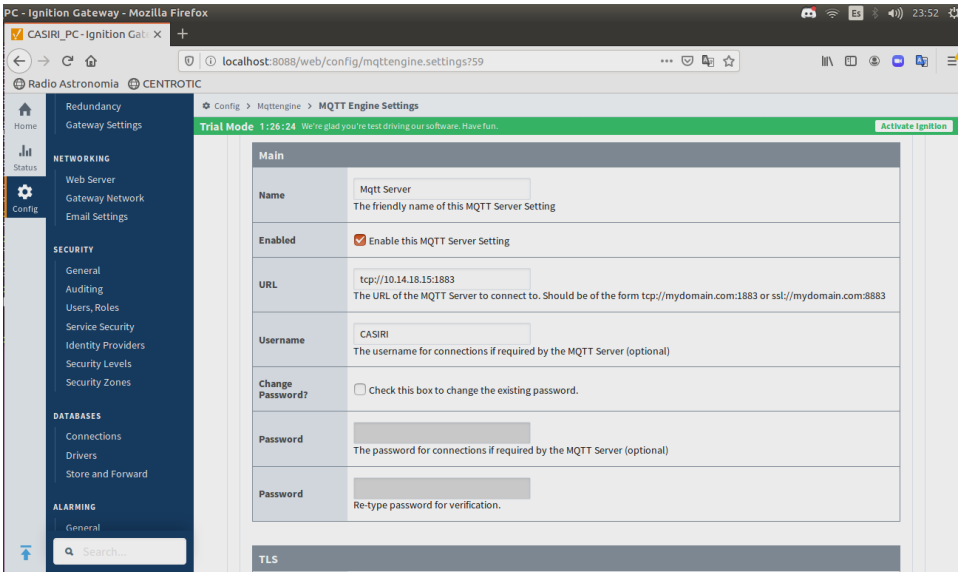


La IP subrayada corresponde a la de la computadora, tener en cuenta que cada vez que se conecte a internet esta dirección va a cambia. Se guarda la IP en el portapapeles con "**Ctrl+Shift+C**" para ser usada en los siguientes pasos.

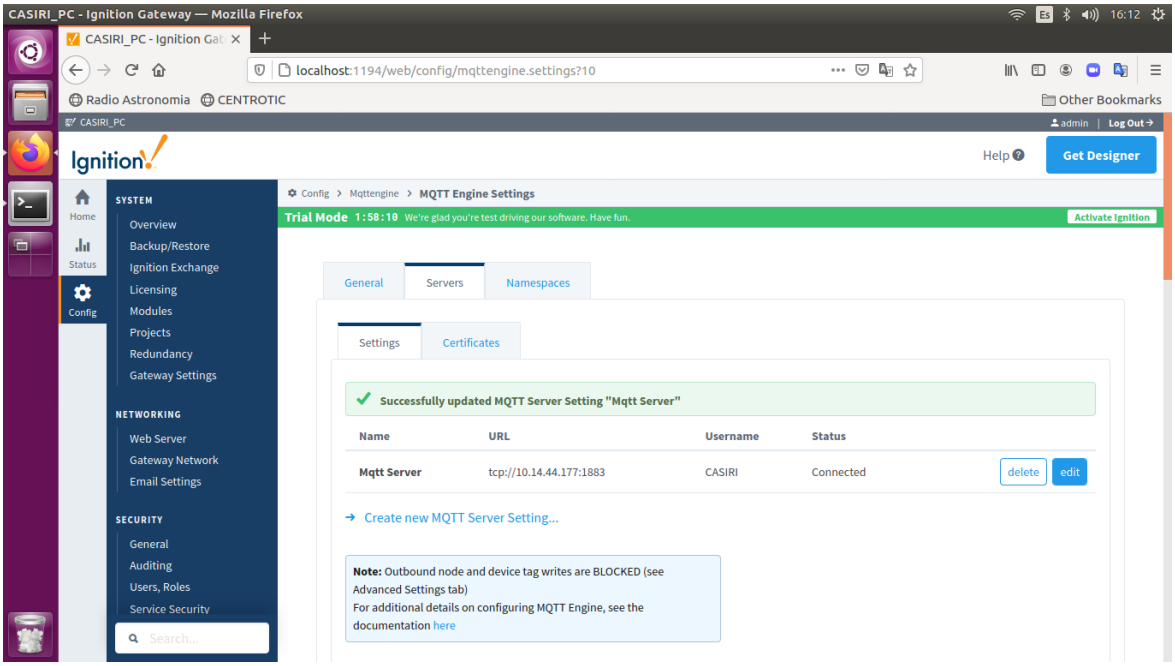
- Abrir el navegador Firefox y colocar en el buscador "**localhost:1194**" para abrir la puerta de enlace, una vez ingresadas las credenciales de acceso dirigirse a la configuración y abrir las opciones de servidores del módulo MQTT Engine.



