Sistema de monitoreo de señales de radio basado en SDR y Node-Red

Elian Mauricio Rojas Poveda y Jhoan Sebastian Bernal Villamizar

Trabajo de Grado para Optar el Título de Ingeniero Electrónico

Director

Homero Ortega Boada

Doctor en Ciencias de la Ingeniería, Radiocomunicaciones

Codirector

Andrés Felipe Rubio Toloza

Ingeniero electrónico

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Ingeniería Eléctrica, Electrónica y de Telecomunicaciones

Bucaramanga

2025

Tabla de contenido

Lista de figuras

lista de tablas

lista de apéndices

(Pueden ser consultados en la base de datos de la biblioteca)

Resumen

Este trabajo de grado mejora la plataforma CASIRI, diseñada inicialmente para la caracterización de sitios para radioastronomía mediante la medición de variables medioambientales. El problema identificado fue la falta de mediciones del espectro radioeléctrico, esencial para la radioastronomía. El proyecto propone integrar un radiotelescopio educativo basado en SDR y GNU Radio al sistema CASIRI, permitiendo ahora la medición remota del espectro, incluso sin conexión a internet. Para ello, se desarrolló la aplicación IoTVirtualCloud, encargada de monitorear los archivos de espectro generados y, al detectar conexión, consume un webservice para subir estos archivos a Google Drive. Se implementó también el webservice "SpectrumViewer", una webapp que visualiza las mediciones de espectro almacenadas en la nube, integrada en Node-RED. Una característica clave del proyecto es la implementación de un gemelo digital, lo que permitió no solo la medición y visualización remota del espectro, sino también el control de los elementos de radioastronomía, como los parámetros del equipo SDR. Para ello, se desarrolló EBSDRADMIN, un sistema de control remoto integrado con Node-RED.

Como resultado, CASIRI ahora permite medir, visualizar y controlar remotamente el espectro radioeléctrico, representando un avance significativo para la radioastronomía y otras aplicaciones de medición de espectro.

*Proyecto de grado

** Facultad de Ingenierías Fisicomecánicas. Escuela de Ingeniería Eléctrica, Electrónica y de Telecomunicaciones. Director: Homero Ortega Boada (Título Académico)

Abstract

This thesis improves the CASIRI platform, initially designed for the characterization of radio astronomy sites by measuring environmental variables. The problem identified was the lack of radio spectrum measurements, essential for radio astronomy. The project proposes integrating an educational radio telescope based on SDR and GNU Radio into the CASIRI system, now allowing remote spectrum measurements, even without an internet connection. To this end, the IoTVirtualCloud application was developed, which monitors the generated spectrum files and, upon detecting a connection, consumes a web service to upload these files to Google Drive. The "SpectrumViewer" web service was also implemented, a web application that visualizes spectrum measurements stored in the cloud, integrated into Node-RED.

A key feature of the project is the implementation of a digital twin, which enabled not only remote spectrum measurement and visualization, but also the control of radio astronomy elements, such as SDR equipment parameters. To this end, EBSDRADMIN, a remote control system integrated with Node-RED, was developed. As a result, CASIRI now enables the remote measurement, visualization, and control of the radio spectrum, representing a significant advance for radio astronomy and other spectrum measurement applications.

*Degree project

** Faculty of Physical and Mechanical Engineering. School of Electrical, Electronic, and Telecommunications Engineering. Director: Homero Ortega Boada (Academic Degree)

Introducción

. La radioastronomía, como disciplina científica, se ha encargado de estudiar el universo mediante la detección de ondas de radio emitidas por fenómenos cósmicos distantes, como estrellas, galaxias y otros cuerpos celestes. A lo largo de los años, esta área ha permitido grandes avances en el entendimiento del origen del cosmos y los fenómenos que lo atraviesan. Las ondas de radio proporcionan una ventana única al universo, ya que, a diferencia de la luz visible, pueden atravesar nubes de gas y polvo que bloquean las observaciones ópticas. Sin embargo, a pesar de su potencial, la radioastronomía enfrenta múltiples retos tecnológicos, particularmente en la captura y procesamiento de señales débiles provenientes del espacio profundo. La interferencia de señales de origen humano, como las emitidas por dispositivos electrónicos y redes de comunicación, constituye un obstáculo importante para obtener mediciones precisas, especialmente en áreas geográficas cercanas a centros urbanos o zonas industriales.

Para superar estos desafíos, han emergido tecnologías innovadoras, como los sistemas de radio definida por software (SDR), que permiten a los investigadores modificar los parámetros de la recepción y el procesamiento de señales sin la necesidad de realizar cambios en el hardware. Los SDR se distinguen por su capacidad de ser reconfigurados a través de software, lo que brinda una mayor flexibilidad y capacidad de adaptación frente a diferentes frecuencias y tipos de señales, todo sin depender de equipos costosos y difíciles de modificar. Esta flexibilidad ha revolucionado no solo la radioastronomía, sino también otras áreas de la comunicación inalámbrica, permitiendo una mayor accesibilidad a herramientas de medición de espectro en tiempo real, desde la educación hasta la industria.

En este contexto, el proyecto de grado presentado ayudara a aumentar las capacidades del sistema CASIRI, que ha sido desarrollado por los grupos de investigación CEMOS, RadioGIS y GISEL de la Universidad Industrial de Santander. CASIRI (Sistema de Caracterización de Sitios en Radio Interferencias) es una plataforma dedicada al monitoreo y la caracterización de interferencias radioeléctricas en la radioastronomía, a través de la medición y gestión de variables ambientales. Este sistema fue diseñado inicialmente para evaluar y mitigar las interferencias generadas por actividades humanas que podrían comprometer la precisión de las mediciones espectrales. Además, CASIRI se utiliza para gestionar los datos de estaciones meteorológicas, lo cual es crucial para las observaciones en entornos al aire libre donde las condiciones climáticas pueden afectar las mediciones.

A pesar de los logros de CASIRI en la gestión de mediciones medioambientales, uno de los principales problemas del sistema es que no ha sido capaz de medir el espectro radioeléctrico, una capacidad esencial para la radioastronomía. La medición del espectro radioeléctrico es crucial porque permite a los investigadores estudiar las señales de radio provenientes de fuentes astronómicas y comprender fenómenos como la emisión de hidrógeno neutro, la radiación cósmica de fondo y otros eventos celestes que solo pueden ser detectados en el dominio de las frecuencias. Este tipo de medición no solo es fundamental para la radioastronomía, sino también para aplicaciones industriales y científicas más amplias que requieren un análisis preciso del espectro en diversas frecuencias.

Por lo tanto, el objetivo del proyecto es ampliar las capacidades de CASIRI para incluir la medición del espectro radioeléctrico mediante la integración de un radiotelescopio educativo basado en SDR y GNU Radio, complementado con una solución remota para el almacenamiento

y visualización de los datos obtenidos. Esto permitirá que CASIRI no solo continúe con su función de medición medioambiental, sino que también brinde herramientas poderosas para la radioastronomía al incorporar la medición del espectro, la visualización remota de datos y el control de dispositivos de manera remota.

Para abordar este reto, la solución propuesta se basa en varias tecnologías clave que se integran de manera innovadora. En primer lugar, se utiliza el radiotelescopio educativo basado en SDR y GNU Radio, que es un prototipo de bajo costo desarrollado previamente en otros proyectos de investigación. El radiotelescopio educativo permite capturar señales de radio de diferentes frecuencias del espectro, procesarlas y generar datos espectrales en tiempo real. La ventaja de usar SDR es que el sistema se puede reconfigurar a través de software para adaptarse a diferentes bandas de frecuencia, lo cual lo hace altamente flexible y adecuado para investigaciones en radioastronomía.

