laSalle ENG
Ramon Llull University

Escola Tècnica Superior d'Enginyeria Electrònica i Informàtica La Salle

Treball Final de Màster

Màster Universitari en Enginyeria de Telecomunicació

Dispositivo wearable para la localización de operarios en entornos ferroviarios

Alumne

Sergio del Horno Torres

Professor Ponent

Joan Lluís Pijoan Vidal

ACTA DE L'EXAMEN DEL TREBALL FI DE CARRERA

Reunit el Tribunal qualificador en el dia de la data, l'alumne

D. Sergio del Horno Torres

va exposar el seu Treball de Fi de Carrera, el qual va tractar sobre el tema següent:

Dispositivo wearable para la localización de operarios en entornos ferroviarios

Acabada l'exposició i contestades per part de l'alumne les objeccions formulades pels Srs. membres del tribunal, aquest valorà l'esmentat Treball amb la qualificació de

Barcelona,

VOCAL DEL TRIBUNAL VOCAL DEL TRIBUNAL

PRESIDENT DEL TRIBUNAL

Resumen

Desde la sección de comunicaciones y *software* de la empresa SENER, surge la oportunidad de participar en uno de sus desarrollos más ambicioso. Se trata del sistema *BlockSAT®*, encargado de monitorizar una línea ferroviaria y gestionar el bloqueo de cantones según la posición de los trenes sobre la línea. En el desarrollo *BlockSAT®*, se identifica la necesidad de añadir un subsistema que permita la representación de los operarios que realicen trabajos en la vía, sobre el sinóptico de la línea representada en el centro de control. Para ello, se debe diseñar el nuevo subsistema y realizar un prototipo funcional que cumpla los requisitos principales establecidos por los responsables del proyecto *BlockSAT®*. Estos son: bajo coste, integrable con *BlockSAT®* y dimensiones reducidas para integrar en el equipo o ropa del operario.

El presente trabajo final de máster consiste en el diseño de este sistema, llamado *BlockSAT® Portable*, y en el desarrollo y fabricación del prototipo. Para lograrlo, en primera instancia, se define un listado de requisitos que debe cumplir el sistema; a continuación, se diseña el sistema; y, por último, se pasa al desarrollo y fabricación del prototipo.

El desarrollo del sistema *BlockSAT® Portable* consiste, por un lado, en un estudio de mercado para la elección de los componentes que mejor se adapten al diseño propuesto, para después adquirirlos y ensamblar el dispositivo. Y por otro lado, en el desarrollo, que se centra en la programación de las aplicaciones que forman el sistema. Este desarrollo *software* se divide en el *firmware* del dispositivo y en una aplicación para *smartphones* que será la interfaz entre el operario y el sistema. Estos desarrollos dotarán al sistema de la capacidad de obtener la posición del operario y de emitir solicitudes para realizar los trabajos en vía. Esta información debe ser transmitida al centro de control del sistema *BlockSAT®* desde donde, con la información de la posición, se podrán enviar respuestas y avisos al operario.

Resum

Des de la secció de comunicacions i *software* de l'empresa SENER, sorgeix l'oportunitat de participar en un dels seus desenvolupaments més ambiciós. Es tracta del sistema *BlockSAT®*, encarregat de monitoritzar una línia ferroviària i gestionar el bloqueig de cantons segons la posició dels trens sobre la línia. Al desenvolupament *BlockSAT®*, s'identifica la necessitat d'afegir un subsistema que permeti la representació dels operaris que realitzin feines a la via, sobre el sinòptic de la línia representada al centre de control. Per a això, s'ha de dissenyar el nou subsistema i realitzar un prototip funcional que compleixi els requisits principals establerts pels responsables del projecte *BlockSAT®*. Aquests són: baix cost, integrable amb *BlockSAT®* i dimensions reduïdes per integrar-se amb l'equip o la roba de l'operari.

El present treball final de màster consisteix en el disseny d'aquest sistema, anomenat *BlockSAT® Portable*, i en el desenvolupament i fabricació del prototip. Per aconseguir-ho, en primera instància, es defineix un llistat de requisits que ha de complir el sistema; a continuació, es dissenya el sistema; i, per últim, es passa al desenvolupament i fabricació del prototip.

El desenvolupament de *BlockSAT® Portable* consisteix, per una banda, en l'estudi de mercat per l'elecció dels components que millor s'adaptin al disseny proposat, per després adquirir-los i acoblar el dispositiu. I d'altra banda, en el desenvolupament, que es centra en la programació de les aplicacions que formen el sistema. Aquest desenvolupament *software* es divideix en el *firmware* del dispositiu i en una aplicació per *smartphones* que serà la interfície entre l'operari i el sistema. Aquests desenvolupaments dotaran al sistema de la capacitat d'obtenir la posició de l'operari i d'emetre sol·licituds per realitzar les feines a la via. Aquesta informació ha de ser transmesa al centre de control del sistema *BlockSAT®* des d'on, amb la informació de la posició, es podran enviar respostes i avisos a l'operari.

Abstract

From communications and software section of SENER, rises up the opportunity to participate in one of its most ambitious development. The *BlockSAT®* system, responsible for monitoring a railway line and managing blockade of cantons according to the position of the trains on the line. In the *BlockSAT®* development appears the need to add a subsystem that allows the representation of the operators that perform works on the track on the synoptic of the line represented in the control center. To do this, the new subsystem must be designed and a functional prototype must be made, that meets the main requirements established by the staff of the *BlockSAT®* project. These are: low cost, integrable with *BlockSAT®* and with reduced dimensions to integrate it in the equipment or clothes of the operator.

The present final master's thesis consists in the design of this system, called *BlockSAT® Portable*, and in the development and manufacture of the prototype. To achieve this, in first instance, is defined a list of requirements that the system must meet; then, the system is designed; and, finally, it is launched the development and manufacture of the prototype.

The development of the *BlockSAT® Portable* system consists, on the one hand, of a market study for the selection of the components that best adapt to the proposed design, to then acquire them and assemble the device. On the other hand, in the development, which is focused on the programming of the applications that make up the system. This software development is divided into the firmware of the device and an application for smartphones that will be the interface between the operator and the system. These developments will provide the system with the ability to obtain the position of the operator and issue requests to carry out the works on track. This information must be transmitted to the control center of the *BlockSAT®* system from where, with the information of the position, responses and warnings can be sent to the operator.

Agradecimientos

Agradezco a las personas de SENER implicadas en el proyecto, que me han introducido y acogido en él, guiándome y enseñándome grandes lecciones durante todo el desarrollo. Especialmente a mi tutora en la empresa, Marta Pérez, gracias por tus correcciones y consejos. También agradecer al director y responsables del proyecto Julio Rodriguez, Hector Garcia, Gerard Garcia y Jaume Llagostera, por confiar en mí y darme la oportunidad y los medios necesarios para llevar a cabo el trabajo.

Igualmente, al Dr. Joan Lluís Pijoan, mi tutor en la Universidad, por resolver mis dudas y ayudarme a encarar el proyecto de la manera correcta. Agradecer también a los profesores que he tenido durante el máster, que han transmitido perfectamente los conocimientos y motivación tan necesarios en esta profesión. Además, dar las gracias a todos los compañeros que me he encontrado a lo largo de mis estudios, con los cuales hemos llegado a formar un gran equipo.

Finalmente agradecer sobre todo, a mi familia, por todo el esfuerzo y confianza que han depositado en mí durante toda mi etapa educativa. En especial a mi madre, Malena, mi padre, Pedro, mi hermano, Pablo y a mi pareja, Alba.

A todos vosotros, muchas gracias.

Lista de contenidos

Lista de figuras

Lista de tablas

Acrónimos

ATT: *Attribute protocol*.

BLE: *Bluetooth Low Energy*.

BSIG: *Bluetooth Special Interest Group*.

CTC: *Control Traffic Center*.

DOORS: *Dynamic Object Oriented Requirements System*.

EDGE: *Enhanced Data rates for GSM Evolution*.

ETSI: *European Telecommunications Standards Institute*.

GAP: *Generic Access Profile*.

GATT: *Generic Attribute Profile*.

GPS: *Global Positioning System*.

GPRS: *General Packet Radio Service*.

GSM: *Global System for Mobile communications*.

HMI: *Human-Machine Interface*.

IBM: *International Business Machines Corporation*.

IDE: *Integrated Development Environment*.

IP: *Internet Protocol*.

LTE: *Long-Term Evolution*.

MMS: *Multimedia Messaging Service*.

RAM: *Random Access Memory*.

SMS: *Short Message Service*.

SRAM: *Static Random Access Memory*.

TFM: Trabajo Final de Máster.

TCP: *Transmission Control Protocol*.

UART: *Universal Asynchronous Receiver-Transmitter*.

UDP: *User Datagram Protocol*.

UMTS: *Universal Mobile Telecommunications System*.

Definiciones

***App* o Aplicación**: Referente a la aplicación móvil para *Android* desarrollada en el presente proyecto.

Dispositivo: Referente al dispositivo electrónico portátil (*wearable*) desarrollado en el presente proyecto y formado por varios componentes.

Blocksat®* o Sistema *Blocksat®: Referente a todos los elementos que forman parte del sistema *Blocksat®* actual desarrollado e implementado por SENER. Excluye aquellos elementos que forman parte del sistema *Blocksat® Portable*.

***Blocksat® Portable* o Sistema**: Referente únicamente a los elementos que forman parte del sistema *Blocksat® Portable* y que son objeto del presente proyecto.

Smartphone: Referente a cualquier dispositivo móvil (tablet o teléfono móvil) con sistema operativo *Android* que cumpla los requisitos para ejecutar la *app*.

Subsistemas: Referente a los diferentes subsistemas que se desarrollan en el presente proyecto y que forman parte del sistema.

1 Introducción

Actualmente la empresa SENER se encuentra en fase de implantación de un sistema para la monitorización y bloqueo de trenes e itinerarios llamado *Blocksat®*, este sistema está basado en la instalación de equipos embarcados en los trenes de una línea controlada y equipos en el centro de control, de manera que es posible controlar el tráfico de una línea, aplicando restricciones y bloqueando cantones según los itinerarios de los trenes, sin necesidad de instalar equipos en la vía o de desplegar personal de operación.

Sin embargo, por estas vías pueden circular, aparte de trenes, otras máquinas que se ocupan del mantenimiento además de personal operando en las vías. Este trabajo final de máster propone añadir una nueva funcionalidad al sistema actual dotando de seguridad añadida a los operarios de la vía.

1.1 Propuesta

El presente proyecto propone el desarrollo de un dispositivo portátil de dimensiones reducidas, de bajo coste, y de una aplicación móvil que permitan transmitir de forma automática la posición de operarios de vía al centro de control en líneas bajo supervisión *Blocksat®*.

Para ello, se va a realizar por una parte, el desarrollo del dispositivo electrónico, con los componentes y desarrollo *software* necesarios y, por otra parte, el desarrollo *software* de una aplicación para *smartphones*.

1.2 Justificación

Este proyecto se desarrolla dentro de la empresa SENER. Se parte del desarrollo interno del sistema *Blocksat®*, a partir del cual se identificaron algunas oportunidades para nuevos desarrollos. Una de estas oportunidades surge de la necesidad de incorporar al sistema *Blocksat®* la monitorización de los trabajos en las vías, ya que el sistema *Blocksat®* actual evita que el bloqueo de las vías con baja densidad de tráfico de trenes sea de forma telefónica, al automatizarlo, pero no tiene en cuenta la ocupación de las vías por otros elementos, como máquinas de vía u operarios. A partir del presente proyecto se pretende replicar esta funcionalidad de automatización en la monitorización de los trabajos de vía, pero sin la complejidad de los equipos embarcados actuales del sistema *Blocksat®* que, al requerir una alta precisión, seguridad y fiabilidad son altamente complejos y más costosos.

1.3 Objetivos

Los objetivos del proyecto se pueden dividir en un objetivo general, donde reside la finalidad del proyecto, y en varios específicos que se han pretendido alcanzar durante la realización del mismo.

1.3.1 Objetivo General

El objetivo general del proyecto es desarrollar un sistema de posicionamiento de operarios de vía centralizado.

1.3.2 Objetivos Específicos

- Definir la arquitectura lógica de un sistema de posicionamiento de operarios de vía centralizado.
- Estudiar las soluciones que el mercado de componentes electrónicos puede proporcionar para la implementación de la arquitectura lógica definida.
- Implementar físicamente un sistema de posicionamiento de operarios de vía centralizado.
- Desarrollar un dispositivo electrónico capaz de: recibir su posición mediante GPS, comunicarse con el centro de control a través de comunicaciones móviles y comunicarse con un *smartphone* mediante BLE.
- Desarrollar una aplicación móvil, en *Android*, capaz de comunicarse mediante BLE con el dispositivo y de lanzar peticiones al centro de control.
- Familiarizarse con los conceptos teóricos en los que se basan las soluciones que se implementan, como son: BLE, GPS, GSM, *Arduino* y *Android*.

1.4 Alcance y limitaciones

El sistema que se desarrolla en el presente proyecto, *BlockSAT® Portable*, forma parte del sistema *BlockSAT®* de SENER. Este sistema ha tenido un largo recorrido tecnológico e incluye diferentes subsistemas y desarrollos que, pese a ser tenidos en cuenta durante el desarrollo del proyecto, su integración con *BlockSAT® Portable* queda fuera del alcance del proyecto.

La empresa ha apoyado y promocionado el desarrollo del proyecto a través de la adquisición de materiales y equipos y ha puesto a disposición del proyecto recursos propios. Pero debido a las limitaciones propias que tiene la realización de un trabajo final de máster, como son el plazo y el presupuesto, el dispositivo y la aplicación que resultan del proyecto deben ser considerados como el primer prototipo de un desarrollo con más posibilidades. De este desarrollo podrán surgir en el futuro, y con una mayor inversión y dedicación, nuevas versiones que podrían llegar a ser industrializables y comerciables.

1.5 Estructura del documento

Inicialmente el documento describe el sistema *Blocksat®*, en el que se basa el proyecto, y su integración con el desarrollo objeto del presente proyecto, *BlockSAT® Portable*. En este mismo apartado se describe, de manera general, la arquitectura del sistema a desarrollar.

Más adelante, se presenta de manera formal el listado de requisitos que deberá cumplir el sistema una vez acabado el proyecto. Estos se dividen según si son requisitos generales, propios del dispositivo o del ámbito de la aplicación móvil.

Una vez detallada la arquitectura y los requisitos, se pasa a describir los elementos desarrollados durante el proyecto (los componentes electrónicos en los que está basado el *hardware* del dispositivo, los desarrollos *software*, que incluyen, el *firmware* del dispositivo y la *app*) y, por último, las interfaces entre los diferentes subsistemas de *BlockSAT® Portable* y el protocolo de comunicaciones utilizado entre ellos.

Para acabar, se ilustrará el funcionamiento final del sistema y las pruebas realizadas, que juntamente con el análisis económico y las conclusiones del proyecto darán por finalizado el documento.

1.6 Plan de trabajo

En el diagrama de tiempo de la Figura 1 se ilustra el cronograma que se ha planteado para el desarrollo del proyecto. En él se observan las diferentes tareas principales en las que se ha dividido el proyecto, que son:

- el desarrollo del sistema electrónico, que incluye una fase de preparación con el estudio de mercado y la elección de los componentes, una fase de montaje y desarrollo de *firmware* y una fase de testeo;
- el desarrollo de la aplicación, que tiene una primera fase de conceptualización donde se definen sus características y una fase de desarrollo;
- la tarea de definir interfaces e integración, que se dedica a unir las diferentes partes del proyecto, configurarlas y probarlas;
- y por último, la tarea transversal de redacción de la memoria del proyecto, que se va realizando según se avanza en el proyecto pero con mayor dedicación en la fase final.

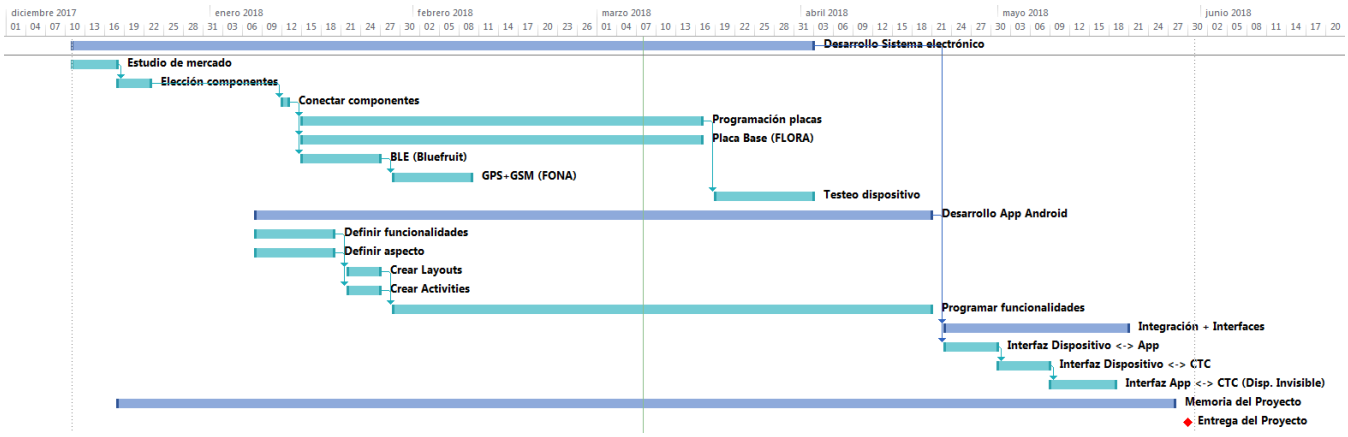

Figura 1. Cronograma del Proyecto.

2 Descripción del Sistema

2.1 BlockSAT®

BlockSAT® es un nuevo sistema de bloqueo y gestión del tráfico ferroviario para líneas de baja densidad, y se fundamenta en la localización por satélite, navegación inercial y comunicaciones inalámbricas. Se trata de un producto actualmente en fase de implementación y patentado por SENER, que propone una nueva tecnología para ejecutar funciones de bloqueo utilizando equipos embarcados en la locomotora que permiten determinar con precisión tanto la localización del mismo en la red ferroviaria, como su velocidad. En el centro de control, el operador del control central de *BlockSAT®* asigna rutas (itinerarios) a cada uno de los trenes de forma segura desde la interfaz hombre-máquina del sistema. [1]

Por otro lado, el equipo embarcado, protege el tren de manera automática e informa al maquinista del estado de la operación. En caso de estar conectado al sistema de freno de la locomotora, el sistema *BlockSAT®* podría activar de manera automática el freno de servicio o de emergencia del tren, en caso de detectarse un incumplimiento de las autorizaciones de movimiento.

BlockSAT® realiza las siguientes funciones principales:

1. determinación de las vías ocupadas,
2. monitorización en tiempo real de la posición y velocidad de los trenes,
3. asignación de autorizaciones de movimiento de acuerdo con la ruta establecida por el operador,
4. verificación del cumplimiento de las autorizaciones y de las limitaciones de velocidad, sin necesidad de instalar equipos de señalización en la vía.

