# Introducción

## Necesidad de los MiniRovers

## KVIs

Los elementos principales que dan valor al proyecto y que se quieren demostrar son:

* Redundancia: en el sector espacial, los fallos de comunicación o de hardware son muy frecuentes. Transmitir los datos por más de un protocolo simultáneamente es una necesidad casi imprescindible.
* Modularidad: reducir el peso y costo de los robots, así como facilitar su independencia es uno de los principales valores del proyecto
* Edge-computing: la descentralización del procesamiento, cuando se tratan de procesos tan complejos como la navegación o el análisis de datos de un sensor muy específico, hace necesario dotar a los rovers de autonomía de procesamiento.
* Optimización de recursos: No obstante, si la carga computacional es demasiado grande, debe tener la posibilidad de derivar ese dato a una capa superior, garantizando que el hardware siempre sea ad-hoc y no exceda las necesidades del dispositivo a controlar.

# Desarrollo del Proyecto

Hemos desarrollado un caso compacto donde la simplicidad se alinea con la diversidad, con el fin de abordar una amplia cantidad de situaciones reales. Para una profunda comprensión de la casuística, proporcionaremos una descripción detallada del escenario.

## Escenario

Hay 4 agentes principales en nuestro escenario

* Rover Sensor: recolecta datos del entorno.
* Rover actuador: extrae muestras del suelo en exploración.
* Ordenador central: secuencia tareas, sirve como punto de recarga y zona de extracción de muestras. También puede analizar datos complejos.
* Red satelital: recibe datos de sensores y computadora central, los almacena en la nube y proporciona a los científicos una interfaz de usuario para interactuar tanto con los rovers como con los datos.

La interconexión es clave. Todos los elementos están conectados entre sí para diferentes propósitos y con protocolos que se adaptan a sus necesidades. Adicionalmente, cada subsistema está provisto de un hardware de procesamiento optimizado para las funciones a realizar (Fig.1).

Diagrama, Esquemático

Descripción generada automáticamente

## Funcionalidades

#### Emergency data sharing

En caso de emergencia, los rovers se comunicarían de manera directa por ESP-NOW, un protocolo rápido, de medio alcance y nativo de estas placas. Para demostrarlo, hemos supuesto que, si uno de los dos detecta un terremoto, se lo comunica al otro mediante ESP-NOW para que ambos cancelen su tarea actual y entren en estado de emergencia, comunicándose así con la central, para que esta actualice sus tareas y puedan dirigirse a la central inmediatamente.

#### Remote control

Para enviar y recibir datos de navegación, los rovers se comunican con la central por LoRa, más apto para distancias largas. Para implementar esta función, la central dispone de un secuenciador de tareas, gracias al cual puede enviar a un rover determinado una ubicación de destino, de manera que, cuando este la alcance (comprobando que sus coordenadas GPS corresponden con las de la ubicación enviada), se ejecute el proceso autónomo del robot. La central, a su vez, recibe datos de la ubicación y estado de los rovers, permitiéndole actuar en caso de timeout y monitorizar el proceso.

#### Triggers

Hay casos específicos donde es necesario un control adicional a la navegación. Por ello, los rovers disponen de varios desencadenantes de emergencia:

* Low Battery: la batería del rover es baja. Se actualiza el estado del rover a “Emergency” y se cancela la tarea actual, a la espera de que la central actualice la secuencia y lo envíe de vuelta al punto de carga y extracción.
* Sample Overflow: se da cuando el rover actuador ha llenado todos los recipientes de extracción de muestras disponibles. Esta emergencia se aborda exactamente igual que la anterior.

Para fomentar el concepto de interconexión, existe un flujo adicional que permite que, pasado un umbral determinado de los sensores del Rover Sensor, se genere un desencadenante (como todos, enviado por LoRa) que permita al secuenciador de tareas crear una tarea que se envíe al Rover Actuador para extraer una muestra de suelo en esa ubicación, pues se considera una ubicación interesante desde el punto de vista científico.