El bloque clave de la solución es la creación de una aplicación en Python denominada IoTVirtualCloud. Esta aplicación tiene como función principal supervisar la conexión a internet y subir los archivos de medición del espectro (en formato CSV) a la nube (específicamente a Google Drive). Esto resuelve el desafío de trabajar en entornos donde la conectividad es intermitente o no está disponible, ya que la aplicación almacena temporalmente los archivos en el equipo local y solo los sube cuando se detecta conexión. Este enfoque permite operar en lugares remotos sin perder la capacidad de almacenar y procesar datos a medida que se establezca la conectividad.

Adicionalmente, se implementó el webservice "SpectrumViewer", una webapp diseñada para visualizar las mediciones de espectro almacenadas en la nube. La webapp ofrece una interfaz gráfica sencilla que permite a los usuarios acceder a los datos y verlos en diferentes formatos,

como gráficos 2D, 3D y Waterflow, facilitando la interpretación de los resultados de las mediciones. Esta herramienta también ha sido integrada dentro de la solución de Node-RED, un sistema de programación visual que se utiliza para gestionar los flujos de datos, y que permite a los usuarios interactuar con el sistema desde una interfaz de usuario amigable.

Una de las características más innovadoras del proyecto es la presentación de la solución como un prototipo de gemelo digital. El concepto de gemelo digital hace referencia a la creación de una representación virtual de un sistema físico, lo que permite realizar un monitoreo, control y simulación de este sistema en tiempo real. En este caso, el gemelo digital no solo representa el radiotelescopio, sino también la infraestructura de medición y los dispositivos que componen CASIRI, integrando el control de estos dispositivos a través de Node-RED.

El uso de este gemelo digital ha permitido abordar un desafío adicional del proyecto: el control remoto de los dispositivos de radioastronomía, como los parámetros de los equipos SDR, la antena y otros elementos. Para resolver este desafío, se desarrolló el producto denominado EBSDRADMIN, una aplicación en Python que permite controlar de forma remota los dispositivos SDR mediante la interfaz de Node-RED. EBSDRADMIN no solo facilita la gestión remota de los parámetros de los dispositivos, sino que también asegura que los usuarios puedan realizar cambios en los parámetros de medición de manera sencilla y eficiente.

Los resultados esperados de este proyecto son diversos y de gran relevancia tanto para la radioastronomía como para el monitoreo de espectro en general. En primer lugar, la integración del radiotelescopio educativo con CASIRI permitirá ahora medir y visualizar el espectro radioeléctrico, lo que representa un avance significativo en la radioastronomía, proporcionando herramientas más accesibles y económicas para las investigaciones. Además, el uso de tecnologías

como SDR, Node-RED, GNU Radio y webservices abre la puerta a futuras ampliaciones del sistema, permitiendo no solo la visualización y el control remoto del espectro, sino también la expansión de capacidades para otros tipos de mediciones.

El impacto potencial de esta solución va más allá de la radioastronomía, ya que la capacidad de medir y visualizar el espectro de manera remota tiene aplicaciones en áreas como las comunicaciones inalámbricas, el monitoreo ambiental y las prácticas educativas. Esta plataforma no solo contribuye al avance de la ciencia, sino que también democratiza el acceso a herramientas de medición espectral, haciendo posible que instituciones con menos recursos puedan participar en investigaciones de vanguardia.

1. Marco Teórico

Este capítulo tiene como objetivo proporcionar los fundamentos esenciales que permitirán al lector comprender de manera fluida los contenidos del informe. En lugar de ofrecer una revisión exhaustiva de todos los conceptos, nos centraremos en aquellos que son clave para entender la solución propuesta, así como su contexto y aplicaciones.

1.1 Radioastronomía y la importancia del monitoreo espectral

La radioastronomía es una rama de la astronomía que se dedica al estudio de señales de radio emitidas por objetos astronómicos, como estrellas, galaxias y otros cuerpos celestes. A diferencia de la astronomía óptica, que utiliza la luz visible, la radioastronomía permite explorar fenómenos que no pueden ser observados con telescopios convencionales. Las señales de radio nos ofrecen información crucial sobre los procesos físicos y químicos que ocurren en el universo, como la formación de estrellas y galaxias, la distribución del gas en el espacio y los fenómenos magnéticos de los planetas.

Para poder analizar estas señales, es esencial contar con instrumentos de medición espectral, que permiten capturar, procesar y analizar las frecuencias de radio. Sin embargo, el monitoreo del espectro radioeléctrico presenta desafíos técnicos, principalmente debido a la interferencia de señales de origen humano, como las emisiones de telecomunicaciones, dispositivos electrónicos y otras fuentes de radiación artificial. Este tipo de interferencia puede distorsionar las mediciones, por lo que es necesario contar con soluciones tecnológicas que ayuden a minimizarla (Complex, 2023).

1.2 CASIRI y la caracterización de interferencias radioeléctricas

CASIRI, que significa Sistema de Caracterización de Sitios en Radio Interferencias, es una plataforma que originalmente se diseñó para caracterizar sitios en los cuales se realizan observaciones astronómicas. Su función principal es evaluar y mitigar las interferencias radioeléctricas que podrían afectar la calidad de las mediciones de espectro. CASIRI se enfoca en la gestión de datos provenientes de estaciones meteorológicas, lo que resulta fundamental para monitorear las condiciones ambientales en áreas remotas donde se encuentran los telescopios.

1.3 Radio Definida por Software (SDR) y GNU Radio

Para poder realizar la medición del espectro de radio de manera efectiva, es necesario implementar tecnologías que permitan capturar y procesar señales de radio en tiempo real. Una de las tecnologías clave para este propósito es la Radio Definida por Software (SDR). Los SDR son sistemas que reemplazan los componentes de hardware tradicionales de los radios por software, lo que proporciona flexibilidad. En lugar de depender de circuitos analógicos para la recepción y procesamiento de señales, los SDR usan programas que pueden ser configurados para ajustar diferentes parámetros, como la frecuencia de operación o el ancho de banda, sin la necesidad de modificar el hardware. (Nationals-instruments, 2023)

GNU Radio es un entorno de desarrollo de código abierto que permite diseñar y simular sistemas de comunicación de radio en tiempo real, utilizando SDR. Esta plataforma se utiliza para procesar las señales recibidas y generar las mediciones espectrales. La flexibilidad de GNU Radio permite crear flujos de trabajo personalizables, adaptados a las necesidades específicas de la

radioastronomía. Gracias a su integración con SDR, se puede obtener información sobre el espectro de radio de manera más eficiente y económica que con los métodos tradicionales.

1.4 Node-RED y la integración de tecnologías en la nube

Para gestionar y visualizar los datos obtenidos por el radiotelescopio, se utilizó la plataforma Node-RED, que es una herramienta de programación visual basada en JavaScript. Node-RED facilita la conexión de dispositivos a través de nodos que permiten transmitir datos, procesarlos y visualizarlos. Esta plataforma es especialmente útil en proyectos IIoT (Internet Industrial de las Cosas), como el de este trabajo, donde se requiere integrar varios sistemas y dispositivos de medición en tiempo real. (Node-Red, s.f.)

En la solución propuesta, Node-RED se utiliza para gestionar los flujos de datos provenientes del radiotelescopio y las mediciones espectrales, integrando tanto las aplicaciones locales como las basadas en la nube. Mediante la implementación de webservices, se permite que los datos procesados se suban a la nube, específicamente a Google Drive, para su almacenamiento y análisis posterior. Además, Node-Red proporciona una interfaz visual que facilita la interacción remota con los sistemas, permitiendo a los usuarios controlar los parámetros del radiotelescopio y visualizar los resultados sin estar físicamente en el lugar de las mediciones.

