Proyecto Final – CyberDragon Telemetry Challenge

Dylan Vasquez Rojas
Shadday Sofia Urrego Martinez
Maria Alejandra Fonseca
Ana Maria Trujillo

***Resumen*— Este proyecto desarrolla un sistema de telemetría inalámbrica para vehículos radiocontrolados, diseñado para el RC Cars Telemetry Challenge 2025 de la Universidad Militar Nueva Granada. Utiliza un transceptor nRF24L01 y una Raspberry Pi Pico 2W en la banda ISM de 2,4-2,5 GHz. El módulo transmisor integra sensores de voltaje, temperatura, GPS, un acelerómetro, un giroscopio y un sensor óptico TCRT5000.**

Los datos se transmiten a un nodo receptor terrestre y se visualizan en tiempo real a través de un panel de control con interfaz Lovable. Además, se garantizan tiempos de vuelta mediante un subsistema Wi-Fi que reporta al servidor de competición. El sistema fue validado con pruebas en pista, demostrando su eficacia en condiciones dinámicas.

***Abstract*— This project develops a wireless telemetry system for radio-controlled vehicles, designed for the 2025 RC Cars Telemetry Challenge at the Nueva Granada Military University. It uses an nRF24L01 transceiver and a Raspberry Pi Pico 2W in the 2.4-2.5 GHz ISM band. The transmitter module integrates voltage, temperature, and GPS sensors, an accelerometer, a gyroscope, and a TCRT5000 optical sensor.**

The data is transmitted to a ground-based receiver node and displayed in real time via a control panel with a Lovable interface. Lap times are also guaranteed by a Wi-Fi subsystem that reports to the competition server. The system was validated with track tests, demonstrating its effectiveness under dynamic conditions.

I. Introducción

La práctica RC Cars Telemetry Challenge 2025 propone el diseño de un sistema de telemetría inalámbrica para un vehículo RC, empleando la banda ISM de 2.4–2.5 GHz y cumpliendo los lineamientos establecidos. El proyecto desarrollado integra una Raspberry Pi Pico 2W como nodo transmisor y un módulo nRF24L01 como enlace RF para enviar información del vehículo hacia una estación de monitoreo en tierra. El sistema incorpora sensores de batería, temperatura del motor, GPS, una unidad inercial (IMU) y un sensor óptico TCRT5000 para la detección del paso por meta.

La estación receptora, basada también en una Raspberry Pi Pico 2W, decodifica los paquetes recibidos y los transmite vía WebSocket a un dashboard diseñado en Lovable, permitiendo visualizar en tiempo real variables como posición, IMU, voltaje y estado de actuadores. Paralelamente, un módulo WiFi adicional en el vehículo reporta los tiempos oficiales de vuelta al servidor de competencia ubicado en 10.10.10.10. Este sistema integra principios de comunicaciones inalámbricas, programación embebida y visualización de datos, permitiendo evaluar el desempeño dinámico del vehículo en condiciones reales de carrera.

La implementación conjunta de telemetría RF y reporte de tiempos mediante WiFi permite analizar el comportamiento dinámico del vehículo, verificar su desempeño en pista, y evaluar la fiabilidad del enlace de comunicaciones. Este proyecto constituye un ejercicio integral que combina electrónica, comunicaciones inalámbricas, programación embebida y visualización de datos, consolidando competencias necesarias en el diseño de sistemas modernos de adquisición y transmisión de información.

DESARROLLO DE LA PRÁCTICA

La metodología empleada para el desarrollo del sistema de telemetría se estructuró en cuatro fases principales: diseño del hardware, implementación del firmware embebido, desarrollo del sistema de recepción y visualización, y validación experimental en pista. Cada etapa se describe a continuación.

I. Diseño del sistema y selección de hardware

Se definió una arquitectura distribuida conformada por tres nodos inalámbricos:

- Control remoto, encargado de enviar comandos de aceleración, dirección y freno al vehículo mediante un enlace nRF24L01.
- Carro (TX), responsable de ejecutar los comandos recibidos y transmitir simultáneamente la telemetría hacia la estación base.
- Estación de telemetría (RX), encargada de recibir los datos, procesarlos y enviarlos a un dashboard en tiempo real.

Adicionalmente, el vehículo incorpora un segundo canal WiFi dedicado exclusivamente al envío de tiempos de vuelta hacia el servidor oficial en 10.10.10.10.

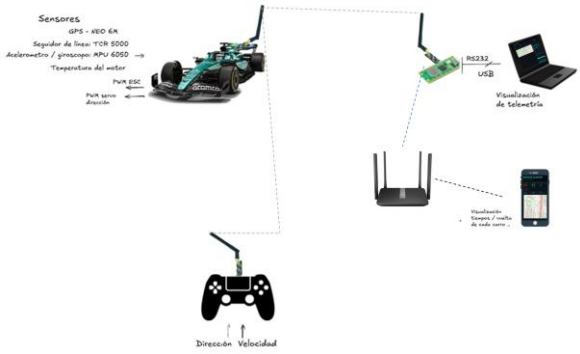


Ilustración. 1. Diseño Del Sistema

II. Implementación del firmware del control remoto (TX de control)

El control remoto está basado en una Raspberry Pi Pico y un transceptor nRF24L01 configurado en modo transmisión. Se integraron dos joysticks analógicos para dirección y aceleración, un botón físico dedicado al freno de emergencia. Para implementar los rangos de aceleración, neutro y retroceso se diseño un codigo de prueba el cual leyera los rangos del joystick para los 3 casos anteriores. Los valores analógicos fueron muestreados, normalizados y empaquetados en un formato binario compacto que se transmite hacia el carro. El canal RF y la dirección del pipe fueron configurados para garantizar exclusividad según los lineamientos de la práctica.

III. Implementación del firmware del vehículo RC (RX de control + TX de telemetría)

El vehículo incorpora una Raspberry Pi Pico 2W que cumple un doble rol:

- Receptor de comandos del control remoto: El nRF24L01 del carro opera en modo escucha para recibir el paquete binario de control proveniente del transmisor. Estos valores se utilizan para generar las señales PWM del motor y del servo de dirección, además de gestionar el freno de emergencia.

