UNIVERSIDAD CARLOS III DE MADRID

Escuela Politécnica Superior

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Diseño e implementación de un vehículo a escala controlado remotamente

Autor: Carlos González Hernández

Tutor: Javier Fernández Muñoz

Madrid, septiembre 2017

Índice

Índice de Tablas

Índice de Ilustraciones

Agradecimientos

Mis agradecimientos son para todas aquellas personas que me han apoyado durante esta etapa universitaria, en especial a mi familia y amigos. Tampoco hay que olvidar a los compañeros que han hecho que estos años se pasen tan rápido.

Resumen

El objetivo de este proyecto de fin de grado es el diseño e implementación de un vehículo a escala controlado remotamente mediante una aplicación desarrollada para teléfonos móviles con sistema operativo Android. El vehículo a escala será diseñado e implementado sobre un microcontrolador Arduino UNO y los elementos hardware de los que dispone. La comunicación entre la aplicación Android y el dispositivo físico se realizará a través de una red wifi. De esta forma se definirá la estructura del proyecto mediante el modelo Cliente-Servidor.

La aplicación Android en el dispositivo móvil será el cliente encargado de realizar las peticiones al servidor con las acciones que quiere ejecutar el usuario. Para ello, con el móvil conectado a la red wifi del vehículo, el usuario podrá controlar el avance y retroceso, las luces y el sonido a través de botones y el giro a través de la rotación del móvil. De esta forma el móvil será utilizado como un mando que resulte sencillo e intuitivo para el usuario.

El servidor será el dispositivo físico que recibe e interpreta las peticiones realizadas por la aplicación y activa los actuadores necesarios para ejecutar las acciones teniendo en cuenta el entorno mediante sensores. Por ello el dispositivo físico es considerado un robot [1]. La comunicación entre Cliente-Servidor a través de la tecnología wifi permitirá explotar sus beneficios como el rango o la conexión desde otras redes.

El objetivo final del proyecto es el diseño e implementación de un prototipo de vehículo a escala que ponga a disposición una red wifi a la que un usuario puede conectarse desde su dispositivo Android y controlarlo a través de la aplicación en tiempo real.

1. Introducción

En la actualidad, la utilización de sistemas distribuidos se ha extendido debido al desarrollo y utilización de microprocesadores de costo y tamaño reducido y al impulso de las redes y comunicaciones inalámbricas [2]. Los sistemas distribuidos podrían definirse como "sistemas cuyos componentes hardware y software se encuentran conectados en red y se comunican y coordinan para un fin concreto ofreciendo una visión de sistema único" [3].

A su vez los sistemas distribuidos se subdividen en función del modelo en procesamiento centralizado, conjunto de servidores y Cliente-Servidor, siendo este último el que predomina en la actualidad. De esta forma el sistema objetivo de este proyecto se podría explicar desde tres puntos: el servidor que sería el robot que representa el vehículo a escala, el cliente que sería la aplicación Android y la red wifi inalámbrica de comunicación entre cliente y servidor.

Como se comentaba anteriormente la reducción de costes y de tamaño de los microcontroladores también ha impulsado el uso de robots. Un robot es un sistema electrónico programable capaz de realizar operaciones reservadas a las personas [4] de forma autónoma, programable y atendiendo al entorno [1]. Las partes de las que se compone el vehículo son las típicas de los robots: sistemas de control y planificadores para una correcta ejecución de acciones, actuadores que generan movimientos, sensores internos que miden y guían a los actuadores, sensores externos que perciben el entorno y una fuente de alimentación que permita su funcionamiento sin conexiones físicas [5].

En la actualidad, el móvil es el dispositivo con mayor popularidad y extensión ya que el 66% de la población mundial dispone de uno y las ventas de smartphones continúan creciendo [6]. Al tratarse de un mercado tan amplio, el desarrollo de aplicaciones móviles ha adquirido una gran importancia. Las aplicaciones varían en función del sistema operativo utilizado por los móviles. Por ello la aplicación cliente será para dispositivos Android ya que su implantación es mayor (81.7% en 2016) frente a su competidor principal iOS (17.9% en 2016) y otros (0.4%) [7].

Los dispositivos interconectados y el Internet de las Cosas (IoT) están creciendo de forma exponencial en los últimos años, tanto en el ámbito industrial como en el uso diario [6]. Las conexiones entre diferentes dispositivos se pueden realizar a través de diferentes tecnologías como por satélite, infrarrojos, GSM, Bluetooth o wifi. Actualmente las conexiones más utilizadas, debido a la expansión de los smartphones antes comentada, son GSM, Bluetooth y wifi.

A partir de la información expuesta, este proyecto propone el diseño y construcción de un robot vehículo terrestre a escala en el que se montará un servidor al que se conectará a través de una conexión inalámbrica una aplicación Android cliente para que el usuario pueda controlarlo remotamente.

1.1. Motivación

Teniendo en cuenta la introducción, la evolución de las tecnologías hardware y las comunicaciones, así como la gran disponibilidad de dispositivos móviles, permiten controlar remotamente cualquier acción que se quiera realizar sobre otro dispositivo situado a cierta distancia.

Los avances tecnológicos permiten diseñar e implementar el control remoto del vehículo a escala disminuyendo la carga de trabajo y los costes derivados de la realización del proyecto. Las diferentes formas de implementación hardware y las distintas posibilidades de comunicación entre dispositivos existentes proporcionan un entorno de desarrollo en tiempo real, sencillo y eficaz para el control remoto.

Los sistemas controlados remotamente son sistemas que se encuentran en expansión especialmente en sectores como la domótica y el control de vehículos como UAV o automóviles. El control remoto se expandirá a otros sectores y en un futuro muchas de las tareas habituales que realizamos se podrán realizar a distancia desde nuestro dispositivo móvil, por lo que comprender su funcionamiento es de gran utilidad.

De esta forma, la principal motivación para la realización de este proyecto de fin de grado es la realización de un vehículo a escala controlado remotamente desde un dispositivo Android de forma sencilla, económica, eficaz y en tiempo real para comprender como se diseñan y se componen este tipo de sistemas distribuidos generando un trabajo abierto a futuras ampliaciones.

1.2. Objetivos

El objetivo de este proyecto realizado como Trabajo de Fin de Grado es el diseño y la implementación de un vehículo terrestre a escala controlado remotamente desde una aplicación Android como se ha expuesto brevemente en los puntos anteriores.

El proyecto pretende diseñar e implementar funcionalidades similares a las de un automóvil de forma que se puedan controlar las acciones desde la aplicación de forma sencilla e intuitiva para los usuarios. Además, se pretenden introducir condiciones de seguridad con el objetivo de asegurar la integridad y protección del vehículo a escala de forma independiente a las decisiones del usuario, como por ejemplo los sensores de distancia para detectar obstáculos.

Otro aspecto que se pretende introducir es el control de algunos aspectos través de los sensores del dispositivo móvil del usuario con el objetivo de que los controles del vehículo se asimilen a un volante o los mandos de cualquier vehículo. Otro objetivo relativo a este tema sería facilitar su integración futura en controles de realidad virtual, campo que en la actualidad está en expansión.

A través del presente documento se pretende ilustrar las fases desarrolladas en la realización del proyecto para alcanzar los objetivos presentados anteriormente y explicar el desarrollo del prototipo de vehículo a escala realizado.

1.3. Estructura del documento

En este apartado se indica como se ha estructurado este documento para explicar la realización del proyecto. La estructura seguida es similar a la indicada en la metodología Métrica Versión 3 [8], propuesta por el Ministerio de Hacienda y Administraciones Públicas del Gobierno de España para la sistematización de las actividades que dan soporte al ciclo de vida del software. Los apartados en los que se divide el documento son:

- **Introducción**: En este apartado se realiza una breve presentación del proyecto y del tema que trata, se presentan las motivaciones para su diseño e implementación y los objetivos que se pretenden alcanzar. También se explica de forma breve y concisa la estructura del documento y se incluye la explicación de algunos acrónimos y términos para facilitar la lectura del documento.

- **Estado del arte**: En este apartado se analiza el campo en el que se enmarca el proyecto: el control remoto de elementos hardware. Se examinarán productos similares existentes en la actualidad, así como las tecnologías utilizadas para su desarrollo. También se estudiarán las alternativas tecnológicas para la realización del proyecto indicando cuales se han utilizado para este proyecto y la razón de su elección.
- **Análisis del sistema**: En este apartado se realizará un análisis detallado de la funcionalidad del proyecto partiendo de una vista general del sistema. Para realizar un análisis en profundidad se definirán y analizarán los casos de uso y los requisitos de usuario para verificar que se han tenido en cuenta todas las necesidades. A partir de la información obtenida de los casos de uso y los requisitos de usuario se definirán los requisitos software del sistema que definirán la funcionalidad final. Para finalizar se realizará un análisis del trabajo realizado a través de matrices de trazabilidad.
- **Diseño del sistema**: En este apartado se realiza la selección de la arquitectura del sistema y se explica el diseño de cada uno de los componentes y la comunicación entre ellos. También se expone el diseño de las clases, el diseño de las interfaces de usuario y la especificación del entorno de desarrollo del proyecto.
- **Implementación**: En este apartado se describen los aspectos más importantes de la construcción del sistema a partir del diseño incluyendo puntos como las tecnologías utilizadas o el resultado final del sistema.
- **Plan de pruebas**: En este apartado se definen las pruebas que se deben realizar sobre el sistema para comprobar que su funcionalidad se corresponde con la definida en el análisis del sistema y el rendimiento del sistema es correcto.
- **Gestión del proyecto**: En este apartado se expone la planificación temporal inicial y real del proyecto para analizar las diferencias entre ellas. También se realiza el presupuesto del proyecto realizando un desglose del montante total.
- **Marco regulador y ético**: En este apartado se realiza el análisis de los aspectos legales de las herramientas utilizadas y de la normativa española y europea que afecta al sistema.

- **Conclusiones y trabajos futuros**: En este apartado se exponen las conclusiones resultantes de la realización del proyecto tanto relativas al sistema como opiniones derivadas de la realización del trabajo de fin grado. También se realiza un análisis de posibles trabajos que se podrían realizar para mejorar el sistema y campos en los que se podría desarrollar.
- **Referencias**: En este apartado se expone la documentación visitada y utilizada para la realización de este proyecto.

1.4. Acrónimos

- **NASA**: *National Aeronautics and Space Administration*
- **MAC**: *Media Access Control*
- **SSID**: *Service Set Identifier*
- **UDP**: *User Datagram Protocol*
- **IoT**: *Internet of things*
- **API**: *Application Programming Interface*
- **VCC**: *Voltage Collector-to-Collector (Voltaje)*
- **GND**: *Ground (Tierra)*

1.5. Definiciones

- **Tecnologia 5G**: Quinta generación de tecnologías de telefonía móvil.
- **UNIX**: Es un sistema operativo portable, multitarea y multiusuario; desarrollado, en principio, en 1969, por un grupo de empleados de Bell de AT&T.
- **Overclock**: Práctica que pretende alcanzar una mayor velocidad de reloj para un componente electrónico, por encima de las especificaciones del fabricante.
- **Actuadores**: Dispositivo capaz de transformar energía hidráulica, neumática o eléctrica en la activación de un proceso con la finalidad de generar un efecto sobre un proceso automatizado.
- **Seguridad WPA/WPA2 PSK**: Implementación de seguridad en las redes inalámbricas wifi que utiliza cifrado AES.
- **Pila TCP/IP**: es una colección ordenada de protocolos organizados en capas. Los protocolos inferiores proporcionan servicios a los protocolos superiores.

2. Estado del Arte

En este apartado se realizará un primer enfoque y un análisis de campo del control remoto de elementos hardware, contexto en el que se enmarca este proyecto. Posteriormente se analizarán algunos de los productos existentes y las herramientas y tecnologías disponibles y utilizadas.

Para poder comprender mejor el desarrollo del proyecto es necesario realizar un análisis previo del campo que comprende la tecnología del control remoto. Tradicionalmente el control remoto ha hecho referencia a los aparatos que a través de la transmisión de ondas infrarrojas permiten hacer llegar información de control a la máquina o sistema principal situado a una cierta distancia con el objetivo de facilitar la comodidad de los usuarios. Los elementos de control remoto tradicionales más conocidos son los mandos de televisión, DVD o equipos de música [9].

De esta forma los usuarios pueden dictar órdenes a través de acciones como presionar botones o mover los controles para manejar los dispositivos totalmente a distancia sin necesidad de que el dispositivo de control y el remoto estén conectados mediante cables. Esta capacidad hace que dichos sistemas sean cómodos y accesibles [10].

Pero en la actualidad, el control remoto dentro de los sistemas ha adquirido una gran importancia en campos como la domótica o la automoción y en dispositivos de la vida diaria. Esto se debe al avance de las telecomunicaciones y tecnologías utilizadas para enlazar dispositivos remotos unido al incremento constante del uso de dispositivos móviles, como se comentaba en la introducción. El auge del control remoto también está estrechamente relacionado con la importancia de los sistemas distribuidos.

Los sistemas distribuidos cuyos componentes hardware y software están en componentes separados conectados en red y se comunican/coordinan las acciones mediante un protocolo preestablecido. Este tipo de sistemas gozan de una gran importancia en la actualidad gracias a Internet, las redes móviles y las redes de área local que permiten el acceso remoto a otros dispositivos para controlarlo desde cualquier punto geográfico con una conexión a Internet. La popularidad actual de los sistemas distribuidos se debe al desarrollo de microprocesadores más potentes, pequeños y baratos unidos al desarrollo de las comunicaciones y redes inalámbricas.

2.1. Productos similares

En este punto se presentan algunos proyectos y productos existentes, partiendo desde los casos más similares al proyecto que se pretende realizar hasta los productos más complejos que tienen como base el mismo fundamento que se tiene para el desarrollo de este proyecto.

2.1.1. Proyectos similares

Al investigar en internet bajo las palabras clave “Arduino”, “Android” y “Car” se obtienen múltiples proyectos similares al que se pretende desarrollar. Estos proyectos corresponden a trabajos académicos y trabajos realizados en casa. La mayoría de estos proyectos exploran la potencia de integración entre Arduino y Android. A continuación, se muestran las imágenes de algunos de los prototipos creados en estos proyectos similares.

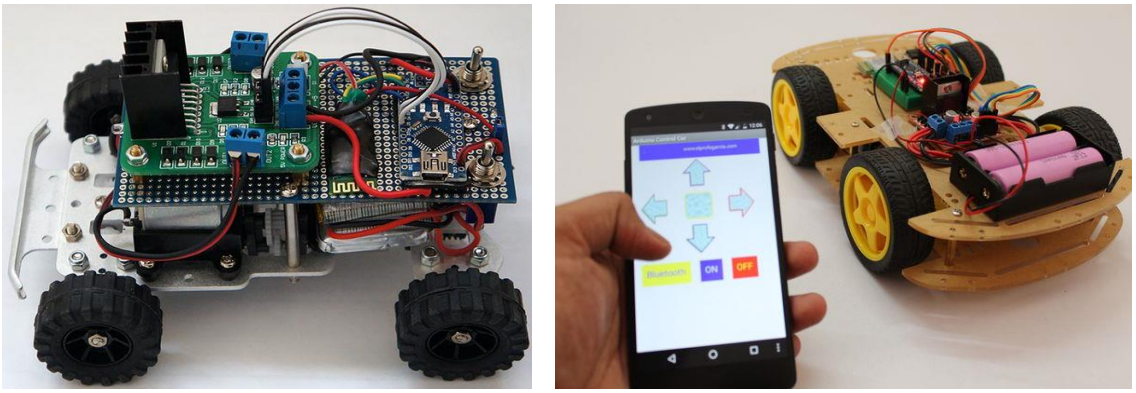

Ilustración 1. Proyectos similares [11]

Estos proyectos implementan el vehículo sobre un chasis donde se sitúa el controlador Arduino que maneja los motores y los sensores. La comunicación entre el vehículo y la aplicación Android se realiza prácticamente en su totalidad a través de Bluetooth. La mayoría de los mandos implementados en la aplicación Android de usuario son botones que traducen su accionamiento en acciones. Un porcentaje muy pequeño de estos proyectos utiliza el acelerómetro del dispositivo móvil o el control gestual para implementar los mandos de los vehículos.

Estos proyectos se tendrán en cuenta como guía de trabajo que se deberá realizar para alcanzar los objetivos e incluso mejorar estos proyectos similares.

2.1.2. Juguetes

Los vehículos controlados remotamente siempre han sido de gran interés tanto para niños pequeños como para personas adultas. Poder controlar un dispositivo a distancia con un mando radiocontrol es algo que llama la atención. Pero poder controlarlo directamente desde una pulsera o desde tu propio dispositivo móvil es aún más llamativo.

Algunos productos comercializados como juguetes se parecen en gran medida a los proyectos similares a este analizados en el punto anterior. Estos sistemas se venden implementados utilizando la tecnología Bluetooth como canal de comunicación con el controlador del vehículo. De esta forma solo se necesita que los compradores desarrollen su propia aplicación móvil, ya sea para Android o para iOS, para controlarlo remotamente. Alguno de estos ejemplos es:

Ilustración 2. Vehículo implementado comercializado [12]

Otros juguetes comercializados ofrecen una solución integral que comprende el vehículo y la aplicación que lo controla desde un dispositivo móvil o gestualmente mediante una pulsera que se comunican a través de Bluetooth, como por ejemplo el producto Droide BB-8 de la saga Star Wars [13]:

Ilustración 3. Juguete comercializado

2.1.3. Automoción

En la actualidad, se ha comenzado a explorar la conducción de automóviles de forma remota al igual que el trabajo que lleva realizando las agencias espaciales como la NASA durante años con sus robots exploradores en el espacio. El avance más destacable en el control remoto de automóviles se presentó en el Mobile Worl Congress (MWC) de 2017 donde Telefónica y Ericsson realizaron la primera prueba de conducción remota con tecnología 5G [14].

2.2. Herramientas y tecnologías utilizadas y alternativas

Como partes importantes del control remoto a través de un sistema distribuido se pueden identificar varias partes necesarias para el desarrollo del vehículo a escala controlado a distancia. Una de las partes que lo compondrá será una aplicación móvil a través de la cual el usuario decidirá las acciones que quiere realizar sobre el vehículo. Estas acciones se enviarán a través de una red inalámbrica al microcontrolador que ejecutará las acciones del vehículo a escala.

Por ello a continuación se analiza la situación de los sistemas operativos de los dispositivos móviles sobre los cuales se ejecutará la aplicación de usuario, los microcontroladores que ejecutarán las acciones recibidas y las redes inalámbricas existentes indicando para cada aspecto la opción que se utilizará en el desarrollo del proyecto.

2.2.1. Sistemas operativos

Como se indicaba en la introducción los dos sistemas operativos principales de dispositivos móviles son iOS y Android. Por ello son los dos sistemas operativos entre los que se opta para que uno de ellos sea la plataforma sobre la que se desarrolle la aplicación de usuario. A continuación, se realizará un breve análisis de cada uno.

iOS [15]

Es el sistema operativo para los dispositivos móviles de la compañía Apple Incl. Su lanzamiento se realizó el 26 de junio de 2007 y desde entonces se han realizado diversas actualizaciones. Originalmente este sistema operativo estaba pensado para los iPhone, pero posteriormente se implementó en el resto de dispositivos Apple.

Este sistema operativo está diseñado para equipos táctiles y se caracteriza por una interfaz dinámica, sencilla e intuitiva con una pantalla principal con las aplicaciones distribuidas en varias páginas y una barra configurable en la parte inferior con las aplicaciones más utilizadas por el usuario. Aun así, ofrece escasas posibilidades de personalización, no más allá de cambios básicos de colores o del fondo de pantalla ya que Apple no permite una variación en la configuración visual al tratarse de una seña de identidad de la compañía.

La multitarea es un aspecto presente en iOS, pero queda limitada a siete posibilidades simultáneas: audio en segundo plano, voz IP, localización en segundo plano, notificaciones push, completado de tareas, notificaciones locales y cambio rápido de aplicaciones.

También hay que destacar que, al tratarse de un sistema cerrado, tan sólo se puede utilizar en los exclusivos dispositivos de Apple caracterizados por ser de gama alta y tener un elevado precio. En cuanto a las aplicaciones, pese a que hasta la versión iOS 8 este sistema operativo móvil no permitía el empleo de Java o de Adobe Flash, Apple cuenta con la tienda de aplicaciones (App Store) más grande del mundo y si Apple no autoriza el uso de aplicaciones de terceros, no es posible disfrutar de ellas en sus productos.

Android [15]

Android es un sistema operativo para dispositivos móviles impulsado por Google, caracterizado por su código abierto basado en Linux. Es el sistema operativo más extendido entre los smartphones y las tabletas de última generación presentes en el mercado. Su historia comienza en 2003 con la fundación de Android Inc. con el fin de desarrollar un sistema operativo para móviles. En 2005, Google compró esta compañía, pero hasta 2008 no se comenzó a utilizar en dispositivos comercializados.

Desde su creación, Android ya cuenta con gran cantidad de versiones. El problema de las actualizaciones es que no son automáticas, sino que dependen de los fabricantes de aparatos móviles que suelen hacerlas incompatibles con los dispositivos más viejos para que los usuarios adquieran nuevos dispositivos.