1.5 Web Service

Un *web service* es un servicio en línea que permite a diferentes aplicaciones comunicarse entre sí a través de la web. Usando protocolos estándar, como HTTP y SOAP o REST, los web services permiten que aplicaciones construidas en diferentes plataformas o lenguajes de

programación intercambien datos de manera sencilla y eficiente. En el contexto de este proyecto, un web service es utilizado para permitir que los archivos de mediciones espectrales, almacenados localmente, sean subidos a la nube. A través de este servicio, los archivos se transmiten de manera segura y se almacenan en plataformas como Google Drive, desde donde pueden ser consultados y visualizados a través de herramientas como la *webapp SpectrumViewer*.

El concepto de *web service* es clave en la construcción de soluciones escalables y flexibles, ya que permite la integración de sistemas y aplicaciones distribuidas, sin la necesidad de realizar modificaciones directas en los sistemas o plataformas de cada usuario. En este proyecto, los web services facilitan la interoperabilidad entre el sistema de medición local (GNU Radio) y las herramientas de visualización en la nube, asegurando que los datos puedan ser accesibles desde cualquier lugar con conexión a internet. (De Zúñiga, 2023,)

1.6 Protocolos de Comunicación: MQTT y API

El protocolo Message Queuing Telemetry Transport (MQTT) es un protocolo de mensajería ligero y eficiente diseñado para la comunicación entre dispositivos remotos, especialmente útil en entornos con conectividad limitada o intermitente. MQTT permite que los dispositivos envíen y reciban datos de forma rápida y segura mediante un modelo de publicación-suscripción, lo que facilita la comunicación bidireccional. En este proyecto, se utiliza MQTT para permitir el control remoto de dispositivos, como los parámetros del SDR en el radiotelescopio, y la transmisión de datos espectrales entre el equipo de medición y las plataformas de visualización en la nube, asegurando así la interoperabilidad entre los sistemas locales y remotos. (Amazon_Web_Services, s.f.)

Por otro lado, una Interfaz de Programación de Aplicaciones (API) es un conjunto de protocolos que permite que diferentes aplicaciones interactúen entre sí. A través de una API, las aplicaciones pueden enviar y recibir datos de manera estándar, facilitando la integración de diferentes tecnologías sin necesidad de modificar los sistemas existentes. En este proyecto, se utilizan APIs para integrar el sistema de medición de espectro con plataformas de almacenamiento en la nube (Google Drive) y con las aplicaciones de visualización de los datos espectrales, lo que permite a los usuarios acceder a los resultados de las mediciones desde cualquier lugar con conexión a internet. (Google_Developes, s.f.)

1.7 Gemelo digital

Un gemelo digital es una representación virtual de un sistema físico que permite su monitoreo, simulación y control en tiempo real. En este caso, el gemelo digital incluye tanto al radiotelescopio como a los dispositivos de medición integrados en el sistema CASIRI. Esta integración permite simular y controlar las mediciones de espectro de manera remota, lo que amplía las capacidades de CASIRI sin necesidad de estar físicamente en el lugar donde se realiza la medición. (IBM, s.f.).

2. Desarrollo de la solución

2.1 Fases de desarrollo

2.1.1 Análisis de Requisitos solicitados

Los requisitos fundamentales para el desarrollo de este proyecto se centran en la necesidad de realizar mediciones del espectro radioeléctrico en lugares remotos, donde las condiciones de conectividad son limitadas o inexistentes. Esta situación plantea el desafío de poder controlar

remotamente el proceso de adquisición y visualización de los datos sin depender de una conexión continua a internet. La solución propuesta permite operar tanto localmente en el equipo de medición, como de manera remota a través de Node-RED, usando una interfaz de *dashboard*. Esta interfaz permite a los usuarios ajustar los parámetros del diagrama de bloques de GNU Radio (el cual es generado como un archivo .py) sin necesidad de modificar directamente el flujo de trabajo, garantizando flexibilidad y control sin intervenir en el proceso técnico subyacente.

Además, otro requisito importante fue la integración con un radiotelescopio diseñado para medir señales a 1420 MHz. La solución también permite acceder rápidamente al historial de datos y mediciones anteriores desde la misma interfaz, facilitando así la consulta y la continuidad de las observaciones. Este sistema está diseñado para ser escalable, permitiendo que futuros proyectos puedan integrarse fácilmente en el sistema, sin necesidad de realizar modificaciones complejas.

En la **figura 1** se muestra la interfaz del menú en el *dashboard* de Node-Red, en donde se visualiza lo descrito anteriormente detallando que cada una tiene su respectivo I D.

figura 1

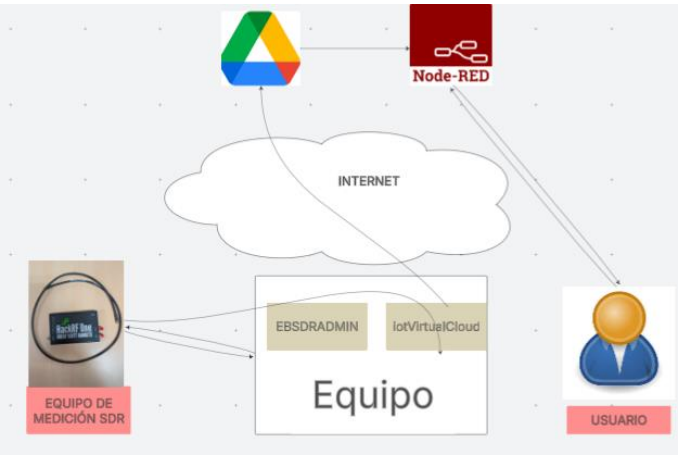
Menú del dashboard de Node-Red

2.1.2 Diseño arquitectónico

Para abordar las necesidades descritas anteriormente, se propuso la solución vista en la **figura 2**, este enfoque se centra en poder manejar la información producida por el equipo de medición SDR. En este contexto, se utilizó el programa de GNU radio utilizado por el proyecto anterior, en el cual a través de bloques de GNU radio genera un archivo CSV con la información obtenida de la antena y el radio definido por *sofwtare* (SDR). Para conocer de mejor manera el funcionamiento y la explicación de este programa busca la documentación (RadioGis, 2024).

figura 2

Diagrama base de la solución



Ya teniendo la información almacenada en un archivo CSV, este se conecta con los 2 programas. El programa IoTVirtualCloud, se encarga de controlar y vigilar si el equipo en uso tiene acceso a internet, en caso de tener internet este lo sube a la nube (Google drive), mientras que el programa EBSDRADMIN es un sistema de control para dispositivos SDR que permite gestionar equipos GNU Radio tanto localmente como de forma remota mediante comunicación **MQTT**, facilitando el ajuste de parámetros clave como frecuencia central y ancho de banda.

Usando un *broker* MQTT (como Mosquitto), el programa se integra con **Node-RED** para habilitar control remoto y monitoreo en tiempo real. Más adelante se explicará más a detalle el funcionamiento de estos programas y sus respectivos anexos y guías para su instalación.

Una vez en la nube, se busca acceder a esta información mediante *web services*, *web hooks* y otras posibilidades que ofrece la nube. Se ha seleccionado Node-RED como la herramienta para procesar los datos, debido al uso del proyecto pasado (González Mendoza & Silva Sepúlveda, 2023) y así poderse complementar mutuamente, y de esta manera desarrollar una interfaz, permitiendo a los usuarios interactuar con los datos y visualizarlos de manera remota. Para lograr esto, se planteó la necesidad de contar con una IP pública, y para ello, se consideró utilizar una máquina virtual proporcionada por Microsoft llamada Azure. En esta máquina virtual, se pueden utilizar servicios como Node-RED, SQL, Mosquito, entre otros servicios en la nube.