#### Specific data collection

En ocasiones, la carga computacional de algunos datos hace complejo su envío directo a la nube o interfaz de usuario. La extracción de muestras, por ejemplo, necesita de un análisis químico y espectroscópico que arroje luz sobre la composición y naturaleza del suelo. Actualmente, robots como el Perseverance no disponen de la aparamenta necesaria para hacer todos los análisis in-situ, obligando a que sus muestras se analicen una vez el rover retorne a la Tierra. Si dispusiéramos de una central con la habilidad de procesar esos datos, no sería necesario devolverlo a nuestro planeta, pues los datos post-procesados se publicarían en la red y podrían visualizarse de manera remota.

Para ejemplificarlo, hemos generado un Render 3D del planeta en exploración, donde se pueden ubicar en tiempo real los Rovers. Naturalmente, esa información no puede ser procesada directamente por la nube, sino que debe ser la central quien analice y genere el Render.

#### Data aggregation

El estado de los rovers y sus mediciones no es la única información científicamente interesante. La secuenciación de tareas o los Renders como el descrito anteriormente también tienen un valor a analizar. Es por eso por lo que la central envía a la nube por MQTT la tabla donde recoge la secuencia de tareas y los datos adicionales que genera, como el Render.

#### Real-time monitoring

Simultáneamente, toda la información sobre los sensores, estado de los robots, emergencias encontradas, etcétera, se envía por MQTT a la base de datos en la nube. Estos datos se pueden visualizar en una interfaz creada con Node-RED.

#### User prompts & Firmware updates

Aunque la autonomía de este sistema es muy alta, es necesario que los científicos dispongan de las herramientas necesarias para que los rovers estén a su servicio, por ello, nuestra nube de Node-RED también controla un chatbot por el cual los científicos pueden enviar consignas. Se prevé la implementación de actualizaciones remotas de firmware, pero no se ha implementado en el proyecto actual.

Curiosamente, todos los protocolos descritos previamente pueden utilizarse en el espacio. Tan solo faltaría disponer de una antena y un conmutador de protocolo CCSDS para poder transmitir la información de nuestro sistema a la Tierra.

Como síntesis de lo explicado anteriormente, se muestra una tabla que relaciona la funcionalidad con el caso implementado.

|  |  |  |
| --- | --- | --- |
| Sender 🡪 Receiver | Function | Showcase |
| Rover 🡨🡪 Rover | Emergency Data | Earthquake (accelerometer) |
| Rover 🡪Central | Triggers | * Low Battery * Sample Overflow * Interesting waypoint |
| Remote Control | * Coordinates * Status |
| Specific Data | * Latitude * Longitude   (Central creates a 3D render) |
| Central 🡪Rover | Remote Control | * Desired Coordinates * Task information |
| Rover 🡪Satellite | Real-time monitoring | * Sensor Data * Rover status |
| Satellite 🡪Rover | Firmware updates | * FOTA (not implemented) |
| Central 🡪Satellite | Data aggregation | * Rover Tasks data * 3D render output |
| Satellite 🡪Central | User prompts | Scientists’ waypoints |