El sistema operativo de Android se compone de los siguientes elementos que dan forma a su arquitectura interna:

- **Aplicaciones.** Escritas en lenguaje Java, destacan como básicas el correo electrónico, el calendario, el programa de SMS, los mapas, el navegador, los contactos y otros servicios. El resto ha de instalarse desde la Play Store.
- **Marco de trabajo de aplicaciones.** Se caracterizan por tener acceso al código de las aplicaciones base comentadas anteriormente para simplificar la reutilización de componentes y permitir al usuario reemplazar dichas aplicaciones.
- **Librerías.** Este sistema operativo integra un conjunto de bibliotecas, que son empleadas por diversos componentes, como la librería de medios o de gráficos.
- **Runtime de Android.** Se trata de un set de bibliotecas que ofrecen las funciones disponibles en las bibliotecas base del lenguaje Java.
- **Núcleo Linux.** Android depende de Linux para seguridad, gestión de procesos y memoria, pila de red y modelo de controladores.

Al igual que iOS, Android dispone de multitarea que permite a los usuarios gestionar diversas aplicaciones al mismo tiempo o incluso trabajar (en tabletas) con dos aplicaciones a la vez e ir cerrándolas cuando haya terminado su trabajo.

Por otro lado, Android permite su instalación en cualquier tipo de dispositivos móviles, independientemente de la marca, potencial y gama del equipo, gracias al código abierto de su software. Por ello, lo encontramos presente en dispositivos de múltiples compañías de telefonía. Dicho código abierto también permite la modificación o resolución de errores por desarrolladores expertos para obtener mejoras en su funcionamiento.

El sistema operativo seleccionado para que sea la plataforma sobre la que se despliegue la aplicación de usuario de este proyecto será Android especialmente porque se encuentra instalada en todo tipo de dispositivos móviles, desde aquellos de gama alta hasta los de gama baja, por lo que de esta forma se reducirán costes. Además, se facilita el desarrollo de aplicaciones de terceros, como sucede en este caso, al disponer de la herramienta Android Studio y utilizar un lenguaje tan extendido como Java.

2.2.2. Microcontroladores

Los microcontroladores son un circuito integrado que en su interior contiene una unidad central de procesamiento (CPU), unidades de memoria (RAM y ROM) y puertos de entrada y salida (Periféricos). Estas partes están interconectadas para formar un microcomputador que requiere de un programa que se almacena en la memoria ROM para que realice una función específica [16].

A continuación, se realiza un breve análisis de los microcontroladores de Arduino y, aunque sea un computador completamente funcional, de Raspberry Pi.

Raspberry Pi [17]

Raspberry PI es una placa computadora de bajo coste, se podría decir que es un ordenador de tamaño reducido desarrollado en el Reino Unido por la Fundación Raspberry PI (Universidad de Cambridge) en 2011, con el objetivo de estimular la enseñanza de la informática en escuelas, aunque no empezó su comercialización hasta el año 2012.

El concepto es el de un ordenador básico formado por una placa que soporta varios componentes necesarios en un ordenador común y es capaz de comportarse como tal. El diseño de la Raspberry Pi incluye:

- Un Chipset Broadcom BCM2835, que contiene un procesador central (CPU) ARM1176JZF-S a 700 MHz (el firmware incluye unos modos Turbo para que el usuario pueda hacerle overclock de hasta 1 GHz sin perder la garantía).
- Un procesador gráfico (GPU) Video Core IV.
- Un módulo de 512 MB de memoria RAM (aunque originalmente era la mitad).
- Un conector de RJ45 conectado a un integrado lan9512 de SMSC que proporciona conectividad a 10/100 Mbps.
- Dos buses USB 2.0.
- Una salida analógica de audio estéreo por Jack de 3.5 mm.
- Salida digital de video y audio HDMI.
- Salida analógica de video RCA.
- Pines de entrada y salida de propósito general.
- Conector de alimentación micro USB.
- Lector de tarjetas SD.

Arduino [18]

Arduino es una plataforma electrónica de código abierto basada en hardware y software fáciles de usar. Las tarjetas Arduino son capaces de leer entradas y convertirlo en una salida. Se puede indicar lo que Arduino debe realizar enviando un conjunto de instrucciones al microcontrolador en un sketch. Para ello se utiliza el lenguaje de programación de Arduino, similar a C, y el software Arduino (IDE).

Arduino nació en el Ivrea Interaction Design Institute como una herramienta fácil para el prototipado rápido, dirigido a estudiantes sin experiencia en electrónica y programación. A lo largo de los años Arduino se ha convertido en la base de proyectos, desde objetos cotidianos hasta complejos instrumentos científicos.

Hay muchos otros microcontroladores y plataformas disponibles para la computación física con ofertas de funcionalidad similar, pero Arduino simplifica el proceso de trabajo y ofrece algunas ventajas sobre otros sistemas por lo que ha sido el hardware seleccionado para el desarrollo de este proyecto:

- **Barato**: Las placas Arduino son relativamente baratas comparadas con otras plataformas. La versión más económica de Arduino puede ser ensamblada a mano, pero existen módulos de Arduino ensamblados que no llegan a 50 euros.
- **Multiplataforma**: El software de Arduino se ejecuta en sistemas operativos Windows, Macintosh OSX y GNU/Linux. La mayoría de los sistemas microcontroladores están limitados a Windows.
- **Entorno de programación simple y claro**: El entorno de programación de Arduino es fácil de usar para principiantes, pero suficientemente flexible para que usuarios avanzados puedan aprovecharlo también. Se basa en el entorno de programación Processing.
- **Código abierto y software extensible**: El software Arduino está publicado como herramientas de código abierto, disponible para programadores experimentados puedan realizar modificaciones. El lenguaje puede ser expandido mediante librerías C++. De forma similar, se puede añadir código AVR-C directamente en los programas de Arduino.
- **Código abierto y hardware extensible**: Arduino está basado en los microcontroladores ATMEGA8 y ATMEGA168 de Atmel. Los planos para los módulos están publicados bajo licencia Creative Commons, por lo que diseñadores experimentados de circuitos pueden hacer su propia versión del módulo, extendiéndolo y mejorándolo. Incluso usuarios relativamente inexpertos pueden construir la versión de la placa del módulo para entender cómo funciona y ahorrar dinero.

2.2.3. Redes inalámbricas

Las redes inalámbricas hacen referencia a la conexión de nodos por medio de ondas electromagnéticas sin necesidad de una red cableada con el objetivo de transmitir y recibir información a través de puertos. De esta forma se consigue la reducción de costes al eliminar las conexiones físicas, pero se debe introducir una seguridad mucho más exigente y robusta para evitar intrusiones [19].

Las tecnologías inalámbricas están alcanzando en la actualidad la madurez necesaria para acceder a una red, sin la necesidad de la utilización de cables. De esta forma se consigue dar soporte a los sistemas distribuidos para realizar el control remoto de sistemas. A continuación, se muestra el principal conjunto de tecnologías de conexión inalámbrica acompañada de una breve explicación obtenida de [20]:

- **GSM (Global System for Mobile communications):** El sistema global para comunicaciones móviles, es un estándar para la comunicación de teléfonos móviles que incorporan tecnología digital. Permite utilizar el sistema SMS (servicio de mensajes cortos), para enviar y recibir mensajes de texto. Es la evolución tecnológica de los teléfonos móviles analógicos.
- **GPRS (General Packet Radio Service):** Es un sistema de transmisión que funciona en el entorno de la telefonía móvil. En este sistema cada llamada de voz o cada conexión de datos, no ocupa de manera exclusiva un canal mientras dure esa llamada o conexión, por tanto, un usuario puede hacer uso de varios canales y un mismo canal puede ser compartido por varios usuarios. Está basado en la conmutación de paquetes y permite la transmisión de datos a alta velocidad para el acceso a Internet.
- **UMTS (Universal Mobile Telecommunications System):** El Sistema Universal de Telecomunicaciones Móviles, permite disponer de banda ancha en telefonía móvil y transmitir un volumen de datos importante por la red. Con esta tecnología de tercera generación son posible las videoconferencias, descargar videos o los intercambios de archivos digitales desde el móvil.
- **WAP (Wireless Application Protocol):** El Protocolo de Aplicaciones Inalámbricas (WAP) es un servicio de mensajes digital inteligente para teléfonos móviles y otras terminales que permiten visualizar contenidos de Internet en un formato de texto especial en un teléfono celular con tecnología GSM. WAP se ha convertido en el estándar global para proveer información a las terminales inalámbricas. Utiliza un microbrowser con el estándar WML (similar al HTML) optimizado para terminales móviles inalámbricas. WAP esconde la complejidad del GSM en las aplicaciones, así como la Web lo ha hecho para Internet. Expande una variedad de opciones de transporte y dispositivos, incluyendo SMS, 9.6 Kbps GSM data y GPRS.

- **WIMAX (Worldwide Interoperability for Microwave Access):** Es el nombre con el que se conoce la norma 802.16a, un estándar inalámbrico aprobado en enero del 2003 en el WiMax Forum que ofrece un mayor ancho de banda y alcance que la familia de estándares wifi, compuesta por el 802.11a, 802.11b y 802.11g. La diferencia entre estas dos tecnologías inalámbricas son su alcance y ancho de banda. Mientras que wifi está pensado para oficinas o zonas relativamente pequeñas, WiMax ofrece tasas de transferencia de 70Mbps a distancias de hasta 50 kilómetros de una estación base.
- **Bluetooth:** Es la norma que define un estándar global de comunicación inalámbrica a cortas distancias, que posibilita la transmisión de voz y datos entre diferentes equipos mediante un enlace por radiofrecuencia. Los principales objetivos que se pretende conseguir con esta norma son:
    - Facilitar las comunicaciones entre equipos móviles y fijos.
    - Eliminar cables y conectores entre éstos.
    - Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre nuestros equipos personales.

  La tecnología Bluetooth comprende hardware, software y requerimientos de interoperabilidad.

- **Wifi (Wireless Fidelity):** Es la tecnología utilizada en una red o conexión inalámbrica, para la comunicación de datos entre equipos situados dentro de una misma área (interior o exterior) de cobertura. Conceptualmente, no existe ninguna diferencia entre una red con cables (cable coaxial o fibra óptica) y una inalámbrica. La diferencia está en que las redes inalámbricas transmiten y reciben datos a través de ondas electromagnéticas, lo que supone la eliminación del uso de cables y, por tanto, una total flexibilidad en las comunicaciones.

Para la realización de este proyecto se utilizará la tecnología wifi. A pesar de que inicialmente el sistema está pensado para su utilización desde un dispositivo móvil cercano, en un futuro y gracias a la tecnología wifi se podría controlar el vehículo desde cualquier dispositivo en cualquier parte del mundo a través de Internet. Por ello la decisión de utilizar wifi es una decisión de futuro.

3. Análisis del sistema

En este apartado se va a realizar el análisis del sistema del vehículo a escala controlado remotamente. Este análisis se realizará siguiendo la estructura propuesta en Métrica Versión 3 [8], que consiste en obtener una especificación del sistema a construir mediante la captación de las necesidades y problemas para resolverlos y realizar un modelado del problema. Este punto será la base del diseño del sistema que se realizará posteriormente.

3.1. Definición del sistema

Esta actividad consiste en describir brevemente el sistema objeto del proyecto, identificando los problemas que se van a resolver, los sistemas que interactuaran en la solución, así como los destinatarios del sistema y el alcance que tendrá.

El sistema que se desarrollará consiste en un vehículo a escala robotizado controlado remotamente a través de una aplicación Android desde un dispositivo conectado a la misma red wifi. El control sobre el vehículo será principalmente el de ejecutar movimientos de forma segura y, secundariamente, realizar otras acciones de las que disponen los vehículos, como las luces o la bocina.

La visión general del sistema se puede observar en el esquema de la siguiente ilustración:

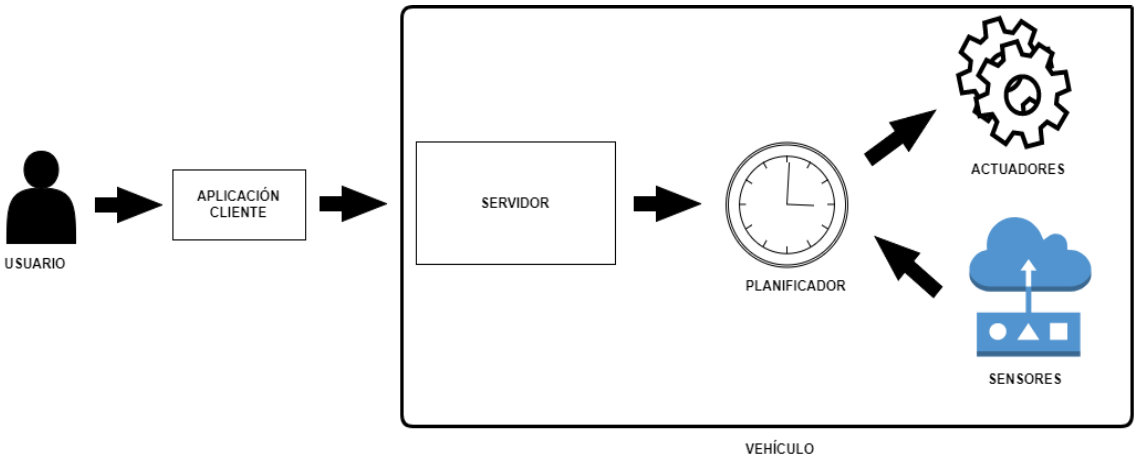

Ilustración 4. Visión general del sistema

Los elementos representados en el esquema y forma parte del sistema son:

- **Usuario**: Es la persona que controla las acciones a realizar por el vehículo a escala de forma remota desde la aplicación Android en su dispositivo móvil.
- **Aplicación cliente**: Es el software implementado para Android que comprueba la conexión con el vehículo y dispone los controles para manejarlo de forma remota a través de un panel de botones y los sensores del dispositivo móvil.
- **Servidor**: Es el elemento encargado de proporcionar un IP y un puerto donde la aplicación cliente se conecta y recibir las peticiones con las acciones que el usuario quiere que el vehículo realice.
- **Planificador**: Es el elemento encargado de leer las peticiones y garantizar que las acciones se realizan cumpliendo los plazos de tiempo y atendiendo a los datos del entorno devueltos por los sensores.
- **Actuadores**: Aquí se engloban todos los elementos del hardware del vehículo que realizan las acciones: motores, zumbador o leds.
- **Sensores**: Son los elementos del hardware del dispositivo que se encargan de recoger información del entorno para que pueda ser utilizada por el planificador en tareas de control.

El alcance del sistema delimita los problemas y necesidades que cubre. De esta forma el sistema permitirá a los usuarios de la aplicación Android conectarse a la red wifi del vehículo a escala y poder controlarlo remotamente. El sistema ejecutará en el vehículo las acciones establecidas por el usuario desde el móvil en tiempo real. La funcionalidad completa del sistema se definirá a lo largo del documento.

Este documento está dirigido en primer lugar al jefe de proyecto (Carlos González Hernández), al supervisor del proyecto (Javier Fernández Muñoz) y a todas las personas interesadas en el desarrollo del prototipo resultante de la realización de este proyecto.

3.2. Análisis de los casos de uso

En este punto se va a realizar un análisis detallado de la funcionalidad que debe ofrecer el sistema a través de la definición de los casos de uso. Los casos de uso se definen a través de la interacción entre el sistema y un agente externo, denominado actor. De esta forma se proporciona una imagen global y de alto nivel del sistema que se utilizará para revelar las necesidades que el sistema debe cubrir.

El estudio de los casos de uso se realizará a partir de las actividades y funciones que los usuarios deberían poder realizar para que el funcionamiento del sistema de control remoto del vehículo a escala fuese correcto. Para definir los diferentes casos de uso perfectamente se utilizará un diagrama y una descripción a través de una tabla con el siguiente formato:

Identificador		**Nombre**	
Actores			
Objetivo			
Escenario			
Condiciones de fallo			

Tabla 1. Formato tabla de especificación de casos de uso

En la tabla se definen los campos:

- **Identificador**: Código único identificativo de los diferentes casos de uso. El formato del identificado será **CU-NN**, siendo NN el numero de la secuencia de casos de uso.
- **Nombre**: Nombre identificativo asignado al caso de uso.
- **Actores**: Hace referencia al tipo/rol de usuario que interactúa en el caso de uso.
- **Objetivo**: Breve explicación del proceso realizado en el caso de uso.
- **Escenario**: Descripción de como el actor interactúa con el sistema y cuál es la respuesta que ofrece el sistema.
- **Condiciones de fallo**: Descripción de los fallos que pueden darse en el progreso del caso de uso y las respectivas respuestas del sistema.

3.2.1. Roles en el sistema

Antes de comenzar con la especificación de los diferentes casos de uso conviene establecer los diferentes roles que serán los actores de los diferentes casos de uso. Los actores de los casos de uso no tienen que ser exclusivamente una persona que interactúa con el sistema, sino que también puede ser un sistema externo. Por ello en este sistema se pueden observar dos roles principales:

- **Usuario final**: Personas que pueden utilizar la aplicación Android para controlar remotamente el vehículo a escala. Las funciones que podrá realizar son conectar/desconectar el smartphone a la red wifi del vehículo; conducir el vehículo (podrá avanzar/retroceder, girar, encender/apagar las luces y accionar la bocina); y consultar la información de utilización.
- **Sensores**: Son los dispositivos hardware conectados al microcontrolador que recogen información del entorno. Su función es modificar las acciones que se ejecutan sobre el vehículo en función de los datos del entorno proporcionados, como por ejemplo la distancia a un obstáculo obtenida por el sensor de ultrasonidos o el porcentaje de luz actual recogido por el sensor de luminosidad.

3.2.2. Especificaciones de los casos de uso

Ilustración 5. Caso de uso CU-01

Identificador	CU-01	**Nombre**	Error de conexión wifi
Actores	Usuario		
Objetivo	El usuario no puede acceder a la funcionalidad de la aplicación ya que no se encuentra conectado a la red wifi con el SSID y la dirección MAC correspondiente.		
Escenario	1. El usuario inicia la aplicación. 2. La aplicación comprueba que el dispositivo no se encuentra conectado a la red wifi correspondiente. 3. Se muestra al usuario la pantalla de error de conexión wifi.		
Condiciones de fallo	- El dispositivo no tiene activada la opción de conexión wifi. Se mostrará la pantalla de error hasta que lo active y se conecte a la red del vehículo. - El dispositivo se encuentra conectado a otra red wifi. Se mostrará la pantalla de error hasta que se conecte a la red del vehículo.		

Tabla 2. Caso de uso CU-01

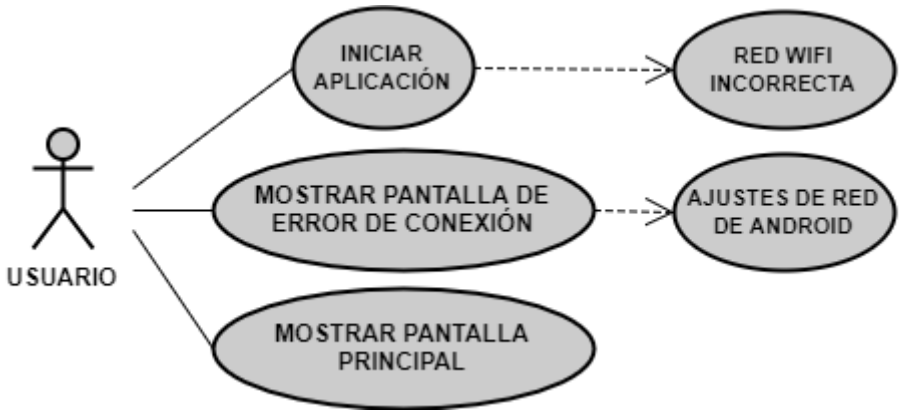

Ilustración 6. Caso de uso CU-02

Identificador	CU-02	**Nombre**	Configuración de red wifi
Actores	Usuario		
Objetivo	El usuario accede a los ajustes de red wifi del dispositivo Android y posteriormente regresa a la aplicación.		
Escenario	1. El usuario inicia la aplicación. 2. La aplicación comprueba que el dispositivo no se encuentra conectado a la red wifi correspondiente. 3. Se muestra al usuario la pantalla de error de conexión wifi y el usuario selecciona la opción de configuración. 4. El usuario configura y se conecta a la red wifi correspondiente al vehículo. 5. El usuario regresa a la aplicación y accede directamente a la pantalla principal.		
Condiciones de fallo	- El dispositivo mantiene desactivada la opción de conexión wifi. Se mostrará la pantalla de error hasta que lo active y se conecte a la red del vehículo. - El dispositivo se ha conectado a una red errónea. Se mostrará la pantalla de error hasta que se conecte a la red del vehículo.		

Tabla 3. Caso de uso CU-02

Ilustración 7. Caso de uso CU-03

Identificador	CU-03	**Nombre**	Salir de la configuración wifi y de la aplicación
Actores	Usuario		
Objetivo	El usuario sale de aplicación desde la pantalla de configuración wifi ante la negativa de configurar la red.		
Escenario	1. El usuario inicia la aplicación. 2. La aplicación comprueba que el dispositivo no se encuentra conectado a la red wifi correspondiente. 3. Se muestra al usuario la pantalla de error de conexión wifi y el usuario selecciona la opción de salir. 4. Se sale de la aplicación.		
Condiciones de fallo	La aplicación detecta la red wifi correspondiente al vehículo. Se saldrá de la aplicación de todas formas.		