En Node-Red, se planifica acceder a la interfaz para poder controlar de manera remota el programa, de esta manera poder controlar el programa a través de esta aplicación, de igual manera en Node-RED se podrá visualizar el espectro tomado por la antena y se podrá ver diferentes aspectos de este. De igual manera más adelante se explicará más a detalle el funcionamiento de esta solución planteada

Al acceder a la IP pública de la máquina virtual en el puerto de Node-RED y su dashboard, los usuarios podrán interactuar con la HMI de forma remota y hacer uso de los servicios que necesitan. Este enfoque integrado busca proporcionar una solución completa y accesible para satisfacer las diversas necesidades de los usuarios en el ámbito de la radioastronomía.

2.2 Solución acotada a los requerimientos del proyecto

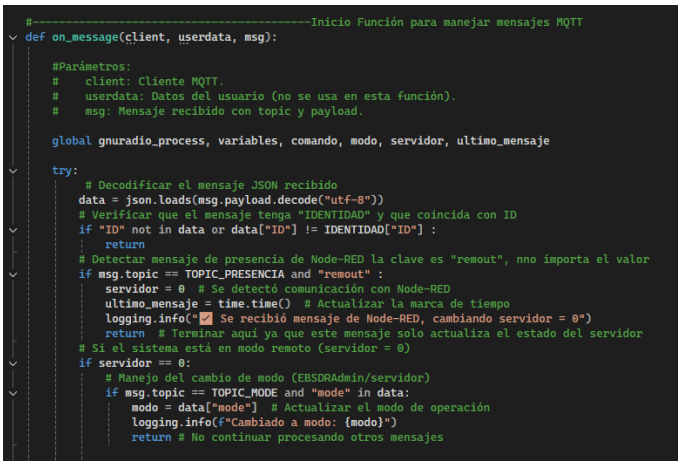
2.2.1 Desarrollo del sistema EBSDRADMIN

El desarrollo del sistema EBSDRADMIN comienza creando la estructura básica en Python, importando las librerías esenciales como paho-mqtt para la comunicación MQTT, tkinter para la interfaz gráfica, *subprocess* para ejecutar GNU Radio, y *threading* para manejar operaciones concurrentes. Inmediatamente configuramos el sistema de *logging* con diferentes niveles de prioridad (DEBUG, INFO, WARNING, ERROR, CRITICAL) que permite monitorear el comportamiento del sistema tanto en desarrollo como en producción. Luego se carga la configuración desde un archivo JSON que contiene las rutas importantes como python_gnuradio y script_gnuradio, el directorio base para almacenamiento, y los parámetros de conexión MQTT incluyendo el broker y puerto, además de un identificador único para manejar múltiples instancias.

El módulo MQTT es el encargado de implementar una comunicación bidireccional mediante cuatro tópicos de recepción: gnuradio/control para actualizar variables como frecuencia y ancho de banda, gnuradio/command para enviar comandos *start/stop,* gnuradio/mode para cambiar entre modos local/remoto, y gnuradio/presencia que actúa como *heartbeat* para verificar la conexión. Por otro lado, tenemos tres tópicos de publicación que envían el estado de ejecución, las variables actuales y el modo operativo. Este módulo incluye mecanismos avanzados como reconexión automática cada 15 segundos, cambio a modo local cuando detecta fallos, y validación de mensajes mediante un ID único para seguridad. En la **figura 3** se puede visualizar la parte del código para su funcionamiento.

figura 3

Código de Python función MQTT

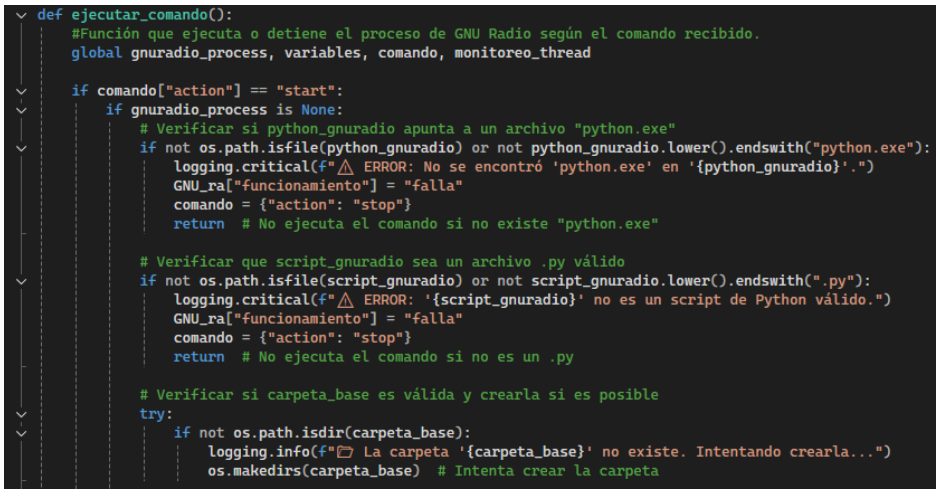


```python
#----------------------------------------Inicio Función para manejar mensajes MQTT
def on_message(client, userdata, msg):

    #Parámetros:
    #    client: Cliente MQTT.
    #    userdata: Datos del usuario (no se usa en esta función).
    #    msg: Mensaje recibido con topic y payload.

    global gnuradio_process, variables, comando, modo, servidor, ultimo_mensaje

    try:
        # Decodificar el mensaje JSON recibido
        data = json.loads(msg.payload.decode("utf-8"))
        # Verificar que el mensaje tenga "IDENTIDAD" y que coincida con ID
        if "ID" not in data or data["ID"] != IDENTIDAD["ID"] :
            return
        # Detectar mensaje de presencia de Node-RED la clave es "remout", nno importa el valor
        if msg.topic == TOPIC_PRESENCIA and "remout" :
            servidor = 0  # Se detectó comunicación con Node-RED
            ultimo_mensaje = time.time()  # Actualizar la marca de tiempo
            logging.info("✅ Se recibió mensaje de Node-RED, cambiando servidor = 0")
            return  # Terminar aquí ya que este mensaje solo actualiza el estado del servidor
        # Si el sistema está en modo remoto (servidor = 0)
        if servidor == 0:
            # Manejo del cambio de modo (EBSDRAdmin/servidor)
            if msg.topic == TOPIC_MODE and "mode" in data:
                modo = data["mode"]  # Actualizar el modo de operación
                logging.info(f"Cambiado a modo: {modo}")
                return # No continuar procesando otros mensajes
```

Para el control de GNU Radio, se implementó un sistema que ejecuta los procesos mediante subprocess.Popen(), realizando validaciones previas como verificar la existencia de python.exe y los scripts, comprobar permisos en la carpeta base, y asegurar el formato correcto de los parámetros numéricos. El sistema monitorea constantemente el proceso de GNU Radio, detectando cierres inesperados y actualizando automáticamente los estados a través de MQTT. La interfaz gráfica, desarrollada con Tkinter, muestra campos para los nueve parámetros configurables que incluyen valores técnicos como BW_hackRF y f_centro, datos geográficos como latitud y longitud, y parámetros operacionales como descripción y nubeVirtual, implementando validaciones en tiempo real para caracteres prohibidos en nombres de archivo y formato numérico, incluyendo notación científica, además de verificar todos los datos antes de permitir la ejecución, como se muestra en la **figura 4** .