De esta manera, *BlockSAT®* es capaz de reducir las necesidades operativas para ejercer la función de bloqueo de una línea ferroviaria; esta reducción se aplica también a las actividades de mantenimiento preventivo de equipos ya que se circunscribe únicamente a los equipos embarcados y al equipo central, sin tener que mantener o renovar equipos en la vía.

2.1.1 Arquitectura del Sistema *BlockSAT®*

BlockSAT® se compone de componentes *hardware* y *software*. Sensores complementarios y unidades de procesado de datos se combinan con un nivel adecuado de redundancia para ofrecer los niveles de disponibilidad y fiabilidad necesarios en la operación ferroviaria. Las unidades de procesado de datos, a partir de los datos obtenidos por los sensores, y utilizando algoritmos de monitorización y de decisión sobre las ambigüedades resultantes, determinan el tramo de vía ocupado por el tren en tiempo real, durante todo el recorrido del tren. Combinando la información de los diferentes trenes, el sistema es capaz de realizar la función de bloqueo.

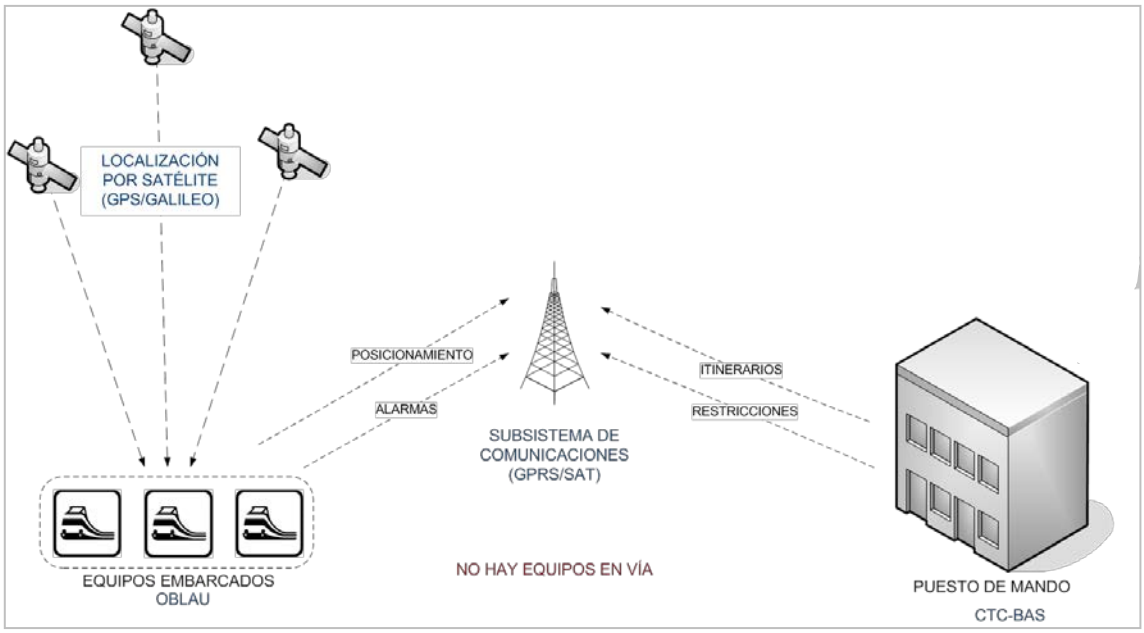

Figura 2. Arquitectura básica del sistema *BlockSAT®*.

Los equipos *BlockSAT®* embarcados en cada uno de los trenes en circulación transmiten datos en tiempo real de su posición y de la velocidad instantánea al módulo instalado en el Centro de Control. Esta transmisión de datos se realiza a través de redes comerciales disponibles de telecomunicaciones inalámbricas. Esto implica que, para implementar el sistema *BlockSAT®*, no es necesario implementar una red de telecomunicaciones específica para la transmisión de datos entre los trenes y el centro de control.

2.1.1.1 Centro de Control

En el centro de control se equipará el módulo de control del sistema *BlockSAT®*, que recibe y procesa los datos provenientes de los equipos embarcados, para poder ser mostrados al operador del centro de control a través de una interfaz gráfica.

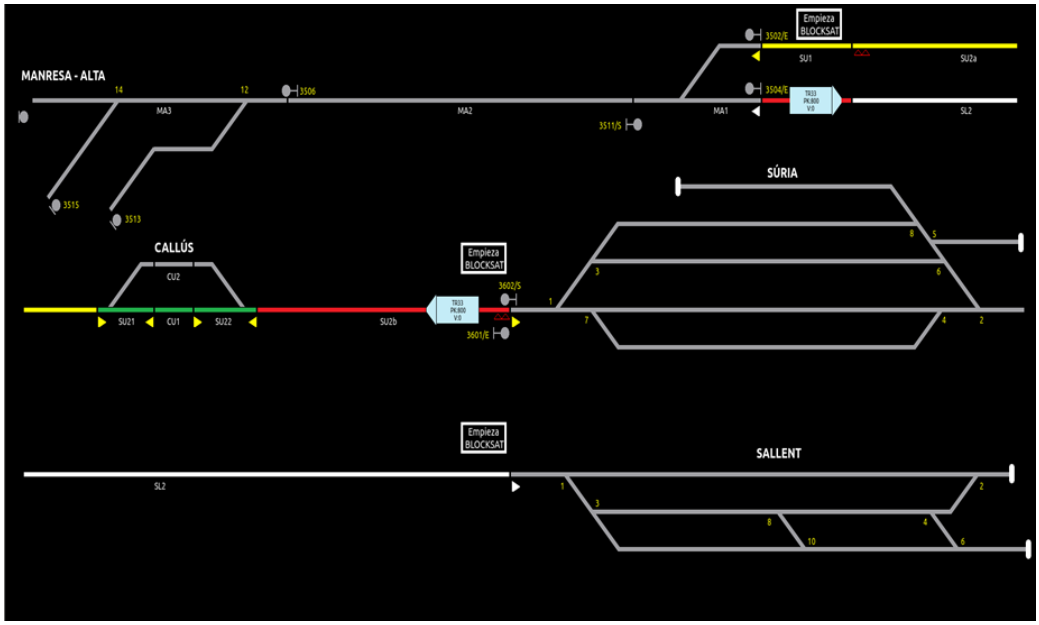

Figura 3. Sinóptico de la interfaz del operador.

El operador del centro de control, a través de la HMI podrá asignar rutas de manera gráfica a los diferentes trenes en operación, pudiendo limitar la velocidad de circulación a las rutas asignadas, si fuera necesario.

La interfaz muestra al operador del CTC las alarmas generadas por el sistema, originadas tanto en el módulo de control como aquellas que se generen en las unidades embarcadas de los trenes.

2.2 *BlockSAT® Portable*

BlockSAT® Portable nace como un complemento al sistema *BlockSAT®* y se define como un sistema para la localización de operarios en entornos ferroviarios. Este sistema añade una nueva funcionalidad al sistema *BlockSAT®*, dotando de seguridad añadida a los operarios de las vías ferroviarias durante el desarrollo de sus tareas, de forma poco intrusiva. Para ello, el presente proyecto propone el desarrollo de un dispositivo *wearable* capaz de posicionar a los operarios de vía de una línea controlada por el sistema *BlockSAT®* y enviar esta posición al centro de control. De esta manera existirá una monitorización por parte del centro de control de los trabajos y zona de trabajo de los operarios y, a través de una interfaz gráfica, los operarios tendrán una plataforma para emitir avisos sobre el progreso de sus tareas. Este sistema, al integrarse con el sistema *BlockSAT®*, sustituye modos más arcaicos de reportar los trabajos de vía y bloquear las vías donde se realizan estos trabajos.

2.2.1 Arquitectura del sistema *BlockSAT® Portable*

BlockSAT® Portable se compone de un dispositivo electrónico portátil de dimensiones reducidas y de una aplicación móvil. El dispositivo debe tener conectividad con diferentes sistemas: GPS para recibir su posición; GPRS para la comunicación con el Centro de Control y; BLE para la comunicación con la aplicación móvil. La *app* será la interfaz entre el operario, el dispositivo y el CTC; desde la misma se podrá configurar el dispositivo, solicitar el inicio de los trabajos en vía y ver el estado y la posición de los trenes próximos al punto de vía donde se sitúa el operario.

El dispositivo debe ofrecer la oportunidad de ser independiente de la aplicación una vez configurado. De esta manera, mientras el operario realiza los trabajos en la vía no es necesario que desvíe su atención hacia el *smartphone* y los avisos tanto de la finalización de los trabajos de vía como de la aproximación de algún tren los hará el dispositivo a través de una vibración y/o señal acústica.

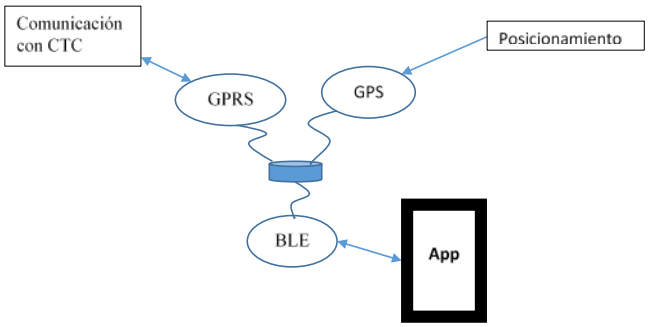

Figura 4. Esquema básico del dispositivo *BlockSAT® Portable*.

El dispositivo se comunicará con el centro de control a través del subsistema de comunicaciones (GPRS) del sistema *BlockSAT®*, en el caso del presente proyecto será una red móvil GSM comercial. A través de estas comunicaciones, el CTC recibirá las coordenadas del operario y, gracias a la base de datos de referencia, aparecerá su posición sobre el sinóptico de la vía que se representa en el HMI del operador del CTC.

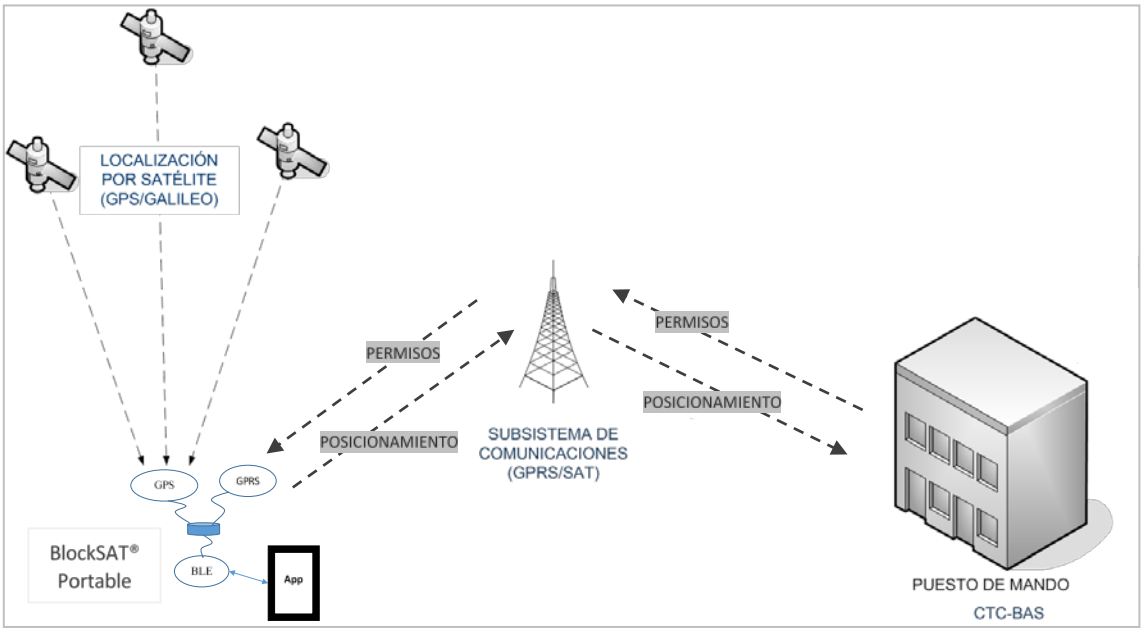

Figura 5. Arquitectura básica del Sistema *BlockSAT® Portable*.

Desde el CTC, el operador tendrá toda la información de los trenes que circulan por las vías controladas y de los operarios de vía que van a realizar los trabajos. De esta manera podrá decidir los itinerarios más óptimos para cada tren y aceptar o denegar los permisos de trabajo que soliciten los operarios de vía desde la aplicación móvil.

3 Requisitos del Sistema

Un objetivo significativo del presente proyecto es definir y gestionar de manera formal y estricta los requisitos del desarrollo. Para ello se propone el uso de la herramienta gestión de requisitos *Rational DOORS* de IBM, por su uso es extendido en el desarrollo de proyectos del ámbito de la ingeniería y especialmente en la industria ferroviaria.

Para la definición de los siguientes requisitos se han tenido en cuenta conceptos de Ingeniería de Requisitos [2]. Se han redactado desde un punto de vista funcional, teniendo en cuenta los objetivos y características del desarrollo. Una vez definidos se han integrado con la herramienta DOORS para poder gestionar los requisitos en una ubicación única y tener una trazabilidad que permita rastrear los requisitos de manera que se enlacen con elementos del diseño.

3.1 Requisitos generales

1.01. El sistema debe permitir la recepción de la posición de un operario en tiempo real en el Centro de Control.

1.02. El sistema debe incluir los controles e interfaces necesarios para que el usuario pueda configurar el dispositivo.

1.03. El sistema debe incluir los controles e interfaces necesarios para que el usuario pueda solicitar permiso para trabajar en la vía al centro de control.

1.04. El sistema debe incluir las interfaces necesarias para que el usuario obtenga la información del estado del dispositivo en todo momento.

1.05. La interfaz entre el dispositivo y el usuario será a través de una aplicación móvil.

1.06. El sistema enviará la información necesaria al CTC para que se realice la función de bloqueo de los cantones ocupados por un operario durante sus operaciones en la vía.

1.07. El sistema debe representar gráficamente las autorizaciones de uso de vía obtenidas por el CTC.

1.08. El sistema debe poder recibir, confirmar, cancelar o terminar una autorización de uso de vía recibida por el CTC.

1.09. El sistema debe tener la función de informar al operario en caso de que el CTC envíe la información de que un tren se encuentra a una distancia mínima predeterminada del operario.

1.10. El sistema debe tener la función de avisar al operario en caso de que el tiempo permitido para los trabajos en la vía esté próximo a agotarse.

1.11. El sistema se integrará al sistema actual *BlockSAT®* y participará como un subsistema más.

1.12. El sistema incluirá una interfaz con las redes de comunicaciones móviles para poder crear un enlace de comunicaciones entre los dispositivos y el centro de control.

3.2 Requisitos del dispositivo

2.01. El dispositivo debe obtener la posición del operario a través de un sistema GPS.

2.02. El sistema GPS del dispositivo debe tener una precisión de posicionamiento por debajo de los 3 metros.

2.03. El dispositivo debe comunicarse con el CTC a través de la red de comunicaciones móviles.

2.04. El dispositivo debe comunicarse con el teléfono móvil del operario a través de un sistema BLE.

2.05. El dispositivo debe procesar la información recibida desde el CTC y enviársela de forma adecuada a la aplicación del teléfono móvil del operario.

2.06. El dispositivo debe procesar la información recibida desde la aplicación del teléfono móvil del operario y enviársela de forma adecuada al CTC.

2.07. El dispositivo debe incorporar un sistema de aviso al operario.

2.08. El dispositivo debe ser ligero e integrable en la ropa de trabajo del operario o como accesorio (*wearable*).

2.09. El dispositivo debe ser capaz de soportar impactos leves y vibraciones y debe ser apto para su uso en un entorno ferroviario.

2.10. El dispositivo debe incorporar una batería recargable.

2.11. El dispositivo debe tener una autonomía mínima de 12 horas.

3.3 Requisitos de la aplicación

3.01. La aplicación será la interfaz entre el dispositivo y el usuario.

3.02. La aplicación debe poder instalarse en un *Smartphone Android*.

3.03. La aplicación debe disponer de una interfaz gráfica intuitiva.

3.04. La aplicación debe mostrar los trenes próximos a la zona donde se está operando.

3.05. La aplicación debe tener un apartado dedicado a la configuración del dispositivo.

3.06. La aplicación debe mostrar la información del estado del dispositivo (apagado, en funcionamiento, nivel de batería, etc.) en todo momento.

3.07. La aplicación debe contener los controles necesarios para que el usuario pueda enviar al CTC una solicitud para iniciar un trabajo en vía.

4 *Hardware* del Sistema

El objetivo de este capítulo es describir el diseño e implementación de los componentes *hardware* que forman el sistema. Para ello, se detalla desde el estudio de mercado realizado hasta la arquitectura del dispositivo final, pasando por la descripción de cada uno de los componentes.

4.1 Estudio de mercado

Teniendo en cuenta la arquitectura del sistema, los componentes *hardware* en los que se va a basar el dispositivo son:

- Controlador programable
- Módulos de comunicaciones:
    - GPRS
    - BLE
- Módulo de posicionamiento GPS

El requisito básico del controlador es que sea de dimensiones y peso reducidos, que se pueda alimentar a través de baterías de litio y que, o bien tenga sistemas de comunicación integrados en el controlador, o tenga la posibilidad de añadir módulos de comunicaciones.

Para la elección de los componentes se han tenido en cuenta diversos fabricantes, haciendo un estudio de mercado a partir de las necesidades del proyecto. Se encontraron varios fabricantes que disponían de componentes que cumplían los requisitos establecidos; de ellos, se filtraron los dos más adecuados y se hizo un análisis más exhaustivo, presentado en los siguientes apartados.

4.1.1 U-Blox

Los productos de este fabricante están dirigidos a aplicaciones profesionales de posicionamiento GPS y están basados en programación *embedded*. Ofrece soluciones precisas y profesionales a cambio de un coste elevado y alta complejidad en su desarrollo. Pese a que hay una gama de los productos de tamaño reducido, no están orientados directamente a aplicaciones *wearable*. A continuación se encuentran los productos de *U-Blox* que más encajan con el proyecto.

4.1.1.1 C027 *Application Board*

Esta *application board* [3] permite desarrollar el proyecto gracias a su micro procesador Cortex-M3 con 512 KB de memoria FLASH y 64 KB de memoria RAM que funciona a 96 MHz. Además, incluye los módulos para comunicación celular (2G/3G) y para localización GPS. Para completar el dispositivo faltaría añadirle el módulo BLE. Sus dimensiones son 53.325 x 96.525 mm. Y su coste aproximado es de 200€.

Figura 6. *U-Blox* C027 *Application board*.

4.1.1.2 NINA B301 *Bluetooth Low Energy Module*

Este módulo de comunicaciones [4] aporta conectividad *bluetooth* a la *application board*. Sus dimensiones son 10.0 x 11.6 x 1.9 mm. Y su coste aproximado es de 30€.

Figura 7. *U-Blox* NINA B301 *BLE Module*.

4.1.2 *Adafruit*

El fabricante *Adafruit* dispone de diferentes gamas de productos que se ajustan a las necesidades del proyecto. Estos productos están basados en *Arduino* y se programan a través de su IDE, que permite cargar el programa a las placas en el mismo entorno. Además en el catálogo del fabricante se ofrecen soluciones para aplicaciones *wearable*, con bajo consumo y de bajo coste. Teniendo en cuenta que lo que se busca es una configuración de componentes que tengan conectividad GSM/GPRS, BLE y GPS y que sean de dimensiones reducidas, en el catálogo de este fabricante podemos encontrar las siguientes opciones.