Tabla 4. Caso de uso CU-03

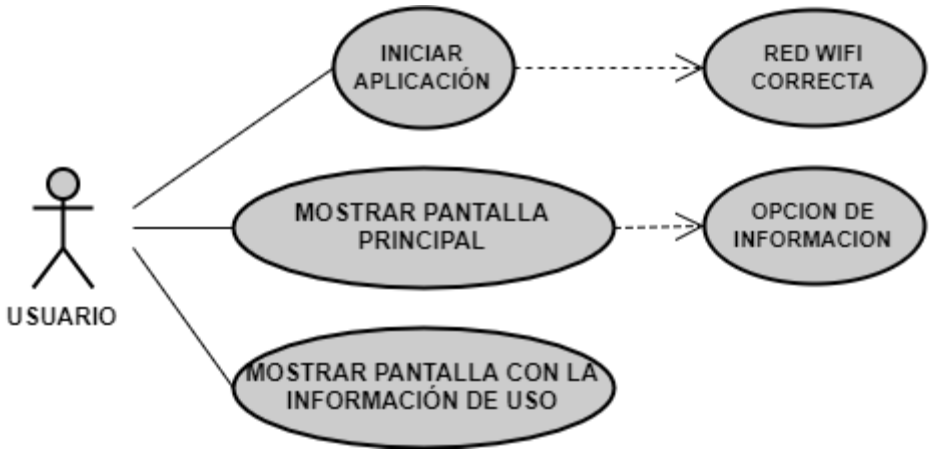

Ilustración 8. Caso de uso CU-04

Identificador	CU-04	**Nombre**	Mostrar información de uso
Actores	Usuario		
Objetivo	El usuario visualiza la información de cómo se controla el vehículo remotamente a través de un pequeño manual de usuario.		
Escenario	1. El usuario inicia la aplicación. 2. La aplicación comprueba que el dispositivo se encuentra conectado a la red wifi correspondiente. 3. Se muestra al usuario la pantalla principal 4. El usuario selecciona la opción de información. 4. Se muestra la pantalla con el manual de uso del vehículo.		
Condiciones de fallo	Imposibilidad de mostrar la información. Se muestra un mensaje de error y se recomienda al usuario reiniciar la aplicación.		

Tabla 5. Caso de uso CU-04

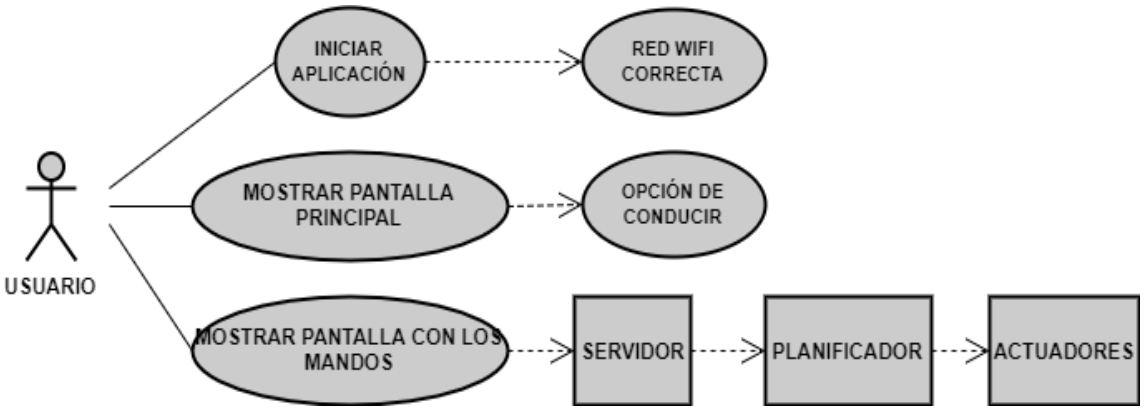

Ilustración 9. Caso de uso CU-05

Identificador	CU-05	**Nombre**	Control del vehículo
Actores	Usuario		
Objetivo	El usuario controla el vehículo remotamente a través de los botones de la pantalla y los sensores del dispositivo móvil.		
Escenario	1. El usuario inicia la aplicación. 2. La aplicación comprueba que el dispositivo se encuentra conectado a la red wifi correspondiente. 3. Se muestra al usuario la pantalla principal 4. El usuario selecciona la opción de conducir. 5. Se muestra la pantalla con los mandos y se comienza el envío de acciones a través de la red wifi para que sean interpretadas y ejecutadas en el vehículo.		
Condiciones de fallo	- Desconexión de la red wifi. Se mostrará la pantalla de error de conexión wifi definida en CU-01. - Error en la recepción de información en la parte servidor. Requerirá el reinicio del vehículo. - Error en el cierre del socket de comunicación al salir de la actividad de conducción. Se cerrará automáticamente al salir.		

Tabla 6. Caso de uso CU-04

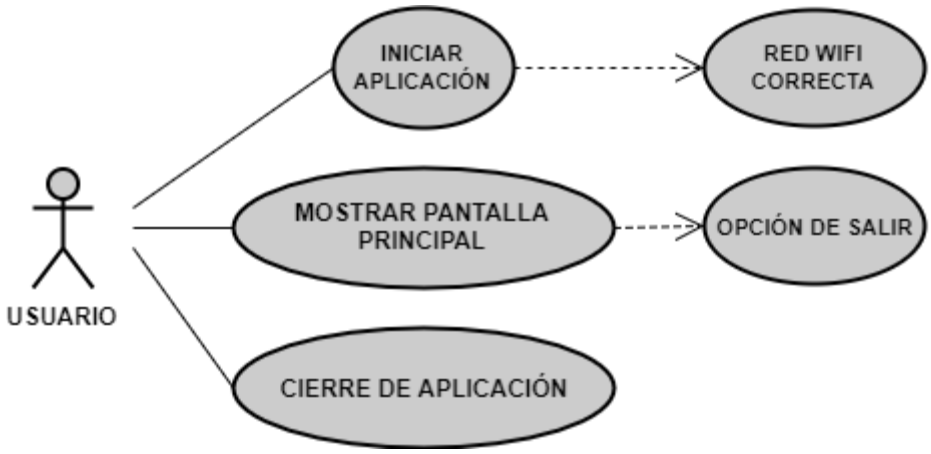

Ilustración 10. Caso de uso CU-06

Identificador	CU-06	**Nombre**	Cierre de la aplicación
Actores	Usuario		
Objetivo	El usuario sale de la aplicación.		
Escenario	1. El usuario inicia la aplicación. 2. La aplicación comprueba que el dispositivo se encuentra conectado a la red wifi correspondiente. 3. Se muestra al usuario la pantalla principal 4. El usuario selecciona la opción de salir. 5. Se sale de la aplicación.		
Condiciones de fallo	Error en la terminación de algún subproceso de la aplicación. Se cierran automáticamente todos los procesos relativos a la aplicación.		

Tabla 7. Caso de uso CU-06

Ilustración 11. Caso de uso CU-07

Identificador	CU-07	**Nombre**	Recogida de información del entorno
Actores	Sensores		
Objetivo	Los sensores recogen información del entorno para que el planificador guie la actividad de los actuadores en función de dicha información.		
Escenario	1. Los sensores recogen información del entorno. 2. El planificador utiliza la información de los sensores para activar o desactivar los actuadores.		
Condiciones de fallo	Error en la información recogida por los sensores. Requerirá un estudio del sensor de forma independiente y en caso de estar defectuoso será necesario sustituirlo.		

Tabla 8. Caso de uso CU-07

3.3. Definición de requisitos de usuario

Esta tarea consiste en la obtención de la información sobre qué es lo que necesita hacer el sistema y las restricciones que se imponen. Para ello se procede a describir y clasificar los requisitos de usuario a través de tablas con el siguiente formato:

Identificador	
Nombre	
Descripción	
Requisitos relacionados	

Tabla 9. Formato tabla especificación de requisitos de usuario

Los campos que forman la tabla de especificación de requisitos de usuario se definen a continuación:

- **Identificador**: Código único identificativo del requisito de usuario. El formato que sigue el identificador es **RX-YY**, siendo:
    - **R**: El identificador de requisito.
    - **X**: El indicador del grupo al que pertenece el requisito que puede ser **C** para los requisitos de capacidad, que describen las operaciones que necesitan realizar los usuarios sobre el sistema, y **R** para los requisitos de restricción, que imponen restricciones sobre el sistema.
    - **YY**: El número de la secuencia de requisitos.
- **Nombre**: Título del requisito.
- **Descripción**: Explicación detallada y completa del requisito.
- **Requisitos relacionados**: Identificadores de los requisitos que tienen alguna relación con el requisito descrito. Si no existe ninguna relación quedara vacío.

3.3.1. Requisitos de capacidad

Identificador	RC-01
Nombre	Conexión
Descripción	El usuario deberá poder configurar y conectar su dispositivo móvil a la misma red wifi que el vehículo.
Requisitos relacionados	

Tabla 10. Requisito RC-01

Identificador	RC-02
Nombre	Información
Descripción	El usuario deberá poder visualizar la información de cómo controlar el vehículo remotamente.
Requisitos relacionados	

Tabla 11. Requisito RC-02

Identificador	RC-03
Nombre	Avance/retroceso
Descripción	El usuario deberá poder hacer avanzar y retroceder al vehículo.
Requisitos relacionados	

Tabla 12. Requisito RC-03

Identificador	RC-04
Nombre	Giro
Descripción	El usuario deberá poder hacer girar a izquierda y derecha al vehículo.
Requisitos relacionados	

Tabla 13. Requisito RC-04

Identificador	RC-05
Nombre	Luces
Descripción	El usuario deberá poder activar/desactivar las luces del vehículo.
Requisitos relacionados	

Tabla 14. Requisito RC-05

Identificador	RC-06
Nombre	Bocina
Descripción	El usuario deberá poder accionar la bocina del vehículo.
Requisitos relacionados	

Tabla 15. Requisito RC-06

Identificador	RC-07
Nombre	Control de distancia
Descripción	El vehículo deberá poder comprobar si tiene obstáculos enfrente.
Requisitos relacionados	

Tabla 16. Requisito RC-07

Identificador	RC-08
Nombre	Control de luz
Descripción	El vehículo deberá poder comprobar la luminosidad existente.
Requisitos relacionados	

Tabla 17. Requisito RC-08

3.3.2. Requisitos de restricción

Identificador	RR-01
Nombre	Conexión
Descripción	La aplicación de usuario y el vehículo solo se deberán conectar a través de una red wifi.
Requisitos relacionados	

Tabla 18. Requisito RR-01

Identificador	RR-02
Nombre	Idioma
Descripción	El idioma utilizado en la aplicación deberá ser el castellano.
Requisitos relacionados	

Tabla 19. Requisito RR-02

Identificador	RR-03
Nombre	Interfaz
Descripción	La navegación a través de la interfaz de la aplicación deberá ser sencilla e intuitiva para el usuario.
Requisitos relacionados	RC-02, RC-03, RC-04, RC-05, RC-06, RR-02

Tabla 20. Requisito RR-03

Identificador	RR-04
Nombre	Compatibilidad
Descripción	La aplicación deberá ser desarrollada para dispositivos Android.
Requisitos relacionados	

Tabla 21. Requisito RR-04

3.4. Definición de requisitos de software

En este apartado se definirán los requisitos software que describen qué debe realizar el sistema a desarrollar. Estos requisitos se obtienen del análisis de los casos de usos (ver 3.2. Análisis de los casos de uso) y de los requisitos de usuario (ver 3.3. Definición de requisitos de usuario). Los requisitos de software se subdividen en dos grupos:

- **Requisitos funcionales:** Requisitos encargados de definir las funcionalidades y propósitos que el software y sus componentes deben cumplir, abarcando cálculos o tratamiento de datos. Para este proyecto, estos requisitos se subdividirán en tres categorías: Aplicación, Hardware y Comunicación.
- **Requisitos no funcionales:** Requisitos que especifican criterios que pueden usarse para juzgar la operación de un sistema, es decir, no definen características de funcionamiento sino aspectos como el rendimiento, la interoperabilidad o la seguridad [21]. De esta forma se han subdivido los requisitos no funcionales de este proyecto en tres categorías: Seguridad, Operación e Interfaz.

Los requisitos de software se definirán y describirán de forma clara, concisa y verificable a través de tablas con el siguiente formato:

Identificador		**Categoría**	
Nombre			
Prioridad	□ Alta	□ Media	□ Baja
Riesgo	□ Alta	□ Media	□ Baja
Verificabilidad	□ Alta	□ Media	□ Baja
Descripción			
Requisitos relacionados			

Tabla 22. Formato tabla especificación de requisitos de software

Los campos de la tabla son:

- **Identificador**: Código único identificativo del requisito de software. El formato que sigue el identificador es **RX-YY**, siendo:
  - **R**: El identificador de requisito.

    - **X**: El indicador del tipo de requisito que puede ser **F** para los Requisitos Funcionales que describen las operaciones a realizar por el sistema, y **NF** para los Requisitos No Funcionales, que indican como realizar las operaciones.
    - **YY**: El número de la secuencia de requisitos.
- **Categoría**: Categoría a la corresponde el requisito.
- **Nombre**: Breve título resumen del requisito.
- **Prioridad**: Identifica el nivel de atención y necesidad del requisito. Hay tres niveles de prioridad:
    - **Alta**: El requisito debe ser satisfecho antes que otros de menor prioridad.
    - **Media**: El requisito debe ser satisfecho después de los de mayor prioridad.
    - **Baja**: El requisito será satisfecho después que el resto de mayor prioridad.
- **Riesgo**: Identifica la relación entre la dificultad y la necesidad del requisito. Sus niveles son los mismos que para la prioridad.
- **Verificabilidad**: Identifica la facilidad de comprobación del cumplimiento del requisito posteriormente. Los niveles de los que dispone son los mismos que para la prioridad.
- **Descripción**: Explicación clara, detallada y completa del requisito.
- **Requisitos relacionados**: Identificadores de los requisitos que tienen alguna relación con el requisito descrito. Si no existe ninguna relación quedara vacío.

3.4.1. Requisitos Funcionales

Identificador	RF-01	**Categoría**	Comunicación
Nombre	Conexión		
Prioridad	☑ Alta	□ Media	□ Baja
Riesgo	□ Alta	□ Media	☑ Baja
Verificabilidad	☑ Alta	□ Media	□ Baja
Descripción	El sistema se comunicará a través de la misma red wifi.		
Requisitos relacionados	RC-01, RR-01, RF-02, RNF-01, RNF-02		

Tabla 23. Requisito RF-01

Identificador	RF-02	**Categoría**	Aplicación
Nombre	Configuración		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	Los usuarios podrán acceder a la configuración de la red wifi del dispositivo desde la aplicación del sistema.		
Requisitos relacionados	RC-01, RR-01, RF-01, RNF-08		

Tabla 24. Requisito RF-02

Identificador	RF-03	**Categoría**	Aplicación
Nombre	Información		
Prioridad	☐ Alta	☐ Media	☑ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema permitirá visualizar la información de control del vehículo a escala controlado remotamente.		
Requisitos relacionados	RC-02, RNF-09		

Tabla 25. Requisito RF-03

Identificador	RF-04	**Categoría**	Aplicación/Hardware
Nombre	Avanzar		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema permitirá al usuario hacer avanzar el vehículo.		
Requisitos relacionados	RC-03, RF-06, RF-07, RF-08, RF-12, RF-17, RNF-10		

Tabla 26. Requisito RF-04

Identificador	RF-05	**Categoría**	Aplicación/Hardware
Nombre	Retroceder		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema permitirá al usuario hacer retroceder el vehículo.		
Requisitos relacionados	RC-03, RF-06, RF-07, RF-08, RF-12, RNF-10		

Tabla 27. Requisito RF-05

Identificador	RF-06	**Categoría**	Aplicación/Hardware
Nombre	Avanzar y retroceder simultaneo		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema dejará inmóvil el vehículo cuando el usuario intente ejecutar simultáneamente las acciones de avanzar y retroceder.		
Requisitos relacionados	RF-04, RF-05, RF-12, RNF-10		

Tabla 28. Requisito RF-06

Identificador	RF-07	**Categoría**	Aplicación/Hardware
Nombre	Giro a izquierda		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema permitirá al usuario hacer girar al vehículo hacía la izquierda tanto al avanzar como al retroceder.		
Requisitos relacionados	RC-04, RF-04, RF-05, RF-12, RNF-11		

Tabla 29. Requisito RF-07

Identificador	RF-08	**Categoría**	Aplicación/Hardware
Nombre	Giro a derecha		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema permitirá al usuario hacer girar al vehículo hacía la derecha tanto al avanzar como al retroceder.		
Requisitos relacionados	RC-04, RF-04, RF-05, RF-12, RNF-11		

Tabla 30. Requisito RF-08

Identificador	RF-09	**Categoría**	Aplicación/Hardware
Nombre	Encendido de luces		
Prioridad	☐ Alta	☑ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema permitirá al usuario encender y mantener encendidas las luces del vehículo.		
Requisitos relacionados	RC-05, RF-10, RF-12, RF-18, RNF-10		

Tabla 31. Requisito RF-09

Identificador	RF-10	**Categoría**	Aplicación/Hardware
Nombre	Apagado de luces		
Prioridad	☐ Alta	☑ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema permitirá al usuario apagar y mantener apagadas las luces del vehículo.		
Requisitos relacionados	RC-05, RF-9, RF-12, RF-18, RNF-10		

Tabla 32. Requisito RF-10

Identificador	RF-11	**Categoría**	Aplicación/Hardware
Nombre	Bocina		
Prioridad	□ Alta	☑ Media	□ Baja
Riesgo	□ Alta	□ Media	☑ Baja
Verificabilidad	☑ Alta	□ Media	□ Baja
Descripción	El sistema permitirá al usuario accionar la bocina del vehículo.		
Requisitos relacionados	RC-06, RF-12, RNF-10		

Tabla 33. Requisito RF-11

Identificador	RF-12	**Categoría**	Aplicación
Nombre	Formato mensajes de acciones		
Prioridad	☑ Alta	□ Media	□ Baja
Riesgo	□ Alta	☑ Media	□ Baja
Verificabilidad	☑ Alta	□ Media	□ Baja
Descripción	El sistema realizará el envío de acciones siguiendo el formato: g=gas&b=brake&d=turn&l=lights&s=sound& siendo los campos: - Gas: Activación de avance. El rango es [0-1]. - Brake: Activación de retroceso. El rango es [0-1]. - Turn: Giro. El rango es [1-8] en función del mapeado y la dirección i (izquierda) y d (derecha). - Lights: Activación de luces. El rango es [0-1]. - Sound: Activación de bocina. El rango es [0-1].		
Requisitos relacionados	RF-04, RF-05, RF-06, RF-07, RF-08, RF-09, RF-10, RF-11, RF-15, RNF-10, RNF-12		

Tabla 34. Requisito RF-12

Identificador	RF-13	**Categoría**	Aplicación
Nombre	Salir de la aplicación		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema permitirá al usuario salir en cualquier momento de la aplicación cerrando todos los procesos activos.		
Requisitos relacionados			

Tabla 35. Requisito RF-13

Identificador	RF-14	**Categoría**	Hardware
Nombre	Configuración servidor		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☑ Media	☐ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema realizará la configuración del servidor cada vez que se inicie el hardware del vehículo.		
Requisitos relacionados	RNF-01, RNF-02, RNF-03		

Tabla 36. Requisito RF-14

Identificador	RF-15	**Categoría**	Hardware
Nombre	Lectura de acciones		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema realizará la lectura de acciones enviadas por la aplicación atendiendo al formato.		
Requisitos relacionados	RF-12, RNF-12		

Tabla 37. Requisito RF-15

Identificador	RF-16	**Categoría**	Hardware
Nombre	Ejecución de acciones		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema ejecutará las acciones leídas a través de un planificador.		
Requisitos relacionados	RF-15, RF-17, RF-18		

Tabla 38. Requisito RF-16

Identificador	RF-17	**Categoría**	Hardware
Nombre	Parada de seguridad		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema cancelará el avance del vehículo cuando detecte un obstáculo frontal a una distancia menor o igual a 15 cm.		
Requisitos relacionados	RC-07, RF-04, RF-16		

Tabla 39. Requisito RF-17

Identificador	RF-18	**Categoría**	Hardware
Nombre	Luces de seguridad		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema encenderá (si están apagadas) automáticamente las luces del vehículo cuando se detecte una luminosidad.		
Requisitos relacionados	RC-08, RF-09, RF-10, RF-16		

Tabla 40. Requisito RF-18

3.4.2. Requisitos No Funcionales

Identificador	RNF-01	**Categoría**	Seguridad
Nombre	Conexión		
Prioridad	☐ Alta	☑ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema se comunicará dentro de la misma red wifi cuyo SSID será “AndruinoCar”, su contraseña “tfgcarlos” y su dirección MAC será “18:fe:35:98:d3:7b”.		
Requisitos relacionados	RR-01, RF-01, RF-14, RNF-02		

Tabla 41. Requisito RNF-01

Identificador	RNF-02	**Categoría**	Seguridad
Nombre	Red		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	La red wifi utilizada utilizará el método de encriptación WPA WPA2 PSK.		
Requisitos relacionados	RF-01, RF-14, RNF-01		

Tabla 42. Requisito RNF-02

Identificador	RNF-03	**Categoría**	Operación
Nombre	Servidor		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☑ Media	☐ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El servidor tendrá la dirección IP del punto de acceso 192.168.4.1 y el puerto de servicio será 1313, permitiendo una única conexión.		
Requisitos relacionados	RF-14		

Tabla 43. Requisito RNF-03

Identificador	RNF-04	**Categoría**	Operación
Nombre	Alimentación		
Prioridad	☐ Alta	☑ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	La alimentación del vehículo se realizará a través de una batería portátil.		
Requisitos relacionados			