figura 4

Codigo de Python para el control de GNUradio



```python
def ejecutar_comando():
    #Función que ejecuta o detiene el proceso de GNU Radio según el comando recibido.
    global gnuradio_process, variables, comando, monitoreo_thread

    if comando["action"] == "start":
        if gnuradio_process is None:
            # Verificar si python_gnuradio apunta a un archivo "python.exe"
            if not os.path.isfile(python_gnuradio) or not python_gnuradio.lower().endswith("python.exe"):
                logging.critical(f"⚠ ERROR: No se encontró 'python.exe' en '{python_gnuradio}'.")
                GNU_ra["funcionamiento"] = "falla"
                comando = {"action": "stop"}
                return  # No ejecuta el comando si no existe "python.exe"

            # Verificar que script_gnuradio sea un archivo .py válido
            if not os.path.isfile(script_gnuradio) or not script_gnuradio.lower().endswith(".py"):
                logging.critical(f"⚠ ERROR: '{script_gnuradio}' no es un script de Python válido.")
                GNU_ra["funcionamiento"] = "falla"
                comando = {"action": "stop"}
                return  # No ejecuta el comando si no es un .py

            # Verificar si carpeta_base es válida y crearla si es posible
            try:
                if not os.path.isdir(carpeta_base):
                    logging.info(f"🗁 La carpeta '{carpeta_base}' no existe. Intentando crearla...")
                    os.makedirs(carpeta_base)  # Intenta crear la carpeta
```

La lógica de operación dual permite alternar entre dos modos: el modo EBSDRAdmin local que ofrece control completo desde la interfaz Tkinter con almacenamiento local en CSV, ideal para entornos sin conexión; y el modo Servidor remoto que permite el control a través de Node-RED con sincronización bidireccional de estados cuando hay conexión MQTT estable. Para garantizar robustez en entornos adversos, el sistema incorpora tolerancia a fallos de red con buffer de mensajes pendientes y reconexión automática, protección de datos mediante validación de archivos CSV y prevención de corrupción, además de seguridad operativa con confirmación para cierre y detección de procesos zombis. Este diseño integral ofrece control dual local/remoto e integración perfecta con plataformas IoT como Node-RED, utilizando estándares abiertos como MQTT y JSON que facilitan su adaptación a diversos escenarios científicos y futuras expansiones en el campo de la radioastronomía y estudios atmosféricos.

Para conocer con mayor profundidad la instalación, el funcionamiento y las características del EBSDRADMIN, en el anexo A se explica a profundidad cada uno de los procesos a realizar

para el correcto funcionamiento, y para conocer con mayor detalle el código, todo se encuentra en el **anexo A.**

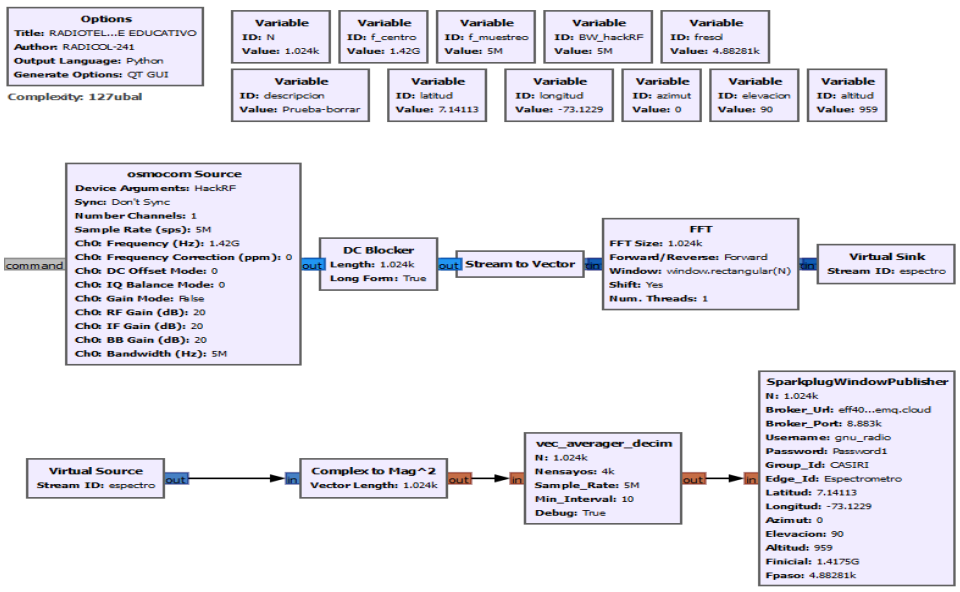
2.2.2 Solución almacenamiento en la nube.

Para la solución implementada en GNU Radio, se utilizó una versión previa desarrollada por el grupo de investigación RadioGIS (Cañizares Carrillo & Guzmán Ortiz, 2024). El SDR y la antena correspondiente al proyecto anterior se conectan al ordenador según el diagrama mostrado en la **figura 5**. La señal recibida será procesada y los datos resultantes se almacenarán en un archivo CSV.

Para una explicación más detallada sobre el funcionamiento de este diagrama, se incluyen los detalles correspondientes en el anexo E, donde también podrán accederse a los archivos relacionados con este proceso.

figura 5

Flujo de GNUradio



El archivo CSV generado se guarda en la carpeta deseada del ordenador utilizado en el momento de la ejecución, en la tabla 1, se ve un ejemplo del archivo generado con sus respectivas variables.

Tabla 1

Archivo CSV generado

Fecha	Latitud	Longitud	Azimut	Elevacion	Altitud	Descripcion	finicial	fpaso	N	Datos
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8324992e-05,2.7722519e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8070563e-05,2.7472666e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.7874157e-05,2.7259897e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8113602e-05,2.7503687e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8619306e-05,2.802314e-(
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.7633632e-05,2.7061913e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8030408e-05,2.7397551e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8276221e-05,2.7674363e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8164062e-05,2.7565704e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.7765187e-05,2.715231e-(
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.7858092e-05,2.7266353e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.894428e-05,2.8328772e-(
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.7648259e-05,2.7076883e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.7983388e-05,2.7351814e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8233258e-05,2.7631924e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.829332e-05,2.7692638e-(
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.7581607e-05,2.7011474e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8052704e-05,2.7418157e
2025-03-25 15:3	7.141879	-73.122193	3	30	5	prueba segun	4117500000	4882.8125	1024	2.8524659e-05,2.7907814e

Las variables usadas son:

- **Fecha**
- **Latitud**
- **Longitud**
- **Azimut**
- **Elevación**
- **Altitud**
- **Descripción**

- **Finicial**
- **Fpaso**
- **N**
- **Datos**

Estas variables deben ser escritas de la misma forma mostrada, con sus respectivas mayúsculas y minúsculas, de otra manera, el programa no funcionará. Además de esto, también se debe tener presente la ubicación del archivo generado y la ubicación del diagrama del diagrama de bloques de la **ilustración 5.** De igual manera, toda la información y la explicación de las variables y su respectiva elección en este proyecto se encuentra en el **apéndice E**

2.2.3 *IoTvirtualcloud*

Ya generado el archivo, se procede a solucionar los casos en donde no haya conexión a internet. Para esto el grupo de investigación radioGis planteó el programa llamado IoTvirtualcloud. Este programa se encarga de vigilar el archivo y verificar si el ordenador está conectado a internet, si está conectado, el archivo se sube automáticamente a la base de datos, en este caso la base de datos es una carpeta estipulada en Google drive. Para conocer más a fondo el funcionamiento y la creación de este módulo, visite en **apéndice D.**