4.1.2.1 FLORA – *Wearable*

FLORA es una plataforma electrónica *wearable* [5] basada en un micro procesador *Arduino* ATmega32U4 con 32KB de memoria FLASH y 2.5 KB de memoria SRAM que funciona a 16 MHz. Con esta placa sería necesario añadir los módulos de comunicación celular, comunicación *bluetooth* y localización GPS. Es de forma circular y sus dimensiones son de 45mm de diámetro x 7mm de alto. Y su coste aproximado es de 12€.

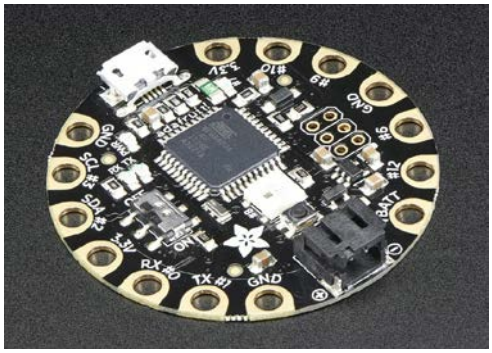

Figura 8. *Adafruit* FLORA - *Wearable*.

4.1.2.2 FLORA *Wearable Bluefruit LE Module*

Este módulo aporta conectividad *bluetooth* a la placa FLORA[6]. También está orientada a aplicaciones *wearable* y tiene la misma forma que la placa FLORA pero con dimensiones más reducidas, por tanto son fácilmente integrables. Sus dimensiones son 30,5mm de diámetro x 4mm de alto. Y su precio aproximado es de 15€.

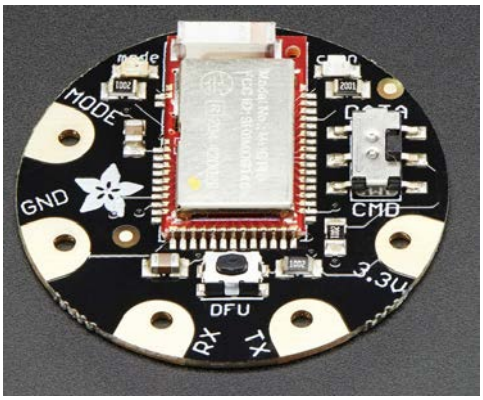

Figura 9. *Adafruit* FLORA *Bluefruit LE Module*.

4.1.2.3 FONA 808 GSM + GPS

Este módulo aporta conectividad GSM para el envío de datos a través de GPRS y localización GPS a la placa FLORA[7]. Pese a no tener la misma forma que la placa FLORA (este módulo es rectangular), es de dimensiones similares y podría adaptarse. Sus dimensiones son 44mm x 43mm x 8mm. Y su precio aproximado es de 43€.

Figura 10. *Adafruit* FONA 808 GSM + GPS *Module*.

4.1.3 Elección del fabricante

Debido a que el objetivo del proyecto es desarrollar un dispositivo ligero, de dimensiones reducidas y de bajo coste, y que las necesidades de procesamiento no son demasiado exigentes para la aplicación a la que va destinada, a la hora de decidirse por un fabricante han primado estos factores. De esta manera, se ha escogido el fabricante *Adafruit* con la placa FLORA y los módulos de comunicaciones *Bluefruit* y FONA para la conectividad *bluetooth* y GSM+GPS, respectivamente.

Tabla 1. Comparativa de fabricantes.

	U-Blox	***Adafruit***
Memoria FLASH (KB)	512	32
Memoria RAM (KB)	32	2.5
Dimensiones totales (mm.)	53.325 x 96.525	44 x 43
Peso total (gramos)	20	19.5
Precio aproximado (€)	230	70

4.2 Descripción de componentes

A lo largo del presente subapartado se detallan uno a uno los componentes elegidos para formar el sistema *hardware* del dispositivo.

4.2.1 Placa Base

FLORA es una plataforma *wearable* electrónica desarrollada por *Adafruit*. Esta plataforma está basada en un chip ATmega32U4 a 16MHz y con soporte USB incorporado. Con unos 4,5 cm de diámetro y su forma circular, es realmente pequeño y adaptable, lo que la hace ideal para proyectos *wearable*, donde se necesitan dispositivos pequeños, ligeros y versátiles.

Esta placa dispone de un pequeño botón de reseteo incorporado para poder reiniciar el sistema fácilmente. Su fuente de alimentación está diseñada para ser flexible y fácil de utilizar. Dispone de un conector de batería JST polarizado con diodo *schottky* de protección incorporado, y se puede utilizar con baterías externas de 3.5v a 9v DC (el voltaje recomendado es entre 3.5V y 5V). Las baterías pueden ser de los tipos *LiIon*/*LiPoly*, *LiFe*, alcalinas o recargables de *NiMh*/*NiCad* de cualquier tamaño.

Las características principales del microcontrolador ATmega32U4 que contiene la placa FLORA son las siguientes:

- 32 KB de memoria flash auto programable *In-System*.
- 2,5 KB de memoria interna SRAM.
- 1 KB de memoria interna EPROM.
- USB 2.0 *full-speed*/*low speed* integrado.
- 12 canales de 10 bits A/D.
- Oscilador interno de 8 MHz

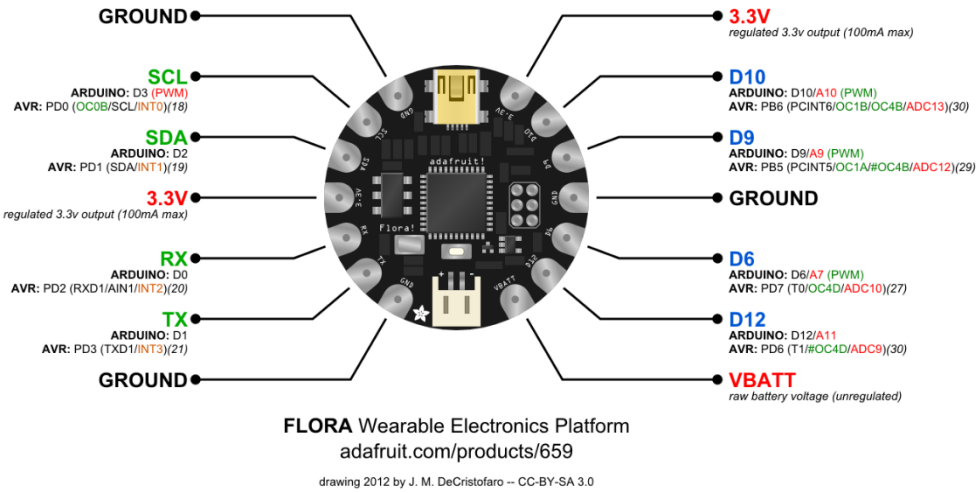

Figura 11. Diagrama de pines de FLORA.

Como se observa en la Figura 11, la placa además del conector *microUSB* y el conector de la batería, dispone de 14 pines (algunos redundantes) a través de los cuales se conectará a los módulos de comunicaciones. Estos pines permiten la interfaz entre la placa base y los dos módulos de comunicaciones.

4.2.2 Placa de comunicaciones GSM y posicionamiento GPS

El módulo encargado de las comunicaciones celulares (a través de una conexión GSM) y de la localización GPS, es el *Adafruit* FONA 808 *Cellular* + GPS *Breakout*. Este módulo mide solo 4,5 x 4 cm y en su interior alberga el potente módulo de comunicaciones GSM SIM808 con GPS integrado. Sus características principales son:

- *Quad-band* 850/900/1800/1900MHz. Conectividad a cualquier red GSM global a través de un tarjeta SIM 2G.
- GPS completamente integrado a través del chip MT3336, con una sensibilidad de -165 dBm.
- Posibilidad de enviar y recibir llamadas y mensajes SMS.
- Posibilidad de enviar y recibir datos GPRS a través de conexiones UDP, TCP/IP, HTTP, etc.
- Posibilidad de controlar un motor vibratorio a través de un pin PWM.
- Interfaz de comandos AT con detección automática de *baudrate*.
- Conexiones *uFL* para antenas externas.

Las características principales del GPS integrado son las siguientes:

- Código GPS L1 C/A
- Sensibilidad:

- *Tracking*: -165 dBm
- *Cold starts*: -147 dBm

- *Time-To-First-Fix*:
  - Cold starts: 30s (typ.)
  - Hot starts: 1s (typ.)

- *Warm starts*: 28s (*typ*.)

- Precisión de posicionamiento: 2.5 metros aproximadamente.

Figura 12. Detalle de pines del módulo GSM+GPS.

Como se observa en la Figura 12, este módulo dispone de conexión *microUSB* y un conector para la batería. Este módulo tiene la peculiaridad de que el único objetivo del conector *microUSB* es realizar la carga de la batería, ya que no está instalado ni para poder alimentar la placa ni para usarse como interfaz con un ordenador. De esta manera, necesita tener una batería conectada al conector JST para alimentar el módulo y que funcione correctamente en todo momento.

Para lograr la conectividad del módulo es necesario incorporarle antenas externas a través de los conectores *uFL*. Estas antenas se han escogido por sus dimensiones reducidas y la compatibilidad con el módulo. La antena GPS (izquierda) es una antena pasiva de 15x15mm y 1dBi de ganancia y la que aparece a la derecha en la imagen inferior es la antena GSM ultrafina de 35x17mm y 3dBi de ganancia.

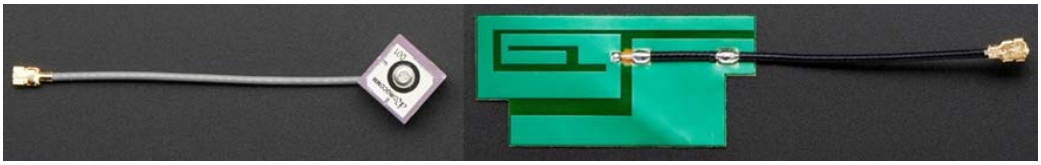

Figura 13. Antenas GPS y GSM, respectivamente.

4.2.3 Placa de comunicaciones *Bluetooth*

El módulo encargado de las comunicaciones *bluetooth* entre el dispositivo y el *smartphone* es el *Adafruit* FLORA *Bluefruit LE*. Este módulo, con la misma forma circular que la placa base FLORA pero con dimensiones más reducidas (3 cm de diámetro), se adapta perfectamente al proyecto. El módulo está basado en el chip nRF51822 de la serie MDBT40 que implementa el estándar de comunicaciones *Bluetooth Low Energy*.

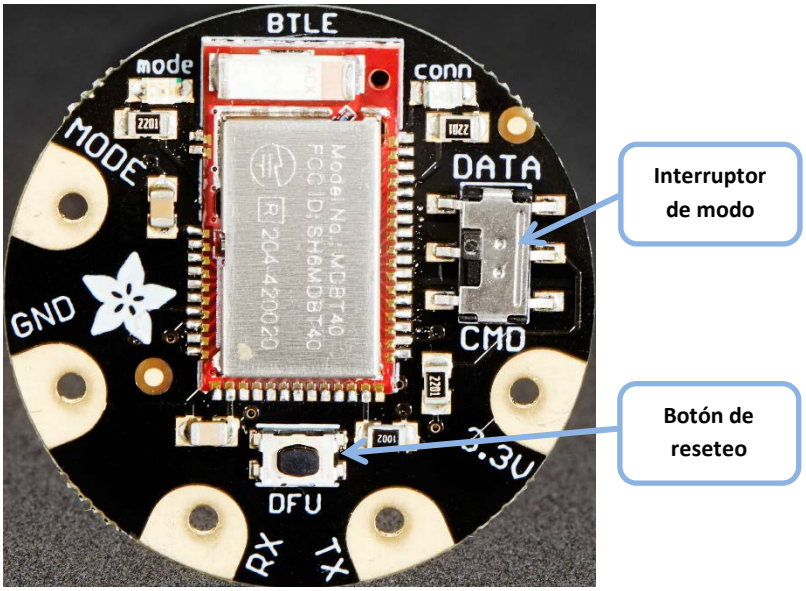

Figura 14. Detalle de pines del módulo BLE.

Como se observa en la Figura 14, el chip se sitúa en el centro del módulo, que incorpora los pines de alimentación (3.3V y GND), los necesarios para comunicarse con la placa base (TX y RX) y el pin para seleccionar modo (*MODE*) que puede estar desconectado y seleccionar entre los modos *DATA* y *COMMAND* a través del interruptor que hay sobre el módulo. Además sobre el módulo también hay *LEDs* informativos y un botón de reseteo (DFU).

4.3 Arquitectura del dispositivo

Los tres componentes descritos en el apartado anterior forman la arquitectura principal del dispositivo. En el esquema de la Figura 15 se representan los bloques que forman el sistema y la placa a la que corresponde cada módulo. Se observa como la placa FLORA es el elemento central que, con el módulo de procesado de datos, se ocupa de analizar las diferentes entradas y generar las salidas en forma de acciones hacia los otros módulos. Esta misma placa dispone de dos interfaces que son las encargadas de dialogar con sus respectivos módulos de comunicaciones:

- La interfaz BLE de FLORA se comunica con el módulo BLE de la placa *Bluefruit LE*, que, a través de su antena integrada, puede comunicarse con el teléfono móvil.
- La interfaz FONA de FLORA se comunica con el módulo de posicionamiento GPS y comunicaciones GSM de la placa FONA, que, a través de sus antenas externas, puede posicionarse a través de la red de satélites y comunicarse con el CTC, respectivamente.

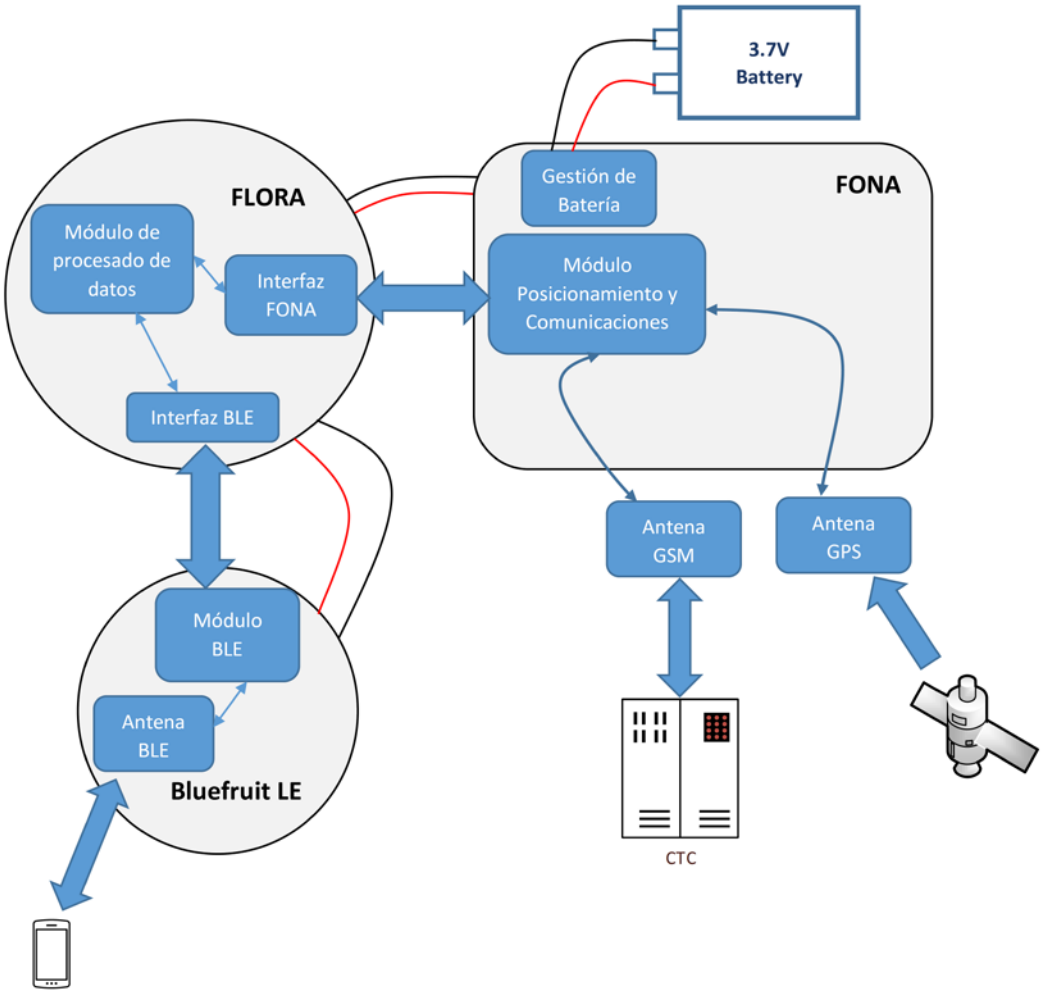

Figura 15. Esquema de bloques del sistema.

Aparte de las interfaces lógicas entre cada subsistema del dispositivo, en este esquema se muestra el recorrido de la alimentación (en rojo/negro). La alimentación entra al sistema a través de la placa FONA, que dispone de un módulo capaz de gestionar y cargar la batería, y, desde este punto se alimenta la placa FLORA que distribuye la energía necesaria a la placa BLE.

4.3.1 Montaje del dispositivo

La arquitectura del dispositivo electrónico se ha desarrollado a partir de la interconexión entre los componentes seleccionados anteriormente. Como se muestra en la Figura 16 además de los componentes básicos explicados en el apartado anterior se han añadido un motor vibrador y la batería que alimentará todo el dispositivo.

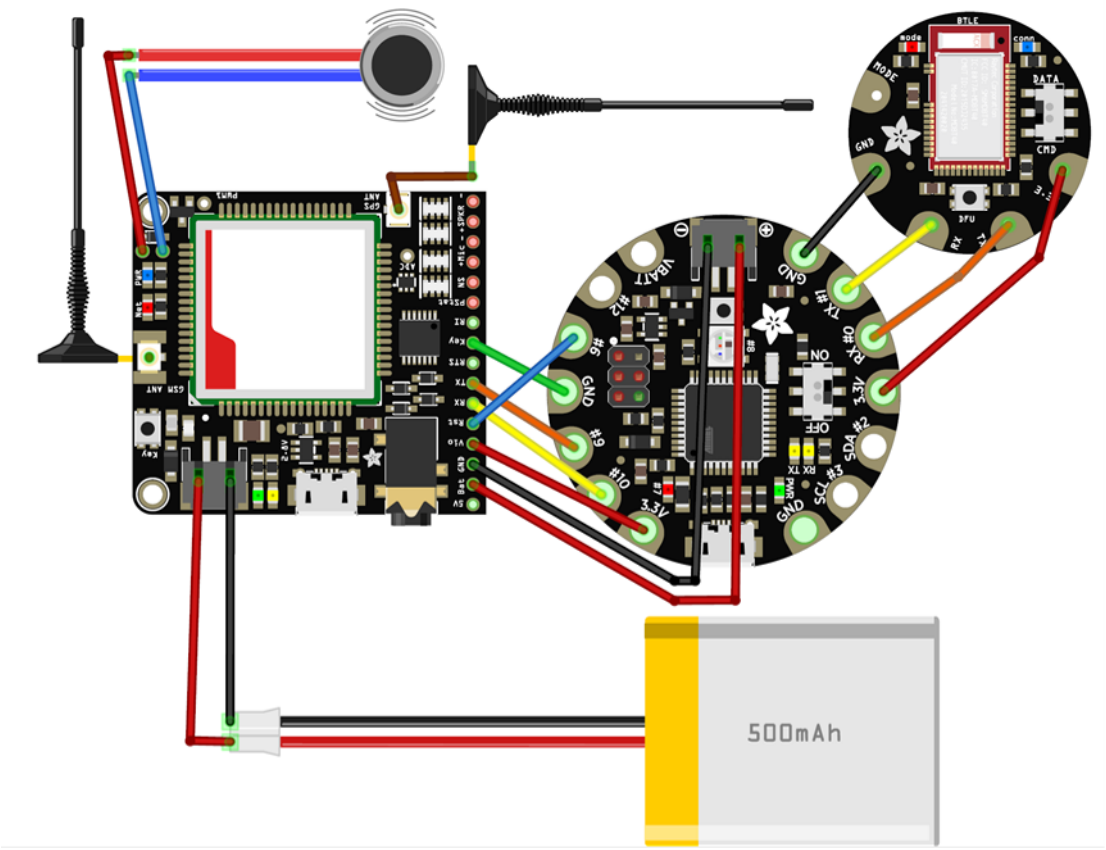

Figura 16. Representación gráfica de conexiones.

