Documento de diseño

PROYECTO DE APLICACIONES DE BIOMETRÍA Y MEDIO AMBIENTE

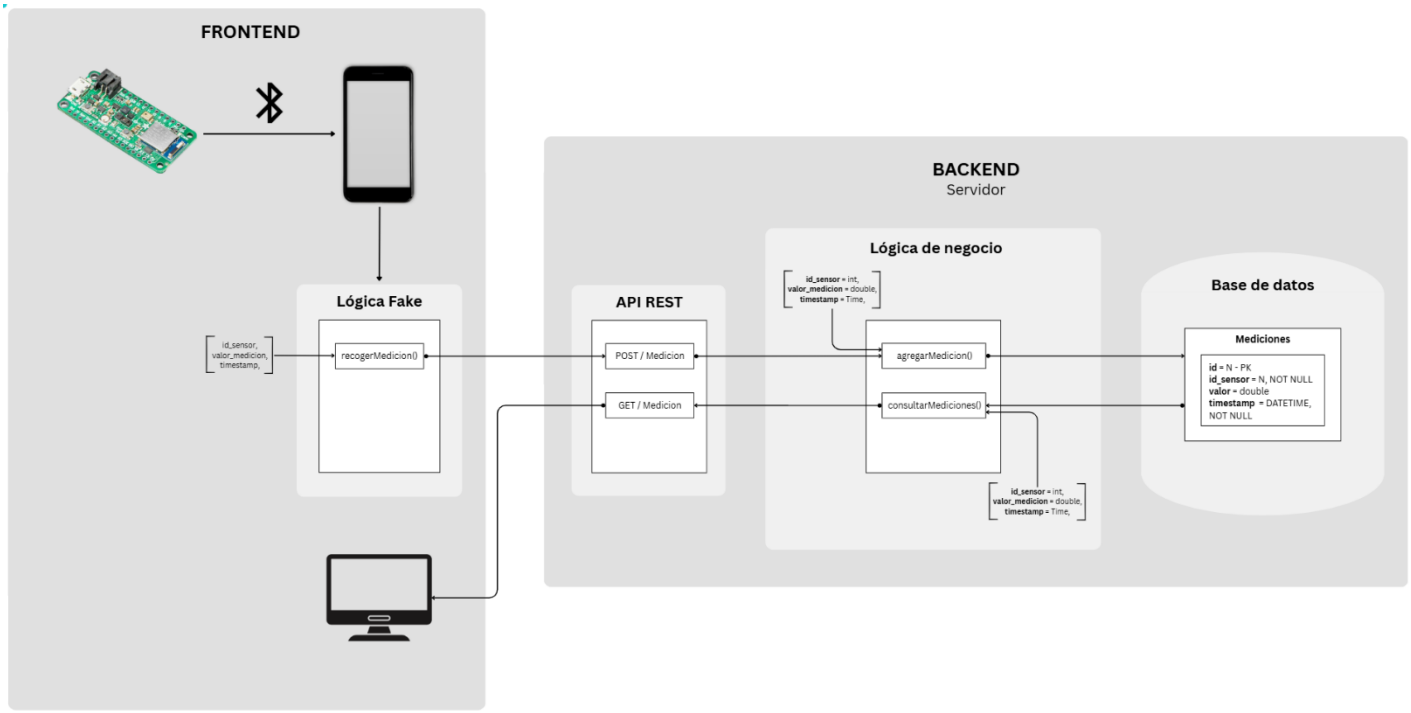
Sprint #0

Autor: Santiago Fuenmayor Ruiz

26 de septiembre de 2025

Tabla de contenido

ARQUITECTURA DEL SISTEMA



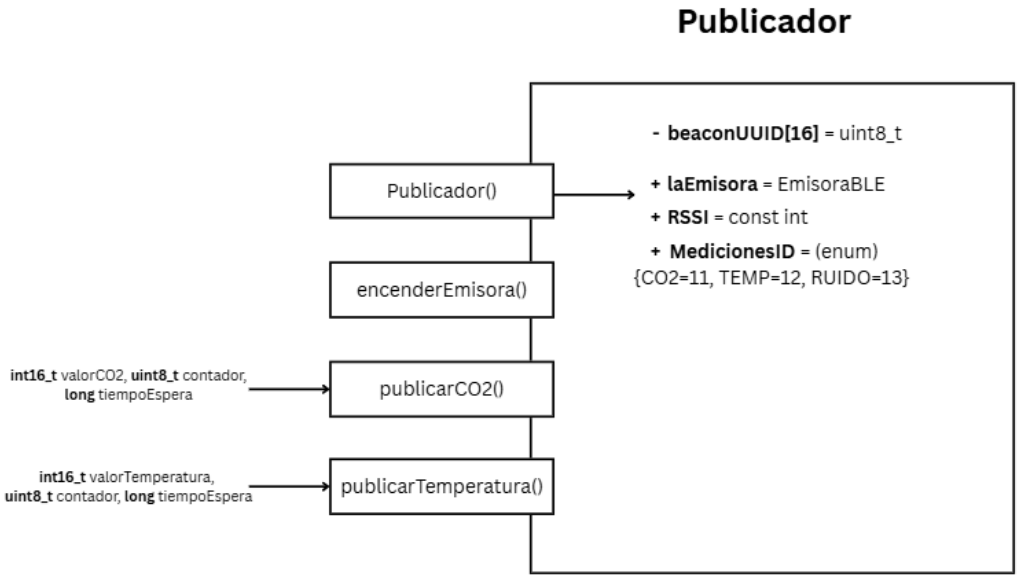
ARDUINO

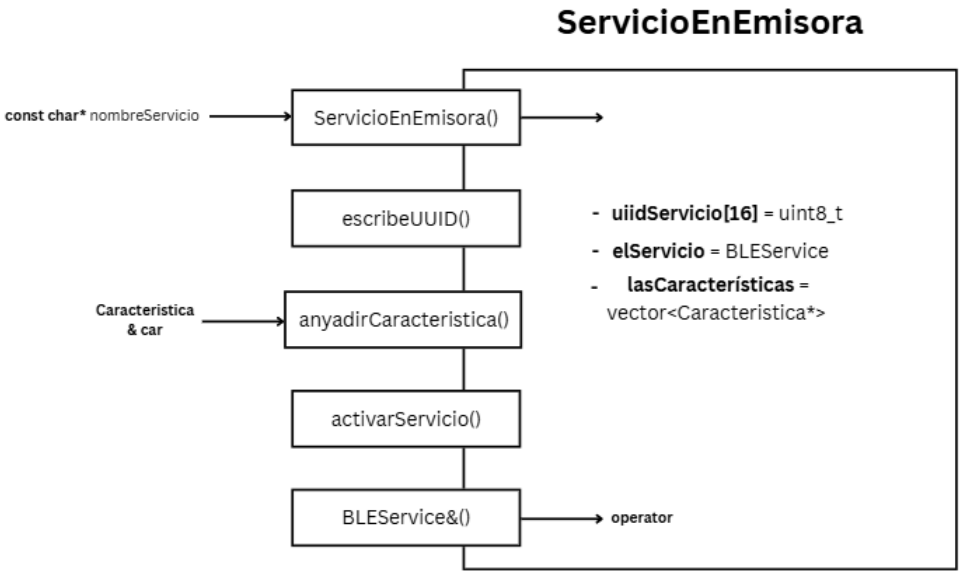
En la parte de Arduino, el código implementa el nodo sensor encargado de simular la lectura de parámetros ambientales y publicarlos mediante Bluetooth Low Energy (BLE) en formato iBeacon. Para ello se estructura en distintas clases con funciones bien definidas:

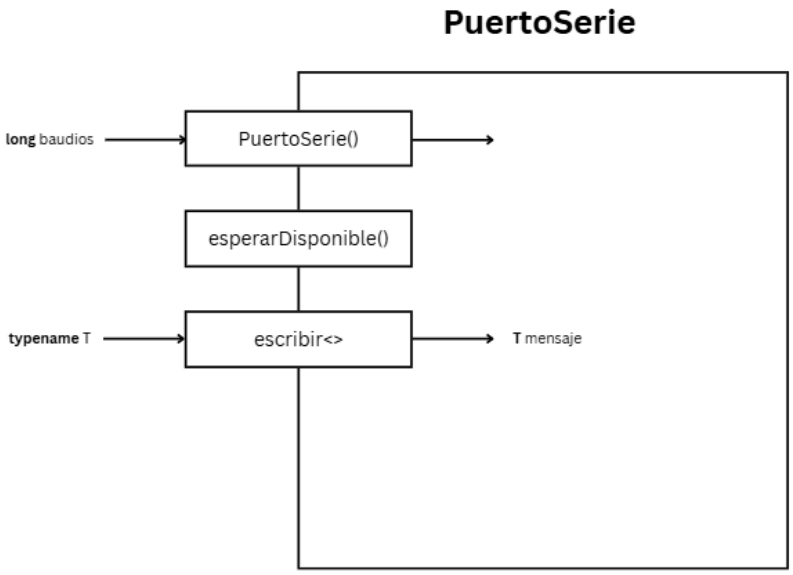
Medidor se encarga de obtener valores de CO_2 y temperatura (actualmente simulados), **Publicador** gestiona la creación de los paquetes iBeacon y su emisión, y **EmisoraBLE** abstrae la configuración del módulo BLE para encenderlo, detenerlo o iniciar la transmisión. Además, se incluyen clases de apoyo como **PuertoSerie**, utilizada para depuración a través del monitor serie, **LED**, que actúa como indicador visual del envío de datos, y **ServicioEnEmisora**, pensada para manejar servicios y características BLE en futuras ampliaciones.