- Transmisor de telemetría: De manera paralela, la Pico adquiere los datos provenientes de los sensores obligatorios (voltaje, temperatura del motor, GPS, IMU y TCRT5000), los empaqueta en un frame binario de 32 bytes y los envía por RF hacia la estación de telemetría.

- Envío de tiempos de vuelta por WiFi: El sensor TCRT5000 detecta los pasos por meta y la Pico envía los tiempos al servidor oficial mediante HTTP.

De este modo, el carro actúa simultáneamente como receptor RF, transmisor RF y cliente WiFi.

IV. Desarrollo del módulo receptor de telemetría (Estación RX) y puente WebSocket

La estación de telemetría se implementó utilizando una Raspberry Pi Pico 2W como receptor RF y un computador portátil encargado del procesamiento y distribución de datos. Este módulo cumple el rol de puente entre el enlace nRF24L01 del carro y el dashboard en tiempo real, mediante una arquitectura basada en Serial + WebSocket.

Para verificar la asignación del canal de transmisión del punto de acceso, se modificó el parámetro CHANNEL en el archivo APWifipico.py, configurándolo inicialmente en el canal 6. Posteriormente, se utilizó la aplicación WiFi Analyzer desde un dispositivo móvil para corroborar la posición espectral del punto de acceso generado por la Raspberry Pi Pico 2W.

- **Para Recepcion en la RaspBerry de telemetria:**

La Pico 2W opera en modo RX del nRF24L01, escuchando continuamente en el canal 76 y en el pipe asignado al vehículo. Cada trama recibida es un paquete binario de 32 bytes, el cual se decodifica localmente para reconstruir las variables de telemetría:

- voltaje de batería
- temperatura del motor
- aceleraciones (ax, ay, az)
- velocidades angulares (gx, gy, gz)
- coordenadas GPS
- PWM del motor y del servo
- estado del sensor TCRT5000.

Estos datos se convierten a JSON y se imprimen por el puerto USB Serial con una frecuencia aproximada de 10–20 Hz.

- **Backend Serial hacia el WebSocket en el computador:**

En el computador se ejecuta un script Python responsable de construir un puente bidireccional entre el puerto serial y el servidor WebSocket. Sus funciones principales son:

a) Apertura y monitoreo del puerto serial: El backend abre el puerto designado (ej. COM6) a 115200 baudios y lee cada línea enviada por la Pico RX. Si la línea no contiene un JSON válido, se descarta automáticamente.

Se incluye manejo de reconexión en caso de que el dispositivo sea desconectado temporalmente.

b) Parseo, validación y limpieza del JSON: Cada paquete recibido es analizado y se verifica la presencia de los siguientes campos mínimos:

```
vbat, temp, lat, lon,
ax, ay, az,
gx, gy, gz,
pwm_motor, pwm_servo,
meta
```

En caso de faltantes o errores de formato, el paquete se descarta para evitar corrupción de datos en el dashboard.

El backend además agrega un timestamp generado por el servidor, el cual es útil para sincronización temporal entre múltiples clientes.

- **Servidor WebSocket asíncrono:**

El backend implementa un servidor WebSocket basado en websockets y asyncio. Este servidor tiene las siguientes características:

- Conexión multi–cliente

Mantiene un conjunto global de clientes conectados (connected_clients).
Cada nuevo navegador que abre el dashboard se convierte en un consumidor adicional del flujo de telemetría.

- Distribución tipo broadcast

Cada paquete JSON proveniente del puerto serial se envía a todos los clientes conectados.
El sistema identifica clientes desconectados y los elimina automáticamente.

- Mantenimiento de conexión

El backend escucha mensajes entrantes desde los clientes (aunque en esta práctica no se utilizan). Esto permite extender el sistema en el futuro para control bidireccional.

V. Desarrollo del dashboard en Lovable

La visualización de la telemetría se desarrolla mediante un dashboard implementado en Lovable, una plataforma que permite crear interfaces dinámicas basadas en JavaScript y componentes UI modernos. El dashboard constituye la capa final del sistema, encargada de interpretar los datos enviados desde el backend WebSocket, presentarlos en tiempo real y facilitar el análisis de telemetría durante la carrera.

El dashboard opera como un cliente WebSocket, conectado al servidor local ejecutado en el computador de la estación de telemetría. Su función principal es:

Abrirse en el navegador.

Establecer un WebSocket hacia:
- ws://localhost:8888

Esperar paquetes JSON provenientes del backend.

Interpretar, actualizar y graficar variables como:

Voltaje de batería, temperatura del motor, coordenadas GPS (mapa), Aceleraciones (ax, ay, az), Giroscopio (gx, gy, gz), PWM de motor y servo, Estado del sensor TCRT5000, Tiempos de vuelta.

Luego muestra los indicadores, barras y gráficas en tiempo real.

Ilustración. 2. Interfaz De Telemetría

MARCO CONCEPTUAL

La telemetría constituye un componente esencial en sistemas embebidos que requieren supervisión remota en tiempo real. Este proceso consiste en medir variables físicas en un nodo remoto y transmitirlas hacia una estación receptora mediante un enlace de comunicaciones digitales [1]. En el contexto de vehículos radiocontrolados, la telemetría permite monitorear parámetros como voltaje de batería, temperatura del motor, aceleraciones lineales, velocidades angulares y posición geográfica, lo que posibilita analizar el comportamiento dinámico del vehículo durante una carrera y ajustar estrategias de control.

El proyecto se desarrolla dentro de la banda ISM de 2.4–2.5 GHz, ampliamente utilizada para enlaces de corto alcance debido a su disponibilidad global, ausencia de requerimientos de licencia y compatibilidad con tecnologías como WiFi, Bluetooth y transceptores propietarios [2]. Sin embargo, esta banda presenta desafíos como interferencia, saturación del espectro y colisiones entre transmisores, por lo que es necesario implementar mecanismos de selección de canal, técnicas de modulación robustas y validación en tiempo real mediante analizadores de espectro para garantizar un desempeño estable durante las pruebas en pista.