## Comunicación y tramas de datos

Para conseguir implementar todas las funciones, nos hemos basado en cinco tramas de datos de una nomenclatura similar, que se trasmitirán simultáneamente por los distintos canales de comunicación dispuestos arriba, y que serán escuchados por el receptor pertinente. De este modo, cada trama tendrá asociado su topic de MQTT, lo cual nos permite clasificarlas de acuerdo con la tabla siguiente:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Usage | Sender 🡪 Receiver | Fields & Content |
| II15/Rovers[[1]](#footnote-1) |  | SensorRover  ActuatorRover | "Sender":"SensorRover” / “ActuatorRover ",  "Timestamp”: integer,  "Latitude”: double  "Longitude”: double  "Status": "Task finished. Available” / ”Task ongoing. Not available” / “Emergency state" |
| Remote control & Triggers | Central |
| Real-time monitoring: Map | Node-RED |
| II15/Sensors |  | SensorRover | "Sender": "SensorRover",  "Timestamp": integer,  "Latitude": double  "Longitude”: double  "Altitude": float  "Temperature": float  "Humidity": float  "Pressure": float  "Radiation": float  "Status": “ “/”Interesting Humidity Pressure Radiation" |
| Triggers: interesting waypoint | Central |
| Real-time monitoring: sensor data | Node-RED |
| II15/Emergency |  | SensorRover  ActuatorRover | "Sender":"SensorRover” / “ActuatorRover ",  "Timestamp” : integer,  "Latitude” : double  "Longitude" : double  "Status": "Earthquake” / “Low battery” / “Sample overflow" |
| Real-time monitoring: Emergency notification | Node-RED |
| Emergency data sharing | SensorRover  ActuatorRover |
| II15/Commands[[2]](#footnote-2) |  | Central | "Sender":"SensorRover”/”ActuatorRover”/”Auto”/”Central”/”User",  "RoverId":"SensorRover/ActuatorRover",  "Timestamp”: integer,  "Latitude”: double  "Longitude”: double  "Priority": 1,  "Timeout": integer |
| Remote control: waypoint sending | SensorRover  ActuatorRover |
| II15/Table |  | Central | "Sender":"SensorRover”/”ActuatorRover”/”Auto”/”Central”/”User",  "RoverId":"SensorRover/ActuatorRover",  "Timestamp”: integer,  "Latitude”: double  "Longitude”: double  "Priority": 1 / 2 / 3…  "Timeout": integer |
| Commands table | Node-RED |

En síntesis, Rovers transmite el estado de los rovers y sus coordenadas, Sensors el valor de los sensores y si se encuentra alguna ubicación de interés, Emergency la ubicación y motivo de la emergencia, Commands las nuevas ubicaciones de destino y el creador de estas, y Table exporta la tabla de comandos completa. De manera adicional, se envía el archivo html del render por MQTT de la Central a Node-RED.

## MiniRovers

Nuestra prueba consiste en dos MiniRovers similares en morfología, pero de distinta funcionalidad.

### Hardware

Los Rovers cuentan con sensores, actuadores, módulos de comunicación y un multiplexor. Todos ellos son controlados por una ESP32, que es el núcleo del rover.

#### Sensores

En el caso concreto del sensor, se han empleado varios módulos[[3]](#footnote-3) para medir distintos valores ambientales que permitan definir umbrales de interés[[4]](#footnote-4). Dichos módulos son:

* BMP180: sensor de presión que utiliza un transductor piezorresistivo para convertir la presión en una señal eléctrica. Esta señal es procesada por un circuito de acondicionamiento para obtener la lectura de presión. El umbral definido es de 1033 hPa.
* MPU-6050: sensor inercial que utiliza un conjunto de microelectromecanismos (MEMS) para medir la velocidad angular (giroscopio) y la aceleración lineal (acelerómetro). Estos MEMS están conectados a un circuito de acondicionamiento que traduce los movimientos físicos en señales eléctricas. El umbral definido es una aceleración superior a 15 m/s
* GUVA-S12SD: sensor de radiación ultravioleta que utiliza un fotodiodo sensible a la radiación UV. Cuando la radiación UV incide sobre el fotodiodo, se generan corrientes eléctricas proporcionales a la intensidad de la radiación. Estas corrientes son amplificadas y convertidas en señales digitales mediante un circuito de acondicionamiento.

#### Actuadores

Para mostrar una demo, se ha utilizado un montaje simple de dos motores con ruedas y el módulo L298N, un controlador de motor dual que utiliza puentes H para controlar la dirección y velocidad de los motores de corriente continua o paso a paso. Permite control bidireccional y regulación de la velocidad mediante señales PWM.

Para ejemplificar la extracción de muestras, se emplea un led que se encenderá cuando la extracción esté en curso.

#### Módulos de Comunicación

* Para obtener la ubicación del rover en coordenadas (latitud y longitud), se ha empleado el NEO-6M: módulo GPS que contiene un receptor GPS y un circuito de procesamiento. El receptor GPS se comunica con múltiples satélites para obtener señales de posicionamiento. Estas señales son procesadas por el circuito, que determina la ubicación geográfica en función de la información recibida de los satélites.
* La transmisión por LoRa ha sido posible gracias al módulo SX1278, un transceptor de largo alcance que recibe datos por LoRa y los transmite a la ESP gracias al protocolo SPI
* Para evitar la saturación del puerto I2C de la placa, se ha empleado un multiplexor I2C que conmute el sensor de lectura.