Tabla 44. Requisito RNF-04

Identificador	RNF-05	**Categoría**	Interfaz
Nombre	Sistema operativo		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	La aplicación del sistema será desarrollada para dispositivos Android.		
Requisitos relacionados	RR-04		

Tabla 45. Requisito RNF-05

Identificador	RNF-06	**Categoría**	Interfaz
Nombre	Idioma interfaz		
Prioridad	☐ Alta	☐ Media	☑ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	La aplicación del sistema tendrá como único idioma el castellano.		
Requisitos relacionados	RR-02		

Tabla 46. Requisito RNF-06

Identificador	RNF-07	**Categoría**	Interfaz
Nombre	Interfaz de inicio		
Prioridad	☐ Alta	☐ Media	☑ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	Al iniciar la aplicación del sistema, se mostrará una pantalla vertical de inicio de 3 segundos que indique que se está comprobando la conexión wifi.		
Requisitos relacionados	RR-03, RNF-01, RNF-08, RNF-09		

Tabla 47. Requisito RNF-07

Identificador	RNF-08	**Categoría**	Interfaz
Nombre	Interfaz de error de conexión		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	En caso de error de conexión, la aplicación del sistema mostrará una interfaz vertical de error que permita salir de la aplicación o acceder a los ajustes de redes wifi de Android.		
Requisitos relacionados	RR-03, RF-02, RNF-07		

Tabla 48. Requisito RNF-08

Identificador	RNF-09	**Categoría**	Interfaz
Nombre	Interfaz de pantalla principal		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	Si la conexión wifi es correcta, el sistema dispondrá de una interfaz principal vertical con 3 botones: acceder a los mandos de conducción, acceder a la información de uso y salir de la app.		
Requisitos relacionados	RR-03, RF-03, RNF-07, RNF-10		

Tabla 49. Requisito RNF-09

Identificador	RNF-10	**Categoría**	Interfaz
Nombre	Interfaz de mandos de control		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☐ Media	☑ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	La interfaz de los mandos de control se dispondrá únicamente horizontalmente y estará compuesta por 3 botones que detectan cuando están siendo accionados para avanzar, retroceder o pitar, y otro botón seleccionable para las luces.		
Requisitos relacionados	RR-03, RF-04, RF-05, RF-06, RF-09, RF-10, RF-11, RF-12, RNF-09, RNF-12		

Tabla 50. Requisito RNF-10

Identificador	RNF-11	**Categoría**	Operación
Nombre	Operación de giro		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☑ Alta	☐ Media	☐ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	Desde la interfaz horizontal de los mandos de control, el giro se realizará a través de los sensores de rotación del dispositivo móvil.		
Requisitos relacionados	RR-03, RF-07, RF-08		

Tabla 51. Requisito RNF-11

Identificador	RNF-12	**Categoría**	Operación
Nombre	Envió de acciones		
Prioridad	☑ Alta	☐ Media	☐ Baja
Riesgo	☐ Alta	☑ Media	☐ Baja
Verificabilidad	☑ Alta	☐ Media	☐ Baja
Descripción	El sistema realizará el envío de acciones unidireccionalmente desde la aplicación al vehículo cada 100 milisegundos, siguiendo un protocolo similar a UDP.		
Requisitos relacionados	RF-12, RF-15, RNF-10		

Tabla 52. Requisito RNF-12

3.5. Matriz de trazabilidad

Para finalizar el análisis del sistema se procede a comprobar si existe inconsistencia, ambigüedad, duplicidad o escasez de información a través del análisis de las funcionalidades y relaciones establecidas en los requisitos de usuario, los casos de uso y los requisitos de software definidos en los apartados anteriores.

Para comprobar si el análisis se ha realizado correctamente se van a construir dos matrices de trazabilidad. La primera matriz contendrá las relaciones entre los requisitos de software funcionales y los no funcionales. La segunda matriz mostrará las relaciones entre los requisitos de software y los requisitos de usuario/casos de uso. Estas relaciones permiten comprobar que los requisitos definidos son completos, consistentes y válidos, y definen todas las funcionalidades establecidas para el sistema.

La siguiente matriz de trazabilidad representa las relaciones entre los requisitos de software funcionales y los no funcionales:

	RF-01	RF-02	RF-03	RF-04	RF-05	RF-06	RF-07	RF-08	RF-09	RF-10	RF-11	RF-12	RF-13	RF-14	RF-15	RF-16	RF-17	RF-18	RNF-01	RNF-02	RNF-03	RNF-04	RNF-05	RNF-06	RNF-07	RNF-08	RNF-09	RNF-10	RNF-11	RNF-12
RF-01		X																	X	X										
RF-02	X																									X				
RF-03																											X			
RF-04						X	X	X				X					X											X		
RF-05						X	X	X				X																X		
RF-06				X	X							X																X		
RF-07				X	X							X																	X	
RF-08				X	X							X																	X	
RF-09										X		X						X										X		
RF-10									X			X						X										X		
RF-11												X																X		
RF-12				X	X	X	X	X	X	X	X				X													X		X
RF-13																														
RF-14																			X	X	X									
RF-15												X																		X
RF-16															X		X	X												
RF-17				X												X														
RF-18									X	X						X														
RNF-01	X													X						X										
RNF-02	X													X					X											
RNF-03														X																
RNF-04																														
RNF-05																														
RNF-06																														
RNF-07																			X							X	X			
RNF-08		X																							X					
RNF-09			X																						X			X		
RNF-10				X	X	X			X	X	X	X															X			X
RNF-11							X	X																						
RNF-12												X			X													X		

Tabla 53. Matriz de trazabilidad entre requisitos funcionales y no funcionales

A continuación, se muestra la segunda matriz de trazabilidad que relaciona los requisitos de software con los requisitos de usuario y los casos de uso:

	RC-01	RC-02	RC-03	RC-04	RC-05	RC-06	RC-07	RC-08	RR-01	RR-02	RR-03	RR-04	CU-01	CU-02	CU-03	CU-04	CU-05	CU-06	CU-07
RF-01	X								X				X	X	X	X	X	X	
RF-02	X								X					X					
RF-03		X														X			
RF-04			X														X		X
RF-05			X														X		
RF-06																	X		
RF-07				X													X		
RF-08				X													X		
RF-09					X												X		X
RF-10					X												X		X
RF-11						X											X		
RF-12																	X		
RF-13															X			X	
RF-14																	X		
RF-15																	X		
RF-16																	X		X
RF-17							X										X		X
RF-18								X									X		X
RNF-01									X				X	X	X		X	X	
RNF-02													X	X	X		X	X	
RNF-03																	X		
RNF-04																	X		X
RNF-05												X	X	X	X	X	X	X	
RNF-06										X			X	X	X	X	X	X	
RNF-07											X		X	X	X	X	X	X	
RNF-08											X		X	X	X				
RNF-09											X			X		X	X	X	
RNF-10											X						X		
RNF-11											X						X		
RNF-12																	X		

Tabla 54. Matriz de trazabilidad entre requisitos de software y requisitos de usuario/casos de uso

Como se puede observar en esta segunda matriz, los requisitos de software completan la funcionalidad del sistema ya que todos los requisitos de usuario y todos los casos de uso tienen al menos un requisito relacionado, de la misma forma que todos los requisitos tienen al menos un requisito de usuario o caso de uso asociado.

4. Diseño del sistema

En este apartado se realizará el diseño cuyo objetivo será resolver el problema planteado y modelado en el punto 3. Análisis del sistema. El proceso que se seguirá es el indicado en Métrica Versión 3 [8].

Para realizar el diseño del sistema se plantearán completa y detalladamente la arquitectura del sistema especificando los componentes del mismo, las clases en que se dividirá y lo implementaran, el diseño de los datos en la comunicación, las interfaces de usuario de la aplicación del sistema y el entorno de desarrollo utilizado.

4.1. Diseño de la arquitectura del sistema

En este apartado se describe la arquitectura seleccionada para alcanzar los objetivos del sistema, es decir, que satisfaga los requisitos. Por ello la arquitectura más adecuada para este sistema es el modelo Cliente-Servidor, ya que define una organización donde los procesos funcionan en máquinas separadas, es decir, el proceso Cliente (la aplicación Android del sistema) consume los servicios de los que provee el proceso Servidor (el vehículo) interactuando por un mecanismo de paso de mensajes preestablecido que se denomina Protocolo [22].

El Cliente es el subsistema que dispone del proceso encargado de reclamar los servicios del Servidor a través de peticiones. El Cliente suele manejar las funciones relacionadas con la manipulación y despliegue de datos, es decir, el procesado de la lógica de la aplicación y el formateado de los datos. Se suele desarrollar para plataformas con interfaz de usuario (GUI) con el objetivo de interactuar con el usuario [23].

El Servidor es el subsistema que contiene el proceso que proporciona los servicios a los Clientes en función de las peticiones realizadas y responde devolviendo a los Clientes el resultado obtenido. El Servidor normalmente realiza las funciones relacionadas con la lógica de negocio. La ejecución del Servidor debe comenzar antes que la del Cliente para poder atender a las interacciones solicitadas [23].

El paso de mensajes [22] es el mecanismo necesario para el soporte de interacciones que permite que un Cliente obtenga un servicio de un Servidor. La utilización de un protocolo de paso de mensajes permite conectar Clientes y Servidores implementados en diferentes plataformas y con distinto lenguaje de programación.

Ilustración 12. Modelo Cliente-Servidor [24]

4.1.1. Cliente

Como se ha explicado anteriormente este subsistema será el medio que utilice el usuario para conectarse con el Servidor. Su funcionalidad principal podría resumirse en la recogida de las acciones que quiere realizar el usuario, procesarlas y enviárselas a través de peticiones al Servidor. En el sistema será la aplicación para dispositivos Android y tendrá la siguiente estructura de componentes definida a partir de las actividades y funciones que deberá realizar:

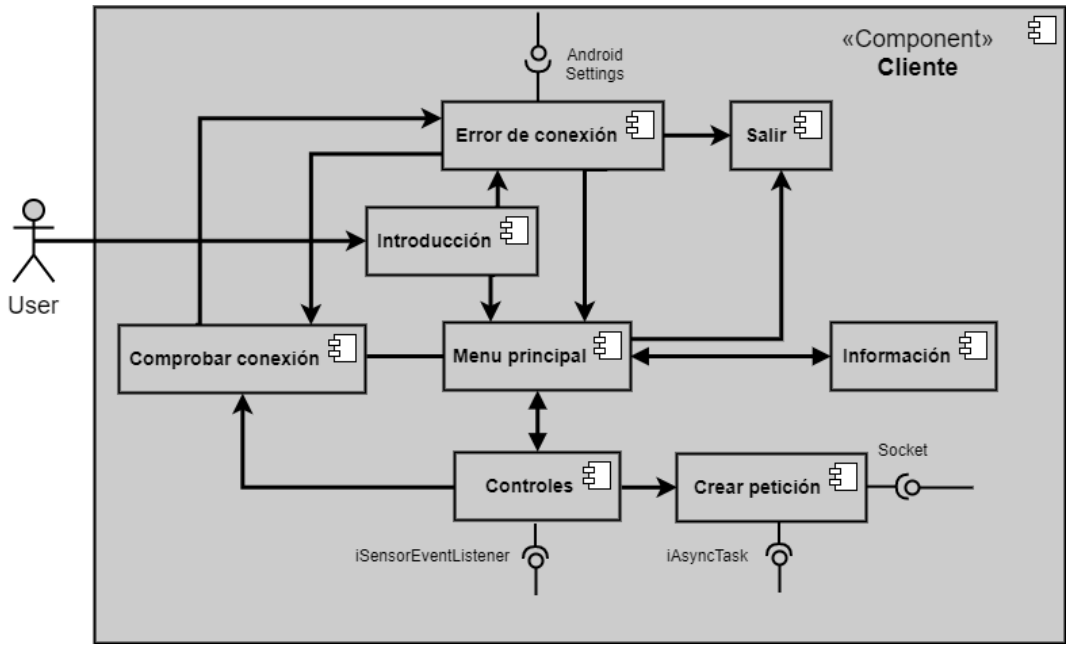

Ilustración 13. Cliente

En la ilustración se pueden observar los componentes pertenecientes a las actividades de la aplicación Android que se explican a continuación:

- **Introducción:** Este componente es la primera actividad Android que se realiza cuando el usuario inicia la aplicación. En ella se muestra una pantalla de inicio que indica que se está comprobando la conexión a la red "AndruinoCar". La funcionalidad consiste en acceder a través de los permisos y API de Android a la configuración wifi para verificar si el dispositivo se encuentra conectado a una red wifi y si el SSID y la dirección MAC de la red coinciden con la red del vehículo. Una vez comprobado, espera 3 segundos y en función del resultado de la comprobación accederá a la actividad de error de conexión si el resultado es negativo o al menú principal si la red es correcta.
- **Error de conexión:** Este componente será una actividad Android que indique que el dispositivo no tiene activado la conexión wifi o la red a la que se encuentra conectado no es la correcta. Desde este punto se ofrecerá la opción de acceder a la configuración wifi utilizando la API de Android o la opción de salir de la aplicación.
- **Salir:** Este componente permite al usuario salir de la aplicación cerrando todos los hilos y procesos asociados a la aplicación que se estén ejecutando en segundo plano.
- **Comprobar conexión:** Este componente se ejecutará en un hilo paralelo para comprobar, a través de los permisos y API de Android, la configuración wifi para verificar si el dispositivo se encuentra conectado a una red wifi y si el SSID y la dirección MAC de la red coinciden con la red del vehículo. Esto lo realizará continuamente. Mientras no encuentre la red del vehículo seguirá mostrando el error de conexión. Una vez que el dispositivo se encuentre conectado a la red wifi correcta se terminará la ejecución del hilo y se accederá al menú principal. El hilo también se cerrará cuando se salga de la aplicación.
- **Menú principal:** Este componente será otra actividad Android que contendrá las opciones que puede realizar el usuario: conducir, leer la información de control remoto del vehículo o salir de la aplicación.

- **Información**: Este componente será una actividad Android que muestra un breve manual de usuario del control remoto del vehículo explicando aspectos como el encendido previo del hardware o el modo de girar. Desde este punto se puede regresar al menú principal.
- **Controles:** Este componente será la actividad Android que mostrará al usuario los botones para manejar el vehículo remotamente. Este componente también utilizara la interfaz de sensores aportada en la API de Android para detectar la rotación del dispositivo y girar de esta forma el vehículo. Desde este punto se vuelve a crear el hilo de comprobar conexión para asegurar que el dispositivo no pierde ni cambia la red wifi del vehículo. Desde los controles se puede volver al menú principal, cerrándose todos los hilos y procesos en segundo plano.
- **Crear petición:** Este componente será el encargado de crear y realizar de forma asíncrona la petición al servidor con los datos y el formato correcto. De esta forma, cada 100 milisegundos, este componente, mediante la utilización del API de Android, abrirá el socket en la dirección IP y puerto conocido del servidor, escribirá la petición con los últimos datos recogidos de los controles y cerrará el socket. Esta tarea se realizará mientras la actividad de los controles este activa.

La forma de comunicación entre este subsistema Cliente y el subsistema Servidor se realiza mediante la escritura en el socket. Los sockets son mecanismos de comunicación entre procesos que incluso se encuentran en distintas máquinas y desde un punto de vista de programación es un "fichero" que se abre de una manera especial [25].

4.1.2. Servidor

Como se ha explicado anteriormente este subsistema será el encargado de recibir las peticiones del Cliente. Su funcionalidad consistirá en leer las peticiones escritas en el socket de comunicación por el cliente, interpretarlas y actuar en función de las acciones recibidas. En el sistema será el vehículo a escala y su diseño se dividirá en dos puntos: el diseño de la parte hardware y el diseño de la parte hardware que controlará las funciones del hardware.

4.1.2.1. Hardware

En este apartado se explica el diseño realizado para la parte Servidor, es decir, el vehículo a escala. Para ello se describirá la funcionalidad y se mostrará el diseño electrónico de los módulos, elementos activos y elementos pasivos que se utilizarán para que el sistema realice la funcionalidad descrita en el punto 3. Análisis del sistema.

Chasis y estructura del vehículo [5]

Dentro de los robots móviles terrestres existen múltiples posibilidades de diseño de la estructura del sistema con ruedas. La seleccionada para este proyecto es la configuración triciclo. Esta estructura aporta una mayor simplicidad en la construcción al utilizar solo tres ruedas (dos motrices y una de apoyo), pero aporta una menor estabilidad al sistema. Mediante esta configuración no se dispone de un eje de dirección y los giros se realizarán mediante el giro a diferente velocidad de las ruedas motrices. También se deberá tener en cuenta la distribución del peso sobre la plataforma para construir un sistema estable.

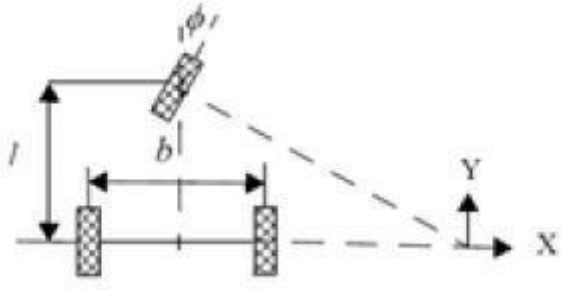

Ilustración 14. Estructura del vehículo

Arduino UNO

Como se explicaba en el punto 2.2.2. Microcontroladores, Arduino UNO es una placa programable con entradas y salidas analógicas y digitales de bajo coste. De esta forma se dispondrá de un pequeño autómata capaz de analizar la información del entorno a través de sensores y realizar acciones mediante actuadores en función de las instrucciones que se le introduzcan en forma de programa que se ejecutará autónomamente. Para garantizar la autonomía del vehículo el controlador se alimentará a través de una batería portátil.

A continuación, se muestran una imagen del componente y del esquema de su circuito electrónico al que se conectarán todos los componentes hardware utilizados:

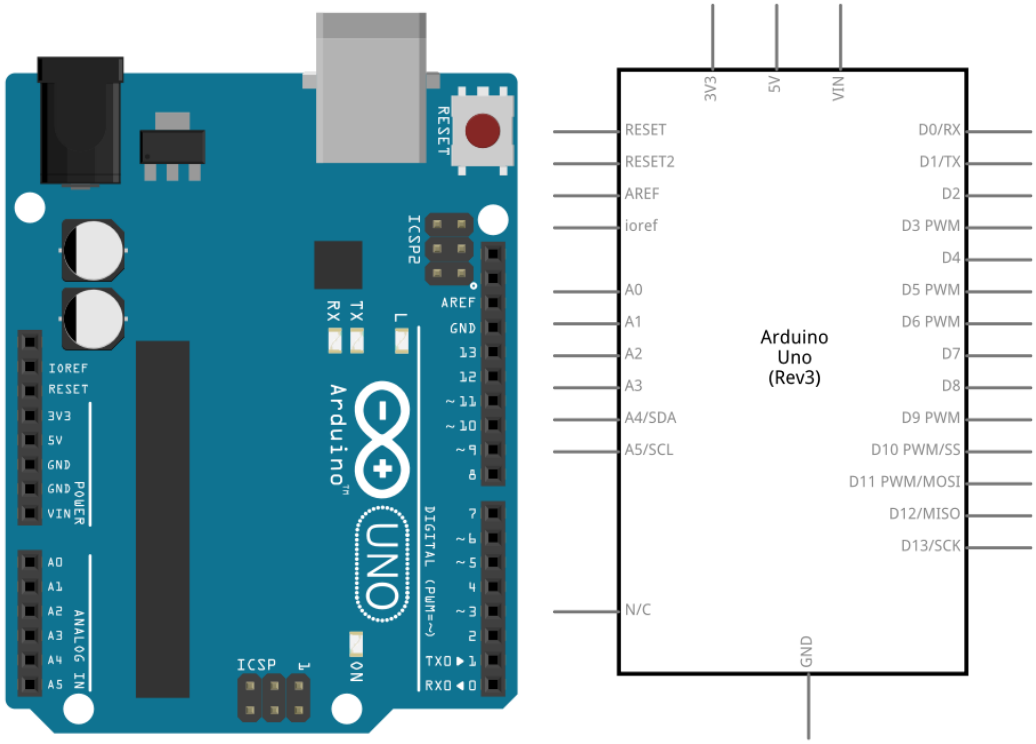

Ilustración 15. Arduino UNO y su circuito electrónico

Módulo wifi ESP8266 [26]

Este módulo es el encargado de proporcionar una red wifi y establecer un servidor sobre el vehículo a escala. Es muy parecido a los módulos Bluetooth comercializados para Arduino e incluye toda la electrónica necesaria para la comunicación por radio frecuencia en la banda wifi, así como la pila TCP/IP y la comunicación a través de un puerto serie. Al igual que los módulos Bluetooth se configura y se trabaja con los comandos AT como se indica en su documentación [27].

El módulo ESP8266 es muy sencillo y está diseñado para implementar dispositivos enmarcados en el paradigma del Internet de las Cosas (IoT), y por eso incluye todo lo necesario para conectarse a un punto de acceso a través de un puerto serie configurable a diferentes velocidades de transferencia medidas en baudios, convertirse en un punto de acceso creando una red o realizar ambas funciones. Una vez configurado se le puede utilizar para recibir información en la dirección IP y puerto que se desee o enviar datos a otro dispositivo con una IP y puertos determinados a través del puerto serie.