La conexión entre la placa FLORA y el módulo BLE es muy directa, solo se deben conectar los cuatro *pads* necesarios del módulo con sus respectivos *pads* en la placa FLORA.

El *pad* MODE del módulo BLE, se utiliza para cambiar el modo de operación BLE, como en el caso del proyecto el módulo tendrá un modo fijo una vez se configure inicialmente, no es necesaria la conexión de ese *pad*. En definitiva, la conexión FLORA-BLE queda de la siguiente manera:

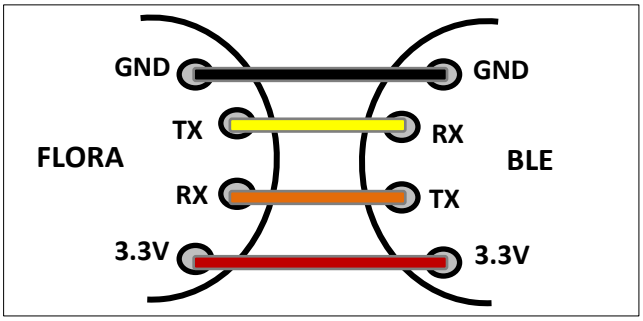

Figura 17. Esquema de conexiones entre FLORA y BLE.

Por otro lado, la placa controladora FLORA se debe conectar con el módulo FONA, encargado de la comunicación GSM y la localización por GPS. Esta conexión también se hace de manera directa en la mayoría de sus *pads*, pero tiene la particularidad de que la placa FONA debe alimentar al resto del sistema (FLORA + BLE). Para ello la conexión de la placa FLORA con los

pads de FONA que distribuyen la alimentación se debe hacer a través del conector de batería JST. En definitiva, la conexión FLORA-FONA queda de la siguiente manera:

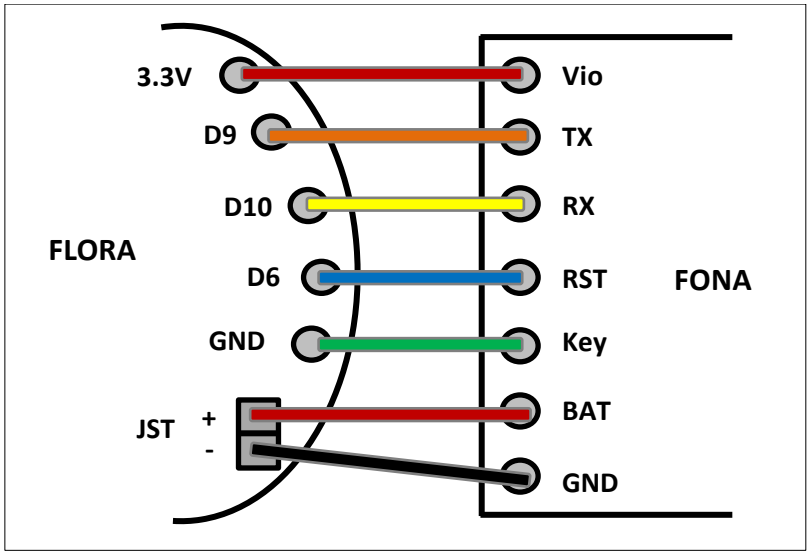

Figura 18. Esquema de conexiones entre FLORA y FONA.

La placa FONA además de ir conectada a FLORA, tiene conectadas a ella las antenas externas de GSM y GPS que van conectadas a los puertos específicos que tiene FONA para las antenas a través de sus conectores *uFL*. También tiene conectado el motor vibrador a dos puertos dedicados a este servicio y que permiten configurar fácilmente, a través del código, las características de la vibración. Por último, se conecta al módulo FONA la batería que alimenta todo el sistema. Las conexiones entre los elementos externos al módulo FONA quedan de la siguiente manera:

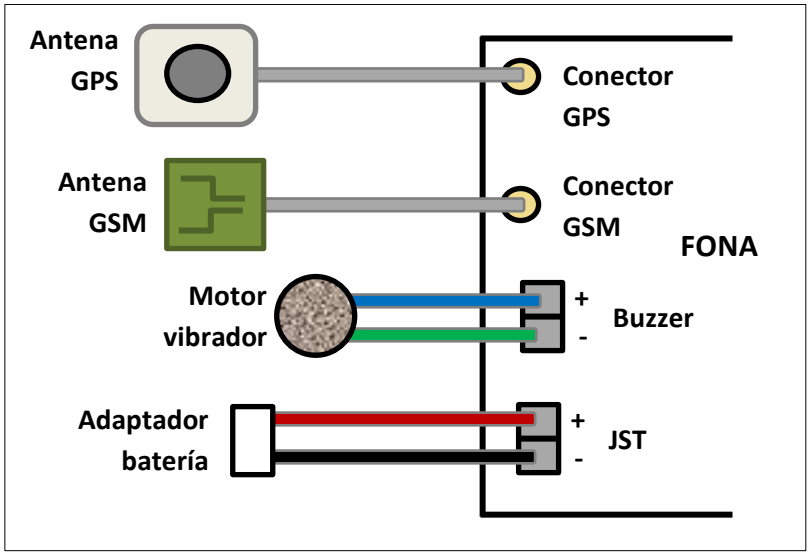

Figura 19. Esquema de conexiones entre FONA y elementos externos.

En la última fase del montaje se añaden dos componentes para completar el dispositivo. Por un lado se añade un interruptor que, conectado entre la batería y la entrada de alimentación del módulo FONA, permite apagar o encender el dispositivo sin tener que manipular directamente las conexiones de la batería.

Figura 20. Interruptor para el encendido del sistema.

Por otro lado, se añade un receptor de carga inalámbrica. Este componente se conecta directamente al conector *microUSB* del módulo FONA y cargará la batería del dispositivo cuando éste se coloque sobre una base de carga inalámbrica. De esta manera, el dispositivo podrá encapsularse en alguna estructura estanca y no será necesario abrir esta estructura ni para cargar ni para encender o apagar el dispositivo.

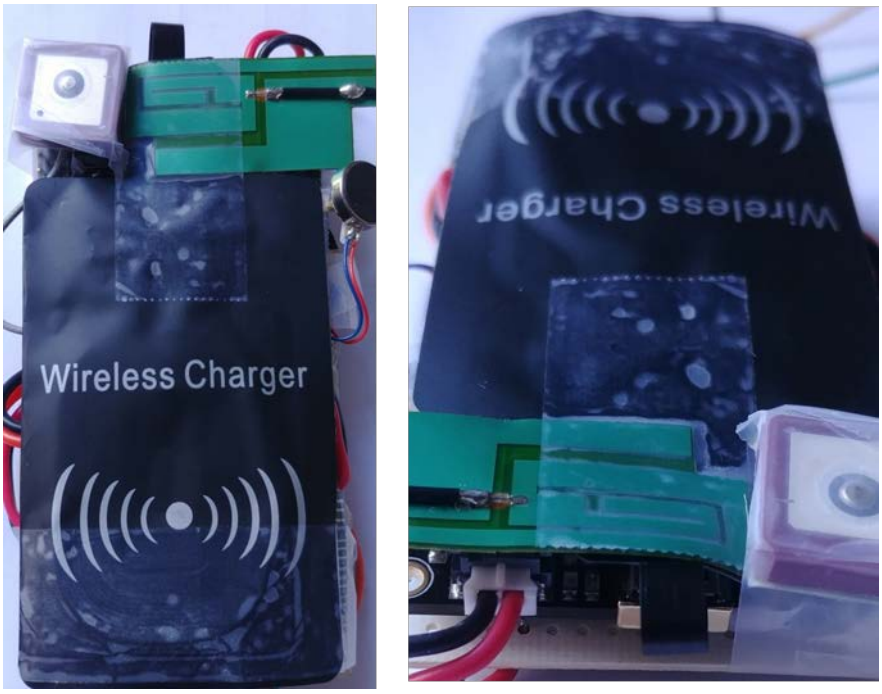

Figura 21. Receptor para carga inalámbrica. Posición y conexión con el dispositivo.

El dispositivo, tal como se indica en los requisitos, debe ser integrable en la ropa o equipos de un operario y estar preparado para soportar leves impactos y vibraciones propias de un entorno ferroviario.

Para lograr esta integración y la posibilidad de que el dispositivo sea *wearable*, se decide ensamblar el dispositivo sobre una placa de topos de manera que el ensamblaje y las conexiones ocupen el mínimo espacio posible. Para lograrlo, se ha aprovechado la placa de topos por ambos lados; por un lado, se ha colocado el módulo FONA, la batería, las antenas de GPS y GSM, y el receptor de carga inalámbrica; y, por el otro, la placa FLORA, el módulo BLE, las soldaduras y las conexiones hacia el interruptor

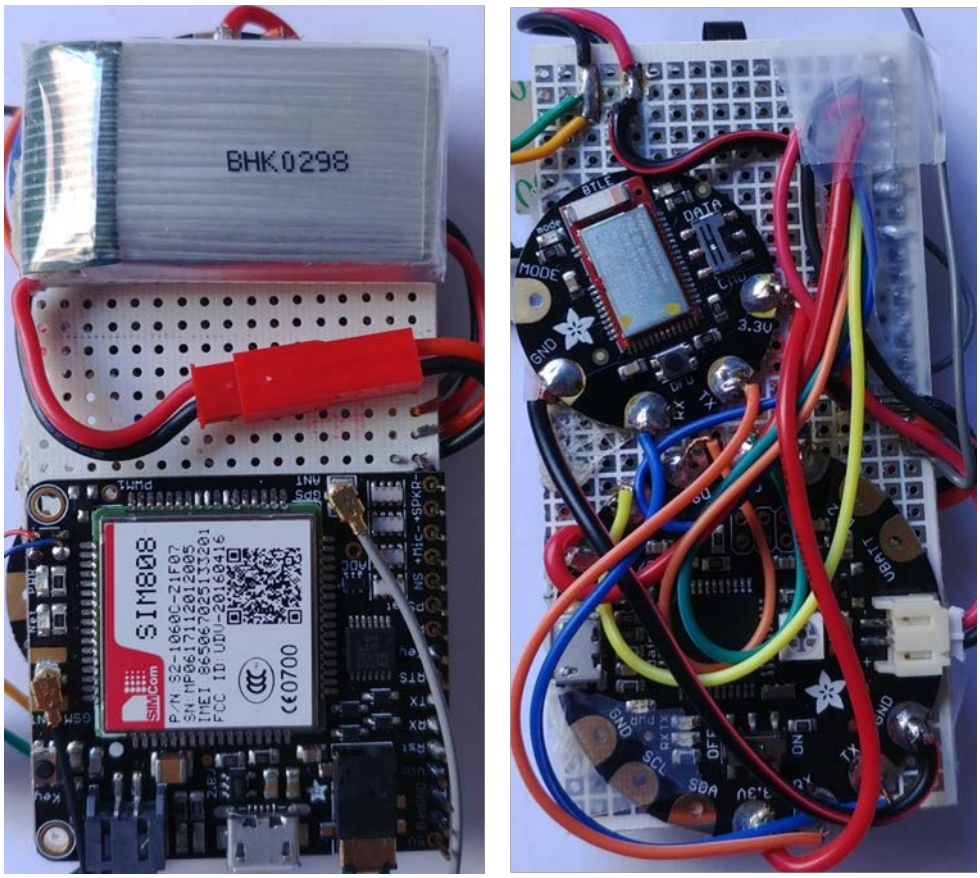

Figura 22. Dispositivo ensamblado. Izquierda (FONA + Batería), derecha (FLORA + BLE+ conexiones).

Para que el dispositivo pueda ser integrable sobre el operario y ser resistente, se ha analizado el uso diversos tipos de encapsulados donde introducir el dispositivo.

El dispositivo puede ir integrado sobre alguna prenda del operario, como el chaleco o el casco, o añadido sobre un elemento extra como un brazalete o una petaca. Es importante que la parte superior del dispositivo, donde se encuentran las antenas, tenga línea de visión lo más directa posible a cielo abierto, de esta manera, la señal GSM será mejor y el tiempo hasta obtener una posición de GPS válida se reducirá.

En la Figura 23 y Figura 24, se muestra la posición en la que se debería colocar el dispositivo sobre el chaleco y el casco del operario.

Figura 23. Posición del dispositivo sobre el chaleco.

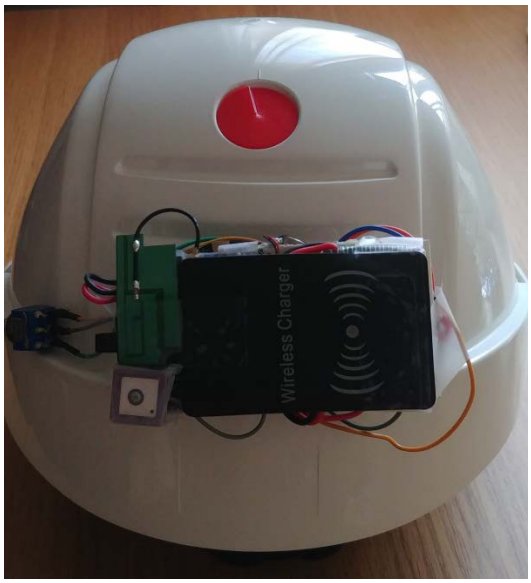

Figura 24. Posición del dispositivo sobre el casco.

Estas opciones que se incluyen sobre prendas del operario, deben tener un encapsulado especial, que incluso podría hacer variar el montaje y la distribución de los componentes sobre el dispositivo. Por tanto, en estas imágenes se muestra, a partir del prototipo fabricado para el proyecto, una propuesta de la posición del dispositivo sobre las prendas, pero para una posible solución final, se deberían analizar los materiales de las prendas y hacer un encapsulado estanco que se integrara de manera óptima y lo menos intrusiva posible con esos materiales.

Por otro lado, en la Figura 25, se muestra el dispositivo dentro de una petaca. Esta solución podría integrarse sobre el cinturón u otro elemento del operario, donde pueda ser integrable y poco intrusivo.

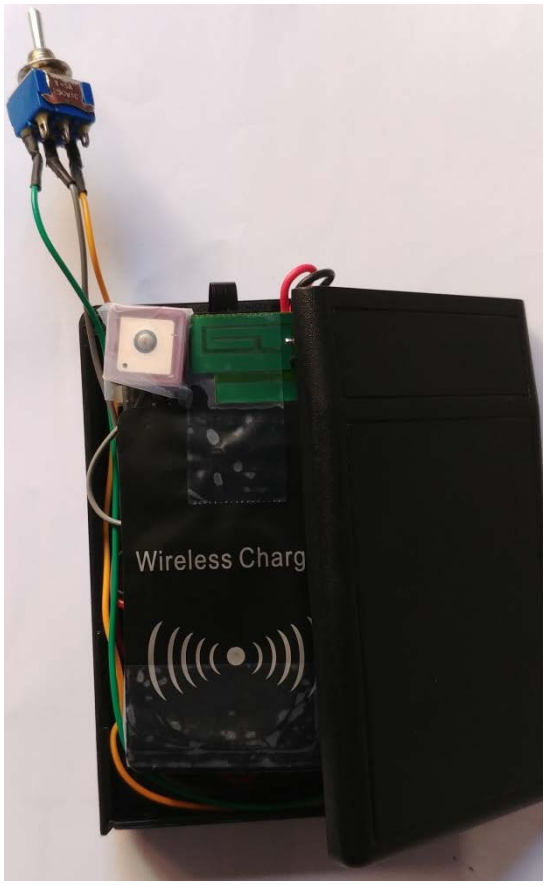

Figura 25. Dispositivo en petaca.

Figura 26. Dispositivo en brazalete.

Por último, en la Figura 26, se muestra la solución en la que el dispositivo iría dentro de un brazalete. Esta opción puede que sea la más intrusiva de las mostradas, ya que obliga al operario a llevar el brazalete equipado siempre que se encuentre realizando trabajos en vía. Pero, debido al reducido peso del dispositivo se considera que es aceptable para el operario.

El proyecto presenta estas opciones como posibles soluciones para la integración del dispositivo sobre el operario, pero es importante tener en cuenta que, al ser un prototipo, todas las opciones deben de pasar un proceso de pruebas e integración más exhaustivas. Además, gracias a la versatilidad de los componentes y a las plataformas sobre las que están desarrollados, pueden encontrarse más soluciones e, incluso, poder ofrecer soluciones a medida para cada tipo de cliente en la fase de comercialización.

5 *Software* del Sistema

El desarrollo *software* del proyecto consta de dos partes diferenciadas. Por un lado se ha desarrollado la aplicación móvil sobre el *IDE Android Studio*, basado en el lenguaje de programación *Java* y, por otra parte, el desarrollo del *firmware* del dispositivo se ha hecho sobre el *IDE Arduino* basado en C/C++. Pese a ser desarrollos independientes y diferentes entre ellos deben existir puntos en común, ya que crearán aplicaciones que deben comunicarse entre ellas. Para que esta comunicación sea eficaz, se define para el proyecto un protocolo de comunicaciones propio, que estructura los mensajes enviados y permite entender los mensajes recibidos.

En los siguientes apartados se describen los desarrollos de *software* realizados.

5.1 *Firmware* del dispositivo

La placa FLORA contiene el microprocesador basado en *Arduino* (*ATMega*) y por tanto es el componente que contendrá y ejecutará el *firmware*. Esta placa se comunica con los módulos de comunicaciones FONA y BLE a través de comandos AT que, en su mayoría, están implementados en las librerías que el fabricante tiene disponibles.