### Comunicaciones

Como se muestra en el diagrama, nuestro sistema emplea diversos protocolos simultáneos, cada uno con un fin específico. Esto ha supuesto un reto técnico, pues la ESP32 sólo dispone de una antena WiFi (por la cual hay que enviar parte de la información por dos protocolos diferentes) y los tiempos de latencia son críticos (ya que el rover debe estar escuchando casi en tiempo real a muchos emisores). A continuación, se describen los protocolos usados, así como los elementos más destacables del código.

#### ESP-NOW

La comunicación básica con ESP-NOW emplea una MAC, que se puede obtener fácilmente con funciones de la librería. Dicha MAC, introducida en Esp\_now\_peer\_info\_t, devuelve todos los datos necesarios para una correcta comunicación entre dos placas ESP32.

Se han definido dos funciones de callback, una para el envío y otra para la recepción de datos, que facilitan la labor de identificar fallos de transmisión de datos. En la callback de recepción, se ha introducido una función de memcpy(), que almacena los datos recibidos en una variable de la placa.

A continuación, basta con inicializar el módulo y, en el bucle principal del programa, enviar los datos pertinentes con Esp\_now\_send(). Nótese que, para el envío en ESP-NOW, es necesario indicar la longitud del mensaje.

#### WiFi

El protocolo WiFi, realmente, se emplea para usar MQTT. Dado que este formato de comunicación se ha usado con asiduidad a lo largo de la asignatura, no se explicará en detalle el código asociado.

No obstante, sí cabe destacar que, en WiFi, existen varias configuraciones de conexión:

* Station: permite que el dispositivo se conecte a una red WiFi existente como cliente.
* Access Point: permite que el dispositivo cree su propia red WiFi a la cual otros dispositivos se pueden conectar.
* APSTA: fusiona ambas funcionalidades simultáneamente.

Dada la naturaleza concurrente del proyecto, para hacer funcionar MQTT y ESP-NOW simultáneamente era necesario emplear el modo APSTA.

##### HANDLE NETWORK

El punto más interesante de este binomio de protocolos es que, como bien se ha dicho previamente, sólo se dispone de una antena WiFi para transmitir dos protocolos simultáneos. Eso implica que, si la conexión del protocolo WiFi fallase, automáticamente se perdería la conexión en ESP-NOW. Para evitarlo, se ha definido un flujo para habilitar ESP-NOW sin WiFi, y sin tener que reiniciar la placa.

[Imagen del flujo]

enable\_esp\_now() modifica el modo de configuración a Station, para habilitar solo el protocolo en cuestión.

#### LORA

LoRa (Long Range) es un protocolo de red de área extensa de baja potencia diseñado para la comunicación a larga distancia entre dispositivos. A diferencia del Wi-Fi, LoRa opera en bandas de frecuencia no licenciadas, minimizando la interferencia en aplicaciones IoT. Es por ello que se ha usado como comunicación principal entre la central y los rovers. Además, es muy sencillo de programar en ESP32. El código empleado se muestra a continuación.

* Init\_lora() define pines y begin
* Lora\_recv\_callback(lora.parsePacket())
  + Mensaje no vacío
  + Address correcto
* Send\_msg\_lora(msg)
  + Lora.beginPacket()
  + Lora.write(receiver\_address)
  + Lora.write(local\_address)
  + Lora.write(msgid)
  + Lora.write(msg.length)
  + lora.print(msg)
  + lora.endpacket()
  + msgid++

### Máquina de estados

Diagrama

Descripción generada automáticamente

El proceso seguido por el robot cuenta con una máquina de estados de 7 etapas, con 3 desencadenantes.

En el flujo principal, el robot comienza en el estado “FINISHED” y, si recibe un mensaje de la central (detectado cuando contiene el campo “Priority”), el desencadenante EXECUTE toma el valor “true”, permitiendo el avance de la máquina a “MOVING”, donde se desplaza al punto solicitado. Una vez haya finalizado la navegación (DONE activado), pasa a tomar la medida o extraer la muestra en “MEASURING”, que, al acabar, vuelve a “FINISHED”.