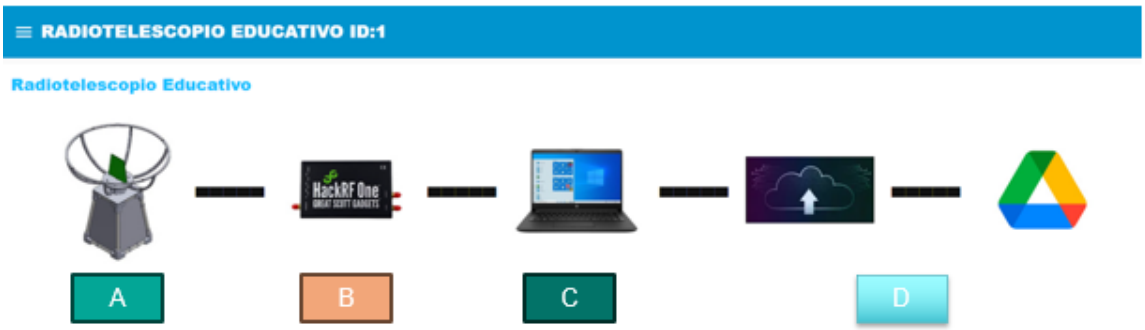
2.2.4 *Desarrollo en Node-Red*

La interfaz en Node-Red a desarrollar debe reunir las partes importantes del proyecto y plantearlas de tal manera que el usuario navegue con facilidad, por lo tanto, se implementó la interfaz mostrada en la **figura 6.** Las cinco imágenes representan cada proceso del problema trabajado, en primer lugar, se encuentra la antena que se encargaran de procesar la señal,

posteriormente tenemos el hackRF, el cual será el panel de control del radiotelescopio educativo, en la tercera imagen se encuentra el equipo de cómputo con el que se toman las mediciones, posteriormente esta información va a la nube y por último va a Google drive que será la base de datos.

figura 6

Interfaz en Node-Red

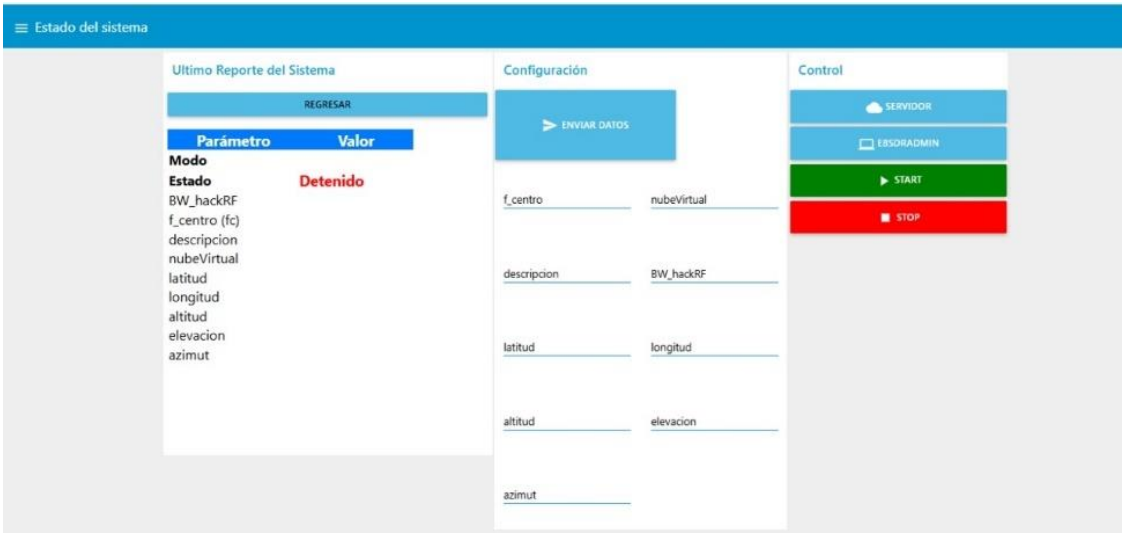


2.2.4.1 Antena. La antena utilizada en este proyecto es el componente principal del radiotelescopio educativo, diseñada para captar ondas de radio emitidas por fuentes astronómicas como el Sol, Júpiter o la Vía Láctea. Su función es recolectar estas señales, que son débiles para ser percibidas por el ojo humano, y convertirlas en datos procesables. A diferencia de los telescopios ópticos, esta antena opera en el espectro radioeléctrico, permitiendo estudiar fenómenos que no son visibles, como emisiones de hidrógeno neutro (línea de 21 cm) o tormentas magnéticas en planetas. La antena a utilizar fue la desarrollada por el grupo de investigación radioGis.

2.2.4.2 HackRF. El desarrollo de esta sección se debe a la necesidad de poder controlar remotamente el sistema desde la herramienta Node-Red. Como se visualiza en la **figura 7,** la interfaz mostrada es similar a la mostrada en **EBSDRADMIN.** La parte central corresponde para la configuración de los parámetros a utilizar en la medición, de igual manera se encuentra el botón de enviar datos**.** En la parte superior derecha se encuentra el modo de control del programa, es decir, en este campo se decide si el control lo tiene el servidor, o lo tiene **EBSDRADMIN** y se encuentran los botones de start o stop**.** En la parte izquierda se encuentra una tabla que muestra los diferentes parámetros que se encuentran en el momento en el programa.

figura 7

HackRF en interfaz de Node-Red



2.2.4.3 Equipo de cómputo. La principal razón de esta sección es mostrar el estado del sistema, es decir, detalla si la conexión en el sistema este encendido o apagada. Esto dependerá exclusivamente del estado de **EBSDRADMIN.** En la **figura 8** se observa un ejemplo de cómo se detalla este estado, en este caso el estado del sistema es **desconectado.**

figura 8

Equipo de Cómputo en interfaz de node-Red

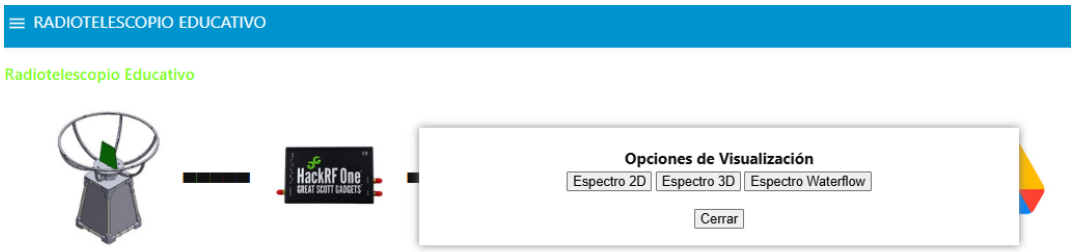
2.2.4.4 Servicio en la nube y Google drive. En este proyecto, Google Drive funciona como el núcleo digital del radiotelescopio educativo, almacenando y organizando todos los datos capturados por la antena. Cuando las ondas de radio del espacio son recibidas y convertidas en espectros de frecuencia, estos se guardan automáticamente en carpetas estructuradas de Google Drive, asegurando que no se pierdan por fallos técnicos locales y permitiendo su acceso desde cualquier lugar con internet. Los estudiantes y profesores pueden visualizar directamente los gráficos de los espectros sin necesidad de descargas complejas, lo que agiliza el análisis de fenómenos astronómicos como emisiones solares.

Además, la visualización del espectro se puede observar en 3 diferentes maneras, en 2d, que muestra la intensidad de la señal en razón de la frecuencia, 3d, que añade un eje adicional (tiempo) para observar cómo varían las frecuencias captadas y *waterflow*, que combina la frecuencia y el tiempo en tonalidades distintas de colores. Estas 3 maneras de ver el espectro hacen

que el análisis y la interpretación de los datos sea más amplio. En la **figura 9** se observa cómo se puede acceder a cada una de ellas.

figura 9

Opciones de visualización del espectro



Si se quiere conocer más a detalle el funcionamiento y los códigos base de cada uno de estos espectros, estos se encuentran en el **apendice B.**

2.3 Resultados

En este apartado, se muestran los resultados obtenidos tras la puesta en marcha y evaluación del sistema, conectándolos con los objetivos específicos establecidos al inicio. Se pretende evidenciar el cumplimiento de los requisitos básicos definidos y examinar cómo los resultados afectan tanto en un ámbito técnico como en uno no técnico.