Para definir el desarrollo del *firmware* se parte de los requisitos del dispositivo que indican las funciones que debe realizar. A continuación se listan de manera sintetizada estas funciones:

- El dispositivo debe comunicarse con la *app* a través del módulo BLE.
- El dispositivo debe comunicarse con el CTC a través de la red de datos GSM.
- El dispositivo debe recibir su localización mediante posicionamiento GPS.
- El dispositivo debe procesar los mensajes tanto de la *app* como del CTC y generar las salidas o respuestas que sean necesarias.
- El dispositivo debe emitir una vibración de aviso al operario cuando la situación así lo requiera.

En la Figura 27 se representa el diagrama de flujo del dispositivo donde se observa inicialmente una fase de *setup* que se encarga de inicializar los sistemas y luego un *loop* donde se ejecuta secuencialmente todo el programa. Este proceso iterativo (*loop*) consta de diversas partes. Primero, el programa lee los estados de todos los subsistemas (BLE, GPS y GSM) y, envía el resultado a la *app* para que el usuario lo pueda visualizar, si alguno de los estados es conflictivo y no da el resultado esperado se vuelve a iniciar esta fase y a intentar la lectura. Una vez todos los subsistemas están funcionando correctamente se envía la posición al CTC y se espera su respuesta; si responde con un ACK, el dispositivo queda a la espera, primero, de que llegue algún aviso por parte del CTC (y si es así, emite la vibración de aviso al operario), y después, de que llegue alguna orden desde la *app*. En caso de que el ACK no llegase se esperaría un tiempo de guarda y se volvería a la fase de envío de la posición al CTC.

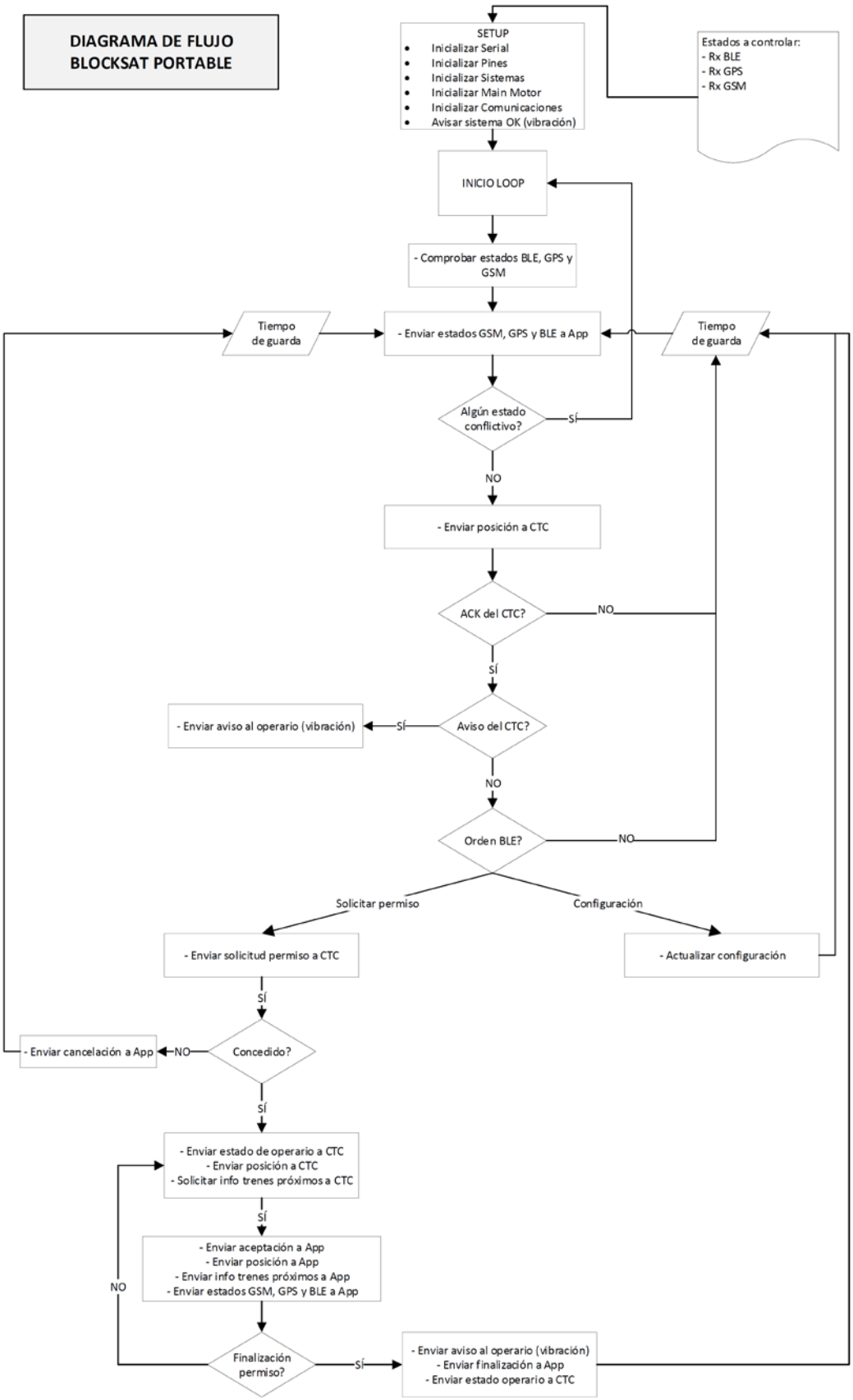

Figura 27. Diagrama de flujo del dispositivo.

En el caso de que no lleguen órdenes ni por parte del operario ni del CTC, el dispositivo irá enviando periódicamente (con un tiempo de guarda configurable) la posición al CTC y la información a la *app*. En cuanto llegue alguna orden desde la *app* el dispositivo actuará para gestionarla. Existen dos tipos de órdenes que pueden llegar y se gestionan de la siguiente manera:

- **Orden de configuración**: Esta orden contiene el valor de las configuraciones del dispositivo, estas son: IP y puerto del CTC, distancia máxima en el que se debe encontrar un tren cercano para que el dispositivo avise al operario, frecuencia de envío de datos al CTC y tiempo de trabajo que se enviará en la solicitud al CTC. Cuando se recibe esta orden, se actualizarán los valores de estas configuraciones en el dispositivo.

- **Orden de solicitud**: Cuando la orden contenga una solicitud de trabajo, el dispositivo actuará de pasarela para enviarla hacia el CTC y esperará su respuesta. Si la solicitud de trabajo se concede por parte del CTC, empezará la fase de trabajo en la que el dispositivo envía periódicamente la posición y el estado del operario (en función de si está trabajando o esperando órdenes) al CTC hasta que se finalicen los trabajos. El dispositivo también mantiene informado al operario a través de la *app* del estado del trabajo (solicitando, concedido, rechazado, trabajando o finalizado) y la información de trenes cercanos que llegue desde el CTC

Los trabajos se pueden dar por finalizados de tres formas diferentes:

- cuando se acabe el tiempo asignado para el trabajo. Este tiempo lo escoge el operario y debe ser aceptado por el CTC,

- cuando el CTC envíe una orden para finalizar el trabajo, o

- cuando el operario envíe una solicitud para finalizar el trabajo, que debe ser aceptada por el CTC.

Una vez finalizados los trabajos se enviará una actualización del estado del operario al CTC y a la *app* y un aviso al operario mediante una vibración. Después del aviso, el dispositivo volverá al *loop* donde se envían los estados a la *app*, la posición a CTC y se esperan de órdenes.

5.1.1 Estructura del código *Arduino*

El desarrollo se ha estructurado en diferentes pestañas dentro del *IDE Arduino* (Figura 28), y, pese a que en *Arduino* el código se ejecuta sin diferenciar las pestañas, queda ordenado de manera que cada pestaña contiene una parte de código correspondiente a un subsistema, además de las pestañas correspondientes a la gestión del procesado.

Figura 28. Pestañas del *firmware* en *Arduino*.

A parte de la pestaña *BlocksatPortable_Main*, donde están las funciones por defecto de *Arduino, setup*() y *loop*(), se han definido el resto de pestañas con una letra antes del nombre. Esto se ha hecho porque el *IDE* de *Arduino* ordena las pestañas alfabéticamente y compila el

código en orden desde la primera pestaña hasta la última, por tanto, si una pestaña quiere utilizar una variable de otra pestaña, ésta debe estar definida en alguna pestaña anterior.

De esta manera, las definiciones de constantes globales se han definido en la pestaña "*A_Definitions"*, dado que deben poder ser utilizadas en cualquier parte del código. Seguidamente, etiquetada con la letra "B", se encuentran las pestañas dedicadas a la inicialización de los diferentes sistemas y al análisis y parseo de los mensajes. Más adelante, con la letra "C" están las pestañas que corresponden a la gestión de los diferentes subsistemas. Por último, aparece la pestaña "*D_MainMotor*" que es la encargada de gestionar el orden de todo el procesado del *loop*(). A continuación se hace un pequeño sumario de las funciones que realizan cada una de las pestañas:

- BlocksatPortable_Main: Contiene las funciones ejecutables de *Arduino*, *setup*() y *loop*(). Estas funciones son las que ejecutará el procesador: en el caso de *setup*(), será una sola vez al iniciarse y, justo después, la función *loop*() se ejecutará cíclicamente mientras la placa esté encendida y no se reinicie.

  La función *setup*() llama a todas la funciones de inicialización definidas en todas las otras pestañas y, si todas se han ejecutado correctamente, emite una vibración para avisar al operario. Después de esto, cíclicamente, se van llamando tres funciones: una que comprueba la recepción de datos BLE, otra que hace lo mismo para la comunicación GSM y por último el MainMotor, que procesa estos datos y genera acciones.

- *A_Definitions*: Define las constantes globales. Estas corresponden a características para parsear los mensajes (cabeceras, separadores, etc.), definiciones de PINs, definiciones de tiempos (*timeouts*, períodos de envío, etc.) y configuraciones para la placa FONA (puerto, IP) y la placa BLE.

- *B_Inits*: Se encarga de inicializar las tres placas del dispositivo: por un lado la inicialización básica de la comunicación serie de FLORA y, por otro, las inicializaciones de las placas de comunicaciones.

- *B_Messages*: Contiene la definición de variables y la función encargada de parsear los mensajes recibidos para poder ser procesados correctamente por el programa.

- *C_BLE*: Contiene las funciones relacionadas con el subsistema BLE; estas son: la función de inicialización de las comunicaciones BLE y las funciones que reciben y envían datos a través de la conexión BLE con la *app*.

- *C_Battery:* Contiene las funciones encargadas de leer la información de la batería a través de comandos AT con la placa FONA y de enviar esta información a la *app*.

- *C_GPRS:* Contiene las funciones relacionadas con el subsistema GPRS; estas son: la función de inicialización de las comunicaciones GPRS, las funciones que se encargan de gestionar la conexión UDP, las funciones que leen el estado y la cobertura de la red GPRS y las que reciben y envían datos a través de la conexión GPRS con el CTC.

- *C_GPS:* Contiene las funciones encargadas de leer la información del GPS a través de comandos AT con la placa FONA y de enviar esta información a la *app* y al CTC.
- *C_Vibration:* Contiene la función encargada de hacer vibrar el avisador del operario.
- *D_MainMotor:* Contiene el motor de la aplicación, que se encarga de gestionar la recepción de mensajes y sus respuestas. También contiene el *MessageHandler*() que analiza el mensaje recibido, una es vez parseado, para lanzar las acciones correspondientes.

5.2 Aplicación Móvil

El sistema operativo móvil más utilizado en los últimos tiempos es *Android* [8] y gran parte de las empresas escogen dispositivos con este sistema operativo como teléfonos corporativos. La aplicación móvil desarrollada para este proyecto se ejecutará sobre los teléfonos corporativos de los operarios, por tanto, el desarrollo del prototipo se ha enfocado únicamente a este sistema operativo.

Para abarcar todo el espectro de dispositivos, o la mayoría de él, se deberían realizar otros desarrollos para el resto de sistemas operativos (como *iOS* o *Windows Phone*) o desarrollar una Web *app* multiplataforma que pueda ejecutarse en cualquier dispositivo. Para este proyecto se ha optado por limitar el prototipo a un solo sistema operativo.

Para definir el desarrollo de la *app* se parte de los requisitos de la aplicación que indican las funciones que debe realizar. A continuación se listan de manera sintetizada estas funciones:

- La aplicación debe disponer de una interfaz gráfica intuitiva.
- La aplicación debe mostrar la información del estado del dispositivo (apagado, en funcionamiento, nivel de batería, etc.) en todo momento.
- La aplicación debe contener los controles necesarios para que el usuario pueda enviar al CTC una solicitud para iniciar un trabajo en vía.
- La aplicación debe mostrar los trenes próximos a la zona donde se está operando.
- La aplicación debe tener un apartado dedicado a la configuración del dispositivo.

5.2.1 Diseño de la apariencia

Inicialmente se hace una propuesta gráfica de las pantallas o *layouts* que debe tener la *app*, para plasmar de forma concreta cómo sería la visualización de las funciones y la navegación del usuario a través de los *layouts*.

Se hace uso de la herramienta online *Ninja Mock* [9] para diseñar los *layouts* de la aplicación con sus diferentes elementos y poder tener cierta simulación de navegación entre las pantallas. En la Figura 29 se observa el diseño preliminar de las diferentes pantallas principales de la aplicación. A continuación se describen estas pantallas, de izquierda a derecha, de la figura:

- **Pantalla principal**: Es la pantalla que aparece al iniciar la *app*. Aparece el título de la *app*, los iconos de estado del dispositivo (cobertura GSM, posición GPS, conectividad BLE y nivel de batería) y el botón de solicitud de inicio de trabajos.
- **Menú lateral**: Aparecerá al desplazar el dedo desde la izquierda de la pantalla o al pulsar sobre los tres puntos que aparecen en la parte superior izquierda de la pantalla y muestra las opciones de navegación de la *app*.
- **Mapa de trenes**: Aparecerá cuando se seleccione esta opción en el menú lateral y muestra la localización del operario y de los trenes cercanos sobre un mapa.
- **Configuración**: Aparecerá al seleccionar la pantalla de configuración en el menú lateral. Permitirá configurar diferentes opciones tanto de la *app* como del dispositivo.

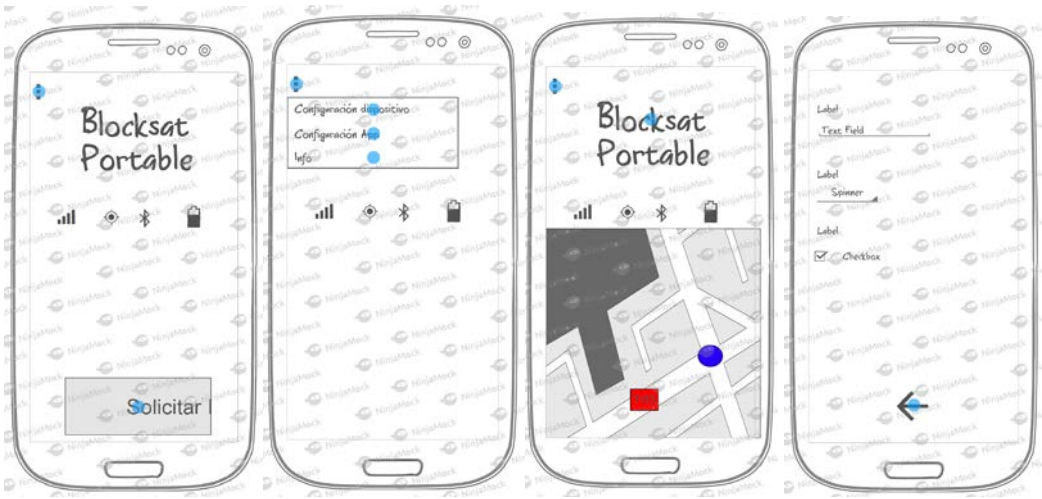

Figura 29. ***Mockup app BlockSAT® Portable*.**

A partir de estos prediseños se trabaja en el desarrollo de la aplicación. Durante el desarrollo se identifican nuevas necesidades que requieren añadir o modificar los *layouts* diseñados inicialmente. De esta manera, se añade un *layout* previo a la pantalla principal para que el operario pueda hacer *login* en la aplicación. Esta pantalla, además de tener los cuadros de texto donde introducir el identificador del operario y su contraseña, dispone de un hipervínculo que direcciona al usuario a una plataforma que permite solicitar el alta de un nuevo usuario en el sistema.

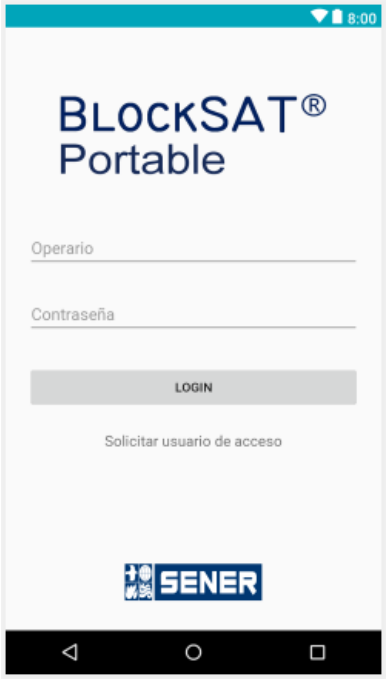

Figura 30. *Layout* de *login*.

Una vez se inicia sesión y se supera este *layout*, la aplicación da paso al *layout* con la lista de dispositivos *bluetooth* detectados.

Figura 31. *Layout* de dispositivos *bluetooth*.

Al seleccionar un dispositivo, la aplicación lanza la pantalla principal donde aparecen los iconos de estado, desde donde se pueden enviar solicitudes de trabajo al CTC y desde donde se accede al menú lateral.

Figura 32. *Layout* de la pantalla principal.

En la pantalla principal se encuentra el botón que permite solicitar el inicio de un trabajo al CTC. Una vez pulsado el botón, sobre esta misma pantalla, se irá actualizando el estado de la solicitud, según las respuestas que se reciban desde el CTC.

Desde el menú disponible arrastrando el dedo desde la izquierda (de izquierda a derecha) en la pantalla principal o pulsando sobre las tres rayas que aparecen en la parte superior izquierda de la pantalla, se puede acceder al resto de *layouts* disponibles, que son: el Mapa de Operación, la Configuración y el *layout* de información.

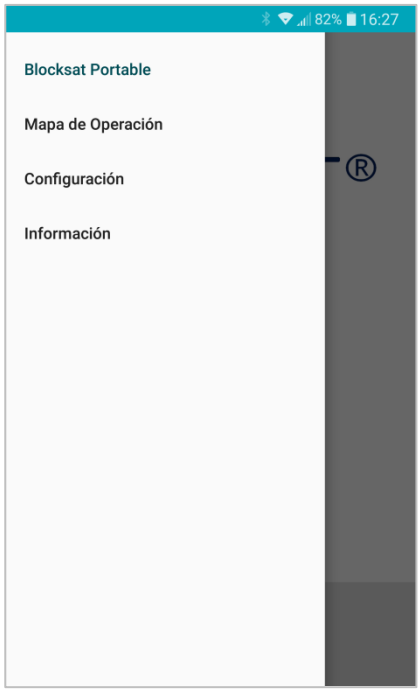

Figura 33. *Layout* con el menú lateral.