La Raspberry Pi Pico 2W es el microcontrolador central del sistema de telemetría. Este dispositivo integra el procesador RP2350, interfaces SPI, I2C, UART y ADC, y un módulo WiFi que permite la implementación de enlaces mixtos de control y comunicación [3]. Su arquitectura dual-core facilita el manejo simultáneo de tareas como adquisición de sensores, generación de señales PWM, comunicación SPI con el nRF24L01 y transmisión de datos, convirtiéndola en una plataforma adecuada para sistemas distribuidos de monitoreo.

El módulo de radio nRF24L01 es un transceptor de bajo consumo diseñado para comunicaciones digitales en la banda de 2.4 GHz. Este módulo emplea modulación GFSK, soporta velocidades de transmisión de hasta 2 Mbps y dispone de mecanismos automáticos de acuse de recibo y retransmisión que mejoran la confiabilidad del enlace [4]. En el proyecto, este dispositivo permite implementar tanto el enlace de control entre el transmisor y el carro como la transmisión de telemetría hacia la estación base, cumpliendo así los requisitos de baja latencia y operación en canales específicos establecidos por el desafío.

El vehículo integra diversos sensores que cumplen funciones fundamentales en la adquisición de datos. El módulo GPS emplea el sistema satelital global de navegación para obtener coordenadas de latitud, longitud y velocidad mediante sentencias NMEA transmitidas por UART [5]. La unidad inercial IMU combina acelerómetros y giroscopios para capturar los movimientos lineales y rotacionales del vehículo, información útil para analizar estabilidad, vibraciones y comportamiento en curvas [6]. El sensor óptico TCRT5000 utiliza un LED infrarrojo y un fototransistor para detectar cambios de reflectividad, permitiendo identificar el paso por la línea de meta y registrar tiempos de vuelta. Adicionalmente, la medición del voltaje de batería y de la temperatura del motor permite evaluar el estado energético del vehículo y prevenir fallas por sobrecalentamiento o descarga excesiva.

El control del vehículo se realiza mediante un servomotor y un controlador electrónico de velocidad (ESC), ambos basados en señales PWM. En los servomotores estándar, el ángulo de posición se determina mediante pulsos de entre 0.6 ms y 2.2 ms con un periodo de 20 ms (50 Hz), donde los valores extremos representan los límites del movimiento mecánico [7]. La correcta generación de estos pulsos es indispensable para la maniobrabilidad del vehículo y se verificó mediante mediciones con osciloscopio, garantizando que la Raspberry Pi Pico produce señales estables y adecuadas bajo carga.

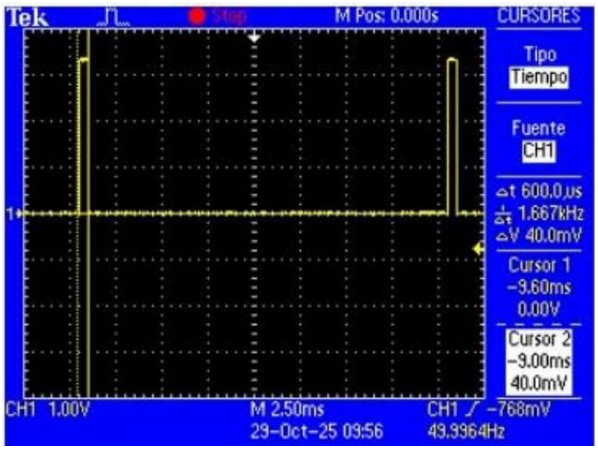
La estación de telemetría recibe los datos provenientes del vehículo mediante un enlace RF y los reenvía a un computador encargado del procesamiento. Un backend implementado en Python actúa como puente entre el puerto serial y un servidor WebSocket, distribuyendo los datos hacia un dashboard desarrollado en una plataforma interactiva. Este tipo de visualización en tiempo real permite monitorear simultáneamente parámetros como aceleraciones, posición GPS, voltaje y temperatura, y facilita el análisis posterior mediante los archivos CSV generados por el sistema [8].

MEDICIONES DE LABORATORIO

Se tomaron las medidas de PWM del servomotor.

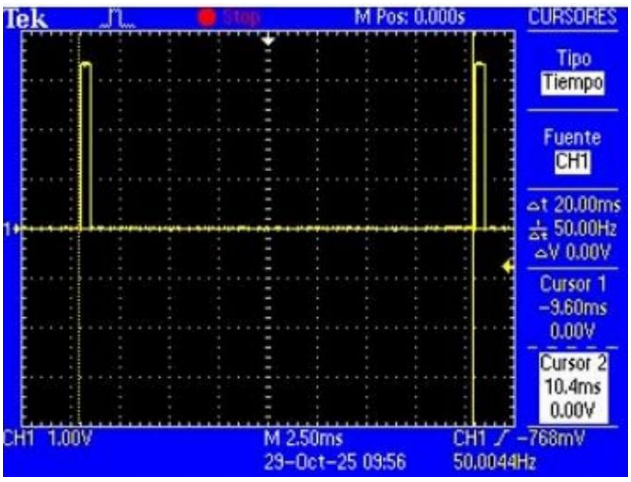
Los servomotores utilizados en el vehículo del proyecto emplean un control basado en modulación por ancho de pulso (PWM), donde la posición del eje depende exclusivamente del ancho del pulso dentro de un período fijo de aproximadamente 20 ms (50 Hz).

Este tipo de señal es fundamental para el sistema de dirección del carro RC, ya que determina los ángulos de giro (0°–180°) y, por tanto, la maniobrabilidad durante la carrera.



