

Solución IIoT con GNURadio

Introducción

La verdad que resultaría muy controversial decir que esta es una solución IIoT. Pero las siguientes razones podrían señalar que al menos si se acerca bastante a eso:

- GNURadio es una plataforma para el desarrollo de soluciones de procesamiento de señales de radio en tiempo real. El término tiempo real se aplica en su máxima expresión. Por ejemplo cuando se envía una señal de televisión tomada de un partido de fútbol, los televidentes necesitan ver el gol justo en el momento en que ocurre.
- Si un desarrollador desea guardar el espectro de una señal en un disco duro, durante unos minutos, pronto verá con horror que el disco se llena ocupando muchos gigas de información.
- Es imposible pensar en usar un protocolo como MQTT para enviar el espectro de una señal desde GNURadio para ser registrado en la nube.

El propósito de esta práctica

No busca aprender términos nuevos, solo reforzar los que ya se tiene y sobre que el estudiante refuerce que en el uso de los web services está la clave para potenciar soluciones increíbles conectando cualquier cosa en la nube. También se busca reforzar las capacidades de comunicación efectiva en los estudiantes a la hora de querer obtener código de Chat GPT.

Prerrequisitos

- Desarrollar esta guía no requiere conocimientos previos, pero sí es habría una mayor ganancia para quienes saben que es GNURadio, los que saben que son los web services y los que han realizado la guía “Solución IoT basada en HTTP”

El gran reto

Crear un flujograma en GNU Radio que permite obtener el espectro de la voz y enviar los resultados del espectro a una base de datos en la nube. Luego crear una aplicación para visualizarlo en varios casos:

- 1) a partir de lo registrado en la base de datos
- 2) visualización en tiempo real

Tarea 1. Preparativos

Paso 1. Comprobar que GNURadio está disponible en el computador. Este paso tiene sentido para aquellos casos en que esta práctica se realice fuera del laboratorio de comunicaciones. Es decir, contamos con un computador improvisado con Windows como sistema operativo (SO) y podría o no podría tener instalado GNURadio.

1) comprueba si está instalado GNURadio así:  >  >



2) Si el ícono anterior de GNU Radio no aparece es porque no está instalado. Debes proceder a realizar la instalación así:

<https://wiki.gnuradio.org/index.php/InstallingGR> > Windows > Radioconda > baja a hasta la sección "Download" > haz click en el instalador para Windows, con lo cual se descarga el instalador > en Windows corre el instalador bajado

2) Baja a tu computador el flujograma de GNU radio y pruebalo: [enlace](#)

3) Baja a tu computador el flujograma de GNU radio y pruebalo: [enlace](#)

4) Baje [archivo de audio](#) que acompaña al anterior flujograma

Paso 2. Preparativos relacionados con el web service a usar y el flujograma.

1) Crea una google sheet y en la primera hoja registra los siguientes encabezados

	A	B	C	D	E	F	G	H
1	Fecha	Latitud	Longitud	Descripcion	N	Datos	finicial	fpaso

Pueden copiar y pegar del siguiente ejemplo para no cometer errores de edición:

Fecha	Latitud	Longitud	Descripcion	N	Datos	finicial	fpaso
-------	---------	----------	-------------	---	-------	----------	-------

2) Revise la [documentación de web service](#) para obtener datos básicos del mismo. Es decir, datos que puedan servir para entrenar a Chat GPT para resolver el problema. Registre estos datos en la tabla siguiente:

En esta tabla se prepara un prompt de entrenamiento para Chat GPT. Tu debes complementar los espacios vacíos
Memorizar: voy a iniciar un nuevo proyecto para lograr registrar el espectro de una señal obtenido mediante GNU Radio, en una hoja de Google Sheets
Memorizar:

1) Ya cuento con un web service, que en adelante llamaré “GoogleSheet CRUD POST”
<p>2) El web service tiene estos datos básicos:</p> <p>a) La URL de “GoogleSheet CRUD POST” es: https://script.google.com/macros/s/AKfycbz7NCKBvRCvhQkuAjcaQUoxhpRQOD_4zOxplrXNP1s2fuiuB-CmPATKADO6ijK__Evm/exec</p>
<p>b) “GoogleSheet CRUD POST” está hecho para: Este servicio web permite realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en una hoja de Google Sheets utilizando solicitudes POST. Las solicitudes deben ser enviadas en formato JSON y el web service devuelve las respuestas también en formato JSON. La importancia de tener un Web service que usa llamadas POST está en que los datos que se envían a la base de datos pueden ser pesados, al menos notablemente más pesados que cuando se usan llamadas GET.</p>
<p>c) El siguiente es un ejemplo, tomado de la documentación del web service, que comprobadamente funciona en python y sirve para crear un registro de datos en una google sheet:</p> <p>Ejemplo. Queremos crear un nuevo registro en la base de datos. Suponemos que tenemos un sensor de temperatura que da un valor discreto por cada medición.</p> <p>Los datos para el registros son:</p> <ul style="list-style-type: none"> • la fecha: 2024-08-07T12:00:00Z • ubicación: Laboratorio de Física • tipo: Datos aleatorios de prueba • el valor medido a registrar es 0.1 <p>Ejemplo de la Información sobre la base de datos:</p> <ul style="list-style-type: none"> • la url de nuestra base de datos (la Google Sheet) es:https://docs.google.com/spreadsheets/d/1aZ02Lg-YHxfcpkfMDu17bzfcCieAcYIJqJRguP8pZGM • de esa Google Sheet vamos a usar la hoja cero (osea, la primera) • los encabezados ya están escritos en la primera fila y son: fecha, ubicación, tipo, valor • la primera columna se dedica a los ID y es la misma fecha <p>Pasos que el código debe seguir:</p> <ul style="list-style-type: none"> • Paso 1. Los datos se organizan como un array 1D, así, para este ejemplo tenemos: ["2024-08-07T12:00:00Z", "Laboratorio de Física", "Datos aleatorios de prueba", 0.1] • Paso 2. Debe crearse un objeto de datos que reúna toda la información para la orden deseada: <pre>{ "ordentipo": "crear", "url": "https://docs.google.com/spreadsheets/d/1aZ02Lg-YHxfcpkfMDu17bzfcCieAcYIJqJRguP8pZGM",</pre>

<pre> "numeroHoja": 0, "filaencabezados": 1, "columnald": 1, "datos": ["2024-08-07T12:00:00Z", "Laboratorio de Física", "Datos aleatorios de prueba", 0.1] } </pre>
<p>3) La Google Sheet en la que queremos registrar el espectro tiene esta URL: https://docs.google.com/spreadsheets/d/1ZI1pO5DJjVnh0ph0jH8sRjnLGqXHOz-jBXJ1ZF875JM/edit?usp=sharing</p>
<p>4) La Google Sheet tiene ocupada la primera fila con los siguientes encabezados para los registros esperados: "Fecha", "Latitud", "Longitud", "Descripcion", "N", "Datos", "finicial", "fpaso"</p> <p>Esos datos se explican así:</p> <ul style="list-style-type: none"> • Latitud y Longitud: son las coordenadas geográficas del lugar donde se realizan las mediciones. Se pueden tomar de google maps, del lugar donde Uds se encuentran • Descripción: está destinada para posibles comentarios a la medición • N: número de muestras espectrales • Datos: los N valores del espectro separados por ",". Por ejemplo, si N=8, esos valores pueden ser: "0.1,0.4,0.8,0.43,0.4,0.2,0.1,0.5" • finicial: es la frecuencia de la primera muestra espectral en Hz • fpaso: es la distancia en Hz entre muestras espectrales
<p>5) Ya cuento con un flujograma en GNU Radio con estas cualidades:</p> <p>a) Hay un bloque que genera el audio, seguido de un bloque "Stream to Vector", seguido de un bloque "FFT" seguido de un bloque "Complex to Mag" que entrega el espectro que obtiene en tiempo real de una señal.</p> <p>b) Hay variables globales como N, latitud, longitud, finicial, fpaso, salto</p>

3) Prepare el prompt para Chat GPT, donde explicas lo que tu necesitas

Al flujograma de GNU Radio estoy agregando un "python block" a la salida del bloque "Complex to Mag".