El *layout* del mapa de operación muestra un mapa de *Google Maps* con un marcador indicando la posición del operario. En el caso de que el CTC enviara la ubicación de algún tren cercano al operario, estos también aparecerían con un marcador en este mapa.

Figura 34. *Layout* del Mapa de Operación.

En la pantalla de configuración aparecen los parámetros del dispositivo que son configurables y un botón para enviar los valores al dispositivo para que se actualicen. Los parámetros son la IP y el puerto del CTC, la distancia a la que puede estar un tren del operario antes de lanzar un aviso, la frecuencia con la que el dispositivo envía la posición al CTC y el tiempo de trabajo con el que se va a enviar la siguiente solicitud al CTC.

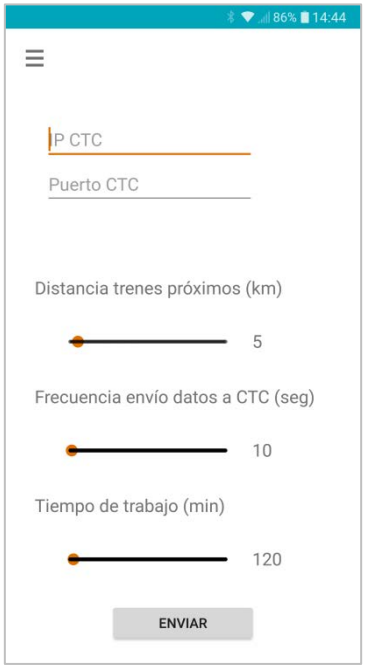

Figura 35. *Layout* de la pantalla de Configuración.

Por último, en la pantalla de información se muestra la versión en la que se encuentra la *app* y el contacto del desarrollador.

Figura 36. *Layout* de la pantalla de Información.

Como se observa en las capturas finales de los *layouts*, el concepto que se ha intentado seguir en el diseño es que tenga un aspecto intuitivo y que las pantallas sean funcionales. Para ello se ha seguido un diseño minimalista, tratando siempre de plasmar en las pantallas la información justa y necesaria para que sean funcionales pero sin cargar demasiado estéticamente el aspecto.

5.2.2 Estructura del código *Android*

Para desarrollar la *app* se ha utilizado el IDE oficial para la plataforma *Android*, *Android Studio*. El proyecto se ha desarrollado siguiendo la estructura de código que impone *Android Studio*. Esta estructura tiene un archivo *xml* con un manifiesto que proporciona información esencial sobre la aplicación al sistema operativo, información que el sistema operativo debe tener para poder ejecutar el código de la *app*. Además, debe tener al menos una clase de Java que contenga la actividad principal de la aplicación; esta es, por defecto, la *Main Activity*. Los elementos visuales como los *layouts* y otros recursos se colocan en el apartado *Resources*, donde también se pueden crear archivos *xml* con información que se utilice durante el código, como constantes globales de texto, colores o estilos.

En la Figura 37 se presenta la manera en la que se ha estructurado el código de la aplicación. Debajo del manifiesto se encuentran todas las clases de Java definidas; éstas se han dividido en carpetas para una mejor identificación del código. A continuación se describen estas carpetas y las clases que contienen:

- ***Activities***: Contiene las actividades de la aplicación. Las actividades contienen el código funcional asociado a las pantallas con las que los usuarios pueden interactuar. Esta carpeta contiene las siguientes clases:

    - *ConfigActivity*: Gestiona el lanzamiento e información de las opciones que forman la configuración de la aplicación y del dispositivo.
    - *InfoActivity*: Muestra la información de la aplicación (versión, contacto).
    - *LoginActivity*: Gestiona el lanzamiento y comprobación de datos del proceso de *login* y alta de un operario.
    - *MainActivity*: Gestiona el lanzamiento de las solicitudes del operario, la actualización de los iconos de estado y algunas opciones de la comunicación BLE.
    - *MapsActivity*: Muestra el mapa con la información de posicionamiento del operario y los trenes que llega desde el dispositivo.
- ***Adapters***: Contiene el adaptador para que la lista de dispositivos *bluetooth* encontrados pueda aparecer correctamente por pantalla.
    - *LeDeviceListAdapter:* Archivo Java que contiene las clases propias del adaptador.
- ***Communications***: Contiene las clases que gestionan la comunicación entre el *smartphone* y el dispositivo.
    - *BluetoothLeService:* Es el servicio que establece, mantiene, gestiona y finaliza la comunicación BLE.
    - *DeviceScanActivity:* Actividad que se encarga de escanear los dispositivos *bluetooth* cercanos y mostrarlos por pantalla.
- ***Others***: Contiene clases con otros propósitos.
    - *Auxiliary*: Contiene diferentes funciones públicas disponibles para ser usadas de manera auxiliar por otras clases.

Además de las clases de Java, una aplicación *Android* está compuesta por un conjunto de recursos. A continuación se describen los 5 grupos de recursos que tiene disponibles la aplicación:

- ***drawable***: Contiene los recursos correspondientes a las imágenes y logos que aparecen a lo largo de la aplicación. Algunos ejemplos son los iconos de estado como los de batería, señal o posicionamiento o los logos que aparecen en las pantallas.
- ***layout***: Contiene los archivos *xml* correspondientes a las pantallas que visualizará el usuario.
- ***menú***: Contiene los elementos que forman el menú lateral de la aplicación.
- ***mipmap***: Contiene el icono de la aplicación en diferentes formatos para adaptarse al dispositivo donde se esté ejecutando.
- ***values***: Contiene los valores que pueden ser utilizados a lo largo del código, como si de constantes globales se tratasen. Estos son colores predefinidos, cadenas de texto y estilos, que se pueden crear en estos archivos para después usarlos en el código.

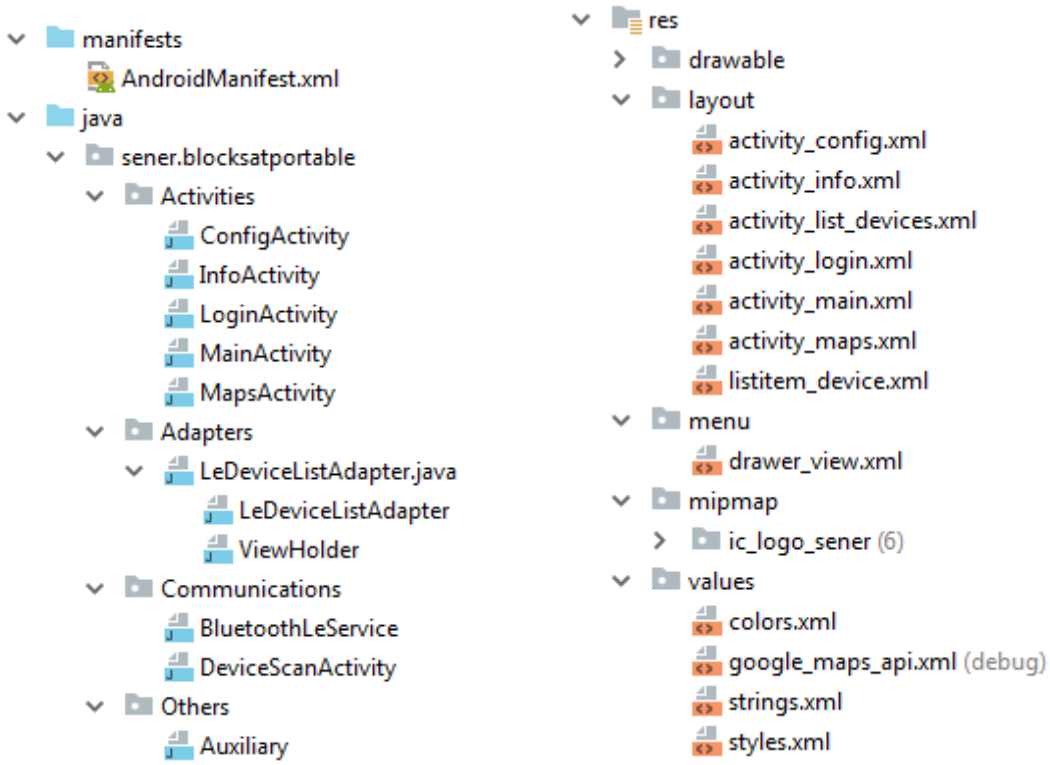

Figura 37. Estructura del código *Android*.

6 Interfaces e integración entre subsistemas

Teniendo en cuenta la arquitectura del sistema, se pueden identificar un total de tres subsistemas que forman *BlockSAT® Portable*. Estos subsistemas son: el dispositivo *wearable*, el *smartphone* y el CTC. Para que el sistema funcione, estos tres subsistemas deben comunicarse correctamente. Para ello, se establecen dos enlaces de comunicaciones bidireccionales desde el dispositivo, por un lado la comunicación entre el dispositivo y el *smartphone* y por otro la del dispositivo con el CTC.

En el presente capítulo se describen las interfaces que existen entre los diferentes subsistemas que forman parte de *BlockSAT® Portable*, el protocolo de comunicaciones diseñado para su integración y un ejemplo de cómo operan las interfaces.

6.1 Comunicación entre la aplicación móvil y el dispositivo

La comunicación entre la aplicación del *smartphone* y el dispositivo se establece a través de un enlace BLE.

Bluetooth Low Energy es la última especificación estable [10] de la tecnología *Bluetooth*, desarrollada por el *Bluetooth Special Interest Group*. Se diseñó como una tecnología complementaria al *Bluetooth* clásico para garantizar un consumo de energía bajo y menor tiempo de establecimiento de conexión. A pesar del uso de la misma banda de frecuencia (2.4 GHz) y las similitudes compartidas con la versión anterior (*Bluetooth* 3.0), BLE debe considerarse un nuevo estándar con objetivos y aplicaciones diferentes.

BLE está diseñado para la transmisión de pequeñas cantidades de datos en tiempos de transmisión muy pequeños y, por lo tanto, de muy bajo consumo de energía. No está pensado para mantener una conexión entre dispositivos por un largo tiempo transmitiendo grandes cantidades de datos a alta velocidad. Esto permite que los dispositivos estén activos solo cuando se les pide la transmisión de datos. Estas características dan lugar a optimizar las comunicaciones de algunos sistemas y encaja perfectamente con los requisitos del presente proyecto.

La topología de red de BLE es de tipo estrella. Los dispositivos *Master* pueden tener varias conexiones de capa de enlace con periféricos (*Slaves*) y simultáneamente realizar búsquedas de otros dispositivos. Por otro lado un dispositivo con el rol de *Slave* solo puede tener una conexión de capa de enlace con un único *Master*. En este proyecto se define como dispositivo *Master* o central al *smartphone* con la aplicación desarrollada y el *Slave* o periférico al dispositivo *wearable*.

Para la gestión de los dispositivos, la conexión y la interfaz de las aplicaciones, el BSIG define una pila de protocolos. La pila del protocolo BLE se divide en tres partes básicas: *Controller*, *Host* y *Application*.

El *Controller* es el dispositivo físico que permite transmitir y recibir señales radio e interpretarlas como paquetes con información. Contiene la capa física (*Physical Layer*), la capa de enlace (*Link Layer*) y la interfaz de control de host (*Host Controller Interface*).

El *Host* es la pila de software que administra cómo dos o más dispositivos se comunican entre ellos. Esta parte de la pila contiene una capa de control de enlace lógico y de protocolo de adaptación (*Logical Link Control and Adaptation Protocol, L2CAP*), administrador de seguridad (*Security Manager*), protocolo de atributo (ATT), perfil de atributo genérico (GATT) y perfil de acceso genérico (GAP).

Por último, la capa de *Application*, gestiona los perfiles de la capa *Host* y dispone de la interfaz ATBLE, con la cual una aplicación o usuario puede comunicarse con el dispositivo a través de comandos AT.

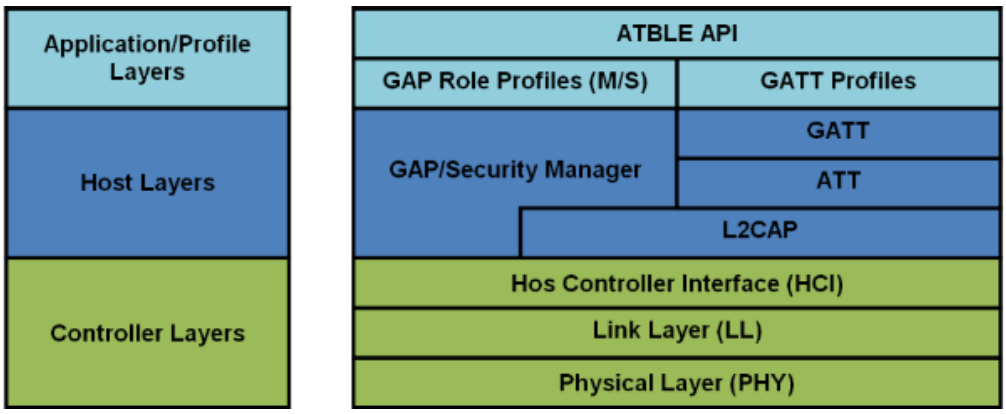

Figura 38. Capas del protocolo BLE [11].

La funcionalidad GATT define dos perfiles de atributo genérico: cliente y servidor. El cliente GATT es un dispositivo que accede a datos en el servidor GATT remoto a través de operaciones de lectura, escritura o notificación. En cambio, el servidor GATT es un dispositivo que almacena datos localmente y proporciona métodos de acceso a datos a un cliente GATT remoto. A diferencia de la distinción *Master*/*Slave* definida anteriormente, un dispositivo podría ser ambas cosas al mismo tiempo, según cómo la aplicación defina la estructura de datos y el flujo para cada lado de la conexión, si bien, lo más común, es que el dispositivo *Slave* (o periférico) sea el servidor GATT y el dispositivo *Master* (o central) sea el cliente GATT. La funcionalidad GATT de un dispositivo está lógicamente separada de la función *Master*/*Slave*. Las funciones *Master*/*Slave* controlan cómo se administra la conexión de radio BLE, y las funciones de cliente/servidor están dictadas por el almacenamiento y el flujo de datos. En el sistema desarrollado se han configurado el dispositivo y el *smartphone* como servidor y cliente GATT, respectivamente.

Para lograr el intercambio de información entre ambos dispositivos se implementa la estructura del protocolo GATT de manera que a los perfiles cliente y servidor se les otorgan ciertos servicios que a su vez están formados por una o más características. Estas características son las que contienen la información, y a través de la suscripción a su servicio un cliente puede tener acceso a esa información e incluso a modificarla.

El BSIG define algunos servicios [12] y características [13] que pueden ser consultados en su web y utilizados libremente en los proyectos BLE. Hay algunos de estos servicios que deben ser implementados obligatoriamente por el servidor; uno de ellos es el *Generic Access Service* [14], que incluye las características *Device Name* y *Appearance*. Aparte de estos servicios

predefinidos, es posible crear servicios y características propios y totalmente personalizados, siempre que usen sus propios identificadores UUID de 128 bits (generados manualmente o a través de algún servicio web gratuito).

Algunos fabricantes incluyen, en sus dispositivos BLE, servicios GATT personalizados a los que llaman "*standard services*". En el caso de la placa FLORA *Bluefruit LE* de *Adafruit* se incluye el servicio UART que es el medio estándar para enviar y recibir datos entre dispositivos conectados, y simula una interfaz UART de dos líneas (una línea para transmitir datos y otra para recibirlos). Está basado en una especificación de servicio UART patentada por *Nordic Semiconductors* [15]. El servicio consta de dos características, TX y RX, que se encargan de enviar y recibir datos entre el dispositivo central y el periférico.

El UUID del servicio UART es: 6E400001-B5A3-F393-E0A9-E50E24DCCA9E [16], este dato es importante a la hora de definir a qué servicio del dispositivo se va a suscribir la *app*.

Para el presente desarrollo se ha decidido utilizar únicamente el servicio UART disponible y definir sobre él un protocolo interno propio con el que se comunicarán el *smartphone* y el dispositivo. La especificación del protocolo BLE requiere que el tamaño máximo de carga útil de datos para los mensajes sea de 20 bytes. Este requisito se ha tenido en cuenta a la hora de definir el protocolo de comunicaciones, que se explica en detalle más adelante en el capítulo 6.3.

6.2 Comunicación entre el dispositivo y el CTC

La comunicación entre el dispositivo y el CTC se establece a través de un enlace GSM. GSM es un estándar desarrollado por el ETSI que describe los protocolos de la segunda generación de comunicaciones móviles digital; su implementación empezó en 1991 y ha llegado a ser el protocolo más extendido y usado en todo el mundo. Pese a que se han desarrollado nuevos protocolos más avanzados como los correspondientes a la tercera (UMTS) y cuarta (LTE *Advanced*) generación, el protocolo GSM es capaz de establecer comunicaciones de voz y de datos de forma optimizada.

En las primeras versiones de GSM aún no estaba disponible el servicio de datos; fue con la incorporación de los servicios GPRS y EDGE con los que se permitió el intercambio de datos a través de la red GSM, no solo servicios de mensajería (SMS y MMS) sino también con protocolos IP como TCP y UDP.

En el contexto del presente proyecto, y en particular en la interfaz con el CTC, se debe tener en cuenta que el sistema *BlockSAT®* ya está desarrollado por parte de la empresa. Esto implica que esta interfaz deberá adaptarse para integrarse correctamente con los sistemas existentes y ya en funcionamiento.

Actualmente la comunicación entre el CTC y los equipos embarcados en los trenes se hace a través de la red comercial GPRS utilizando el protocolo UDP disponible. Para integrar *BlockSAT® Portable* al sistema, en este proyecto, se propone utilizar la misma red y protocolos que ya están implementados en el sistema actual.

El protocolo UDP basa la comunicación de un sistema en el intercambio de datagramas entre dispositivos. Permite el envío de estos datagramas a través de la red sin que se haya

establecido previamente una conexión, ya que el propio datagrama incorpora suficiente información de direccionamiento en su cabecera para llegar a su destino. Este protocolo no tiene confirmación ni control de flujo, por lo que los paquetes pueden adelantarse unos a otros; y tampoco se sabe si ha llegado correctamente, ya que no hay confirmación de entrega o recepción. Esto hace que sea un protocolo ligero y rápido, pero obliga a establecer un control de flujo propio en el caso de que sea necesaria la confirmación de recepción del mensaje en el sistema.

El protocolo UDP permite definir el cliente y servidor de una comunicación. En el proyecto se define al dispositivo como el cliente y el CTC como servidor. Para establecer la conexión, el cliente se conectará al servidor a través de su IP y su puerto y, en el mensaje de conexión, indicará cuál es su IP para que el servidor pueda contestar a sus peticiones y poder enviar mensajes al cliente. Una vez establecida esta conexión el sistema ya estará preparado para intercambiar mensajes entre el dispositivo y el CTC.

Para esta interfaz también se ha tenido que definir un protocolo interno propio con el que se comunicarán el CTC y el dispositivo. La especificación del protocolo UDP requiere que el tamaño máximo de carga útil de datos para los mensajes sea de 508 bytes. Esta limitación sobrepasa de manera muy sobrada las necesidades de mensajería entre el dispositivo y el CTC; igualmente, se tendrá en cuenta en la definición del protocolo de comunicaciones propio para evitar llegar al límite o gestionar los mensajes en el caso de que se sobrepase.