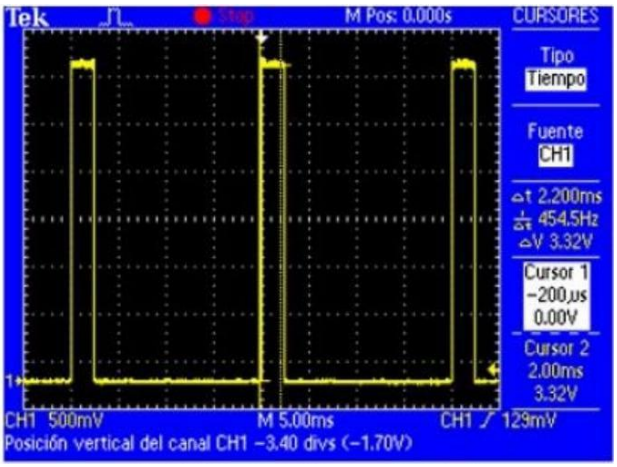
- En el sistema de control del vehículo, este ancho de pulso representa N Es fundamental verificar este valor para evitar:
- Forzar mecánicamente el servo
- Pérdida de control
- Sobrecorriente del ESC o deformación del mecanismo

La medición confirma que los límites de control fueron correctamente mapeados desde los valores del joystick enviados por RF.



Aquí se observa la señal completa donde:

- Período medido ≈ 20 ms → 50 Hz, valor estándar
- Se confirma que la Raspberry Pi Pico genera la señal con la periodicidad adecuada
- La estabilidad del período garantiza que el servo no sufra jitter ni vibraciones

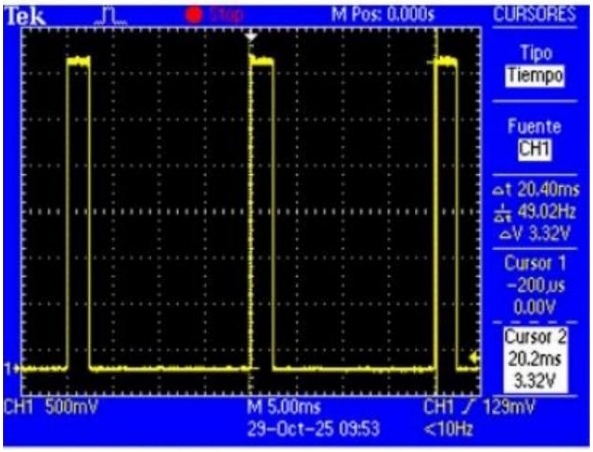


El osciloscopio registra:

- Pulso alto ≈ 2.2 ms, valor típico del ángulo máximo
- Restante bajo hasta los 20 ms
- Frecuencia entre 49 a 50 Hz

Este pulso representa el límite derecho del servo. La medición confirma que:

- El mapeo entre joystick → paquete RF → Pico → PWM funciona correctamente
- El servo tiene recorrido completo útil para la pista
- El sistema respeta los límites mecánicos del vehículo.



Se valida que:

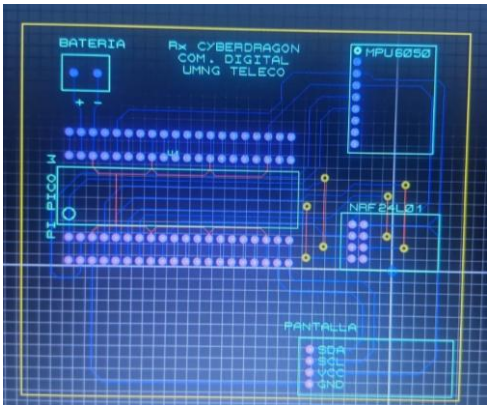
- Frecuencia fija de 50 Hz
- Señal estable incluso en ancho máximo
- Correcto funcionamiento del generador PWM de la Pico bajo carga del servo
- Interpretación para el proyecto

Este comportamiento comprueba que:

- El servo responde de manera lineal al control remoto
- El enlace RF no introduce jitter
- La Raspberry Pi Pico mantiene precisión temporal aun cuando:
    - Recibe paquetes del control remoto
    - Procesa sensores
    - Envia telemetría

Las mediciones realizadas en el osciloscopio permiten validar experimentalmente el funcionamiento del sistema de control del vehículo RC. Se confirma que la Raspberry Pi Pico genera una señal PWM estándar de 50 Hz, con anchos de pulso de 0.6 ms para 0° y 2.2 ms para 180°, lo que demuestra que el mapeo de los comandos provenientes del control remoto se ejecuta correctamente. La estabilidad del período y del ancho de pulso garantiza que el servo opere sin vibraciones, responda adecuadamente durante la carrera y se integre de forma confiable dentro del sistema general de control y telemetría del proyecto.

DISEÑO PCB



Este es el diseño de la PCB donde se pueden observar los principales componentes como la bateria ubicada en la parte superior, el sensor MPU6050, la raspberry Pi Pico W, el modulo NRF24L01 que ayuda a la comunicación inalambrica y por ultimo la pantalla que muestra los datos. Esto facilita el funcionamiento.

La PCB permite conectar de forma organizada el microcontrolador (Pico 2W), el módulo nRF24L01, sensores como el MPU6050 y los circuitos de acondicionamiento.

La PCB funciona como una plataforma inteligente que:

1. Distribuye energía a todos los componentes mediante planos de alimentación y tierra.
2. Conduce señales digitales y analógicas por trazas diseñadas con longitudes, anchos y separaciones adecuadas.
3. Estabiliza el sistema mediante planos de masa, capacitores de desacoplo y topologías de baja interferencia.
4. Garantiza integridad de señal para módulos sensibles como IMU o radiofrecuencia.
5. Integra sensores y módulos en una sola placa para reducir ruido eléctrico y mejorar la confiabilidad.

Las ventjaas de utilizar una PCB son:

- Reducción de fallas por cableado suelto o conexiones inestables.
- Mayor resistencia mecánica, ideal para entornos con vibraciones como un carro RC.
- Mejor integridad de señal en líneas I2C, SPI, UART y señales analógicas.
- Organización limpia y compacta del hardware.

- Facilita la distribución de energía mediante planos de alimentación y tierra.
- Disminuye el ruido eléctrico gracias al diseño controlado de pistas.
- Mayor confiabilidad al tener componentes soldados de forma permanente.
- Disipación térmica más eficiente que un protoboard.
- Reducción del tamaño total del sistema, al integrar varios módulos en una sola placa.
- Facilidad de replicación y producción, útil para fabricar varias unidades del proyecto.
- Aspecto profesional en comparación con cableado Dupont o protoboard.
- Mantenimiento más sencillo, ya que cada componente queda en un lugar claro y fijo.
- Menor interferencia electromagnética, especialmente importante para el módulo RF.
- Mayor durabilidad del sistema, incluso en condiciones de uso prolongado.
- Posibilidad de añadir serigrafía con nombres de pines, zonas y conexiones.

