Prototipo Costo-Eficiente y Escalable para el Monitoreo del Espectro Radioeléctrico en Colombia mediante Radio Definido por Software y Aprendizaje Profundo

B.E.A.M. - Broad Electromagnetic Activity Monitoring

September 30, 2024

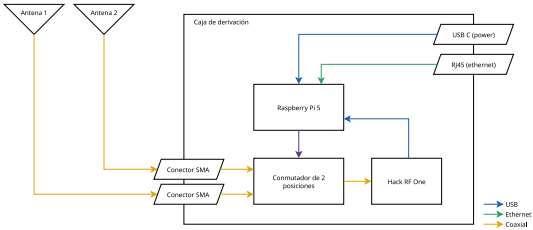
Universidad Nacional de Colombia
Signal Processing and Recognition Group - SPRG
Manizales, Colombia

September 30, 2024

Outline

Overall Design Overview

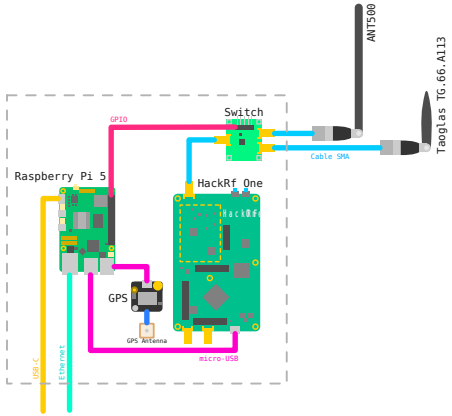
Key Hardware Components of the Prototype



- **Raspberry Pi 5**: Central processing unit.
- **HackRF One**: SDR device for signal acquisition.
- **Antennas (ANT500 and TG.66.A113)**: ANT500 for low frequencies, TG.66.A113 for high frequencies to cover the full range.
- **GPS Module**: Provides geolocation to complement signal acquisition.

Hardware Architecture Overview

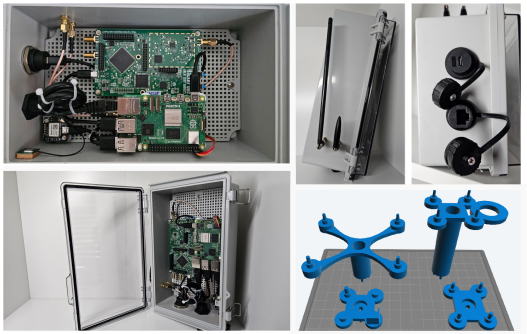
Hardware Interconnection and Data Flow



- **Raspberry Pi 5** connects to **HackRF One** via USB for signal acquisition.
- **GPS Module** connects to the Raspberry Pi via USB to provide location data.
- **Antennas** are controlled by the Raspberry Pi via GPIO to switch between SDR and GPS.

Prototype Construction

Assembly and Enclosure Details

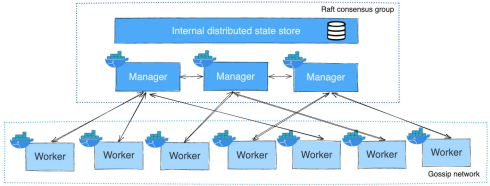


- **LeMotech Electrical Box:** IP67 waterproof enclosure for outdoor protection.
- **Sealed connectors**: Ethernet and USB for secure connections.
- **External antennas**: ANT500 for SDR. Taoglas TG.66.A113
- **Power button**: Push button for safely turning the Raspberry Pi on and off.
- **OLED display**: For real-time debugging information and system status.
- **Internal organization**: Components mounted on perforated panel for stability.
- **3D Printed Components**: Custom-designed mounts and holders printed in 3D for securing parts during assembly.

Service Management

Foundation and Docker Swarm Integration

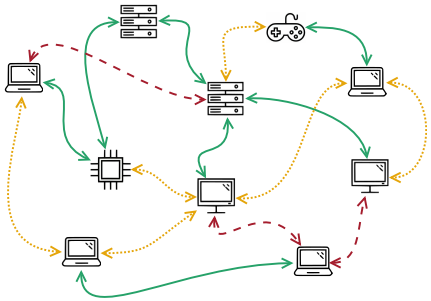
- **Foundation**: Manages creation, scaling, and monitoring of services in Docker Swarm.
- **python-libhackrf**: Essential for communicating with the HackRF SDR for signal acquisition.
- **FastAPI**: Provides endpoints for system configuration and interaction.
- **Workers**: Execute specific tasks like data acquisition and real-time processing.
- **Watchdogs**: Monitors services to ensure uninterrupted acquisition and system stability.



Data Communication and Streaming

Real-Time Communication with Chaski-Confluent

- **Chaski-Confluent**: Manages real-time data streaming between system components.
- **Low-Latency Data Flow**: Ensures minimal delays between data acquisition, processing, and visualization.
- **Communication Pipelines**: Establishes dedicated pipelines for efficient data handling across nodes.
- **Data Buffering**: Buffers data during acquisition to prevent data loss in case of network delays.
- **Monitoring Dashboard**: Tracks system performance, signal quality, and data transmission status in real time.
- **Logging System**: Centralized logging for tracking data flow and identifying any issues during streaming.
- **Error Handling**: Mechanisms to handle communication errors and ensure data integrity.



Data Acquisition and Processing

Frequency Scanning with python-libhackrf

- **python-libhackrf**: Manages communication with the HackRF for configuring and executing frequency sweeps.
- **Sweep Scanning**: Captures frequency data across defined bands for spectrum analysis.
- **Key Parameters**: Frequency bands, sample rate, step width, and buffer sizes.
- **Asynchronous Operation**: Allows non-blocking frequency scanning, enabling continuous data acquisition.