📘 Documentación Técnica - Sistema de Simulación y Localización de Trackers vía MQTT

# 1. Descripción General del Proyecto

Simulación de posicionamiento de transbordadores, activos y lógica de aguja mediante mensajes MQTT.

# 2. Tecnologías Utilizadas

- Node.js: Lógica de simulación y backend.  
- Mosquitto (MQTT broker): Comunicación de mensajes en tiempo real.  
- MySQL: Base de datos de persistencia.  
- Sequelize: ORM para Node.js.

# 3. Estructura del Proyecto

📁 mqtt-simulation/  
 ├── 📁 transbordador-simulation/  
 │ ├── publisher.js  
 │ ├── listener.js  
 │ ├── utils.js  
 ├── 📁 aguja-simulation/  
 │ ├── cases/  
 │ ├── publisher.js  
 │ ├── listener.js  
 │ ├── utils.js  
  
📁 models/  
 ├── TrackerTransbordador.js  
 ├── BalizasTransbordador.js  
 ├── TrackerActivos.js  
  
📁 services/  
 ├── TrackerTransbordador.Service.js  
 ├── BalizasTransbordador.Service.js

├── TrackerActivos.Service.js  
  
📁 sqlScripts/  
 ├── Creacion\_tablas\_tracker\_agujas\_activos\_balizas.sql

├── Insertar\_tablas\_tracker\_agujas\_activos\_balizas.sql

# 4. Simulación MQTT

## 4.1 Objetivo

Simular el comportamiento de los transbordadores, balizas y activos mediante envío de datos vía MQTT.  
- Determinar:  
 • En qué vía está el transbordador según intensidad de balizas.  
 • Si un transbordador está acoplado a otro transbordador.  
 • Si un activo está en un transbordador.

• Determinar la ubicación de un tracker mediante posición de agujas y señal de baliza.

## 4.2 MQTT Topics

|  |  |
| --- | --- |
| Topic | Descripción |
| 'position/tracker' | Envía datos de balizas de transbordadores |
| position/asset | Envía datos de balizas de activos |
| simulation/tracker | Cambios en el estado de las agujas |

## 4.3 transbordador-simulation

- publisher.js: simula envío de balizas de transbordadores.  
- listener.js: escucha las señales simuladas para validación.  
- /utils: funciones comunes como generación de payloads o intensidades aleatorias.

## 4.4 aguja-simulation

- cases/: casos de prueba que modifican el estado de las agujas.  
- publisher.js: publica cambios de estado de aguja.  
- listener.js: escucha las señales que afectan el camino de un tracker.  
- /utils: lógica de decisión de vía según intensidades y configuración de aguja.

# 5. Modelos y Servicios

## 5.1 TrackerTransbordador

Modela los transbordadores con sus trackers asociados.

## 5.2 BalizasTransbordador

Modela las balizas Bluetooth fijas en los transbordadores.

## 5.3 TrackerActivos

Modela los activos (cargas u objetos) que se montan en los transbordadores.

Cada servicio (\*.service.js) contiene CRUD completo y funciones auxiliares como búsqueda por cercanía, o por ID de tracker.

# 6. Base de Datos

* Script: Creacion\_tablas\_tracker\_agujas\_activos\_balizas.sql  
  Tablas incluidas:  
   trackerstranbordador  
   balizastransbordador  
   trackeractivos

balizas  
 agujas

# Cómo Probar el Servicio MQTT

Para realizar pruebas del sistema de simulación MQTT y verificar el funcionamiento de los transbordadores y las agujas, se deben seguir los siguientes pasos:

## ✅ Prueba de Transbordadores

1. Navegar a la ruta del proyecto:  
  
 backend\mqtt-simulation\transbordador-simulation\

2. Ejecutar el simulador de transbordadores con el siguiente comando:  
  
 node runAll.js

Este script inicia la simulación completa de los transbordadores, incluyendo la publicación de datos MQTT y el manejo de eventos.

## ✅ Prueba de Agujas

Tienes dos opciones para probar la simulación de las agujas:

* Opción 1: Ejecutar el simulador completo

1. Navegar a la carpeta:  
  
 backend\mqtt-simulation\aguja-simulation\

2. Ejecutar el simulador de agujas:  
  
 node runAguja.js

* Opción 2: Ejecutar casos individuales

1. Ir a la subcarpeta con los casos de prueba:  
  
 backend\mqtt-simulation\aguja-simulation\cases\

2. Ejecutar cualquier archivo individual según el caso que se desee probar:  
  
 node nombreDelArchivo.js  
 Por ejemplo:  
 node simulateCase1.js