La función que nos interesa de este módulo es la de recibir ya que no realizará respuestas como se explica posteriormente en 4.1.3. Comunicación. Cuando se trata de recibir internamente el módulo limpia todo el empaquetado TCP/IP y reenvía por el puerto serie la información limpia, por lo que permite olvidarse de las gestiones de la pila TCP/IP y de las demandas de procesador y memoria que suponen. Por ello no se trata exactamente una conexión wifi ya que no permite el acceso directo a la pila o al socket IP, pero para Arduino es casi una ventaja.

El esquema de conexión se presenta a continuación. Como se puede observar se configurarán los pines 2 y12 para que sean el puerto serie que utilice el módulo. También se puede comprobar que la tensión de trabajo es de 3,3V ante el riesgo de deterioro si se conecta a tensiones de 5V (tensión de trabajo de Arduino) tal y como se indica en su documentación.

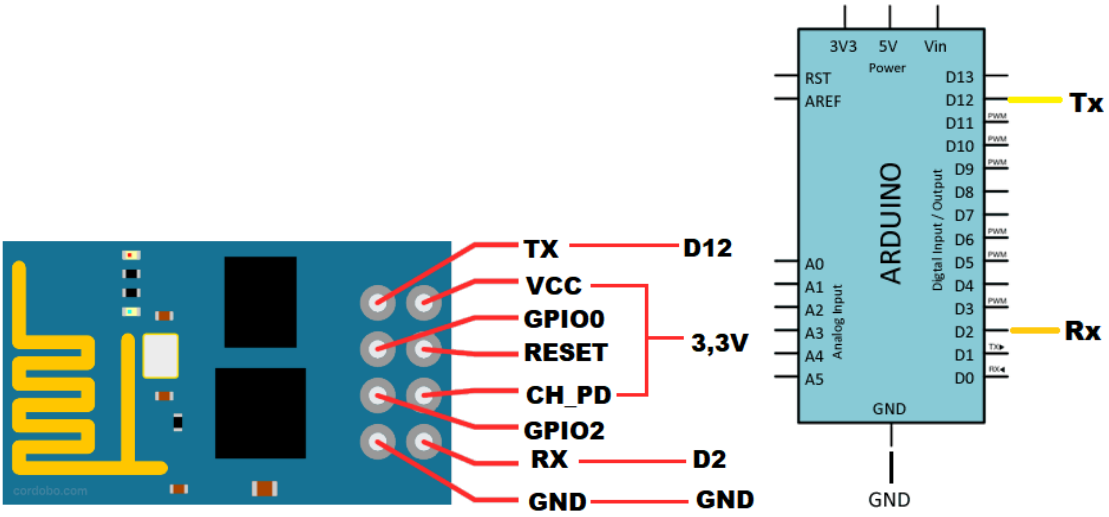

Ilustración 16. Esquema electrónico módulo ESP8266

Una vez conectado se debe realizar la configuración a través de los comandos AT como se indicaba anteriormente y esta configuración se debe realizar cada vez que se inicie ya que los cambios se guardan en su memoria flash y se borran cuando se corta la alimentación del módulo:

- **AT+CIPSTAMAC="18:fe:35:98:d3:7b"**. Este comando establece la dirección MAC del módulo.
- **AT+CWMODE=2**. Este comando establece el módulo como un punto de acceso a la red en lugar de una estación que usa una red.

- **AT+CWSAP=“AndruinoCar”, “tfgcarlos”, 3, 4**. Este comando configura el punto de acceso estableciendo como SSID el nombre AndruinoCar, como contraseña tfgcarlos, como identificador del canal el 3 y una seguridad WPA/WPA PSK.
- **AT+CIPMUX=1**. Este comando permite que múltiples dispositivos puedan conectarse a la red que ofrece este punto de acceso. Este paso es necesario para poder levantar un servidor en el módulo.
- **AT+CIPSERVER=1,1313**. Este comando crea un servidor cuyo puerto será el 1313.
- **AT+CIOBAUD=9600**. Este comando cambia la velocidad de comunicación del módulo de 115200 baudios (por defecto) a 9600 baudios para facilitar la comunicación entre el puerto serie del módulo y el puerto serie de Arduino.

Controlador L298N [28]

El L298N es un controlador (driver) de motores, que permite encender y controlar dos motores de corriente continua desde Arduino, pudiendo invertir la dirección de giro y controlar la velocidad de giro. Puesto que Arduino no dispone de suficiente potencia para mover actuadores, la mejor opción es utilizar drivers como este para que realicen el trabajo.

Básicamente el controlador L298N consiste en dos puentes H, uno para la salida A y otro para la salida B. Un puente H es un componente ampliamente utilizado en electrónica para alimentar una carga de forma que se puede invertir el sentido de la corriente que le atraviesa. Internamente es una formación de 4 transistores, conectados entre VCC y GND, con la carga a alimentar entre ellos. Dibujado en esquema el conjunto tiene forma de “H”, de la que recibe su nombre su nombre.

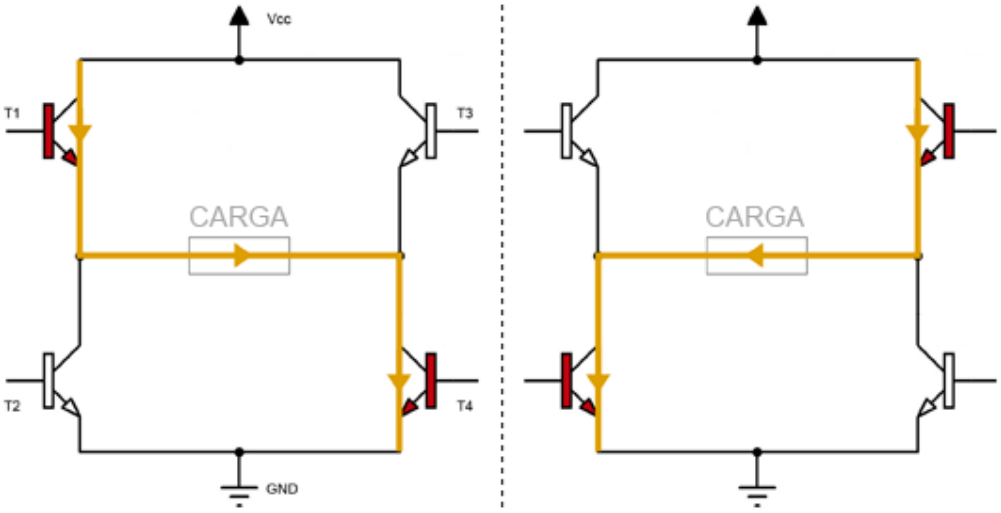

Ilustración 17. Esquema de funcionamiento de un puente H

Activando los transistores opuestos en diagonal de cada rama, se puede variar el sentido de la corriente atraviesa la carga, tal y como se muestra en la imagen. Conectando simultáneamente los transistores superiores o inferiores, se puede poner la carga en VCC o GND respectivamente, configuración que se usará como freno. Por último, nunca se deben encender ambos transistores de un mismo ramal (izquierda o derecha), ya que se producirá un cortocircuito entre VCC y GND.

La placa L298N incorpora electrónica que simplifica la conexión de los dos puentes H que integra, agrupando las conexiones en 3 pines accesibles por cada salida y eliminando la posibilidad de generar un cortocircuito. Dos de estos pines, IN1 y IN2 para la salida A y IN3 y IN4 para la salida B, controlan el encendido de los transistores de cada una de las dos ramas, encendiendo el ramal superior o inferior de la misma. El tercer pin, ENA para la salida A y ENB para la salida B, desactiva simultáneamente todos los transistores del puente-H desconectando la carga por completo, pero también se pueden conectar a una señal PWM para controlar la velocidad de giro.

La corriente máxima que este componente puede suministrar a los motores es, en teoría, 2A por salida (hasta 3A de pico) y una tensión de alimentación entre 3V y 35V. Sin embargo, el controlador L298N tiene una eficiencia baja y su electrónica provoca una caída de tensión de unos 3V. Estas pérdidas se disipan en forma de calor lo que se traduce en que es difícil obtener más de 0.8-1A por fase sin exceder el rango de temperatura de funcionamiento. De esta forma la idea de alimentar todo el sistema a partir de las salidas de 5V y 3,3V de Arduino se tiene que cambiar para añadir una alimentación separada y adicional a los motores.

El L298N incorpora protecciones contra sobre intensidad, sobre temperatura, y diodos de protección contra corrientes inducidas. El controlador L298N es ampliamente usado en proyectos electrónico y robótica, por su sencillez de uso, bajo coste, y buena calidad precio.

Una vez explicado el fundamento y funcionalidad de este componente se procede a mostrar su esquema de conexión para el control de dos motores. Como se ha comentado anteriormente los pines ENA y ENB se deben conectar a salidas con señal PWM [29] para poder controlar la velocidad de giro. También se muestra que se ha tenido que añadir una fuente de alimentación supletoria a pilas, ya que el voltaje lógico de 5V no era suficiente para mover los motores.

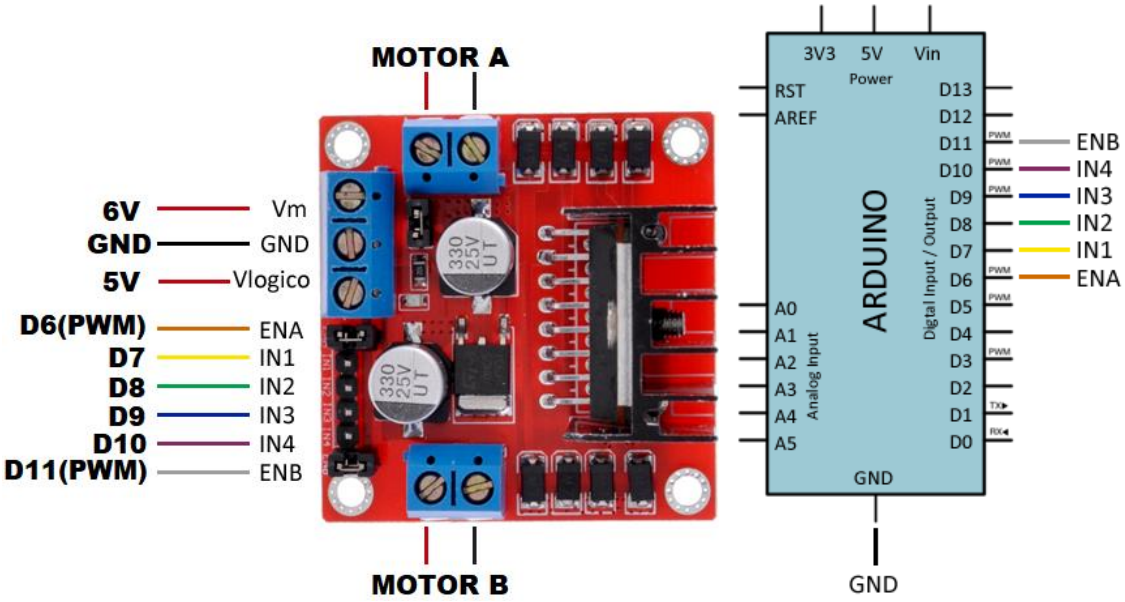

Ilustración 18. Esquema electrónico controlador L298N

De esta forma el control de los motores se realizará siguiendo la tabla:

	Avance	Retroceso	Parada
IN1/IN3	HIGH	LOW	LOW
IN2/IN4	LOW	HIGH	LOW

Tabla 55. Control giro motores

Sensor HC-SR04 [30]

Este elemento se trata de un sensor de ultra sonidos para medir distancias. Su funcionamiento se basa en el envío de un pulso de alta frecuencia, no audible por el ser humano, para que rebote en los objetos cercanos y refleje hacia el micrófono para esa frecuencia que dispone el sensor. De esta forma a partir del tiempo entre pulsos y conociendo la velocidad del sonido (343m/s) se puede estimar la distancia al objeto sobre el que impacto el impulso ultrasonido.

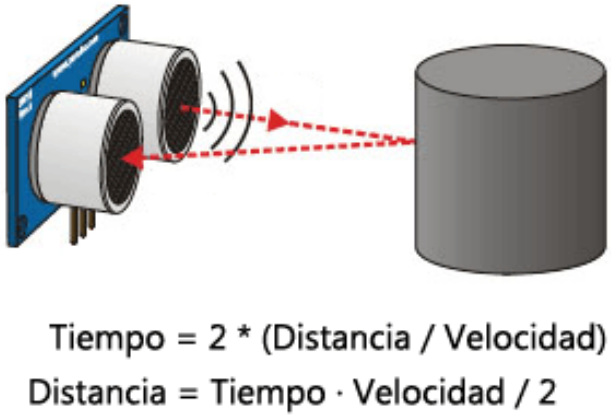

Ilustración 19. Esquema de funcionamiento de un sensor HC-SR04

Los sensores de ultrasonidos son sensores baratos, y sencillos de usar. El rango de medición teórico es de 2 cm a 400 cm, con una resolución de 0.3 cm. En la práctica, sin embargo, el rango es mucho más limitado, entre 20 cm y 200 cm. Los sensores de ultrasonidos son sensores de baja precisión. La orientación de la superficie a medir puede provocar que la onda se refleje, falseando la medición. Además, no resultan adecuados en entornos con gran número de objetos, dado que el sonido rebota en las superficies generando ecos y falsas mediciones. Tampoco son apropiados para el funcionamiento en el exterior y al aire libre. Para una mayor precisión se utilizarán sensores por infrarrojos y sensores ópticos.

El esquema de conexión de este elemento será:

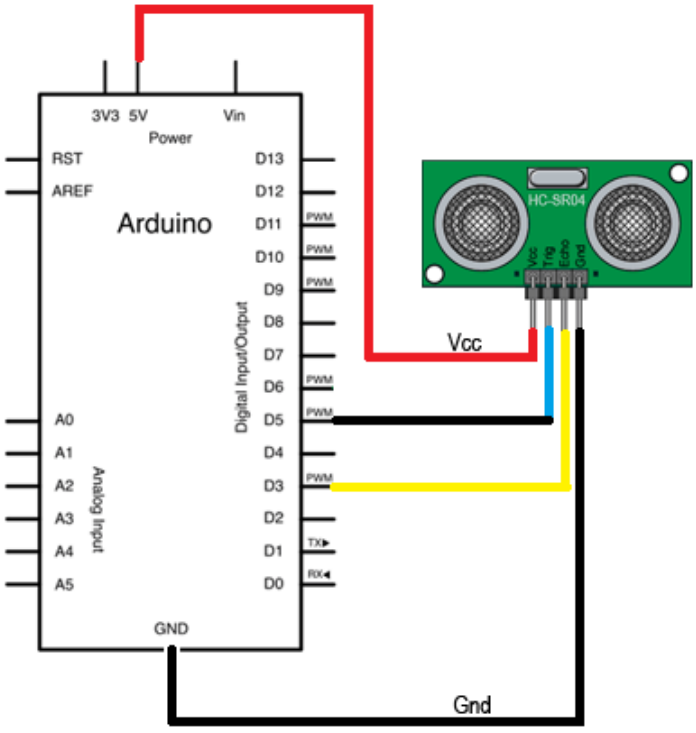

Ilustración 20. Esquema electrónico sensor HC-SR04

Buzzer [31]

Los buzzer, denominados zumbadores, son dispositivos que generan un sonido de una frecuencia determinada y fija cuando son conectados a tensión. La calidad del sonido que producen dista bastante de la alta fidelidad. Pero es suficiente para generar tonos audibles para simular la bocina del vehículo. Su esquema de conexión es el siguiente:

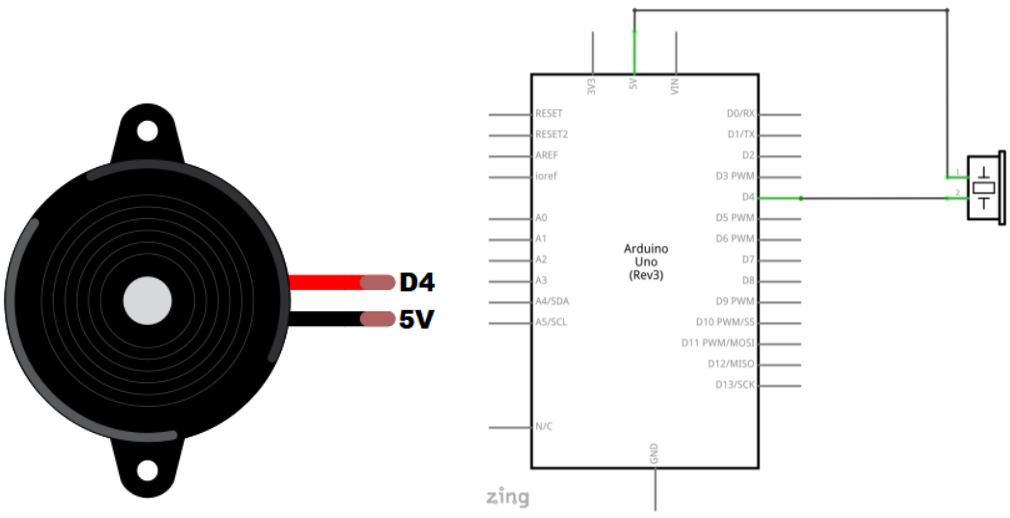

Ilustración 21. Esquema electrónico buzzer

Leds y resistencia LDR [32]

Una resistencia LDR (Light Dependent Resistor), es un dispositivo cuya resistencia varía en función de la luz recibida. Podemos usar esta variación para medir, a través de las entradas analógicas, una estimación del nivel de luz. El rango que devuelve se encuentra entre 0 (máximo) y 1024 (mínimo), por lo que si el nivel de luz que detecta es mayor de 512 se encenderán los leds automáticamente. A continuación, se muestra el esquema de la fotorresistencia y de los leds que simularán las luces del vehículo:

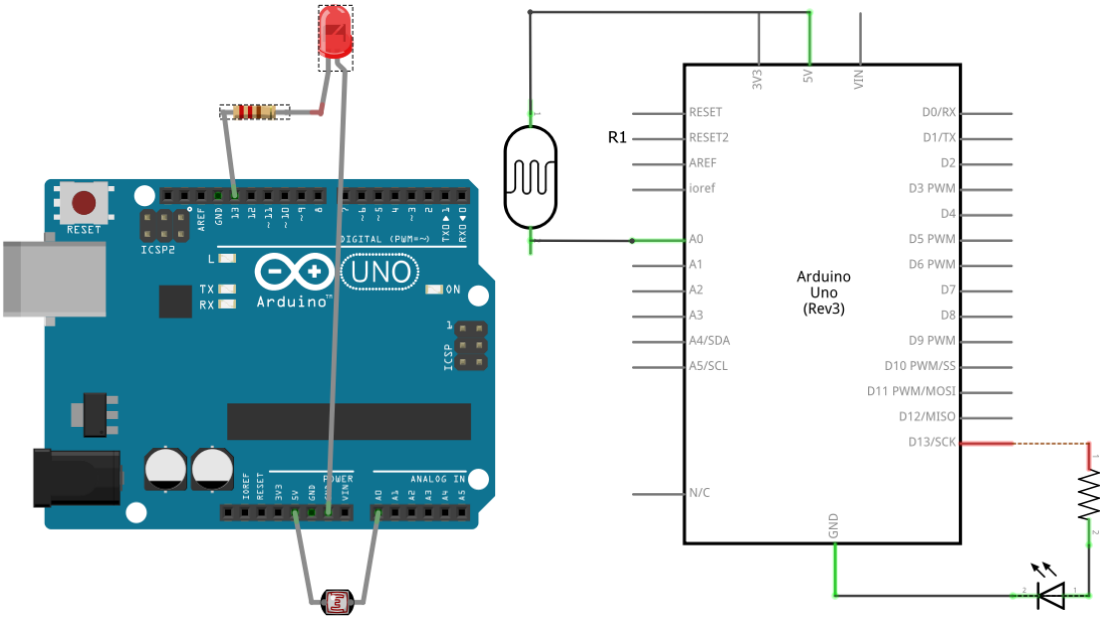

Ilustración 22. Esquema electrónico LDR y leds

4.1.2.2. Software

En este apartado se explica el diseño del programa o skecth que se cargara al controlador Arduino UNO para realizar la parte Servidor. Su funcionalidad podría resumirse en la lectura del socket a través del módulo wifi que pasa su información por el puerto serie creado y conectado a la placa, su formateo para obtener las acciones a realizar y la llamada al planificador para que accione los actuadores o elementos explicados anteriormente. En el Sistema será el vehículo a escala y tendrá la estructura de componentes definida a continuación:

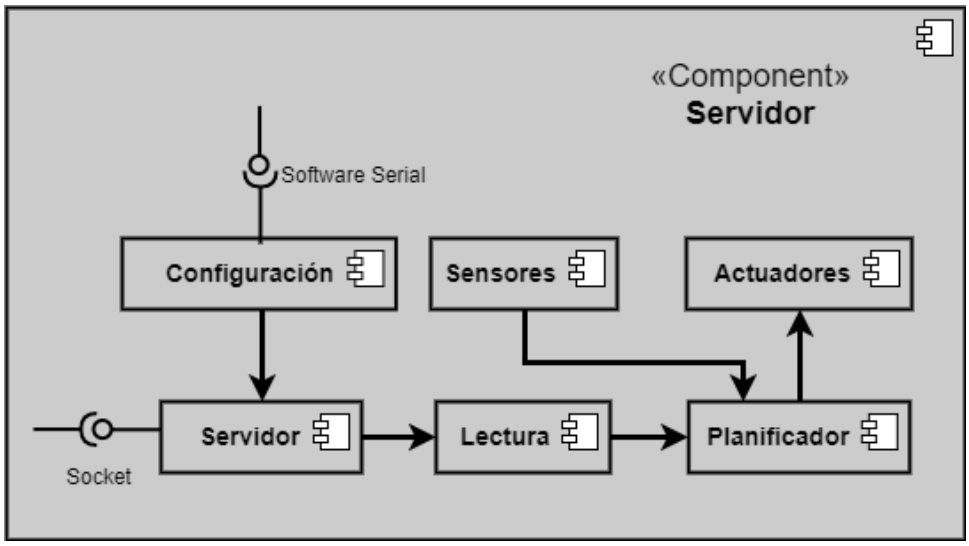

Ilustración 23. Servidor

En la ilustración se pueden observar los componentes pertenecientes a las partes que realizan funciones principales en la parte Servidor para que el vehículo a escala funcione a distancia:

- **Configuración**: Este componente es el encargado de configurar la red wifi y el servidor cada vez que se inicia o reinicia el hardware estableciendo una conexión por puerto serie con el módulo wifi.
- **Servidor**: Este componente es el encargado de recibir la información enviada por el Cliente a través del socket y pasar la información de la petición directamente a la lectura a través del puerto serie configurado.
- **Lectura**: Este componente es el encargado de sacar la información de la petición mediante un analizador de la trama recibida.
- **Planificador**: Es el encargado de distribuir y realizar las llamadas a los actuadores en función de las acciones leídas, dando una mayor prioridad a las acciones de movimiento que al resto. Esta unidad también toma decisiones de ejecución de acciones en función de los datos de los sensores y decidiendo si es posible.
- **Sensores**: Este componente recogerá información del entorno para introducir cierta seguridad, como por ejemplo la detección de obstáculos frontales o el encendido de luces de emergencia cuando la luminosidad del entorno sea baja.
- **Actuadores**: Es el conjunto de componentes que se encargan de accionar o parar los elementos explicados anteriormente.