Los siguientes son requerimientos para este bloque:

1) se llamará "Envia a Google Sheet"

2) tendrá los siguientes parámetros externos:

```

url_google_sheet='https://docs.google.com/spreadsheets/d/1ZI1pO5DJjVnh0ph0jH8sRjnLGqXHOz-jBXJ1ZF875JM/edit?usp=sharing',
latitud=7.141879,
longitud=-73.122193,
N=8,
finicial=1417000000,
fpaso=1421000000,
salto=40000

```

- 3) El bloque dejará pasar sin procesar “salto” ventanas espectrales y para la siguiente ventana registrará los N valores del espectro en la google sheet usando el web service. Luego repetirá lo mismo una y otra vez con las siguientes ventanas espectrales.
- 4) En el constructor de los parámetros deben tener valores por defecto
- 5) El bloque no tiene salidas, solo tiene una entrada.
 - 6) Es importante importar librerías que usa gnu radio como:

```
import numpy as np
from gnuradio import gr
```

Escriba aquí el código obtenido para el “python block”

```
import numpy as np
from gnuradio import gr
import json
import requests
from datetime import datetime

class blk(gr.sync_block): # Heredando de gr.sync_block
    def __init__(self,
url_google_sheet='https://docs.google.com/spreadsheets/d/1ZI1pO5DJjVn
h0ph0jH8sRjnLGqXH0z-jBXJ1ZF875JM/edit?usp=sharing',
        latitud=7.141879, longitud=-73.122193, N=8,
        finicial=1417000000, fpaso=1421000000, salto=40000):
    gr.sync_block.__init__(self,
        name="Envia a Google Sheet",
        in_sig=[(np.float32, N)], # Recibe un vector de tamaño N
        out_sig=None) # No tiene salida

# URL del servicio web de Google Apps Script
self.url_web_service =
'https://script.google.com/macros/s/AKfycbz7NCKBvRCvhQkuAjcaQUoxhpRQO
D_4zOxplrXNP1s2fuiuB-CmPATKADO6ijK__Evm/exec'
self.url_google_sheet = url_google_sheet
self.latitud = latitud
self.longitud = longitud
self.N = N
self.finicial = finicial
self.fpasso = fpaso
self.salto = salto
self.contador_entradas = 0 # Contador de entradas

# Contador de ventanas espectrales procesadas
self.entradas_procesadas = 0

def registrarDatos(self, data):
    """
    Función que organiza y envía el JSON al servicio web.
    """
    # Convertir los datos del espectro a una cadena separada por
comas
    data_str = ','.join([str(d) for d in data])

    # Obtener la fecha y hora actuales
    fecha_actual = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

    # Crear el objeto JSON para la solicitud
```

```

        json_data = {
            "ordentipo": "crear",
            "url": self.url_google_sheet, # URL de la Google Sheet
            "numeroHoja": 0,
            "filaencabezados": 1,
            "columnaId": 1,
            "datos": json.dumps([ # Envolviendo en json.dumps para
convertirlo en una cadena JSON válida
                fecha_actual,
                self.latitud, # Coordenada latitud
                self.longitud, # Coordenada longitud
                "dagd_TestHome",
                self.N, # Número de muestras espectrales
                data_str, # Datos espectrales
                self.finicial, # Frecuencia inicial
                self.fpasso # Paso de frecuencia
            ])
        }

        # Enviar el JSON como solicitud POST al servicio web
        try:
            response = requests.post(self.url_web_service,
json=json_data)
            response.raise_for_status() # Verificar que no haya
errores
            print(f"Datos enviados exitosamente: {response.text}")
        except requests.exceptions.RequestException as e:
            print(f"Error enviando datos: {e}")

    def work(self, input_items, output_items):
        """
        Función principal que procesa las entradas y llama a la
función de registro.
        """
        # Incrementar el contador de entradas procesadas
        self.entradas_procesadas += len(input_items[0])

        # Verificar si se han procesado suficientes entradas
(definido por "salto")
        if self.entradas_procesadas >= self.salto:
            self.entradas_procesadas = 0 # Restablecer el contador

            # Recibir los datos del espectro (vector de tamaño N)
            data = input_items[0][0]

            # Registrar los datos en la Google Sheet
            self.registrarDatos(data)

            return len(input_items[0]) # Indica que se procesaron todas
las muestras de entrada