- Si la dirección IP de la computadora no es la misma que la de la URL del servidor se debe cambiar la configuración para realizar la conexión con Mosquitto.



- Una vez cambiada la dirección IP en la URL se comprueba el estado del servidor en conectado. De esta manera ya se puede pasar al envío de datos.



- Una vez conectado cerrar las terminales usadas.

3.3. Toma de datos

- Abrir una terminal presionando las teclas “Crtl+Alt+T”.
- Ejecutar la instrucción “source estacion_ambiental/bin/activate”.
- En la misma terminal ejecutar “cd /home/comdiguis/Desktop/CASIRI/ESTACION/cods/”
- Ejecutar el comando “gedit estación.py”.
- Una vez abierto el archivo dirigirse a la línea 32, borrar la dirección IP y colocar la usada en los pasos anteriores.

- Por último, guardar la edición y cerrar para ejecutar en el terminal anterior el comando “python estacion.py” de esta manera ya se está enviando de forma periódica los datos de los sensores ambientales de CASIRI.

NOTA: No cerrar la terminal donde se está ejecutando el código estacion.py.

3.4. Visualización

3.4.1. Visualización en computadora local

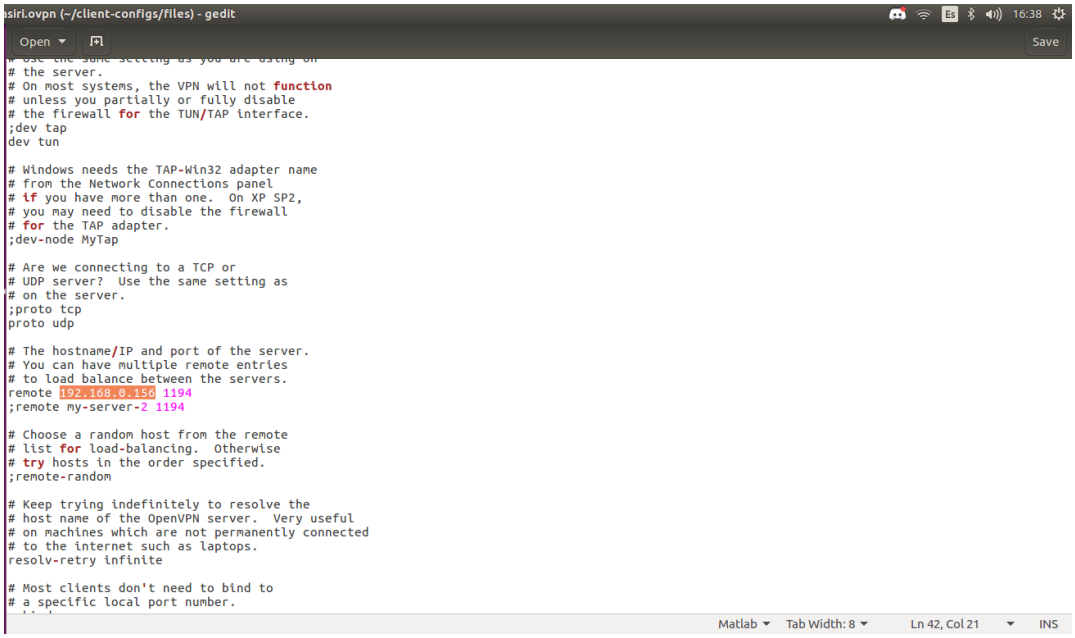
- Abrir el navegador Firefox y escribir en la barra de búsqueda localhost:1194, esto lo redirigirá para ingresar a la Gateway de Ignition.

- Ingresar con el usuario y contraseña proporcionado en el repositorio. Comprobar si la versión TRIAL esta activada.
- Abrir una pestaña en el navegador y colocar en la barra de búsqueda http:localhost:1194/data/perspective/login/proyecto". Ingresar con las credenciales del proyecto.