4.1.3. Comunicación

La comunicación entre el Cliente y el Servidor se realizará a través la red wifi inalámbrica. La forma en que se comunican es a través de un socket, que como se explicaba anteriormente son mecanismos de comunicación entre procesos que incluso se encuentran en distintas máquinas [25] como en este caso. Para realizar la comunicación entre ambas partes se seguirá un protocolo propio basado en los protocolos no orientados a la conexión y los protocolos sin estado.

El protocolo diseñado será no orientado a la conexión, de forma similar a UDP, y la aplicación cliente abrirá y cerrará el socket cada vez que quiera realizar una petición ya que no interesa mantener una conexión abierta entre Cliente y Servidor porque se restaría rendimiento al sistema y lo que prima sobre todo para que la comunicación y el control remoto sea en tiempo real es que ninguna parte se quede bloqueada. De esta forma se pueden recibir peticiones incorrectas o incluso perder peticiones, pero esto no importa ya que la aplicación cliente realizará constantemente cada 100 ms peticiones y el vehículo actuará en función de la última petición que reciba descartando las incorrectas.

Este protocolo también se basa en que el Servidor no mantiene el estado del Cliente ya que como se ha explicado no mantiene una conexión abierta. Por ello cada petición debe ser autocontenida, es decir, debe incluir en todas las peticiones las acciones que se desean realizar ya que el servidor no recordará las acciones que se han realizado en la petición anterior.

También hay que realizar un idioma de comunicación común para que Cliente y Servidor. Esto se refiere a que el formato en que el Cliente escribe la información en el socket debe ser conocido por el Servidor para separar la trama y entender las acciones que se realizarán para atender la petición. Para ello se utilizará el Cliente escribirá en el socket las acciones mediante la cadena g=gas&b=brake&d=turn&l=lights&s=sound& donde:

- Gas: Tomará los valores 0 (Avance desactivado) o 1 (Avance activado).
- Brake: Tomará los valores 0 (Retroceso desactivado) o 1 (Retroceso activado).
- Turn: El valor del giro tomará valores entre 1 y 8 en función del mapeado realizado sobre la rotación del dispositivo móvil mostrado en la siguiente imagen acompañado del indicativo i (izquierda) o d (derecha).

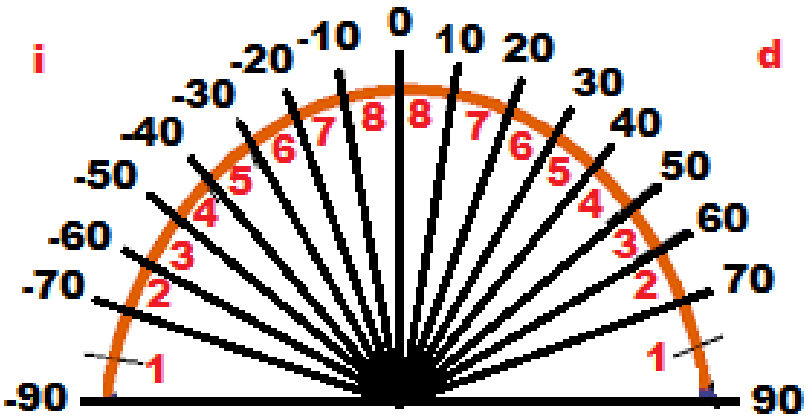

Ilustración 24. Mapeado de la rotación del dispositivo móvil para girar

- Lights: Tomará los valores 0 (Luces desactivadas) o 1 (Luces activadas).
- Sound: Tomará los valores 0 (Bocina desactivada) o 1 (Bocina activada).

4.2. Diseño de clases

En este apartado se va a realizar el diseño de clases del sistema recogiendo la especificación detallada de cada clase indicando sus atributos, métodos y relaciones. Para ello se tendrán en cuenta los escenarios de los casos de uso y la arquitectura del sistema explicada anteriormente.

4.2.1. Diagrama de Clases

El sistema basado en la arquitectura Cliente-Servidor se puede descomponer en las clases relacionadas como se muestra en la siguiente ilustración. Como se puede observar la parte Servidor está formada por una única clase que sería el sketch de Arduino con los pines y el vector de char de lectura como atributos y los métodos propios de Arduino (setup y loop) y los utilizados para tratar las peticiones que el Cliente realiza a través del socket.

La parte Cliente sería el resto de clases que formarán la aplicación del sistema. En ellas se muestran los atributos utilizados y los métodos, que como se puede ver son los propios de Android. En cuanto a lo relativo con la interfaz de la aplicación solo se han incluido los atributos que representan botones.

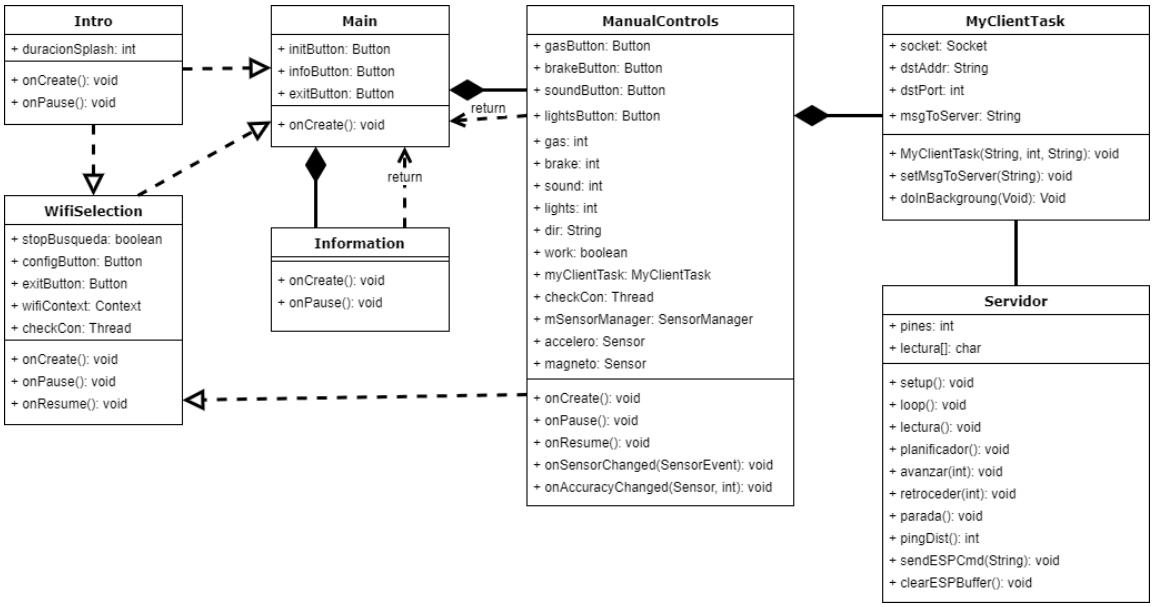

Ilustración 25. Diagrama de clases

4.2.2. Identificación de atributos y métodos

En este apartado se van a especificar con mayor profundidad las clases definidas en el diagrama de clases del sistema junto con sus atributos y métodos. Para ello se van utilizar tablas con el siguiente formato:

CL-XX	
Nombre de la clase	
Descripción	
Atributos	
Métodos	

Tabla 56. Formato tabla descripción de clases

Los campos que constituyen la tabla se definen a continuación:

- **CL-XX**: Código único identificativo de cada una de las clases. Su formato corresponde al identificador de clase **CL** acompañado del número que representa la clase.
- **Nombre de la clase**: Nombre de la clase tal y como aparece en el diagrama de clases.
- **Descripción**: Breve resumen de la funcionalidad y objetivo de la clase.
- **Atributos**: Enumeración de los atributos de la clase.
- **Métodos**: Enumeración de los métodos de la clase.

A continuación, se muestran las clases del sistema con mayor profundidad:

CL-01	
Nombre de la clase	Intro
Descripción	Esta clase muestra una pantalla de inicio de 3 segundos mientras comprueba que la conexión wifi es correcta.
Atributos	- int duracionSplash
Métodos	- void onCreate() - void onPause()

Tabla 57. Clase CL-01

CL-02	
Nombre de la clase	WifiSelection
Descripción	Si la conexión wifi no es correcta, se accederá a esta clase que muestra una pantalla de error que da las opciones de salir de la aplicación o de acceder a la configuración wifi de Android. También ejecuta un hilo en segundo plano que se encuentra comprobando la conexión constantemente.
Atributos	- Button configButton - Button exitButton - Context wifiContext - boolean stopBusqueda - Thread busquedaParalela
Métodos	- void onCreate() - void onPause() - void onResume()

Tabla 58. Clase CL-02

CL-03	
Nombre de la clase	Main
Descripción	Si la conexión wifi es correcta, se accede a esta clase que muestra el menú principal de la aplicación.
Atributos	- Button initButton - Button infoButton - Button exitButton
Métodos	- void onCreate()

Tabla 59. Clase CL-03

CL-04	
Nombre de la clase	Information
Descripción	Esta clase es la relativa al mostrado de la información de utilización del Sistema
Atributos	
Métodos	- void onCreate() - void onPause()

Tabla 60. Clase CL-04

CL-05	
Nombre de la clase	ManualControls
Descripción	Esta clase contiene los mandos del vehículo, tanto los botones como la detección de giro. Será la encargada de estar recogiendo la información de las acciones constantemente y dar el formato de envió. También realiza la instancia de la clase MyClientTask que realiza la escritura en el socket.
Atributos	- Button gasButton, brakeButton, soundButton, lightsButton - int gas, brake, sound, lights - String dir - boolean work - MyClientTask myClientTask - Thread búsquedaParalela - SensorManager mSensorManager - Sensor accelerometer, magnetometer
Métodos	- void onCreate() - void onPause() - void onResume() - void onSensorChanged(SensorEvent event) - void onAccuracyChanged(Sensor s, int a)

Tabla 61. Clase CL-05

CL-06	
Nombre de la clase	MyClientTask
Descripción	Esta clase es la encargada de abrir y cerrar el socket en la dirección IP y puerto del servidor para escribir en él la información más reciente recogida por la clase ManualControls. La tarea de esta clase se realiza en segundo plano.
Atributos	- Socket socket - String dstAddr - int dstPort - String msgToServer
Métodos	- void MyClientTask(String ip, int port, String msg) - void MsgToServer(String newMsg) - void doInBackground()

Tabla 62. Clase CL-06

CL-07	
Nombre de la clase	Servidor
Descripción	Esta clase se corresponde con el sketch de Arduino. Su función es configurar la red wifi y levantar el servidor para que reciba peticiones y las trate adecuadamente para ejecutar las acciones que solicita la aplicación cliente.
Atributos	- int pines (todos los pines utilizados) - char lectura[]
Métodos	- void loop() - void setup() - void lectura() - void planificador() - void avanzar(int speed) - void retroceder(int speed) - void parada() - int pingDist() - void sendESPCmd(String cmd) - void clearESPBuffer()

Tabla 63. Clase CL-07

4.2.3. Diagramas de secuencia

En este apartado se muestran los diagramas de secuencia relativos a los casos de uso para describir las interacciones del Sistema explicadas en el apartado 3.2.2. Especificaciones de los casos de uso:

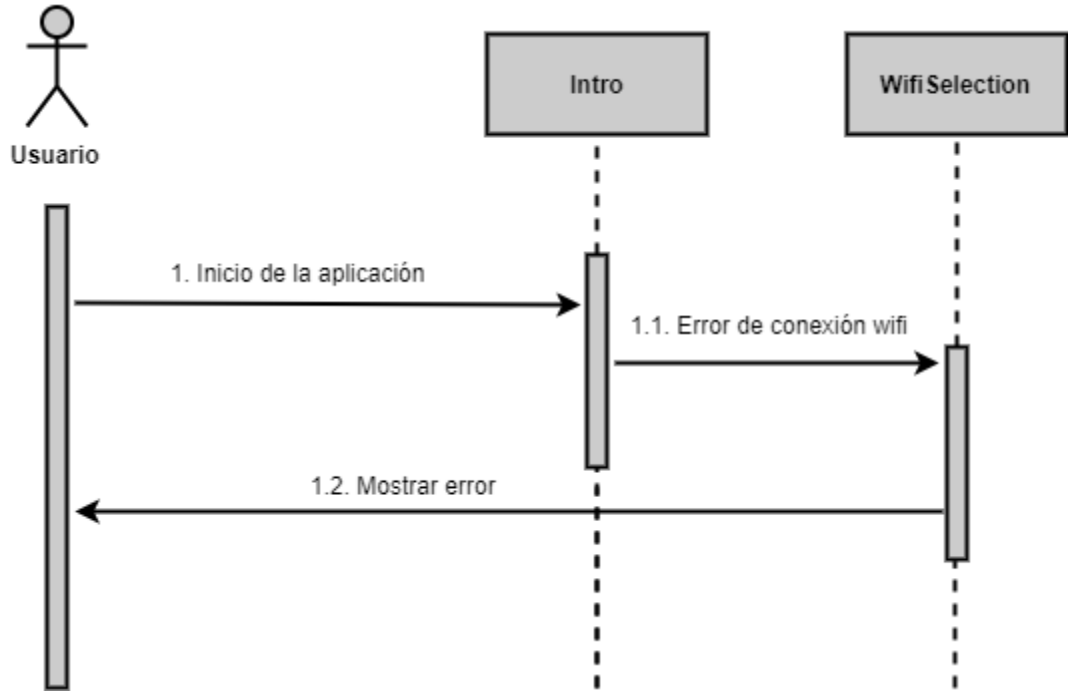

Ilustración 26. Diagrama de secuencia CU-01

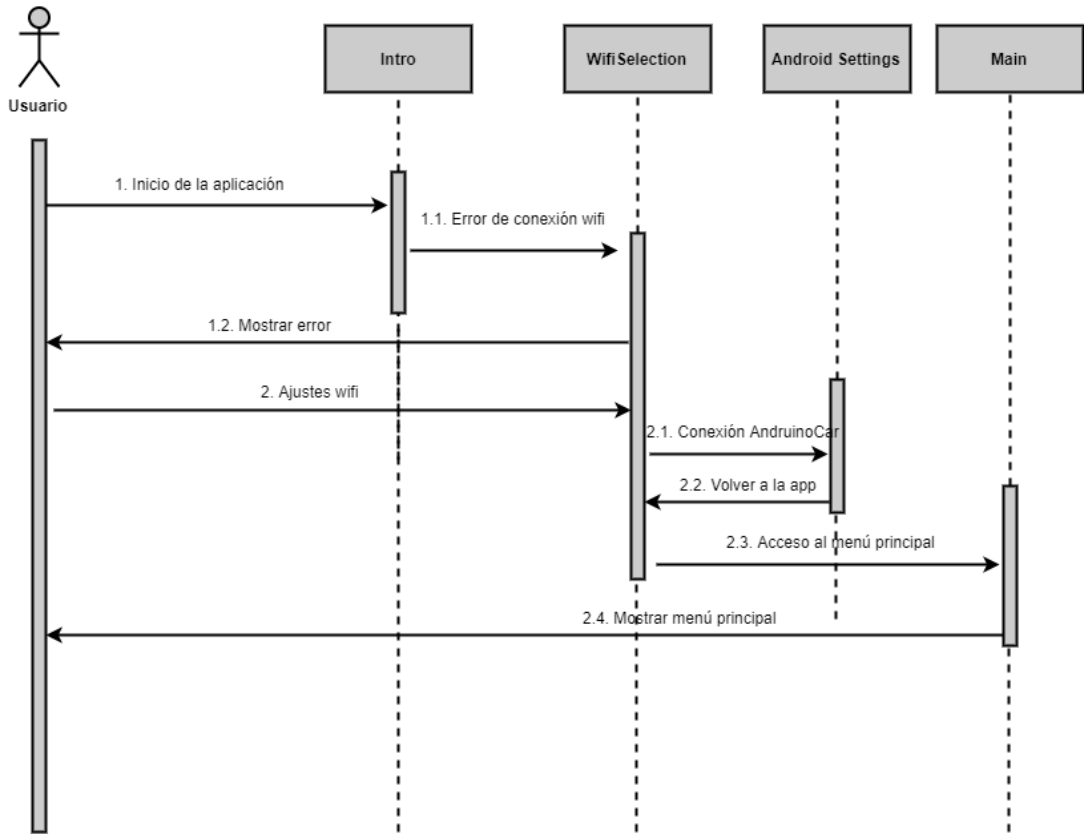

Ilustración 27. Diagrama de secuencia CU-02

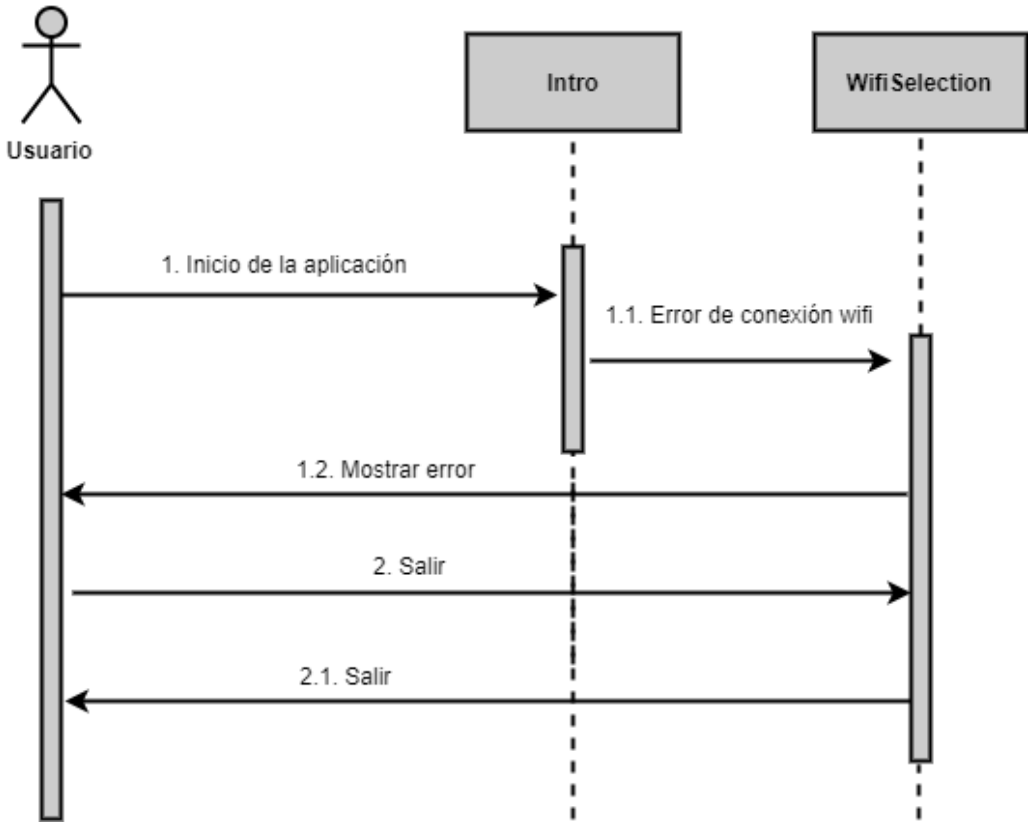

Ilustración 28. Diagrama de secuencia CU-03

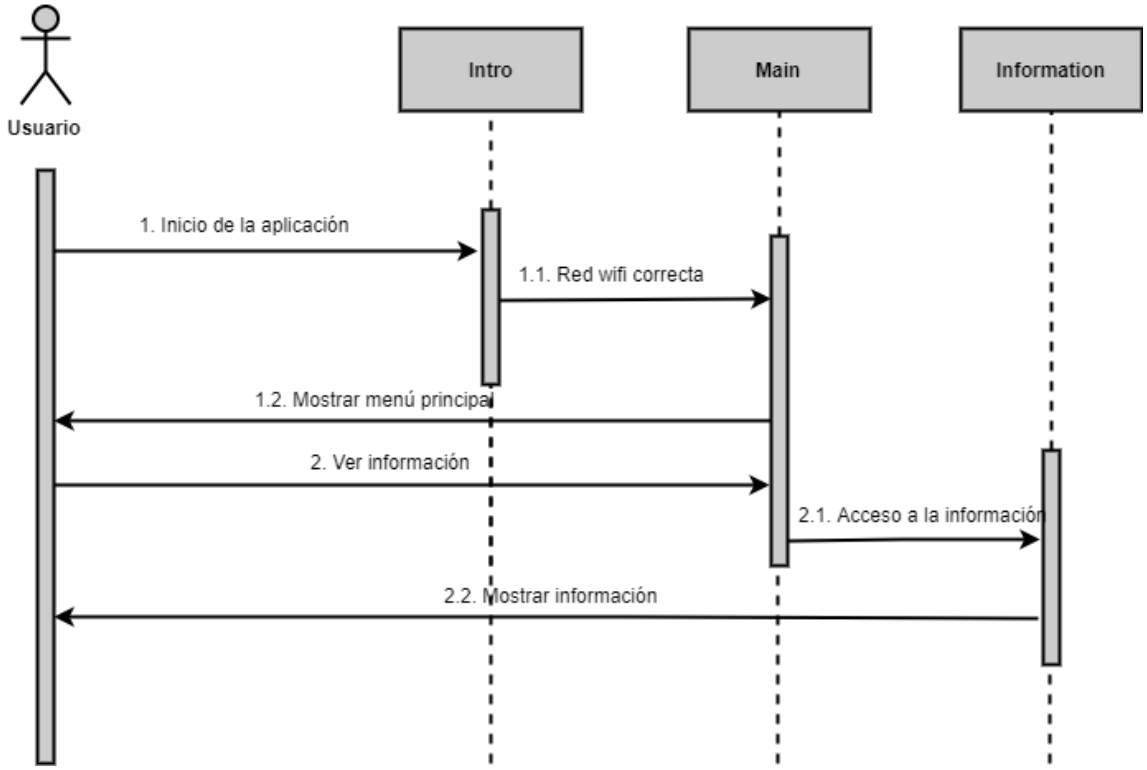

Ilustración 29. Diagrama de secuencia CU-04

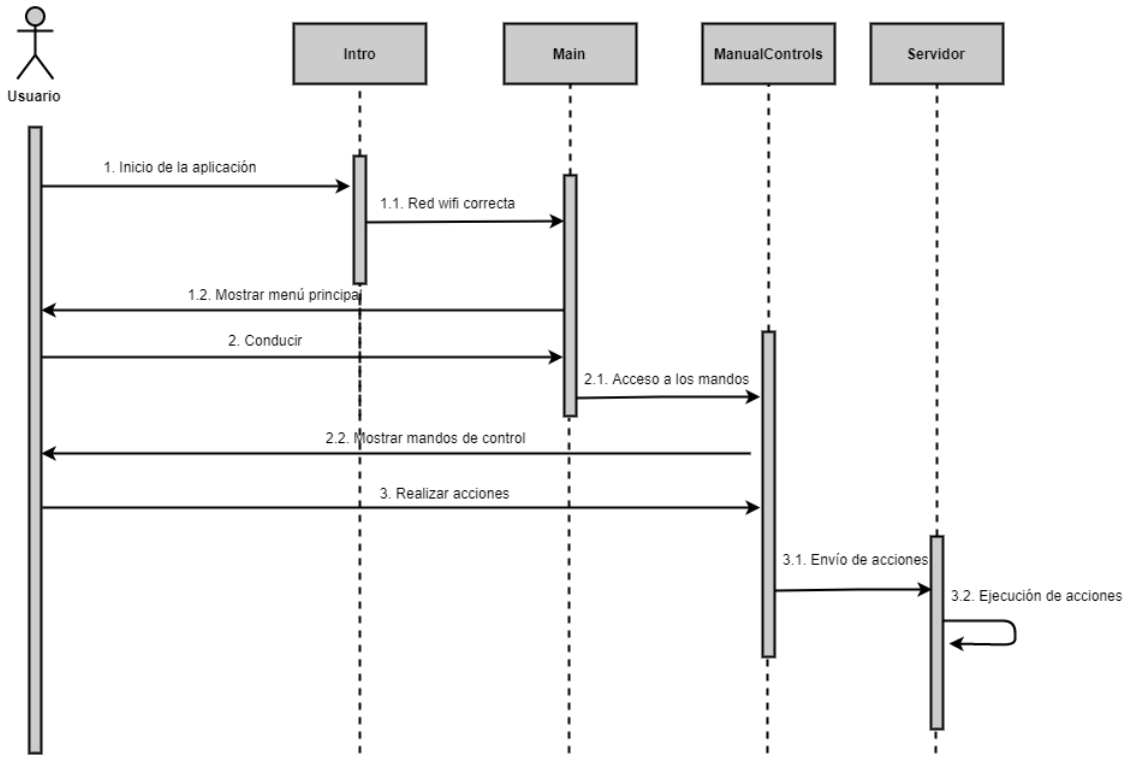

Ilustración 30. Diagrama de secuencia CU-05

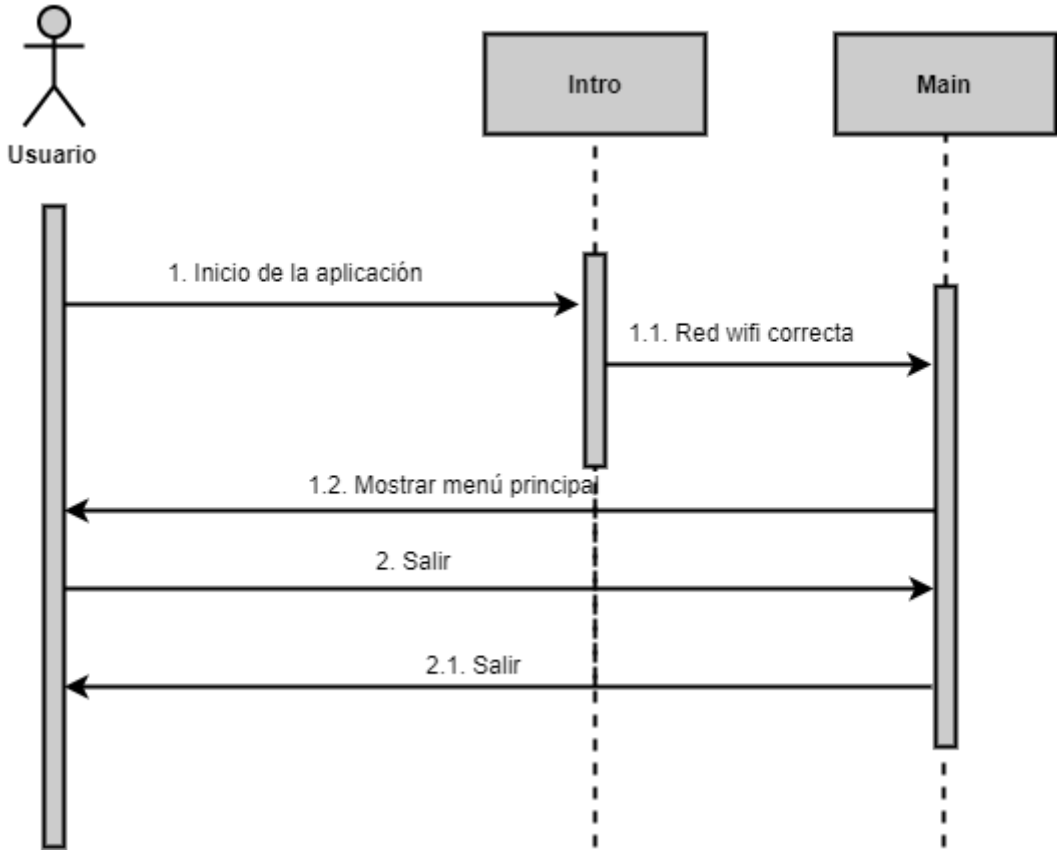

Ilustración 31. Diagrama de secuencia CU-06

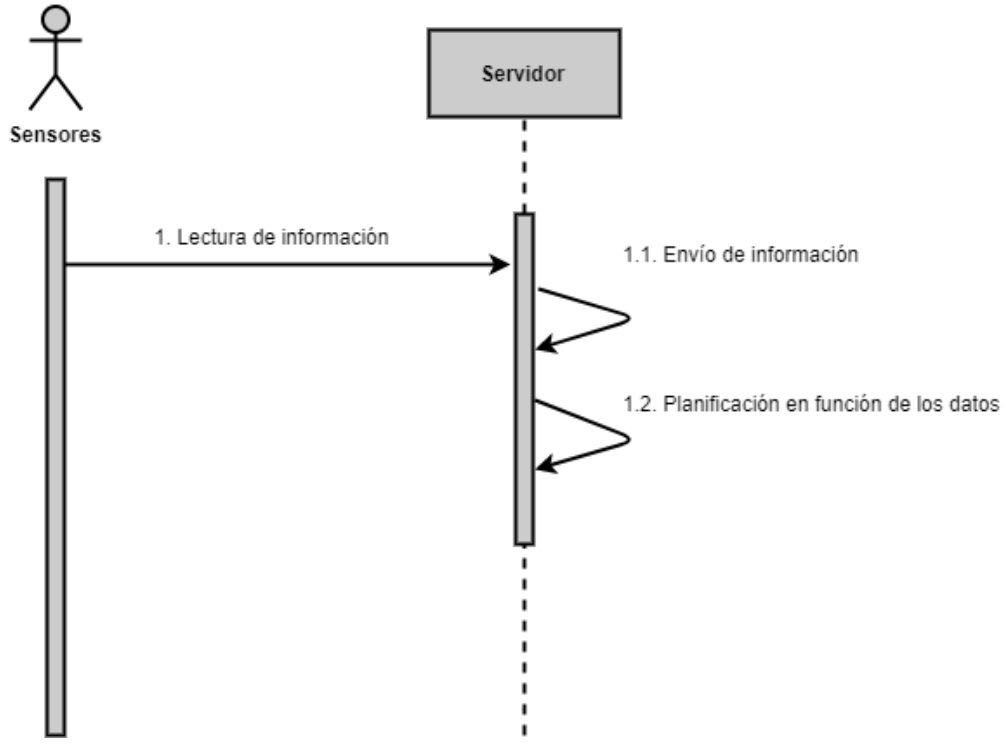

Ilustración 32. Diagrama de secuencia CU-07

4.3. Diseño de interfaz de usuario

En este apartado se mostrará el diseño de las interfaces de usuario de la aplicación cliente a través de las cuales el usuario interactuará con el sistema. Todas las interfaces se mostrarán de forma vertical (Portrait) únicamente excepto la pantalla de los mandos de control que únicamente se mostrará horizontalmente (Landscape) para simular la utilización de un volante. Los colores predominantes serán los representativos de Arduino (azul) y Android (verde). El sistema presentará las siguientes interfaces:

4.3.1. Pantalla de introducción

Esta pantalla se muestra durante 3 segundos mientras la aplicación comprueba la conexión wifi.

Ilustración 33. Interfaz de introducción

4.3.2. Pantalla de error wifi

Esta pantalla se muestra cuando el dispositivo no se encuentra conectado correctamente a la red wifi del vehículo o pierde la conexión. Desde ella se puede salir o acceder a los ajustes wifi de Android.

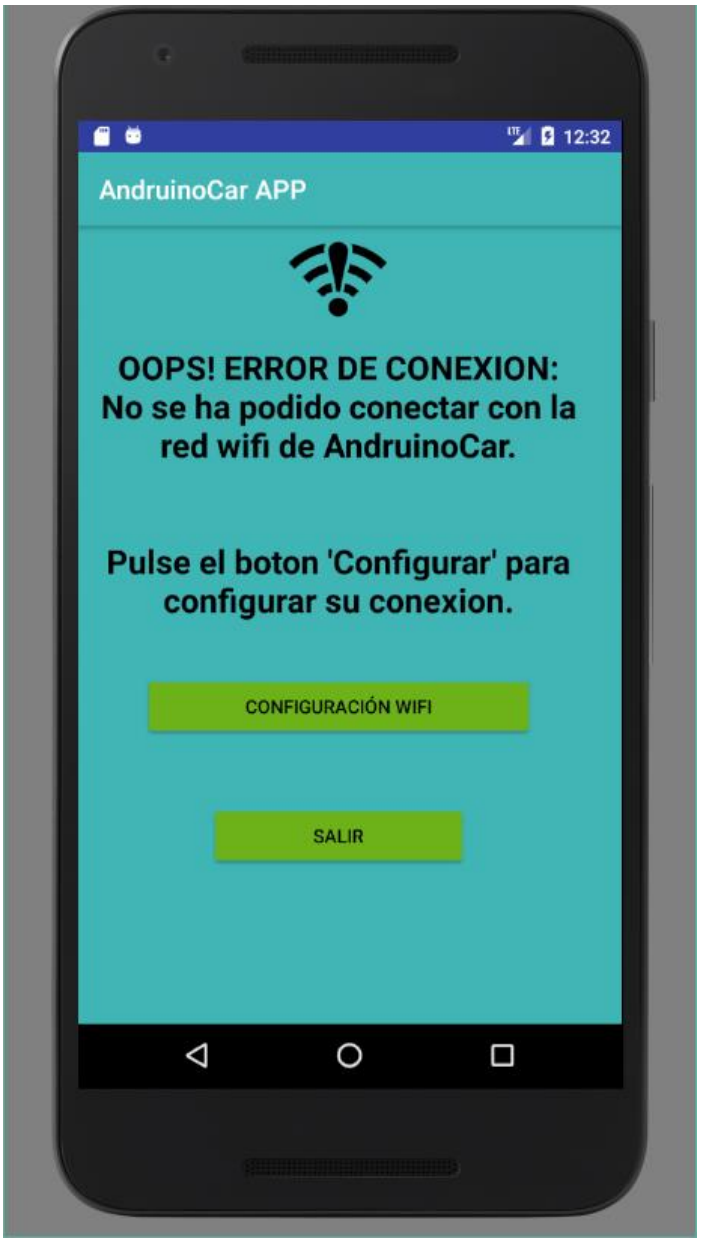

Ilustración 34. Interfaz de error de conexión

4.3.3. Pantalla del menú principal

Esta pantalla se muestra si la conexión al iniciar el programa es correcta. En ella se muestran las diferentes opciones que puede seleccionar el usuario: iniciar conducción, visitar la información de uso o salir de la aplicación.

Ilustración 35. Interfaz del menú principal

4.3.4. Pantalla de información

Esta pantalla muestra la información de utilización del vehículo.

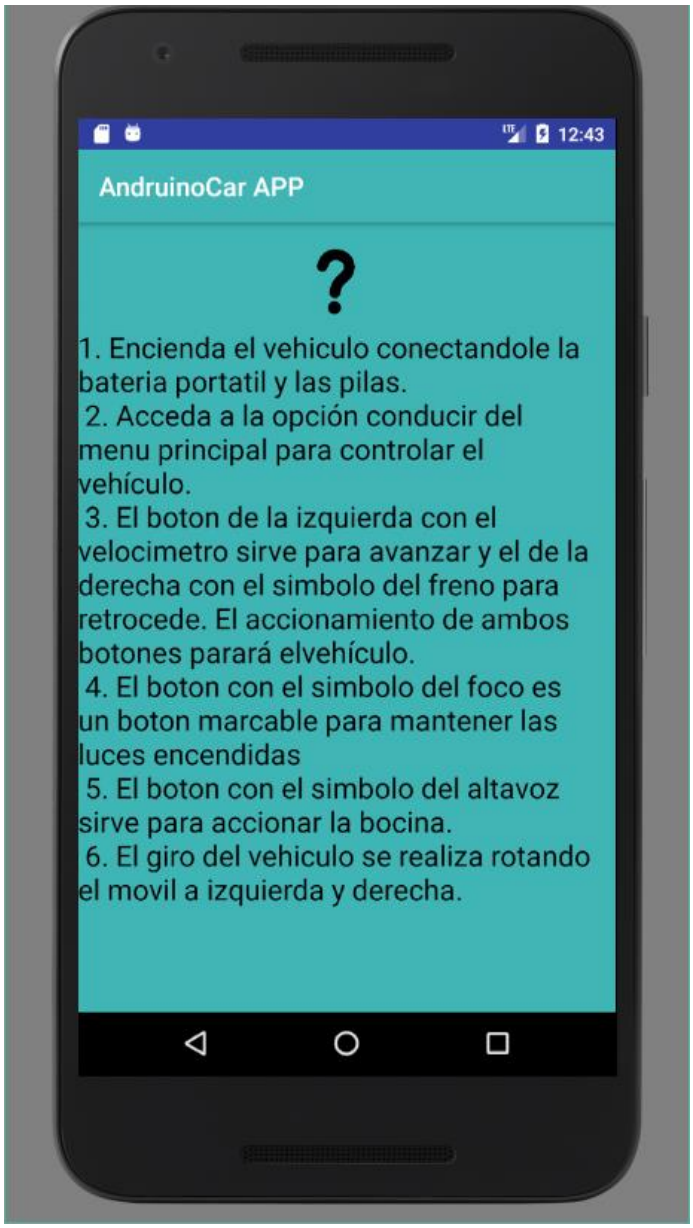

Ilustración 36. Interfaz de información

4.3.5. Pantalla de mandos de control

Esta pantalla contiene los botones necesarios para el manejo del vehículo de forma remota. También detecta las rotaciones del dispositivo para controlar el giro.

Ilustración 37. Interfaz de mandos de control

4.4. Especificación del entorno de desarrollo

En esta sección se presentan las herramientas que se han utilizado para el desarrollo del proyecto. La selección de estas herramientas son una parte del diseño necesario para la implementación del sistema. Dentro de estas herramientas se encuentran los dispositivos hardware utilizados para la implementación, los sistemas operativos del entorno de desarrollo y del sistema, los lenguajes de programación usados y sus entornos de desarrollo.

4.4.1. Hardware

Los dispositivos utilizados para el desarrollo del proyecto y los elementos hardware utilizados en la implementación del sistema se muestran a continuación:

Modelo	Ordenador personal MSI GP62 2QE
Procesador	Intel(R) Core(TM) i5-4210H 2.90 GHz
Memoria RAM	8 GB 2133 MHz
Disco Duro	1TB
Tipo Sistema	Sistema operativo de 64 bits, procesador x64
Sistema Operativo	Windows 10 Pro Versión 1703

Tabla 64. Ordenador personal

Modelo	Móvil personal Meizu M3 Note
Procesador	Procesador Helio P10 (ARM Cortex-A53 de 1.8 GHz x 4 + ARM Cortex-A53 de 1 GHz x 4)
Memoria RAM	2 GB
Disco Duro	16 GB ROM
Tipo Sistema	Sistema operativo móvil de 64 bits, procesador x64
Sistema Operativo	Android Lollipop Versión 5.1

Tabla 65. Móvil personal

Modelo	Arduino UNO (ver 4.1.2.1. Hardware)
Procesador	ATMEGA 328
Memoria RAM	2KB
Disco Duro	32 KB
Tipo Sistema	Sistema de 8 bits
Sistema Operativo	-

Tabla 66. Microcontrolador

Modelo	ESP8266 (ver 4.1.2.1. Hardware)
Procesador	Tensilica L106
Memoria RAM	Instrucción de 64 KB, datos de 96 KB
Disco Duro	-
Tipo Sistema	32 bit
Sistema Operativo	-

Tabla 67. Módulo ESP8266

Además de estos elementos hardware se han utilizado el controlador L298N, el sensor HC-SR04, un buzzer, una resistencia LDR y leds que se explican en el apartado 4.1.2.1. Hardware.

4.4.2. Sistemas operativos

En este punto se presentan los sistemas operativos utilizados y las razones por las que se ha decidido utilizarlos. Los sistemas operativos utilizados para el desarrollo del proyecto, ya sea para la implementación, pruebas o documentación han sido:

- **Windows 10**: Este sistema operativo de la compañía Microsoft es uno de los sistemas más extendidos e implantados del mercado. Se puede instalar en cualquier dispositivo compatible a través de la compra de una licencia. Se ha utilizado este sistema por su amplio conocimiento además de la compatibilidad del software utilizado para desarrollar el sistema. También se ha utilizado por el paquete de ofimática Microsoft Office 2016 que ofrece.
- **Android Lollipop (Version 5.1)**: Esta versión del sistema operativo Android desarrollado por Google se lanzó en 2014 con cambios internos para mejorar y optimizar el rendimiento interno y la batería de los teléfonos móviles compatibles. Este sistema operativo se ha utilizado para ejecutar la aplicación cliente y probar su correcto funcionamiento dentro del sistema comprobando a su vez la visualización de la interfaz de usuario.

4.4.3 Lenguajes de programación

En este punto se explican los lenguajes de programación utilizados para el desarrollo del sistema:

- **Java [33]**: Java es un lenguaje de programación y una plataforma informática comercializada por primera vez en 1995 por Sun Microsystems. Hay muchas aplicaciones y sitios web desarrolladas y en desarrollo con leguaje Java y necesitan la plataforma para funcionar. Java es rápido, seguro y fiable. Java se encuentra en todas partes: portátiles, centros de datos, consolas para juegos, súper computadoras, teléfonos móviles o Internet lo utilizan.

  Java no es un lenguaje desconocido ya que se ha utilizado en otros proyectos desarrollados en años anteriores por lo que su utilización será más sencilla. Además, se contará con la ayuda del entorno de desarrollo Android Studio.