6.3 Protocolo de comunicaciones

Como se ha mencionado en los capítulos anteriores, se describe un protocolo de comunicaciones propio para el correcto envío e interpretación de los mensajes que se intercambian los diferentes subsistemas. Este protocolo contempla la estructura y formato de los datagramas de información, es decir, qué contienen los mensajes que se envían y cómo están divididos.

Cada mensaje, independientemente de su origen y destino, debe estar formado por una cabecera, un parámetro y uno o varios valores que correspondan al parámetro. Cada una de estas palabras irá debidamente posicionada y entre ellas se separarán a través de unos separadores predefinidos. Debido a que la longitud de los mensajes está limitada (20 bytes en BLE, 508 bytes en GSM) se define un carácter que indica el final de un mensaje, de esta manera, el receptor no parseará el mensaje hasta que no identifique el último carácter. A continuación se muestra la estructura de los mensajes y el ejemplo de un mensaje con datos de posición GPS entre el dispositivo y la *app*.

[Cabecera]_[Parámetro]:[Valor1];[Valor2];[…];[ValorN]/
"DA_gsp: 41.490272; 2.107008/"

Figura 39. Estructura y ejemplo de un mensaje.

En las siguientes tablas se listan y describen en detalle todos los elementos que pueden formar parte de un mensaje. Así mismo, se describen los separadores que se utilizan para estructurar

el mensaje, las posibles cabeceras del mensaje, que dependen del origen y destino de la información, y el conjunto de parámetros y valores que pueden contener los mensajes según quién sea el emisor y el destinatario.

Tabla 2. Separadores del protocolo de comunicaciones.

Separadores	
Carácter	**Descripción**
"_"	Separador entre la cabecera y el parámetro
":"	Separador entre el parámetro y los valores
";"	Separador entre valores (no obligatorio)
"/"	Separador que indica el final del mensaje

Tabla 3. Cabeceras del protocolo de comunicaciones.

Cabeceras	
Caracteres	**Descripción**
"AD"	Indica que es un mensaje desde la Aplicación al Dispositivo
"DA"	Indica que es un mensaje desde el Dispositivo a la Aplicación
"DC"	Indica que es un mensaje desde el Dispositivo al CTC
"CD"	Indica que es un mensaje desde el CTC al Dispositivo

Tabla 4. Parámetro y valores del protocolo de comunicaciones.

Parámetros y Valores			
De *App* a Dispositivo [AD]			
Nombre	**Parámetro**	**Valores**	**Descripción**
Request	"req"	"str"	Solicita inicio de trabajo al CTC
		"stp"	Solicita fin de trabajo al CTC
		"pse"	Solicita una pausa en el trabajo al CTC
Initialization	"ini"	"ok"	Indica que la inicialización desde la *app* es correcta
		"nok"	Indica que la inicialización desde la *app* no es correcta
Configuration	"cfg"	"[*IP_CTC*]; [*Port_CTC*]; [*dist*];[*freq*]; [*time*]"	Envía al dispositivo los parámetros de configuración introducidos por el operario en la *app*.
De Dispositivo a *App* [DA]			
Nombre	**Parámetro**	**Valores**	**Descripción**
Initialization	"ini"	"ok"	Indica que la inicialización desde el dispositivo es correcta
		"nok"	Indica que la inicialización desde el dispositivo no es correcta
GCPS state	"gss"	"on"	GPS funcionando
		"off"	GPS no funcionando
		"sch"	GPS buscando posición
GPS Position	"gsp"	"[*lat*];[*lon*]"	Latitud y longitud de la posición recibida por el GPS
Battery Level	"blv"	"[*level*]"	Porcentaje con el nivel de batería restante
GPRS State	"grs"	"[*state*]"	El estado del sistema GPRS que reporta FONA
GPRS Level	"gri"	"[*rssi level*]"	El nivel de señal GPRS en dBm que reporta FONA
CTC Gateway	"cts"	"wai"	Indica que se está esperando una respuesta del CTC

state		"grn"	Indica que el CTC ha concedido permisos de trabajo
		"den"	Indica que el CTC ha denegado permisos de trabajo
		"end"	Indica que el CTC ordena la finalización de los trabajos
		"wrk"	Indica que el operario está dentro del rango de trabajo
CTC Gateway trains	"ctt"	"[*id*];[*lat*];[*lon*];[*dist*]"	Indica el identificador, latitud, longitud y distancia a la que se encuentra un tren según la información del CTC
De Dispositivo a CTC [DC]			
Nombre	**Parámetro**	**Valores**	**Descripción**
GPS Position	"gsp"	"[*lat*];[*lon*]"	Latitud y longitud de la posición recibida por el GPS
Request	"req"	"str"	Solicita inicio de trabajo al CTC
		"stp"	Solicita fin de trabajo al CTC
		"pse"	Solicita una pausa en el trabajo al CTC
		"trn"	Solicita información de trenes cercanos al CTC
New operator	"opn"	"[*id*]"	Envía el alta con el identificador de un nuevo operario al CTC
Operator state	"ops"	"wai"	Indica que el operario está esperando alguna acción
		"wrk"	Indica que el operario está trabajando
De CTC a Dispositivo [CD]			
Nombre	**Parámetro**	**Valores**	**Descripción**
New operator ok	"opn"	"[*id*];[*resp*]"	Responde a la solicitud de un nuevo operador con la respuesta ("ok" o "nok")
Request response	"rqr"	"[*req*];[*resp*]"	Responde a las solicitudes recibidas desde el dispositivo. La respuesta puede ser "ok" o "nok"
End of work	"wrk"	"end"	Envía un aviso conforme se deben acabar los trabajos
Trains info	"trn"	"[*id*];[*lat*];[*lon*];[*dist*]"	Envía el identificador, latitud, longitud y distancia a la que se encuentra un tren
Warning to operator	"wng"	"[*level*]"	Hace que le llegue un aviso al operador (vibración) según el nivel de gravedad (de 1 a 3)

Nota: en la columna "Valores" aparecen sin corchetes los valores literales que asigna el protocolo y, entre corchetes, el nombre del valor que deberá asignar cada programa.

6.4 Interfaces en operación

Una vez descritas todas las interfaces y cómo se realiza su integración, en este apartado se muestra un ejemplo de operación donde aparecen todos los elementos del sistema y el flujo de información y mensajes entre ellos.

La siguiente figura muestra un modo de operación normal donde el operador de vía enciende el dispositivo y solicita permiso para iniciar los trabajos al CTC, que responde afirmativamente y recibe la información y posición del operario hasta que finalizan los trabajos.

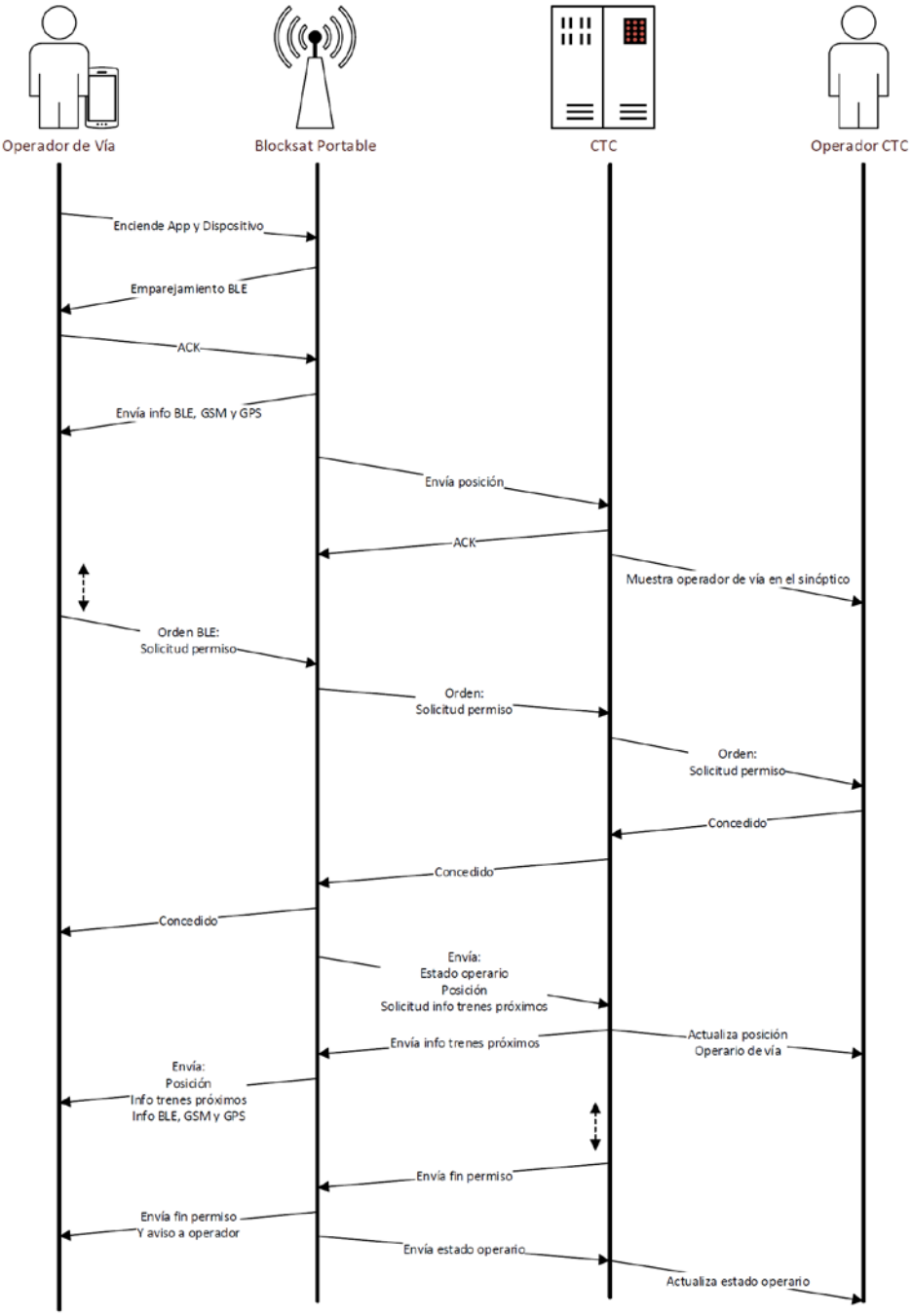

Figura 40. Ejemplo de interfaces entre dispositivos.

7 Prueba del Sistema

Una vez finalizado el desarrollo de todos los subsistemas, se pasa a una fase de pruebas del sistema, de manera que se pueda verificar el funcionamiento y se obtengan los resultados esperados.

A lo largo del desarrollo se han ido realizando pruebas unitarias de cada uno de los subsistemas y de la comunicación entre ellos. Se han probado cada uno de los componentes del dispositivo; primero, ejecutando códigos de test proporcionados por el fabricante, y, más adelante, ejecutando el código del dispositivo. Se empezó con la placa FLORA junto al módulo BLE, comunicándolo con una aplicación orientada para este propósito [17]. Más adelante, se le añadió el módulo FONA, en el que se ejecutó un código orientado a testear todos los sistemas [18].

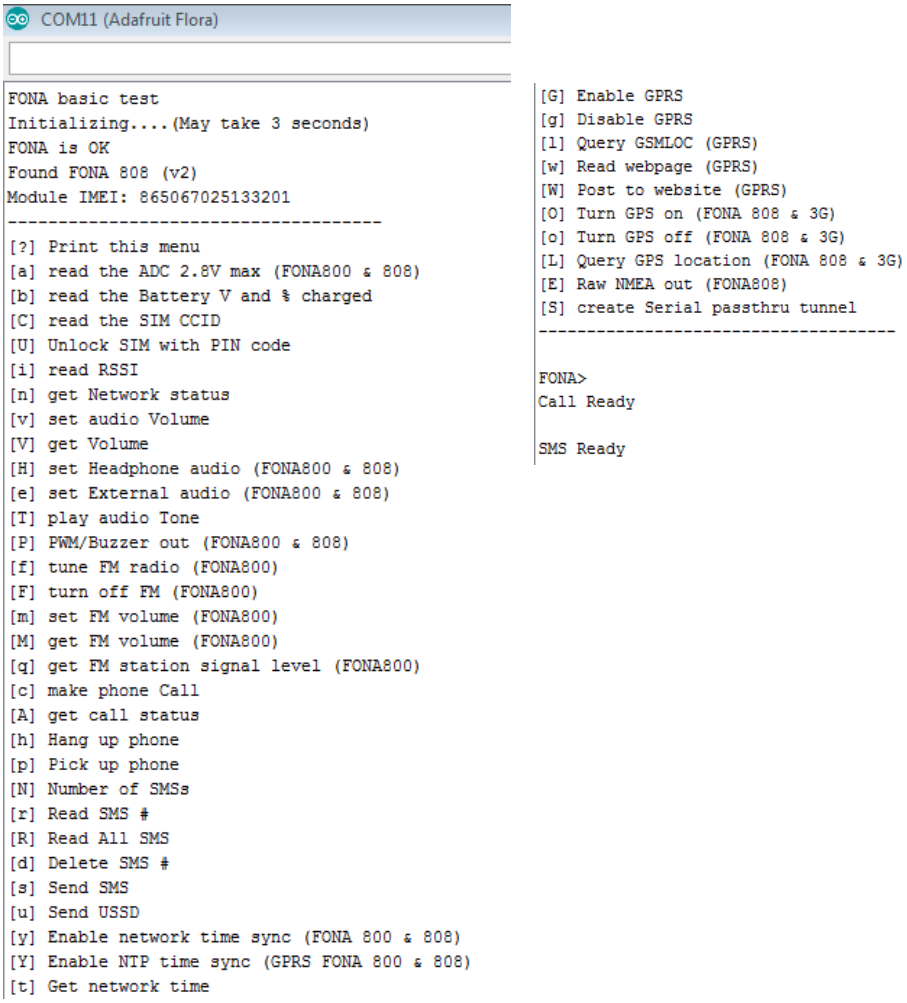

Figura 41. Captura del programa para realizar pruebas sobre la placa FONA.

Cuando se certificó el correcto funcionamiento de los componentes uno a uno se instaló el código con el *firmware* del sistema y se probó la comunicación con la *app*. En la Figura 42 se muestran las capturas hechas durante la realización de las pruebas unitarias sobre la comunicación entre el dispositivo y el *smartphone*. En estas capturas aparece; a la izquierda, el proceso de carga e inicio del *firmware* sobre la placa FLORA y la inicialización de los módulos

BLE y FONA; arriba a la derecha, el proceso de inicio de comunicaciones entre el dispositivo y la *app* y los primeros mensajes con datos que se envían; y, abajo a la derecha, el *layout* de la pantalla principal de la *app* al recibir datos desde el dispositivo, donde se observan los diferentes iconos con el estado de la red GSM, el sistema GPS, la conectividad BLE y el nivel de batería, además, en esta fase de pruebas, se habilitó una función que mostraba por pantalla los mensajes recibidos desde el dispositivo.

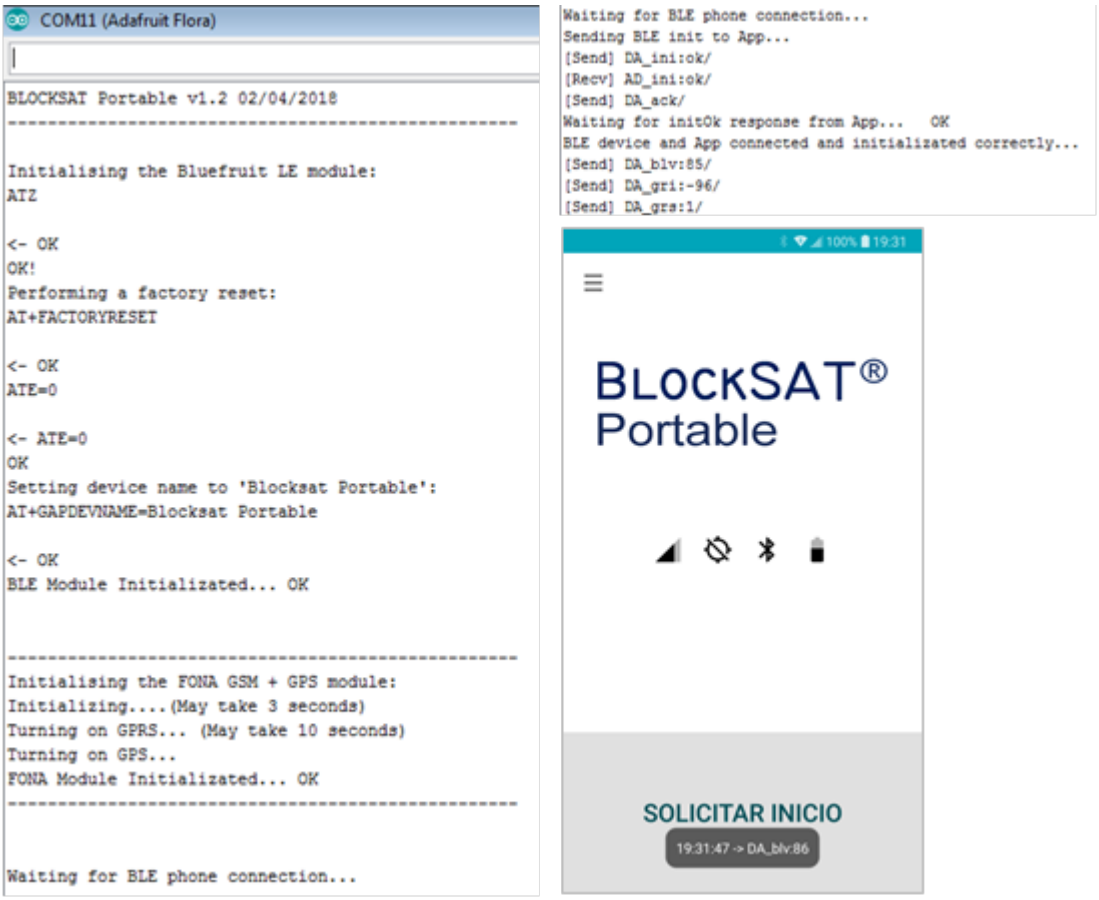

Figura 42. Capturas de las pruebas unitarias del sistema,

La prueba final del sistema consiste en hacer una demostración en un entorno lo más real posible siguiendo el ejemplo de operación mostrado en el capítulo anterior. En la prueba un usuario arrancará el dispositivo y utilizará la aplicación para solicitar un permiso de trabajo al CTC. El CTC deberá recibir los mensajes desde el dispositivo con la posición del operario y las solicitudes correspondientes, a las que responderá afirmativamente, en este punto el usuario recibirá la confirmación y empezará los trabajos en vía. Cuando el CTC envíe información de trenes cercanos aparecerán en el mapa de la aplicación. Este funcionamiento seguirá su curso hasta la finalización de los trabajos, que se producirá cuando el operario reciba un aviso a través de la vibración del dispositivo.

Debido a que el sistema *BlockSAT®* está actualmente en fase de implantación y que la integración con *BlockSAT® Portable* no entra en el alcance del proyecto, para realizar la prueba global del sistema se ha tenido que hacer una simulación del CTC. Para realizar esta simulación se generarán manualmente las respuestas del CTC hacía el dispositivo y el resto de mensajes que sean necesarios, siguiendo el protocolo de comunicaciones definido en el proyecto.