FOTOS, MEDIA Y EVIDENCIA DEL PROTOTIPO

Esta figura presenta el montaje completo del sistema de control y telemetría implementado en el proyecto. En el lado izquierdo se observa el vehículo RC con su servomotor de dirección, cableado de potencia y la PCB principal instalada en el chasis, encargada de adquirir datos de los sensores y controlar los actuadores mediante la Raspberry Pi Pico 2W. En la parte derecha se encuentran dos tarjetas electrónicas diseñadas por el equipo, cada una equipada con un módulo nRF24L01 y una Pico 2W, correspondientes al control remoto y a la estación de telemetría. También se visualiza el joystick analógico utilizado para enviar comandos de aceleración y dirección. La imagen evidencia la integración completa del hardware, el uso de PCBs propias y la implementación física del enlace de comunicaciones en la banda ISM de 2.4 GHz, cumpliendo con los requisitos del RC Cars Telemetry Challenge 2025.

Este corresponde al disñeo exterior, sin dejar cableado o la estructura de los compoenentes para evitar que se dañen en el momenteo de la carrera.

Módulo del Vehículo RC (Carro – Nodo TX/RX)
En la parte izquierda de la imagen se observa la electrónica instalada sobre el chasis del carro. Allí se distinguen:

• Servomotor de dirección:
Responsable del giro del eje delantero. Está conectado a la Raspberry Pi Pico mediante una señal PWM, cuya operación fue verificada mediante osciloscopio en las mediciones previas.

• Cableado de alimentación y distribución de energía:
Incluye líneas hacia el ESC del motor, el servo y los sensores. La organización del cableado garantiza una conexión segura sin falsos contactos durante la carrera.

• PCB principal del vehículo:
Montada bajo el chasis, esta tarjeta integra:

- Conexiones hacia el nRF24L01 (SPI)
- Entradas del TCRT5000
- Líneas del GPS, IMU y sensores
- Puertos de alimentación regulada

La imagen evidencia la instalación física real del hardware diseñado por el equipo.

PCB con Raspberry Pi Pico 2W + nRF24L01 para Control Remoto y Telemetría

En la parte derecha aparecen dos placas verdes PCBs diseñadas por el equipo, cada una equipada con:

• Raspberry Pi Pico 2W

Encargada de procesar los datos, generar señales PWM y manejar sensores y actuadores. Su montaje sobre la PCB garantiza rigidez mecánica y conexiones confiables.

• Transceptor nRF24L01 con antena externa

Este módulo es el encargado del enlace RF en la banda ISM de 2.4 GHz. La antena de alta ganancia incrementa la estabilidad y el alcance del enlace, especialmente durante la telemetría.

• Conectores y pads de expansión

Se observan múltiples conectores para:

- Joysticks del control
- Sensor IMU
- GPS
- Sensores adicionales

Joystick del Control Remoto

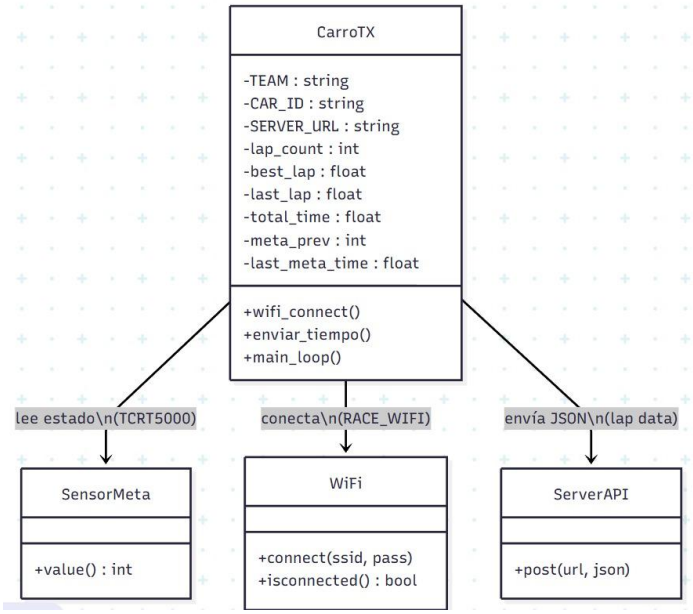
Se aprecia un módulo de joystick analógico, el cual forma parte del control remoto del vehículo. Este dispositivo envía información de:

- Aceleración
- Dirección
- Freno de emergencia

Los valores analógicos se leen desde la Pico, se normalizan y se transmiten como paquetes binarios por nRF24L01 hacia el carro.

DIAGRAMAS UML

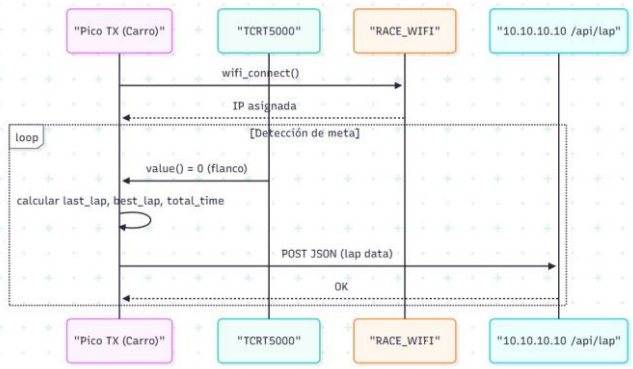
CASOS DE USO



El diagrama de casos de uso muestra las funciones principales que intervienen en el módulo TX del carro dentro del sistema de telemetría. En este esquema se representan los actores “Usuario”, “Carro RC” y “Estación Base”, así como las acciones específicas que cada uno realiza. El usuario ejecuta los comandos de dirección, velocidad y freno; el carro detecta el cruce por meta y envía la información de las vueltas; y la estación base recibe, procesa y registra los datos. Este diagrama permite visualizar de manera general las responsabilidades de cada componente y la interacción funcional entre ellos dentro del proceso de registro de tiempos de carrera.