```

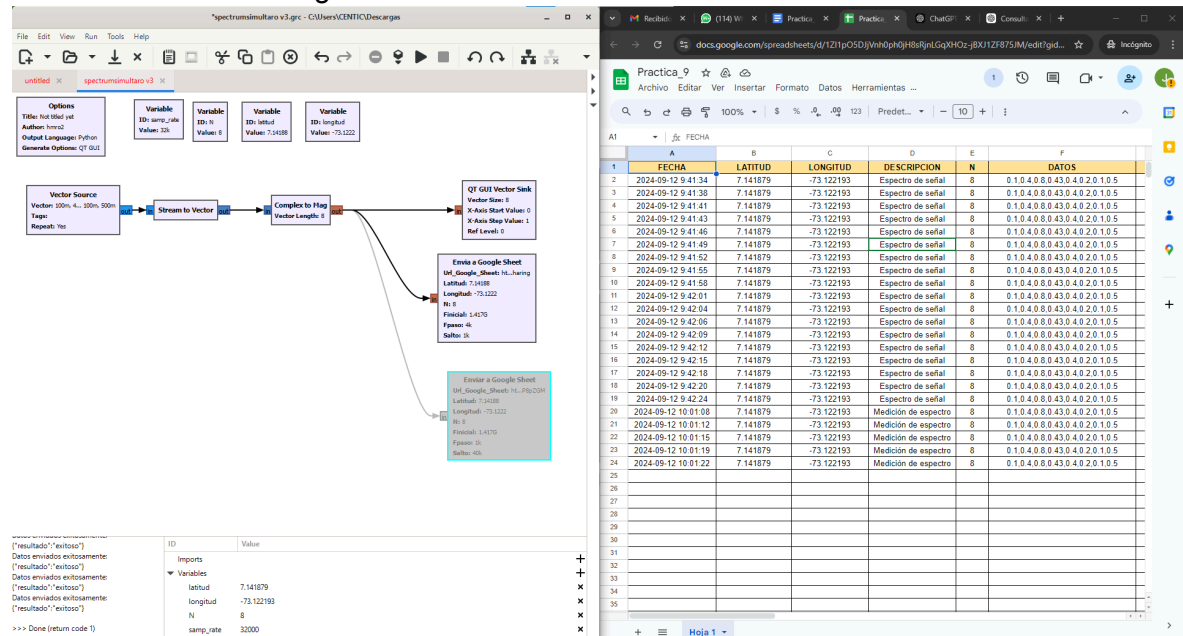
Paso 3. Completa el flujograma (el primer flujograma)

- 1) Conecta un python block a la salida del bloque "Complex to Mag". Luego ábrelo > Open In Editor > pega el código dado por Chat GPT.
- 2) Realiza las pruebas. Ten cuidado de usar la URL de tu archivo de Google Sheet

3) Si no tuviste éxito, puedes usar [el código que proporciona el profesor](#) y analiza qué fallas has cometido

Evidencias de las pruebas

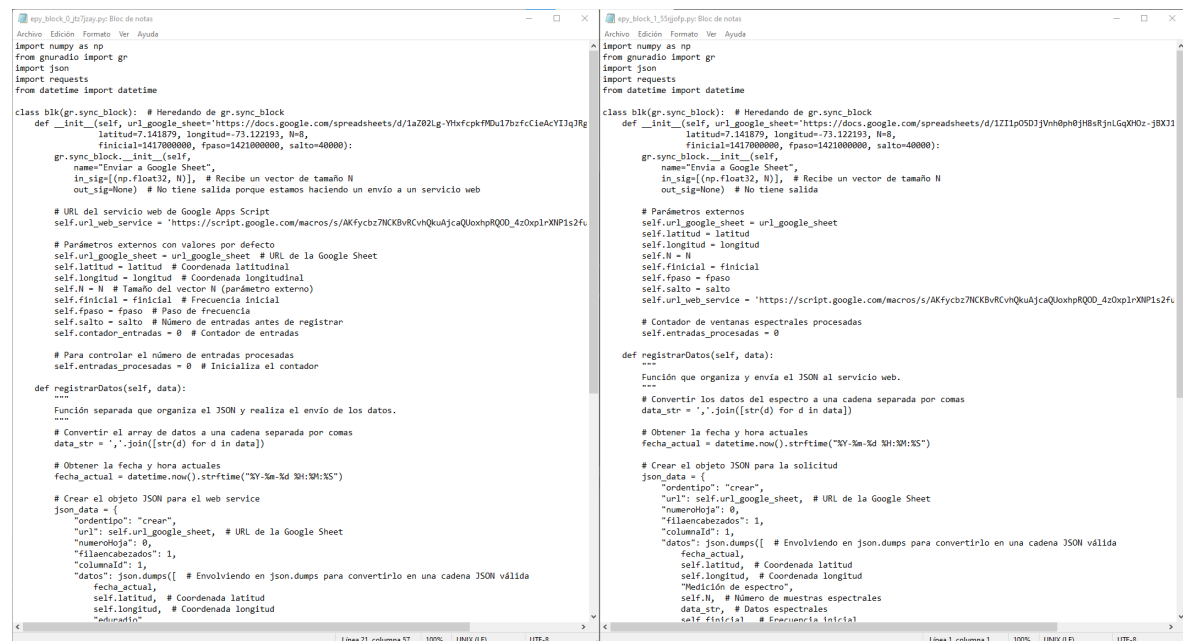
En clase se realizó lo siguiente:



The screenshot shows a QGIS Vector Source tool configuration. The tool is set to stream data from a vector source to a Google Sheet. The Google Sheet is titled "Practica_9" and contains a table with the following columns: FECHA, LATITUD, LONGITUD, DESCRIPCION, N, and DATOS. The table is populated with data for various locations and dates.

FECHA	LATITUD	LONGITUD	DESCRIPCION	N	DATOS
2024-09-12 9:41:34	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:41:38	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:41:41	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:41:43	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:41:46	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:41:49	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:41:52	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:41:55	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:41:58	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:01	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:04	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:06	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:09	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:12	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:15	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:18	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:20	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 9:42:24	7.141879	-73.122193	Espectro de señal	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 10:01:08	7.141879	-73.122193	Medición de espectro	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 10:01:12	7.141879	-73.122193	Medición de espectro	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 10:01:15	7.141879	-73.122193	Medición de espectro	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 10:01:19	7.141879	-73.122193	Medición de espectro	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5
2024-09-12 10:01:22	7.141879	-73.122193	Medición de espectro	8	0.10.40.0.0.43.0.4.0.2.0.1.0.5

La siguiente imagen es una comparación del código generado por GPT y el código funcional que tiene el profe:



The image shows a comparison of two Python code snippets. The left snippet is a basic implementation of a data streamer, and the right snippet is a more complex implementation with additional features like JSON handling and error handling.

```
class blk(gr.sync_block): # Heredando de gr.sync_block
    def __init__(self, url_google_sheet="https://docs.google.com/spreadsheets/d/1a2LgZg-YMfcPkP9Du17bfcCiaAcY1JqHg
        latitude=7.141879, longitude=-73.122193, N=8,
        ffinal=1417000000, fpasso=1421000000, salto=400000):
        gr.sync_block.__init__(self,
            name="Enviar a Google Sheet",
            in_sig=(np.float32, N)), # Recibe un vector de tamaño N
            out_sig=None) # No tiene salida porque estamos haciendo un envío a un servicio web

# URL del servicio web de Google Apps Script
self.url_web_service = "https://script.google.com/macros/s/AKfycb7NCKBvRCvHkQaJcaQlXohpRQ00_4z0p1rXNP1s2Fu

# Parámetros externos con valores por defecto
self.url_google_sheet = url_google_sheet # URL de la Google Sheet
self.latitude = latitude # Coordenada latitudinal
self.longitude = longitude # Coordenada longitudinal
self.N = N # Tamaño del vector N (parámetro externo)
self.ffinal = ffinal # Frecuencia inicial
self.fpasso = fpasso # Paso de frecuencia
self.salto = salto # Número de entradas antes de registrar
self.contador_entradas = 0 # Contador de entradas

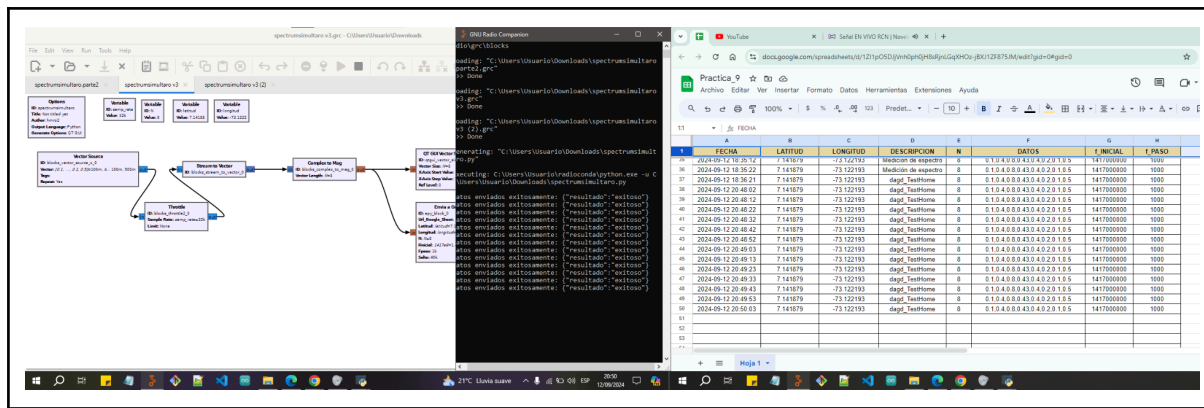
# Para controlar el número de entradas procesadas
self.entradas_procesadas = 0 # Inicializa el contador

def registrarDatos(self, data):
    """
    Función separada que organiza el JSON y realiza el envío de los datos.
    """
    # Convertir el array de datos a una cadena separada por comas
    data_str = ','.join([str(d) for d in data])

    # Obtener la fecha y hora actuales
    fecha_actual = datetime.now().strftime("%Y-%m-%d %H:%M:%S")

    # Crear el objeto JSON para el web service
    json_data = {
        "orden": "crear",
        "url": self.url_google_sheet, # URL de la Google Sheet
        "numero": 0,
        "filas": 1,
        "columnas": 1,
        "datos": json.dumps([ # Envolviendo en json.dumps para convertirlo en una cadena JSON válida
            {
                "fecha": fecha_actual,
                "latitud": self.latitude, # Coordenada latitud
                "longitud": self.longitude, # Coordenada longitud
                "medicion": self.final, # Frecuencia inicial
            }
        ])
    }
```