3.4.2 Visualización en cliente con computadora Windows

Los siguientes pasos se deben ejecutar en la computadora del proyecto:

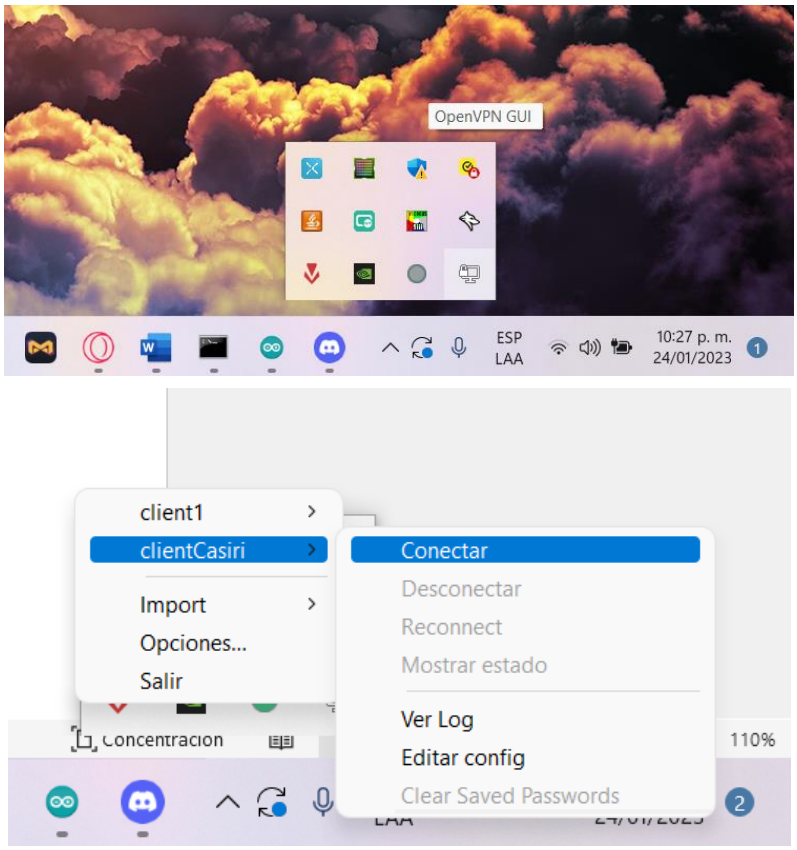
- Abrir el navegador Firefox y escribir en la barra de búsqueda localhost:1194, esto lo redirigirá para ingresar a la Gateway de Ignition
- Ingresar con el usuario y contraseña proporcionado en el repositorio. Comprobar si la versión TRIAL esta activada.
- Tener a mano la dirección IP de la computadora.
- Abrir una terminal con el comando "Ctrl+Alt+T" y ejecutar los siguientes comandos.
- Ejecutar "sudo systemctl status openvpn", comprobar que el servicio este corriendo, de no ser así, ejecutar "systemctl start openvpn" y volver a verificar el estado del servicio. Para finalizar el proceso de status presionar "Ctrl+C".
- En la misma terminal ejecutar el comando "cd /home/comdiguis/client-configs/files/" al momento de estar en la carpeta files ejecutar "gedit clientCasiri.ovpn" y en la línea 42 cambiar la dirección IP por la actual de la computadora.



- Guardar y cerrar el archivo para enviarlo a la computadora en la que se desea visualizar la interfaz.

Los siguientes pasos se realizan en la computadora del cliente.

- Copiar el archivo clientCasiri.ovpn en la carpeta config de la instalación del programa OpenVPN en la carpeta config.
- Ejecutar el programa OpenVPN, y buscar en los iconos ocultos el programa y seleccionar el cliente para luego conectar. Si la dirección IP en los pasos anteriores es la misma de la computadora principal el VPN se conectaría.



NOTA: Por ahora se puede realizar la conexión entre los maquina en la misma red wifi.

- Una vez conectado el servicio se abre el navegador de preferencia y en la barra de búsqueda colocar "http:**IP_COMPUTADORA_CASIRI**:1194/data/perspective/login/proyecto". Como resultado se obtiene la pagina de la interfaz, donde el siguiente paso es ingresar las credenciales de acceso para visualizar los datos.

Capítulo 4

4 Repositorio

Los entregables de este proyecto corresponden al disco dura de 500GB que contiene la imagen de CASIRI junto con una memoria USB para inicial el software Clonezilla.