DIAGRAMA DE SECUENCIA

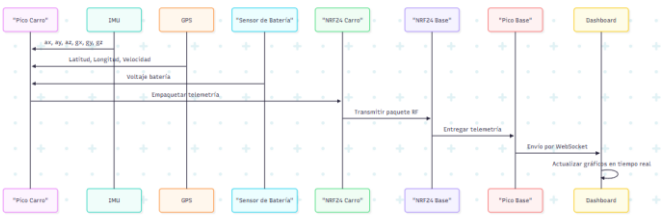
CARRO - CONTROL



El diagrama de secuencia ilustra paso a paso cómo el carro detecta un cruce por meta, procesa los tiempos de vuelta y realiza la comunicación con el servidor. En él se observa que, tras una lectura válida del sensor TCRT5000, la Raspberry Pi Pico calcula el tiempo transcurrido desde la última vuelta, actualiza las variables internas del sistema y construye un mensaje JSON. Posteriormente, este paquete es enviado a través de la red WiFi hacia el servidor 10.10.10.10, el cual responde confirmando la recepción. Este diagrama describe claramente el flujo temporal de ejecución del firmware y la secuencia de eventos que garantizan que cada vuelta quede registrada en tiempo real.

DIAGRAMA DE SECUENCIA

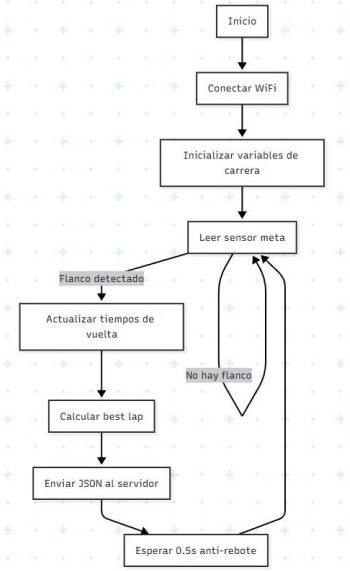
CARRO – ESTACION BASE



Este diagrama de secuencia detalla la interacción entre el carro, los sensores integrados y la estación base durante el envío de telemetría. El flujo inicia con la lectura de la IMU, GPS y sensor de batería, cuyos valores son procesados por el microcontrolador del carro. Luego, el paquete con los datos es transmitido mediante el módulo WiFi hacia la estación base, donde es recibido, decodificado y enviado al dashboard para su visualización. El diagrama permite comprender el proceso completo de adquisición, transmisión y visualización de la

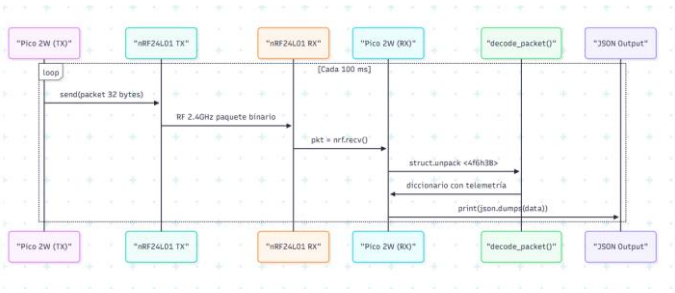
información relacionada con el rendimiento del vehículo, siguiendo la lógica del firmware real.

DIAGRAMA DE LÓGICA



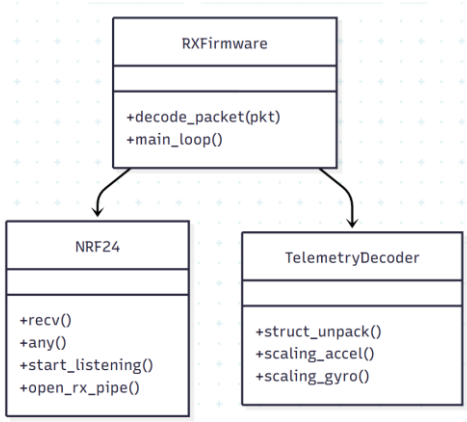
El diagrama de flujo resume la lógica operativa del programa principal en el carro TX. Comienza con la conexión a la red WiFi y la inicialización de variables, seguido por un ciclo continuo de lectura del sensor de meta. Cuando se detecta un flanco válido, el firmware calcula la vuelta, actualiza los tiempos relevantes y envía un mensaje JSON al servidor. Finalmente, implementa un breve mecanismo anti-rebote antes de reiniciar la lectura. Este diagrama sintetiza la lógica de control del firmware y explica visualmente el comportamiento repetitivo del sistema durante la carrera.

SECUENCIA RX- TX



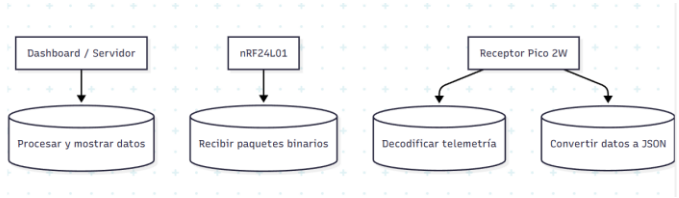
Este diagrama describe el proceso completo de comunicación entre el transmisor y el receptor. El módulo TX empaqueta la telemetría en un bloque binario de 32 bytes mediante struct.pack() y lo envía a través del nRF24L01. El receptor escucha el canal RF, recupera el paquete cuando está disponible y lo entrega al firmware, que lo decodifica mediante struct.unpack("<4f6h3B"). Finalmente, el RX convierte los datos a formato JSON y los envía al backend o dashboard. El diagrama representa fielmente el flujo temporal real del código.