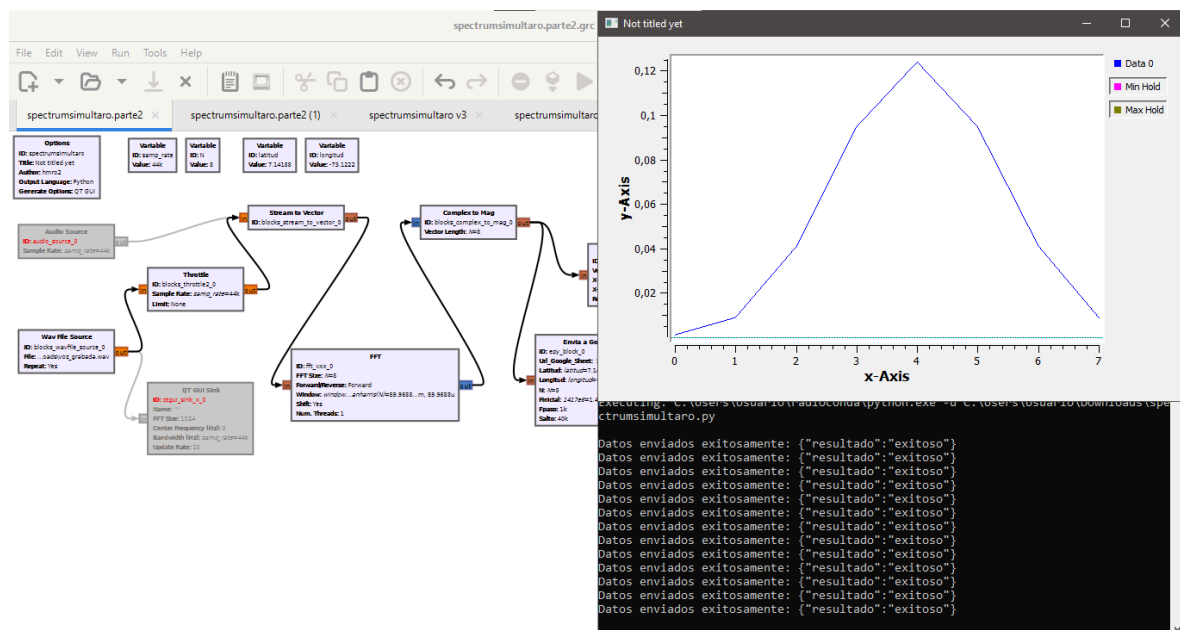
A continuación se muestra otra prueba:



Paso 4. Completa el flujograma (el segundo flujograma).
Reusa el python block anterior en el segundo flujograma