La prueba seguirá el hilo de operación mostrada en la Figura 40, pero sin tener en cuenta la parte que corresponde al operador del CTC, ya que solo se podrán analizar los mensajes que llegan al CTC simulado y no los que deberían llegar al HMI del operador. A continuación se muestran las capturas y resultados de este proceso.

1. **Encendido de *app* y dispositivo y emparejamiento entre ambos.** El dispositivo ilumina el LED al encenderse y al inicializarse emite una vibración. Cuando se realiza el emparejamiento entre el dispositivo y la *app*, en la *app* se actualizan los estados y empiezan a llegar mensajes y el dispositivo vuelve a emitir una vibración para avisar de que está a la espera de órdenes.

Figura 43. Encendido y emparejamiento.

2. **Envío de posición al CTC y espera de ACK.** En el terminal del puerto serie de la placa FLORA se ve como, primero, el CTC recibe las coordenadas GPS del dispositivo y, luego, como se recibe por UDP el ACK del CTC.

```
Received in CTC: DC_gsp:41.411682,2.139275/
[Send] DA_blv:81/
[Send] DA_gri:-86/
[Send] DA_grs:1/
UDP received: CD_ack/
```

Figura 44. Establecer comunicación con CTC.

3. **Solicitud de permiso y respuesta desde CTC.** En la *app* se pulsa sobre el botón de solicitud de permiso, en el dispositivo se recibe la orden y se envía al CTC, que la recibe y actualiza el estado a “concedido”.

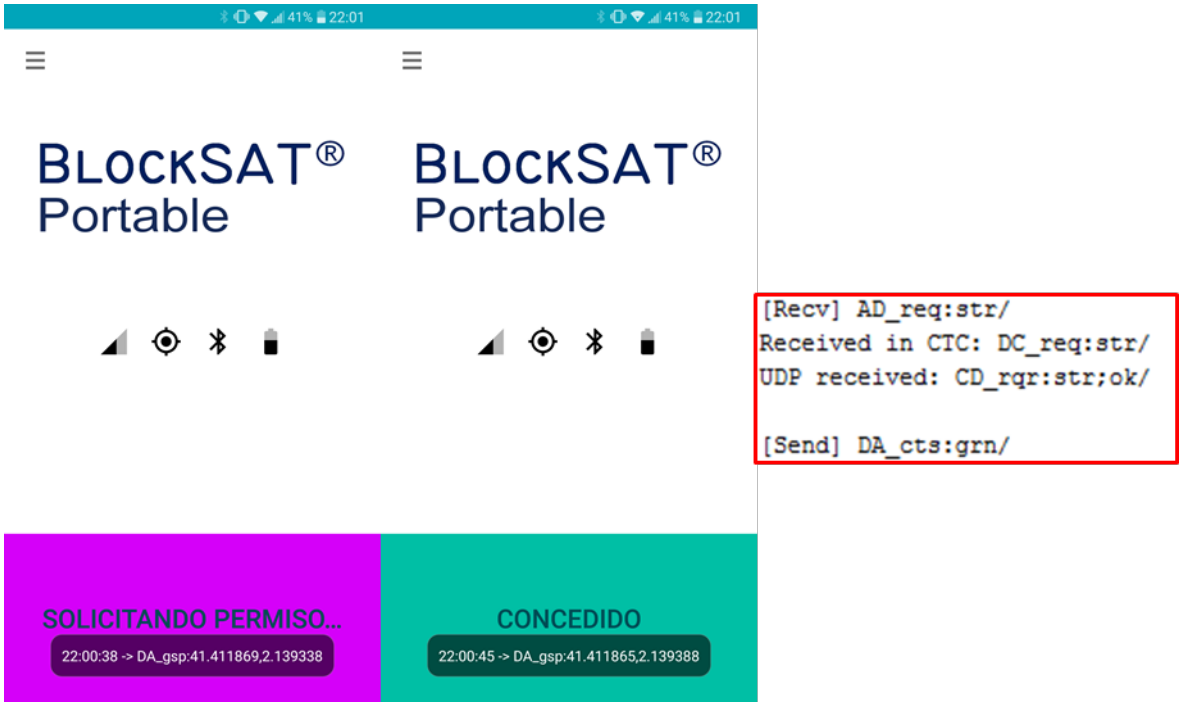

Figura 45. Gestión de permiso de trabajo.

4. **Comunicación entre el dispositivo y el CTC durante los trabajos.** Una vez concedido el permiso de trabajo, el operario puede empezar sus tareas. El dispositivo envía periódicamente una petición de información de trenes cercanos al CTC, el estado del operario y su posición. El CTC contesta con la información de trenes cercanos, si los hay.

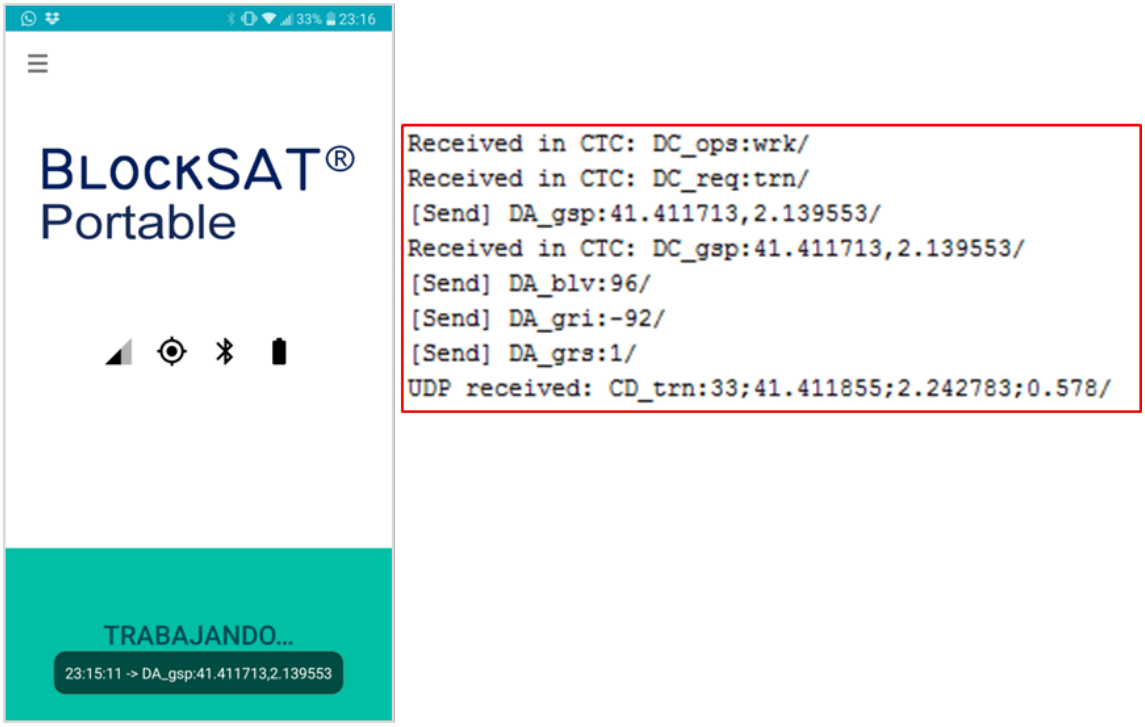

Figura 46. Comunicación dispositivo - CTC durante trabajo.

Dado que el CTC ha enviado un mensaje con información de trenes cercanos (“*CD_trn*”), esta información se reenvía a la app y, si se accede al Mapa de Operación, aparece el tren en el punto que ha enviado el CTC.

Figura 47. Captura del mapa con un tren cercano al operario.

5. **Rutina de fin de trabajo.** Cuando el permiso de trabajo expira, el CTC envía un mensaje indicando el fin del trabajo al dispositivo, éste transmite la información a la *app* y emite una vibración para avisar al operario. Por último, envía el estado del operario al CTC, donde le indica que el operario queda a la espera de alguna orden.

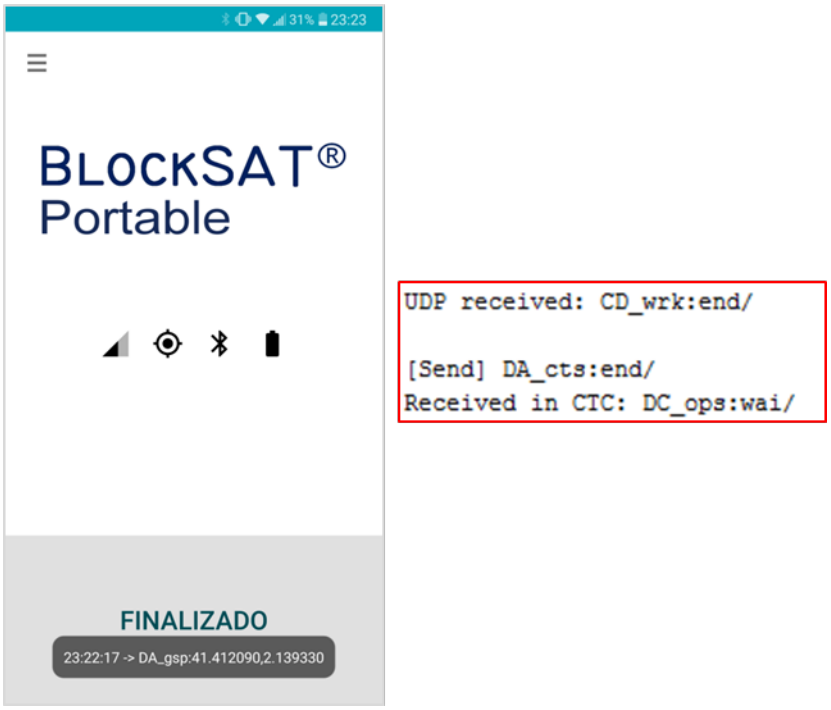

Figura 48. Rutina de fin de trabajo.

8 Análisis económico

El presente proyecto consiste en el diseño y desarrollo de un prototipo del sistema *BlockSAT® Portable* llevado a cabo por un ingeniero de desarrollo junior durante 5 meses a media jornada con una dedicación total de 400 horas. Para hacer el análisis económico del proyecto se deben tener en cuenta varios elementos, desde el propio material utilizado para la fabricación del prototipo, hasta el precio hora estimado de un ingeniero junior, pasando por los equipos necesarios para hacer el desarrollo.

A continuación, en la Tabla 5, se desglosa el coste de los componentes del prototipo electrónico. Se incluyen todos los componentes que forman parte del equipo final que se ha fabricado.

Tabla 5. Coste del prototipo electrónico.

Prototipo electrónico			
Concepto	**Cantidad**	**Precio**	**Coste Total**
Adafruit FLORA Wearable	1	12,71€	12,71€
FLORA Wearable Bluefruit LE Module	1	14,88€	14,88€
Adafruit FONA GSM+GPS	1	42,46€	42,46€
Passive GPS Antenna uFL 1 dBi gain	1	3,36€	3,36€
GSM Quad-Band Antenna uFL 3dBi	1	2,51€	2,51€
Vibrating Mini Motor Disc	1	1,66€	1,66€
Receptor Cargador Inalámbrico *Qi* conector *microUSB*	1	7,99€	7,99€
Batería LiPo (conector JST) 3.7V 750mAh	1	3,4€	3,4€
Brazalete deportivo para encapsulado	1	7,99€	7,99€
		TOTAL	**96,96€**

Aparte de los costes de los componentes del prototipo, el proyecto tiene otros costes derivados del desarrollo y pruebas del propio proyecto; en la Tabla 6 se desglosan estos costes. Estos costes, incluyen los equipos y programas necesarios para el desarrollo y las pruebas, y el coste propio del desarrollo en concepto de los recursos humanos utilizados.

Tabla 6. Coste de desarrollo y pruebas.

Desarrollo y pruebas			
Concepto	**Cantidad**	**Precio**	**Coste Total**
PC de desarrollo	1	500€	500€
*IDE Android Studio**	1	0€	0€
*IDE Arduino**	1	0€	0€
Smartphone Android con *BLE*	1	100€	100€
Cableado y adaptadores	1	10€	10€
Base para carga inalámbrica *Qi*	1	10€	10€
Ingeniero de desarrollo junior	400h	20€/h	8000€
		TOTAL	**8620€**

*Software libre

8.1 Industrialización

Uno de los objetivos del desarrollo del proyecto es la posibilidad de que el sistema *BlockSAT® Portable* acabe siendo producido de una manera industrial para su comercialización y uso junto al sistema *BlockSAT®*. En ese caso el análisis económico del prototipo realizado en el capítulo anterior debería variar para tener en cuenta el nuevo alcance.

Se debería tener en cuenta que gran parte del diseño del sistema surge del presente proyecto, por tanto algunos costes de desarrollo se podrían reducir. Sin embargo, se debería contar con una nueva partida de desarrollo para poder evolucionar el desarrollo del prototipo a un dispositivo final pensado para ser industrializable. Además, se debería crear un nuevo entorno de pruebas para dicho desarrollo final.

Por otra parte, en el mercado de componentes electrónicos los fabricantes ofrecen descuentos cuando se realizan compras de un gran número de unidades del mismo producto. Como ejemplo, el fabricante *Adafruit*, que fabrica la mayoría de componentes del prototipo, ofrece un 20% de descuento en todos sus productos si se compran más de 100 unidades. De esta manera el coste de fabricar el dispositivo se podría reducir alrededor de ese porcentaje, es decir, el coste por dispositivo sería de unos 80€.

Otra de las características del mercado electrónico, es su flexibilidad en la configuración de los componentes que se adquieren cuando son pedidos de gran escala. Algunos fabricantes ofrecen componentes personalizados ya sea en apariencia o en el diseño electrónico. De esta manera, se podría realizar un diseño electrónico personalizado para reducir el tamaño de los componentes y adaptarlo al encapsulado más eficiente para el operario.

El análisis de estos casos quedan fuera del alcance del presente proyecto pero debería tenerse en cuenta a la hora de evolucionar el sistema a un dispositivo industrializable y comercializable.

9 Conclusiones

La redacción de las conclusiones del presente proyecto puede ser tomada desde dos puntos de vista diferentes; por un lado están los conocimientos y hábitos de trabajo adquiridos durante su realización, y por otro lado, el cumplimiento de los requisitos del proyecto.

En lo que corresponde al primer aspecto, en los objetivos específicos marcados al inicio del proyecto se encuentra la voluntad de estudiar y aprender, desde un punto de vista tanto teórico como práctico, las tecnologías que se han usado en el desarrollo del proyecto. Al finalizar el desarrollo, se puede concluir que estos objetivos se han cumplido ya que la realización del proyecto ha permitido este aprendizaje.

Un proyecto individual de estas características permite alcanzar nuevas aptitudes en la investigación y puesta en práctica de tecnologías actuales, como BLE, *Arduino* o *Android*. Además del aprendizaje técnico, cabe destacar la experiencia que este proyecto ha aportado en la gestión y seguimiento de un proyecto de estas características, orientado a llegar a unos objetivos marcados por la empresa y con la vista puesta en la posible continuación del proyecto hasta ser comercializable.

Por otra parte, se debe evaluar el cumplimiento de los requisitos y del objetivo principal del proyecto, es decir, el diseño del sistema y la fabricación y puesta en marcha del prototipo. Teniendo en cuenta los resultados obtenidos, se puede concluir que el proyecto cumple los requisitos funcionales marcados al inicio del proyecto y, por tanto, se ha llegado al objetivo principal, que era desarrollar un sistema de posicionamiento de operarios de vía centralizado.

Como conclusión final, se destaca que en el proyecto se ha logrado diseñar y desarrollar un prototipo de un sistema complejo que contiene varios aspectos claves en la ingeniería de las telecomunicaciones. Esto incluye, el conocimiento de los componentes electrónicos para el diseño y montaje del dispositivo, el dominio de las herramientas y lenguajes de programación implicados en el desarrollo del *firmware* y de la aplicación móvil, y la aplicación de tecnologías inalámbricas como BLE, GPS y GSM para lograr una integración y funcionalidad completa de todos los subsistemas.

9.1 Líneas futuras

Las líneas futuras que surgen del presente proyecto se pueden dividir en dos grandes bloques: por una parte, las mejoras que se podrían proponer sobre el prototipo o el propio diseño del sistema, y, por otra parte, la evolución del sistema a producto final.

Un sistema de estas características suele ir acompañado de varias fases de prototipaje; por limitaciones temporales y de presupuesto, propias de un trabajo final de máster, este desarrollo se ha limitado a un único prototipo. En futuras fases de prototipaje se podrían incluir nuevas características al dispositivo, como podrían ser:

- La inclusión un altavoz para emitir avisos sonoros al operario, además de la vibración;
- la fabricación de un encapsulado con diseño propio a través de impresión 3D para una mejor integración en el entorno del operario; y,

- la ampliación de los servicios que ofrece la aplicación móvil de manera que, además de visualizar los trenes próximos y enviar solicitudes al CTC, se puedan realizar informes de pruebas sobre la aplicación y enviarlos directamente al CTC o reportar sobre el mapa incidencias o defectos en la vía o alrededores.

Por otro lado, surge la posibilidad de que el sistema supere la fase de prototipo y llegue a ser industrializable y comercializable. Para ello se debería contar con el interés y apoyo de la empresa para lanzar el desarrollo al siguiente nivel.

10 Referencias

[1] Memoria descriptiva Sistema *BlockSAT®*. SENER. Noviembre 2016.

[2] Ingeniería de Requisitos. Curso 2008-2009. Gonzalo Méndez. Dpto. de Ingeniería de Software e Inteligencia Artificial. Facultad de Informática. Universidad Complutense de Madrid.

[3] https://www.u-blox.com/en/product/c027 [Mayo 2018]

[4] https://www.u-blox.com/en/product/nina-b3-series [Mayo 2018]

[5] http://www.adafruit.com/products/659 [Mayo 2018]

[6] https://www.adafruit.com/product/2487 [Mayo 2018]

[7] https://www.adafruit.com/product/2542 [Mayo 2018]

[8] https://www.idc.com/promo/smartphone-market-share/os [Mayo 2018]

[9] https://ninjamock.com/ [Mayo 2018]

[10] https://www.bluetooth.com/bluetooth-technology/radio-versions [Mayo 2018]

[11] http://www.blueradios.com/hardware_LE4.0-S2.htm [Mayo 2018]

[12] https://www.bluetooth.com/specifications/gatt/services [Mayo 2018]

[13] https://www.bluetooth.com/specifications/gatt/characteristics [Mayo 2018]

[14] https://www.bluetooth.com/specifications/gatt/viewer?attributeXmlFile=org.bluetooth.service.generic_access.xml [Mayo 2018]

[15] https://developer.nordicsemi.com/nRF5_SDK/nRF51_SDK_v8.x.x/doc/8.0.0/s110/html/a00072.html [Mayo 2018]

[16] https://learn.adafruit.com/introducing-adafruit-ble-bluetooth-low-energy-friend/uart-service [Mayo 2018]

[17] https://play.google.com/store/apps/details?id=com.adafruit.bluefruit.le.connect&hl=es [Mayo 2018]

[18] https://github.com/adafruit/Adafruit_FONA/tree/master/examples/FONAtest [Mayo 2018]