DIAGRAMA RX



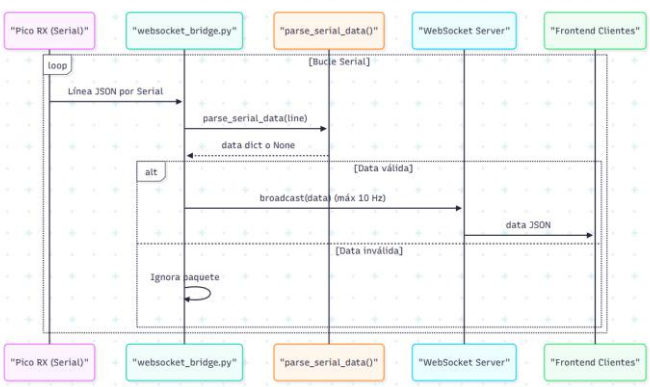
El diagrama de clases modela la estructura interna del receptor. Incluye las clases RXFirmware, responsable del bucle principal y de la recepción; NRF24L01, encargada del manejo del radio; y TelemetryDecoder, encargada de interpretar el paquete binario y aplicar los factores de escala a IMU, PWM y meta. Este diagrama evidencia modularidad y la separación de responsabilidades del firmware.

DIAGRAMA CASOS DE USO – RX



El diagrama de casos de uso presenta las funciones principales del receptor dentro del sistema de telemetría del vehículo RC. El RX recibe paquetes binarios desde el módulo nRF24L01, decodifica la telemetría, la convierte a JSON y la envía al backend encargado de visualizar y registrar los datos. Este diagrama facilita la comprensión general de las responsabilidades del RX dentro de la arquitectura global del proyecto.

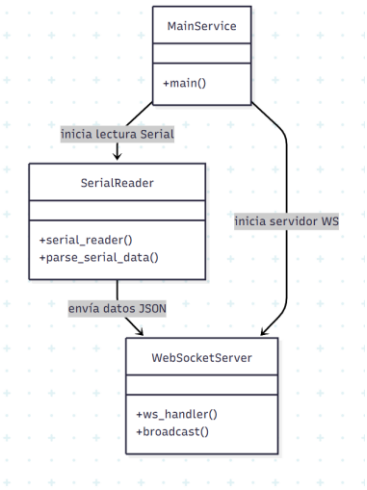
DIAGRAMA DE SECUENCIA SERIAL – WEBSOCKET



Este diagrama representa el flujo completo de comunicación dentro del puente entre la Pico 2W y los clientes WebSocket. El script escucha continuamente el puerto serial, recibe líneas JSON provenientes del receptor de telemetría y las envía al

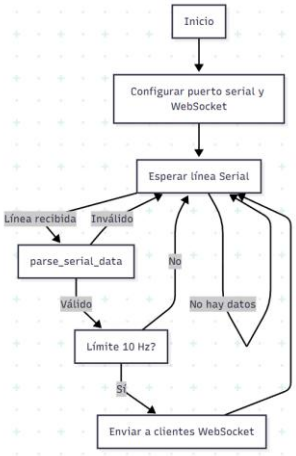
parseador para validar su estructura. Si los datos son correctos, el sistema los difunde a todos los clientes WebSocket conectados respetando un límite de 10 Hz para evitar saturación.

DIAGRAMA DE CLASE BRIDGE



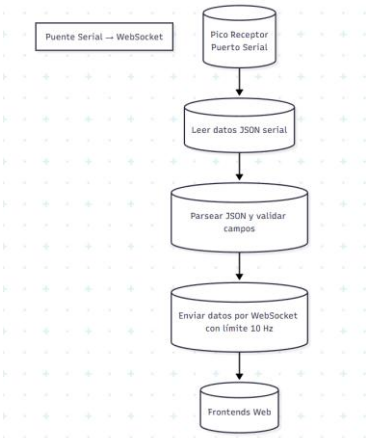
Este diagrama muestra la estructura lógica del sistema, dividido en tres módulos principales: SerialReader, encargado de gestionar la lectura del puerto serial y el análisis del JSON; WebSocketServer, responsable de mantener las conexiones y transmitir los datos a los clientes; y MainService, que inicializa ambos componentes y coordina su ejecución concurrente. Representa claramente la separación de responsabilidades del puente.

DIAGRAMA DE FLUJO – BRIDGE



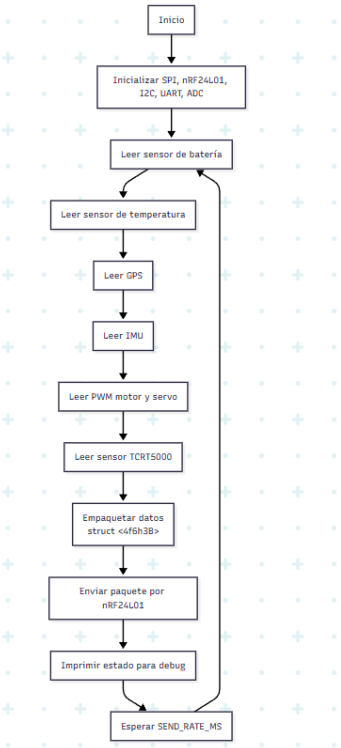
El diagrama de flujo resume la lógica principal del script: apertura de puertos, lectura serial, parseo de datos, validación y envío controlado por tasa hacia los clientes WebSocket. Se observa visiblemente cómo el sistema filtra datos inválidos, respeta un límite temporal mínimo entre envíos y mantiene un ciclo continuo de trabajo. Es una representación ideal para documentación técnica.

DIAGRAMA CASOS DE USO - BRIDGE



El diagrama de casos de uso muestra cómo el puente actúa como intermediario entre la Pico y los clientes web. Sus funciones básicas son leer la telemetría desde el puerto serial, validar y formatear esos datos, y enviarlos hacia el frontend vía WebSocket. El diagrama evidencia claramente cómo el bridge cumple el rol de middleware dentro del ecosistema de telemetría002E