La ejecución del proyecto logró crear una conexión sólida entre los dispositivos SDR y el sistema CASIRI, lo que facilitó no solo la recopilación de datos espectrales, sino también su análisis, visualización y almacenamiento en la nube. Los hallazgos obtenidos confirman que es posible usar Node-RED como la plataforma principal para unir diversos componentes, mostrando su efectividad en condiciones reales, incluso en situaciones con conectividad inestable como en

las misiones en la Antártida. A continuación, se detallan los logros conseguidos en relación con cada objetivo específico, resaltando tanto los aspectos técnicos como las soluciones aplicadas.

2.3.1 Conexión entre EBSDRADMIN Y NODE-RED

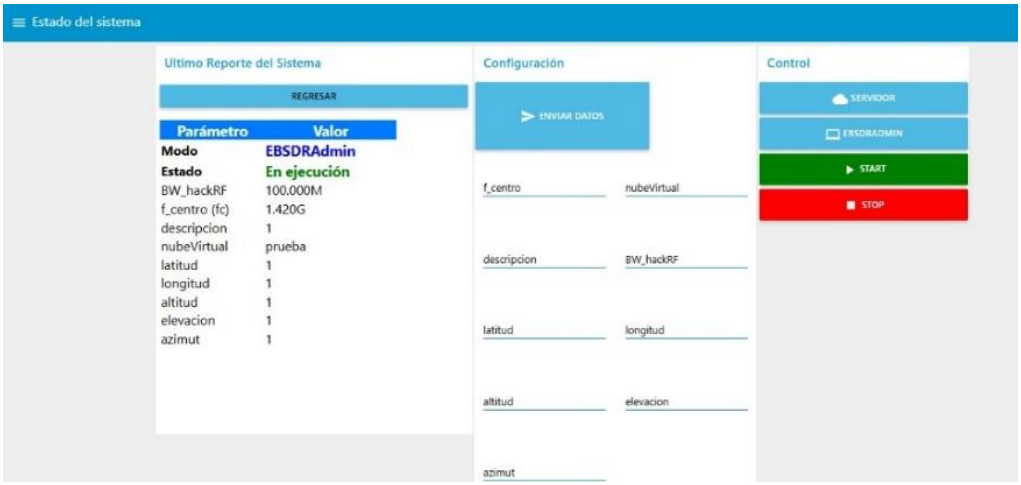
Para asegurar una conexión correcta entre EBSDRADMIN y Node-RED, ambos sistemas se comunican mediante el protocolo MQTT, estableciendo una conexión bidireccional que permite el control remoto de los parámetros del SDR (como la frecuencia central y el ancho de banda) y la recepción de datos en tiempo real.

Para verificar el correcto funcionamiento del proceso, se realizaron pruebas iniciales utilizando EBSDRADMIN para controlar los parámetros del SDR. En un primer paso, se enviaron datos aleatorios a Node-RED, y posteriormente se utilizaron datos específicos del proyecto, lo que permitió asegurar que la integración y transmisión de datos se realizaban de manera correcta.

Ya teniendo claro el funcionamiento, se accede a la IP en Node-Red y se demuestra el funcionamiento con node-red manteniendo el control, y su operación bidireccional. En la **figura 10** se detalla en Node-Red que el control lo tiene el módulo, por lo que en este caso los datos serán enviados desde este,y en el servidor no se podrán introducir ningunas variables.

figura 10

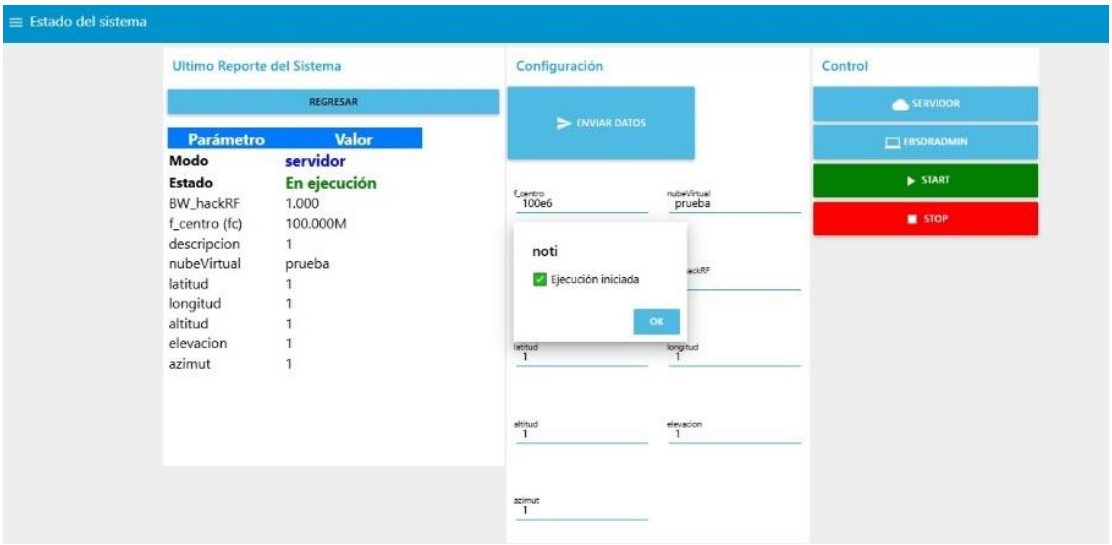
Control de EBSDRADMIN



Ahora, para que el servidor pueda tomar el control, se cambia de modo. En la **figura 11** se detalla que el funcionamiento lo tiene el servidor, y en este caso si se pueden introducir las variables desde esta interfaz, y del mismo modo se puede iniciar o parar el proceso. Toda la información sobre esta sección se encuentra de manera mas detallada en el anexo C. (C, 2025)

figura 11

Control desde el servidor



2.3.2 Subida de los archivos CSV a drive y visualización del espectro

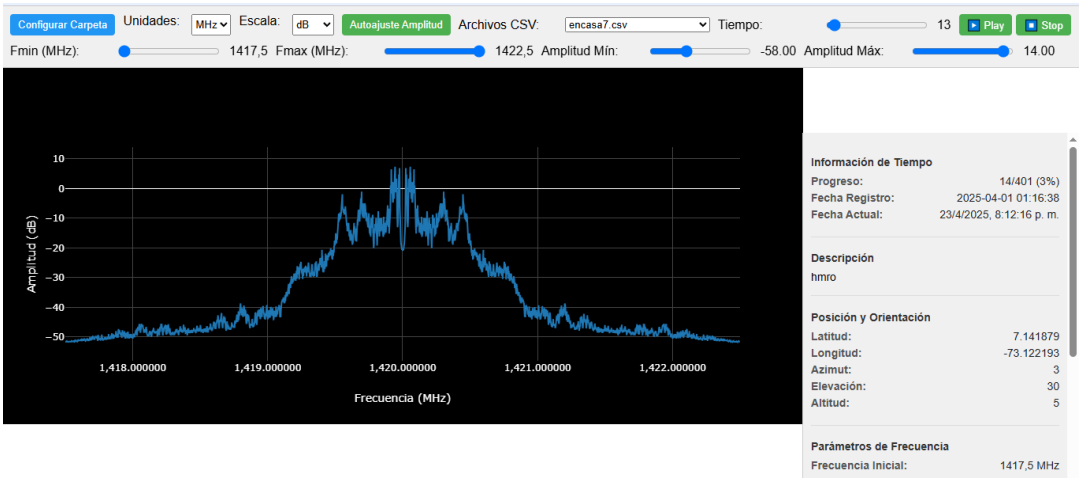
Ahora, para comprobar el funcionamiento de la aplicación, IoTVirtualCloud, se hizo varias pruebas para demostrar su correcto funcionamiento y probar cual era la mejor configuración a usar, en este caso se decidió extender el tiempo de vigilancia a 3 minutos o más, esto se debe a no llenar por completo la base de datos, y más bien que la subida a esta sea intermitente.