El flujo secundario se da si sucede alguno de los casos de emergencia o desencadenante asociado (“Low Battery”, “Sample Overflow” y “Earthquake”). Así, si en alguno de los casos se da un caso de emergencia, el rover que detecta el fallo lo envía a la central, que actualiza las tareas y lo envía de vuelta a origen. Una vez llegado, e independientemente de la naturaleza de la emergencia, el rover carga y, en el caso del ActuatorRover, descarga sus muestras. Una vez acabado todo el proceso, está listo para salir de nuevo.

Existen dos desencadenantes adicionales para el control:

* SEND: usado para enviar emergencias por mqtt solo cada 5 segundos
* FROM\_PARTNER: usado para evitar que el estado de emergencia entre en bucle y que los rovers se envíen indefinidamente que hay un terremoto.

Para simular la casuística descrita, a falta de recorrer la distancia suficiente con los rovers para generar una consigna, y a falta también de tener un sensor de batería, se ha empleado una función que usa el botón de la ESP32 para generar los flags, button\_handler, que en caso de click, envía un “DONE”, en caso de click largo, supone un “Low Battery”, y en caso de doble click, genera un “EXECUTE”.

### Tramas y envío de datos

#### II15/Rovers

Los campos asociados a este topic se mandan por LoRa y MQTT cada 5 segundos.

* Sender será SensorRover o ActuatorRover según quién transmita la información a Central
* Timestamp tomará el valor de sincronización inicial, más los milisegundos pasados desde entonces, obtenidos con la función millis()
* Latitude y Longitude corresponderán a los datos del GPS
* Status corresponderá con “Task finished…” si está en el estado FINISHED, “Task ongoing…” si está en MOVING o MEASURING y “Emergency State” si está en EMERGENCY, ya sea por “Low Battery”, “Sample Overflow” o “Earthquake”

#### II15/Sensors

Los campos asociados a este topic se mandan por LoRa y MQTT cada vez que se llega a la ubicación de destino.

* Sender siempre será SensorRover
* Timestamp se define de la misma forma que en el topic anterior.
* Latitude, Longitude, Altitude, Temperature, Humidity, Pressure y Radiation corresponderán a los datos de los sensores
* “Status” se envía vacío si la ubicación no es de interés científico, y con “Interesting” más el nombre del campo del que se haya obtenido un valor interesante en caso de que algún valor supere un umbral.

#### II15/Emergency

Los campos asociados a este topic se enviarán por LoRa o MQTT cada vez que se dé un estado de emergencia.

* Sender será SensorRover o ActuatorRover según quién transmita la información a Central
* Timestamp se define igual que en los topics anteriores
* Latitude y Longitude también
* Status tomará el valor del tipo de emergencia encontrado (“Low Battery”, “Sample Overflow” o “Earthquake”).

## Chatbot de Telegram

El Chatbot tiene tres funciones principales:

1. Consultar las coordenadas actuales de los rovers
2. Enviar una consigna para que uno de los rovers se dirija a un punto determinado
3. Descargar como .csv los datos de los sensores para analizarlos en el laboratorio

Diagrama

Descripción generada automáticamentePara desarrollarlo, se ha recurrido a Node-RED, implementando el flujo que se muestra a continuación:

1. El “Status” de Rovers indica a la Central que hay un caso de emergencia, pero no se indica cuál, porque es indiferente para la navegación. Sí se indica en Emergency, que se envía a Node-RED, para notificar a los científicos [↑](#footnote-ref-1)
2. En el caso específico de commands, RoverId corresponde al rover al que se quiere mandar el comando, mientras que Sender es quien lo genera. [↑](#footnote-ref-2)
3. Nótese que ninguna de la sensórica usada es apta para el uso espacial. No obstante, la obtención de módulos funcionales en ambientes extraterrestres es costosa y requeriría de un testeo, procesado e inversión que excede el ámbito de la asignatura. [↑](#footnote-ref-3)
4. Los umbrales definidos para activar el desencadenante de “ubicación de interés” no tienen validez científica, sino meramente ilustrativa. [↑](#footnote-ref-4)