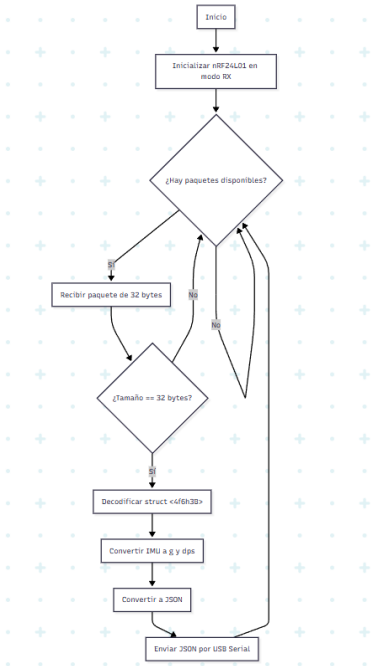
DIAGRAMAS DE FLUJO RX – CARRO



El diagrama de flujo del transmisor describe el proceso cíclico que ejecuta la Raspberry Pi Pico 2W para obtener la telemetría del carro. El sistema se inicializa configurando los buses SPI, I2C, UART y ADC. En cada ciclo se leen los valores de batería, temperatura, GPS, IMU, PWM y sensor TCRT5000. Posteriormente, la información se empaqueta en un frame binario de 32 bytes utilizando la estructura <4f6h3B> y se transmite mediante el nRF24L01. Finalmente, se imprime un

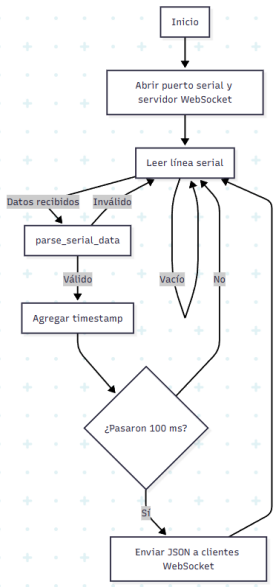
mensaje de depuración y se repite el proceso según la frecuencia establecida.

DIAGRAMA DE FLUJO – RX – ESTACION BASE



Este diagrama muestra el flujo continuo del receptor de telemetría. Una vez configurado el módulo nRF24L01 en modo escucha, la Pico verifica si hay paquetes disponibles. Cada paquete recibido es validado en tamaño y luego decodificado usando la estructura <4f6h3B>. Los datos del acelerómetro y giroscopio se convierten a unidades reales, y toda la telemetría se transforma a formato JSON. Finalmente, la información es enviada por el puerto serial hacia el computador para su posterior procesamiento.

DIAGRAMA DE FLUJO – WEBSOCKET – BRIDGE



El WebSocket Bridge funciona como intermediario entre la Pico y el dashboard web. El sistema abre el puerto serial y el servidor WebSocket, y luego lee continuamente cada línea entrante. Los datos recibidos se validan mediante el parseador JSON, descartando paquetes incompletos. Una vez procesados, se asigna un timestamp y se aplica un límite de frecuencia de 10 Hz para evitar saturación. Los paquetes válidos se envían mediante WebSocket a todos los clientes conectados.

CONCLUSIONES

- La arquitectura de telemetría desarrollada demostró ser robusta y modular, permitiendo la integración de múltiples sensores (IMU, GPS, temperatura, batería y TCRT5000) sin afectar la estabilidad del sistema. El uso de la Raspberry Pi Pico 2W como unidad central permitió un procesamiento eficiente y un empaquetamiento uniforme mediante una trama binaria compacta de 32 bytes.
- El enlace inalámbrico basado en nRF24L01 resultó adecuado para la transmisión en tiempo real, logrando un canal estable en 2.4 GHz y una comunicación a 10 Hz con mínima pérdida de paquetes. Esto valida la efectividad del protocolo simple de transmisión unidireccional, especialmente en aplicaciones donde la latencia es un factor crítico.
- La estación base cumplió exitosamente su función de decodificación y normalización de datos, convirtiendo la trama binaria en información estructurada en JSON. Esto garantiza la compatibilidad con otros sistemas y facilita la integración con interfaces de visualización, dashboards web o aplicaciones de análisis.
- El puente WebSocket implementado en Python se consolidó como una capa fundamental para la transmisión hacia el frontend, garantizando envío estable, validación de paquetes y control de frecuencia (rate limiting). Esta interfaz permitió que múltiples clientes pudieran acceder simultáneamente a la telemetría en tiempo real sin saturar los recursos del sistema.
- Los diagramas de flujo y de bloques permitieron comprender claramente la secuencia de operaciones, desde la adquisición de datos en el carro hasta su visualización final. La documentación gráfica evidencia una arquitectura bien estructurada que facilita la escalabilidad, mantenimiento y posible expansión del sistema hacia nuevas funcionalidades.
- El sistema completo demuestra un enfoque integral de ingeniería, donde electrónica, comunicaciones, programación embebida y programación backend convergen en un mismo proyecto. Esto permitió implementar una plataforma de telemetría realista y funcional, similar a sistemas utilizados en aplicaciones profesionales como robótica móvil, vehículos autónomos y automodelismo de competencia.
- La modularidad del sistema permite expandir el proyecto con facilidad, ya sea añadiendo nuevos sensores, cambiando el transporte inalámbrico (LoRa, WiFi o Bluetooth), integrando un almacenamiento

local o incluyendo algoritmos avanzados de análisis de datos. El diseño actual sirve como una base sólida para futuras mejoras.

REFERENCIAS

[1] MicroPython, machine — functions related to the hardware, Documentación oficial, [En línea]. Disponible en: https://docs.micropython.org/en/latest/library/machine.html. [Accedido: 14-ago-2025].

[2] Raspberry Pi Foundation, Getting started with MicroPython on Raspberry Pi Pico, Documentación oficial, [En línea]. Disponible en: https://www.raspberrypi.com/documentation/microcontrollers/micropython.html. [Accedido: 14-ago-2025].

[3] MathWorks, *Communications Toolbox User's Guide*, MATLAB R2008a Documentation. Disponible en: https://www.mathworks.com/help/com

[4] ANSI/TIA, TIA-232-F: Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange, Telecommunications Industry Association, 1997.

[5] A. S. Tanenbaum and D. Wetherall, Computer Networks, 5th ed. Boston, MA, USA: Pearson, 2011.

[6] Vasquez Rojas, D. (s.f.). *dylanrojas04 - Overview*. GitHub. https://github.com/dylanrojas04

[7] R. G. Gallager, "Low-Density Parity-Check Codes," *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[8] J. Rugeles, Repositorio de prácticas de comunicación digital – I²C con Raspberry Pi Pico, Universidad Militar Nueva Granada. [En línea]. Disponible en: https://github.com/jrugeles/I2C