Para la visualización del espectro, se utilizó Google Apps Script, que permitió crear una herramienta capaz de mostrar el espectro de los archivos CSV seleccionados. Esta herramienta ofrece tres tipos de visualización del espectro: 2D, 3D y Waterflow. Cada una de estas aplicaciones demostró una gran capacidad para mostrar el espectro de manera detallada.

La herramienta fue efectiva para visualizar las muestras espectrales tomadas en diferentes momentos y lugares, y en la mayoría de los casos, el rendimiento fue aceptable, sin presentar errores. Los errores mayormente presentados provenían del uso de las variables, debido a que si una de las variables era escrita de forma diferente a como se mencionaron anteriormente, el código no la identificará y no tendrá la capacidad de mostrar la gráfica. En **la ilustración 12** se muestra el espectro en 2D y sus respectivos controles e información necesaria.

figura 12

Espectro en 2d



3 Conclusiones

Se comprobó y se realizó la interoperabilidad entre los dispositivos SDR y la plataforma existente de casiri, realizando una interfaz mediante la implementación de **EBSDRADMIN**, que actuó como puente entre el hardware SDR (como HackRF) y el ecosistema de Node-RED. Se logró una comunicación estable a través del protocolo **MQTT**, permitiendo el control remoto de parámetros clave (frecuencia, ancho de banda) y la transmisión de datos en tiempo real.

Se logró satisfactoriamente analizar e interpretar los datos de la información recopilada, esto se hizo a través del diagrama de flujo de GNU Radio que procesaba la señal y mediante el visualizador de espectro que se incorporó a node-red, poder analizarlo.

Se estableció un sistema en la nube de almacenamiento, en este caso Google drive, esto se debe a su fácil manejo, a su fácil conexión para poder elegir un archivo deseado y a su facilidad de poder recibir archivos de cualquier clase.

Se realizaron pruebas que comprueban el funcionamiento correcto del programa, se comprobó la correcta conexión entre el módulo EBSDRADMIN y Node-Red, además de la correcta visualización del espectro y sus respectivas funciones.

Se realizaron manuales y documentaciones para el correcto manejo para la continuidad del proyecto, en esas documentaciones se pueden encontrar los códigos y las instrucciones para la correcta descarga de los programas y varias recomendaciones a seguir para un uso futuro.

4 Recomendaciones

- Optimizar el código existente. Existe posibilidad de que existan maneras de las cuales se puedan realizar algunas de las acciones del código de forma más eficiente.
- Mejora referente a los parámetros que se configuran en el JSON. Se debe diseñar una manera más eficiente y sencilla por medio de la cual los usuarios puedan configurar dichos parámetros de forma más eficiente.
- EBSDRADMIN cuando se reinicia no almacena las variables que se usaron en una ultima ejecución. En casos como por ejemplo un apagón, se apagarían todos los sistemas y por ende la ejecución de EBSDRADMIN se detendría y al reiniciarlo se requeriría configurarlo nuevamente y esto implica que el usuario tiene que tener un registro externo de los parámetros usados.
- EBSDRADMIN no se inicia automáticamente. En casos como un apagón lo ideal es que cuando se restablezca el servicio y los equipos se reinicien EBSDRADMIN se iniciara y reiniciara el .py con las ultima configuración que guardo antes del apagón.

- Optimizar el cómo se manejan los errores ya que actualmente si una configuración del JSON es incorrecta EBSDRADMIN debe cerrarse corregir el JSON y volver a ejecutar el programa

Referencias bibliográficas.

Amazon Web Services. (s.f.). *¿Qué es IoT? Explicación del Internet de las cosas*. https://aws.amazon.com/es/what-is/iot/

Beservices. (2023, 28 noviembre). *¿Qué es Google Apps Script y para qué sirve?* https://blog.beservices.es/blog/que-es-google-apps-script-para-que-sirve

Dario, J. (2016, 27 abril). *Programación visual con Node-RED: conectando el Internet de las cosas con facilidad*. Toptal Engineering Blog. https://www.toptal.com/nodejs/programacion-visual-con-node-red-conectando-el-internet-de-las-cosas-con-facilidad

De Zúñiga, F. G. (2023, 31 octubre). *Servicios web: qué son y qué tecnología usar en su desarrollo*. Blog de arsys.es. https://www.arsys.es/blog/web-services-desarrollo

NASA's Madrid Deep Space Communication Complex (MDSCC). (n.d.). *Radioastronomía*. NASA. Recuperado el 7 de septiembre de 2023, de https://www.mdscc.nasa.gov/index.php/radio-astronomia/

Google Developers. (s.f.). *Google Sheets API*. https://developers.google.com/sheets/api

Amazon Web Services. (s.f.). *¿Qué es MQTT?*. **https://aws.amazon.com/es/what-is/mqtt/**

Infante, D. C. H. (2023, 19 abril). *Qué es Node.Js: casos de uso comunes y cómo instalarlo*. Tutoriales Hostinger. https://www.hostinger.co/tutoriales/que-es-node-js

Ramírez, I. (2020, 31 enero). *Máquinas virtuales: qué son, cómo funcionan y cómo utilizarlas*. Xataka. https://www.xataka.com/basics/maquinas-virtuales-que-son-como-funcionan-como-utilizarlas

IBM. (s.f.). *¿Qué es un gemelo digital?*. **https://www.ibm.com/es-es/think/topics/what-is-a-digital-twin**

National Instruments. (n.d.). *¿Qué es un radio definido por software USRP?* National Instruments. Recuperado el 7 de septiembre de 2023, de https://www.ni.com/es/shop/wireless-design-test/what-is-a-usrp-software-defined-radio.html?srsltid=AfmBOorSBQQiDYMh2F_BQHv-GbgJw7Psv9ejROLWRDWBi5-e38mx8oQt

Mosquitto.org. (s.f.). *Eclipse Mosquitto: An open source MQTT broker*. https://mosquitto.org/

RadioGis. (2024). *Anexo E, Documentación de diagrama de flujo de GNU Radio.* "https://docs.google.com/document/d/1E8Jz3fCoILGNFlw7cxcRdXaT96U6jgK6/edit#heading=h.m28cm1isu695"

González Mendoza, B. H., & Silva Sepúlveda, C. E. (2023). Solución IoT basada en Node-RED para la visualización, almacenamiento y procesamiento en la nube para la estación meteorológica CASIRI [Trabajo de grado, Universidad Industrial de Santander]. Repositorio Institucional UIS. https://noesis.uis.edu.co/items/97b8e532-2084-45f2-9dc2-304c0b88a6e2.

Cañizares Carrillo, A. F., & Guzmán Ortiz, K. D. (2024). Prototipo de radiotelescopio para la banda de 1420 MHz en apoyo de la componente social ejecutada en el Páramo de Berlín en el marco de un proyecto de radioastronomía financiado por MinCiencias - VIE UIS [Trabajo de grado, acceso restringido]. Repositorio Institucional UIS. https://noesis.uis.edu.co/items/495294ea-1c13-450e-a325-2136afa305df.