Evidencias de las pruebas

Cómo se rehusó el python block, el código usado es el mismo presentado anteriormente, al ejecutar el flujograma se obtiene el siguiente resultado:



A continuación se observan los datos cargados correctamente en Google Sheets:

practica 9

Archivo Editar Ver Insertar Formato Datos Herramientas Extensiones Ayuda

52:64 12/9/2024 21:06:14

	A	B	C	D	E	F	G	H
1	FECHA	LATITUD	LONGITUD	DESCRIPCION	N	DATOS	f_INICIAL	f_PASO
47	2024-09-12 20:49:33	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
48	2024-09-12 20:49:43	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
49	2024-09-12 20:49:53	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
50	2024-09-12 20:50:03	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
51	2024-09-12 20:50:13	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
52	2024-09-12 21:06:14	7.141879	-73.122193	dagd_TestHome	8	6e-05.0.0001798582.0.0005207429.0.0009820443.0.0005207429.0.00015558451.0.020981008.0.07099857.0.1310354.0.15704179.0.1310354.0.07099857.0.020981008.0.0005055368.0.03326605.0.18837869.0.47287992.0.632941.0.47287992.0.18837869.0.03326605.0.0004146993.0.03788573.0.1825449.0.42897558.0.56353366.0.42897558.0.1825449.0.03788573.0.90087764.0.008810906.0.025845092.0.045413516.0.053958632.0.045413516.0.025845092.0.008810906.0.006670058.0.01172009.0.066433676.0.16671106.0.22309554.0.16671106.0.066433676.0.01172009.0.00024175753.0.0018188372.0.004998873.0.006839746.0.004998873.0.0018188372.0.00024175753.0.004436612.0.0458841.0.23940708.0.57787627.0.7644228.0.57787627.0.23940708.0.0458841.0.0017673373.0.036641356.0.2055717.0.51430684.0.68772346.0.51430684.0.2055717.0.036641356.0.0010857657.0.003537897.0.013661756.0.02974458.0.03811109.0.02974458.0.013661756.0.003537897.0.0022453032.0.013354128.0.0426722.0.1823036.0.084076226.0.07318236.0.0426722.0.013354128.0.0012559891.0.049141977.0.24067176.0.5981403.0.7192564.0.5981403.0.24067176.0.049141977.0.18378198.0.010910597.0.028974904.0.040414743.0.039918646.0.040414743.0.028974904.0.010910597.0.010774359.0.06717033.0.22469239.0.39420864.0.45577365.0.39420864.0.22469239.0.06717033	1417000000	1000
47	2024-09-12 20:49:33	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
48	2024-09-12 20:49:43	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
49	2024-09-12 20:49:53	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
50	2024-09-12 20:50:03	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
51	2024-09-12 20:50:13	7.141879	-73.122193	dagd_TestHome	8	0.1040.0.8.0.43.0.4.0.2.0.1.0.5	1417000000	1000
52	2024-09-12 21:06:14	7.141879	-73.122193	dagd_TestHome	8	6e-05.0.0001798582.0.0005207429.0.0009820443.0.0005207429.0.00015558451.0.020981008.0.07099857.0.1310354.0.15704179.0.1310354.0.07099857.0.020981008.0.0005055368.0.03326605.0.18837869.0.47287992.0.632941.0.47287992.0.18837869.0.03326605.0.0004146993.0.03788573.0.1825449.0.42897558.0.56353366.0.42897558.0.1825449.0.03788573.0.90087764.0.008810906.0.025845092.0.045413516.0.053958632.0.045413516.0.025845092.0.008810906.0.006670058.0.01172009.0.066433676.0.16671106.0.22309554.0.16671106.0.066433676.0.01172009.0.00024175753.0.0018188372.0.004998873.0.006839746.0.004998873.0.0018188372.0.00024175753.0.004436612.0.0458841.0.23940708.0.57787627.0.7644228.0.57787627.0.23940708.0.0458841.0.0017673373.0.036641356.0.2055717.0.51430684.0.68772346.0.51430684.0.2055717.0.036641356.0.0010857657.0.003537897.0.013661756.0.02974458.0.03811109.0.02974458.0.013661756.0.003537897.0.0022453032.0.013354128.0.0426722.0.1823036.0.084076226.0.07318236.0.0426722.0.013354128.0.0012559891.0.049141977.0.24067176.0.5981403.0.7192564.0.5981403.0.24067176.0.049141977.0.18378198.0.01091059		