El flujo básico del sistema consiste en inicializar la emisora y el puerto serie, simular las medidas, empaquetarlas en un anuncio iBeacon y transmitirlas durante un tiempo definido, con la posibilidad de indicar el proceso mediante un parpadeo del LED. De este modo, el Arduino actúa como un beacon emisor de datos ambientales, que posteriormente son detectados por la aplicación móvil y enviados al servidor para su almacenamiento y consulta.

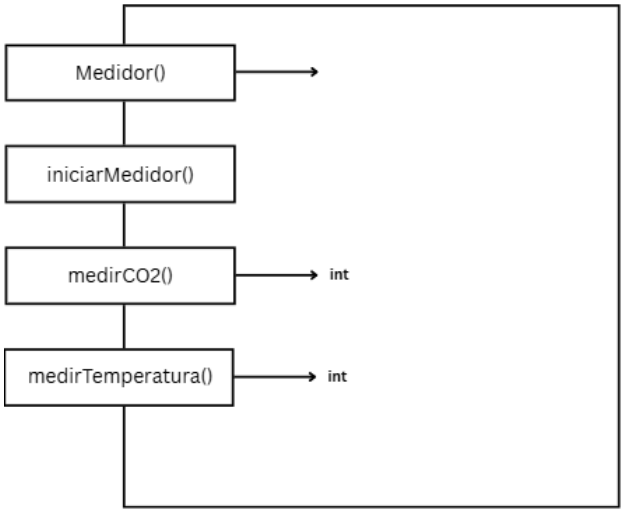
INGENIERÍA INVERSA



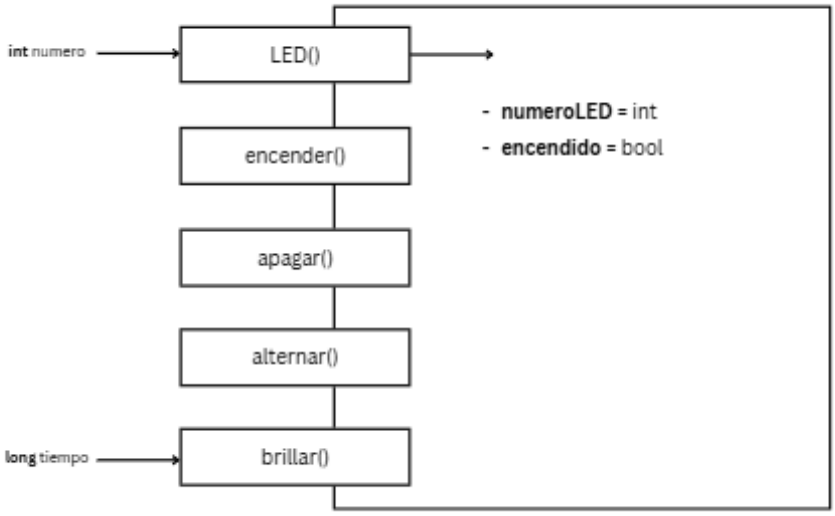




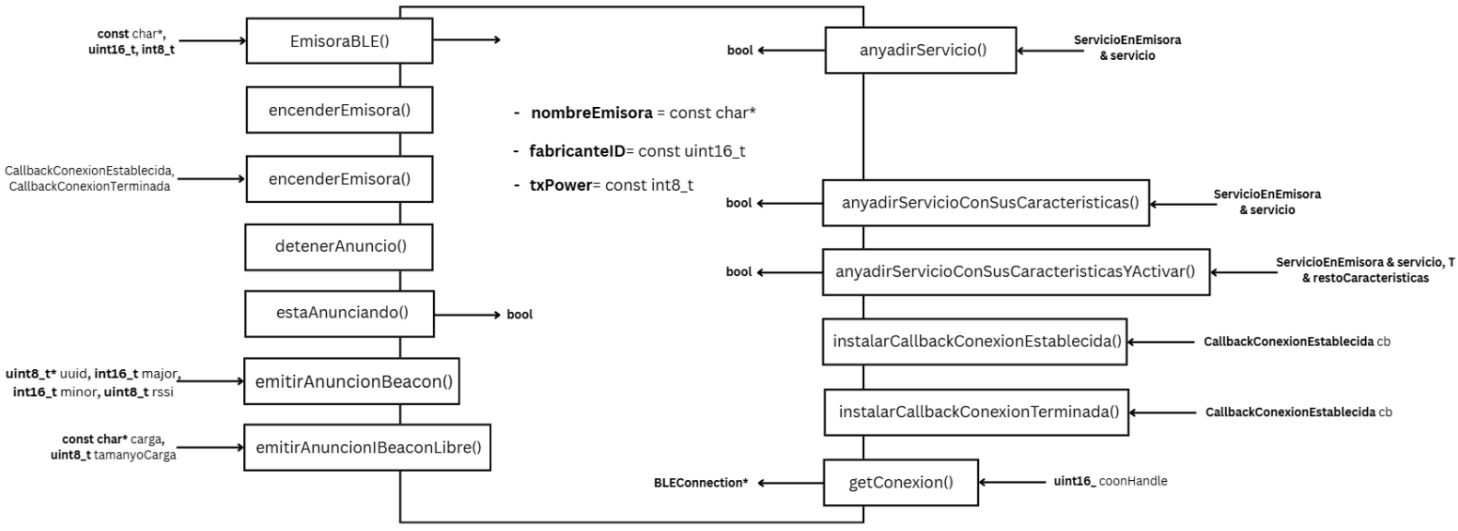
Medidor

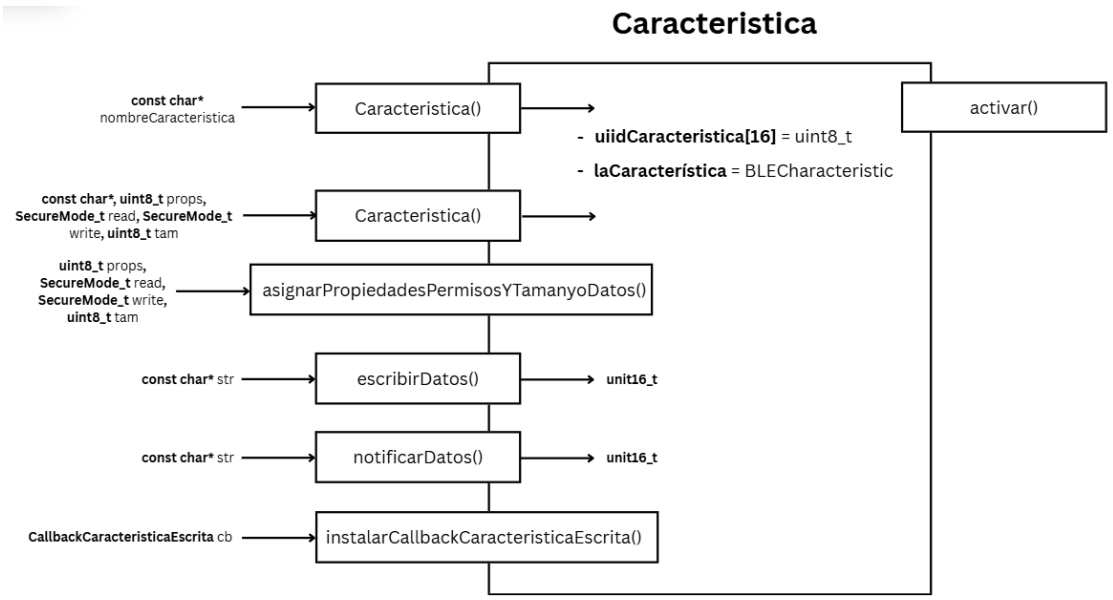


LED



EmisoraBLE




Caracteristica
const char*
nombreCaracteristica
Caracteristica()
- uiidCaracteristica[16] = uint8_t
- laCaracterística = BLECharacteristic
activar()
const char*, uint8_t props,
SecureMode_t read, SecureMode_t
write, uint8_t tam
Caracteristica()
uint8_t props,
SecureMode_t read,
SecureMode_t write,
uint8_t tam
asignarPropiedadesPermisosYTamanyoDatos()
const char* str
escribirDatos()
unit16_t
const char* str
notificarDatos()
unit16_t
CallbackCaracteristicaEscrita cb
instalarCallbackCaracteristicaEscrita()

TELÉFONO

En la parte de **Android**, la aplicación funciona como receptor de los anuncios iBeacon enviados por el Arduino, interpretando las tramas y mostrando los valores medidos. Se apoya en las clases MainActivity para la gestión de la interfaz y detección, TramaIBeacon para decodificar los paquetes recibidos y Utilidades para funciones auxiliares, mientras que el AndroidManifest.xml define los permisos de Bluetooth y localización. En este Sprint 0 su funcionamiento es básico o simulado, mostrando cómo se prepararían los datos para su posterior envío al servidor.

INGENIERIA INVERSA