- **C**: Este lenguaje de programación fue creado por Brian Kernighan y Dennis Ritchie a mediados de los 70 sobre un computador con sistema operativo UNIX. C es un lenguaje de propósito general que proporciona una gran flexibilidad de programación [34]. Se trata de un lenguaje de medio nivel que trata con objetos básicos como caracteres o números a la vez que con bits y direcciones de memoria. Posee una gran portabilidad y se utiliza para la programación de sistemas [35].

  Este lenguaje de programación es la base de la programación de Arduino y ha sido también utilizado en otros proyectos anteriores por lo que la única dificultad que supone es la utilización de los elementos hardware a través del código. En este caso se utilizará el IDE de Arduino que contienen el compilador y el cargador del código.

- **XML [36]**: Extensible Markup Language (Lenguaje de Etiquetado Extensible) es un lenguaje de etiquetas muy simple y estricto fundamental en el intercambio de una gran variedad de datos. Su función principal es describir datos, aunque también se puede utilizar para mostrarlos de forma similar al lenguaje HTML. Este formato sirve para estructurar, almacenar e intercambiar información.

  Este lenguaje es el que utiliza Android Studio para realizar y almacenar las interfaces de usuario. En este lenguaje no se tiene mucha experiencia, pero el entorno de desarrollo ayuda a implementar las interfaces a través del editor visual que proporciona.

4.4.4. Software de desarrollo

En este punto se presentan las herramientas software utilizadas para el desarrollo del sistema:

- **Android Studio [37]:** es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android. Este entorno propone un entorno unificado para el desarrollo de aplicación para todos los dispositivos Android. El potente editor de códigos y las herramientas para desarrolladores ofrecen funciones como un emulador o herramientas y frameworks de prueba que aumentan la productividad durante el desarrollo de aplicaciones para Android.

- **IDE de Arduino [38]:** es el entorno de código abierto que pone a disposición Arduino para facilitar la escritura de código y subir los programas a la placa. Es compatible con Windows, Mac OS X y Linux. El entorno está escrito en Java y basado en software de código abierto. Este software se puede utilizar con cualquier tarjeta Arduino.

5. Implementación

En este apartado se explica con muy poca profundidad la implementación del Sistema, ya que se ha seguido detalladamente el diseño realizado en el apartado 4. Diseño del sistema. El objetivo principal de esta sección es presentar las tecnologías empleadas en la implementación del sistema y el resultado final del sistema.

5.1. Tecnologías utilizadas

Como se ha comentado anteriormente el componente Cliente será la aplicación en el dispositivo móvil Android del usuario implementada en Java y XML a través del entorno de desarrollo Android Studio. De esta forma la aplicación cliente puede instalada y ejecutada en cualquier dispositivo Android.

El componente Servidor es la parte hardware del sistema y se ha implementado a través de un sketch del IDE de Arduino, entorno de desarrollo que utiliza el lenguaje de programación C/C++. El programa se cargará en la placa Arduino UNO con los componentes conectados. Este programa no es exportable a otros microcontroladores y no es compatible con todo Arduino ya que se necesitará que el montaje hardware sea igual, es decir, todos los elementos estén conectados y configurados en los mismos pines.

5.1.1. Cliente

El componente Cliente se ha desarrollado utilizando Android Studio. Este entorno permite desarrollar la funcionalidad de las diferentes clases definidas en el diseño mediante el lenguaje de programación Java y asociarlo a las interfaces de la aplicación desarrolladas con el lenguaje de etiquetas XML. De esta forma se consigue una integración de la interfaz y de la funcionalidad de la aplicación.

Este entorno también permite crear dispositivos Android virtuales para probar la funcionalidad de la aplicación a la vez que va mostrando las trazas que se han introducido para comprobar el correcto funcionamiento de las clases. La instalación en el dispositivo simulado se realiza de igual forma que en un dispositivo real: se carga el archivo apk y se instala la aplicación. Una vez que la aplicación tiene un correcto funcionamiento se eliminan las trazas introducidas para que no resten rendimiento a la aplicación.

5.1.2. Servidor

El componente servidor se ha desarrollado utilizando el IDE de Arduino que utiliza un lenguaje C/C++. Para su implementación hay que definir los pines para cada elemento tal y como se han conectado en el esquema electrónico (ver 4.1.2.1. Hardware). En el método setup se configura el módulo ESP8266 y los pines (INPUT/OUTPUT). Una vez que el servidor este levantado se comprueba constantemente si hay algo que leer. En cuanto hay algo que leer, lo recoge y lo formatea para actuar en función de los datos leídos.

Para su implementación se utilizaron trazas a través del puerto serie del ordenador. Una vez finalizada la implementación correcta de la funcionalidad del servidor se deben eliminar para evitar cualquier retraso o bloque en esos puntos y no perder rendimiento. El rendimiento del Servidor es muy importante porque, a pesar de que tolera ciertos fallos, tiene que trabajar en tiempo real y los fallos graves degenerarían el sistema.

5.2. Resultado final del sistema

En este apartado se muestra el aspecto final del sistema, tanto de la aplicación cliente como del vehículo a escala que contiene el servidor:

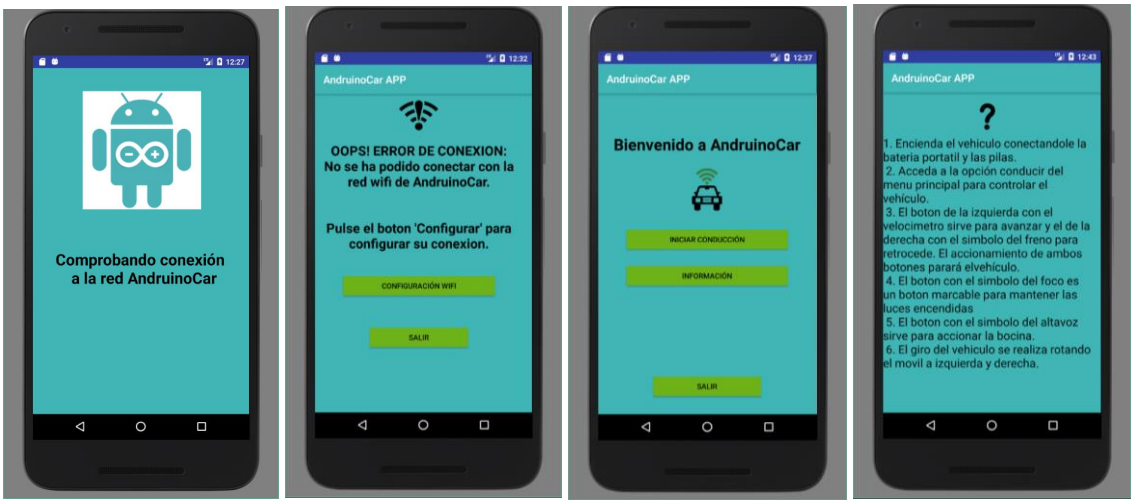

Ilustración 38. Resultado final de la aplicación del sistema

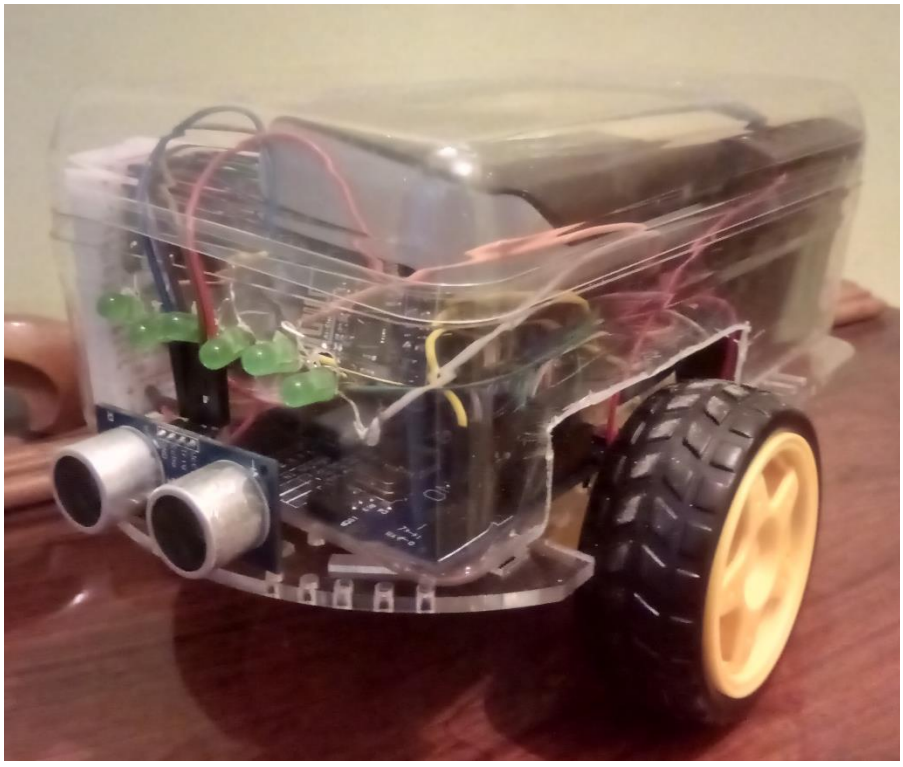

Ilustración 39. Resultado final del vehículo a escala

6. Plan de pruebas

Este apartado tiene como objetivo describir el proceso de comprobación de que tras la implementación del sistema se han incluido y desarrollado todas las funcionalidades definidas en el apartado 3. Análisis del sistema. Para realizar esta batería de pruebas se ha utilizado el hardware definido en 4.4.1. Hardware con la ayuda del dispositivo Android simulado que ofrece Android Studio.

6.1. Definición del alcance de las pruebas

Las pruebas que se van a realizar para comprobar el correcto funcionamiento del sistema implementado se pueden dividir en dos categorías:

- **Pruebas unitarias**: Comprenden las verificaciones asociadas a cada componente del sistema con el objetivo de comprobar la funcionalidad y estructura de cada componente individual.
- **Pruebas del sistema**: Son pruebas de integración del sistema completo. Su objetivo es verificar las especificaciones técnicas y funcionales del sistema en su conjunto.

6.2. Especificación de pruebas

6.2.1. Pruebas unitarias

En este apartado se definen las pruebas unitarias sobre los diferentes componentes del sistema a través de tablas con el siguiente formato:

PU-XX	
Título	
Objetivo	
Clase	
Método	
Implementación	
Resultado	

Tabla 68. Formato tabla pruebas unitarias

Los campos que componen la tabla se definen a continuación:

- **PU-XX**: Código identificativo único donde **PU** es del identificador de pruebas unitarias y **XX** es el número de la secuencia de pruebas.
- **Título**: Breve descripción de la prueba.
- **Objetivo**: El objetivo que persigue la realización de la prueba.
- **Clase**: Clase del sistema que contiene el elemento a probar.
- **Método**: Nombre del método sobre el que se realiza la prueba.
- **Implementación**: Parámetros que utilizan para la prueba.
- **Resultado**: Resultados esperados a partir de los parámetros introducidos.

PU-01	
Título	Comprobar conexión wifi inicial
Objetivo	Si el dispositivo se encuentra conectado a la red wifi se mostrará la pantalla principal sino mostrará la pantalla de error.
Clase	Intro
Método	onCreate
Implementación	Iniciar la aplicación
Resultado	-Pantalla principal si la conexión wifi es correcta -Pantalla de error wifi si la conexión es incorrecta

Tabla 69. Prueba unitaria PU-01

PU-02	
Título	Comprobar construcción trama de acciones
Objetivo	Comprobar que la aplicación cliente introduce los datos correctos dentro del tiempo establecido
Clase	MyClientTask
Método	doInBackground
Implementación	Colocar trazas que imprima la cadena de acciones junto al tiempo que tarda en realizarlo
Resultado	-Cadena de acciones

Tabla 70. Prueba unitaria PU-02

PU-03	
Título	Comprobar el servidor recibe trama de acciones
Objetivo	Comprobar que los datos recibidos del cliente se corresponden con los enviados
Clase	Servidor
Método	lectura
Implementación	Colocar trazas que imprima la cadena de acciones recibida
Resultado	-Cadena de acciones recibida

Tabla 71. Prueba unitaria PU-03

PU-04	
Título	Avance
Objetivo	Comprobar que los motores se pueden mover hacia delante
Clase	Servidor
Método	avance
Implementación	Llamar al método
Resultado	-Movimiento de avance de las ruedas -Error: Mantiene las ruedas inmóviles

Tabla 72. Prueba unitaria PU-04

PU-05	
Título	Retroceso
Objetivo	Comprobar que los motores se pueden mover hacia atrás
Clase	Servidor
Método	retroceso
Implementación	Llamar al método
Resultado	-Movimiento de retroceso de las ruedas -Error: Mantiene las ruedas inmóviles

Tabla 73. Prueba unitaria PU-05

PU-06	
Título	Bocina
Objetivo	Comprobar que el buzzer emite sonido
Clase	Servidor
Método	planificador
Implementación	Poner la acción de bocina a 1
Resultado	-Emisión de sonido -Error: Silencio

Tabla 74. Prueba unitaria PU-06

PU-07	
Título	Luces
Objetivo	Comprobar que los leds se encienden
Clase	Servidor
Método	planificador
Implementación	Poner la acción de luces a 1
Resultado	-Encendido de leds -Error: No se encienden los leds

Tabla 75. Prueba unitaria PU-07

PU-08	
Título	Distancia
Objetivo	Comprobar que el sensor de distancia funciona correctamente
Clase	Servidor
Método	pingDist
Implementación	Imprimir la distancia que devuelve el método
Resultado	-Distancia aproximada correcta -Error: 0

Tabla 76. Prueba unitaria PU-08

PU-09	
Título	Distancia
Objetivo	Comprobar que el sensor de distancia funciona correctamente
Clase	Servidor
Método	pingDist
Implementación	Imprimir la distancia que devuelve el método
Resultado	-Distancia aproximada correcta -Error: 0

Tabla 77. Prueba unitaria PU-09

PU-10	
Título	Luminosidad
Objetivo	Comprobar que la resistencia LDR funciona correctamente
Clase	Servidor
Método	planificador
Implementación	Tapar la resistencia y comprobar si se encienden los leds
Resultado	-Encendido de leds -Error: Los leds permanecen apagados.

Tabla 78. Prueba unitaria PU-10

6.2.2. Pruebas del sistema

Una vez probados los componentes del sistema por separado se comprueba cómo funciona el sistema con dichos componentes integrados. En este apartado se definen las pruebas del sistema a través de tablas con el siguiente formato:

PS-XX	
Nombre	
Objetivo	
Implementación	
Resultado	

Tabla 79. Formato tabla pruebas del sistema

Los campos que componen la tabla se definen a continuación:

- **PS-XX**: Código identificativo único donde **Ps** es del identificador de pruebas del sistema y **XX** es el número de la secuencia de pruebas.
- **Nombre**: Breve descripción de la prueba.
- **Objetivo**: El objetivo que persigue la realización de la prueba.
- **Implementación**: Parámetros que utilizan para la prueba.
- **Resultado**: Resultados esperados a partir de los parámetros introducidos.

PS-01	
Nombre	Inicio aplicación
Objetivo	Acceder a la aplicación y me lleve a la pantalla correcta en función de la red wifi
Implementación	Abrir la aplicación
Resultado	-Pantalla de error wifi -Pantalla del menú principal

Tabla 80. Prueba del sistema PS-01

PS-02	
Nombre	Configuración wifi
Objetivo	Acceder a la configuración wifi del teléfono
Implementación	Abrir la aplicación con el wifi desconectado y en la pantalla de error wifi seleccionar la opción de configuración para acceder a los ajustes del teléfono y activar el wifi. Volver a la aplicación.
Resultado	-Pantalla de error wifi (red incorrecta) -Pantalla del menú principal (red correcta)

Tabla 81. Prueba del sistema PS-02

PS-03	
Nombre	Salir
Objetivo	Salir de la aplicación
Implementación	Pulsar en los botones de salir que ofrece la aplicación o salir directamente.
Resultado	-Se cierra la aplicación

Tabla 82. Prueba del sistema PS-03

PS-04	
Nombre	Desplazamiento
Objetivo	Mover el vehículo remotamente
Implementación	Acceder a la opción de conducir que ofrece la aplicación y pulsar los botones de avanzar, retroceder, luces o bocina. Utilizar la rotación del dispositivo para girar
Resultado	-El vehículo avanza y retrocede girando tanto como se rote el dispositivo móvil, enciende los leds o emite sonido. -El vehículo permanece estático (se han pulsado avanzar y retroceder simultáneamente)

Tabla 83. Prueba del sistema PS-04

PS-05	
Nombre	Sensor de distancia
Objetivo	Parada de seguridad
Implementación	Dirigir el vehículo a través del avance hacia un obstáculo
Resultado	-El vehículo se detiene y solo permite retroceder.

Tabla 84. Prueba del sistema PS-05

PS-06	
Nombre	Sensor de luminosidad
Objetivo	Luces de seguridad
Implementación	Se apagan las luces de la habitación
Resultado	-El vehículo enciende los leds

Tabla 85. Prueba del sistema PS-06

PS-07	
Nombre	Apagado wifi
Objetivo	Al pagar el wifi mientras se realizan acciones de conducción el vehículo se detiene y la aplicación lleva a la pantalla de error wifi.
Implementación	Apagar el wifi desde el dispositivo móvil
Resultado	-Pantalla de error wifi y detención del vehículo.

Tabla 86. Prueba del sistema PS-07

PS-08	
Nombre	Salir de los mandos de control
Objetivo	Volver al menú principal y detener el vehículo
Implementación	Pulsar el botón de atrás
Resultado	-Pantalla del menú principal y detención del vehículo.

Tabla 87. Prueba del sistema PS-08

6.3. Análisis de consistencia

Las pruebas realizadas tanto unitarias como del sistema cubren todos los requisitos de software que definen la funcionalidad del sistema. Además, el resultado obtenido para todas las pruebas es el esperado y correcto por lo que se da por finalizado el prototipo de este proyecto.

7. Gestión del proyecto

En este apartado se detalla la realización de la planificación y gestión del proyecto que se ha llevado a cabo para su realización, teniendo en cuenta la planificación que se realizó inicialmente y la duración real del proyecto. También se realiza el presupuesto del proyecto realizando un desglose por costes. El objetivo de esta actividad es realizar un seguimiento y control de las actividades y recursos que intervienen en el desarrollo del sistema.

7.1. Planificación temporal

En este apartado se realiza la planificación en el tiempo de las actividades y tareas que se realizan para el desarrollo del sistema. La técnica utilizada para la planificación es la representación de diagramas de Gantt, una herramienta para planificar y programar tareas a lo largo de un período determinado que permite visualizar las actividades a realizar, la interdependencia entre ellas y su planificación en el tiempo del proyecto [39].

7.1.1. Planificación inicial del proyecto

En este punto se expone la planificación realizada al iniciar el proyecto el 01 de mayo de 2017, estableciéndose como fecha de finalización el 15 de septiembre de 2017 (duración inicial estimada aproximadamente de 4 meses y medio). De forma aproximada, el trabajo diario invertido en la realización del proyecto se estableció en una media de 4 horas diarias de lunes a viernes y los fines de semana libres.

Teniendo en cuenta la duración del proyecto y la carga aproximada de trabajo diario se realizó la división del proyecto en tareas y la planificación de cada tarea se muestra en la siguiente ilustración (ver Ilustración 40). En ella se puede observar la división del proyecto en tareas y subtareas y como se organizan en el tiempo. También se puede comprobar como algunas actividades necesitan la realización de otra previamente para poder iniciarse, por ejemplo, para poder iniciar la implementación era necesario la recepción del hardware comprado.

Ilustración 40. Diagrama de Gantt de la planificación inicial

7.1.2. Desarrollo real del proyecto

Tras la planificación real presentada en el apartado anterior, en este punto se presenta la planificación real que se ha seguido ya que existen tareas que han sufrido modificaciones en su duración con respecto a lo estimado inicialmente. Esto se debe a la poca experiencia en la estimación de periodos de duración de las actividades de un proyecto unido a que en un principio no se tuvieron en cuenta los periodos de exámenes ordinario y extraordinario, además de la inexperiencia en la programación de componentes hardware.

Atendiendo a la planificación real realizada (ver Ilustración 41), se puede observar que la entrega final del proyecto ha sufrido un ligero retraso de una semana, siendo la nueva fecha de finalización el 22 de septiembre de 2017. También se han visto modificadas otras tareas debido a que se realizaron más rápido de lo estimado inicialmente como por ejemplo es estudio del estado del arte. Otras tareas se alargaron más de lo estimado como por ejemplo la implementación e integración de los componentes hardware y software debido a la falta de experiencia en montaje y programación de elementos electrónicos.

También se ha visto modificada la compra de los elementos hardware, ya que esta tarea se separó de la actividad de diseño estando únicamente asociada al diseño de la arquitectura del sistema y pudiéndose realizar en paralelo con el resto de tareas de diseño. Por ello la actividad de implementación también vario ya que la implementación de la parte cliente no necesitaba que se hubiese completado la recepción de la compra realizada.

Otra tarea que también se ha extendido ligeramente es el plan de pruebas, ya que al realizar las pruebas se detectaron algunos fallos que se fueron solventando sin ser representados en el diagrama. Si esta tarea de corrección de errores se mostrará tendría como resultado que la fase de implementación se alarga hasta el final del plan de pruebas o sería necesario añadir otra tarea de corrección de errores.

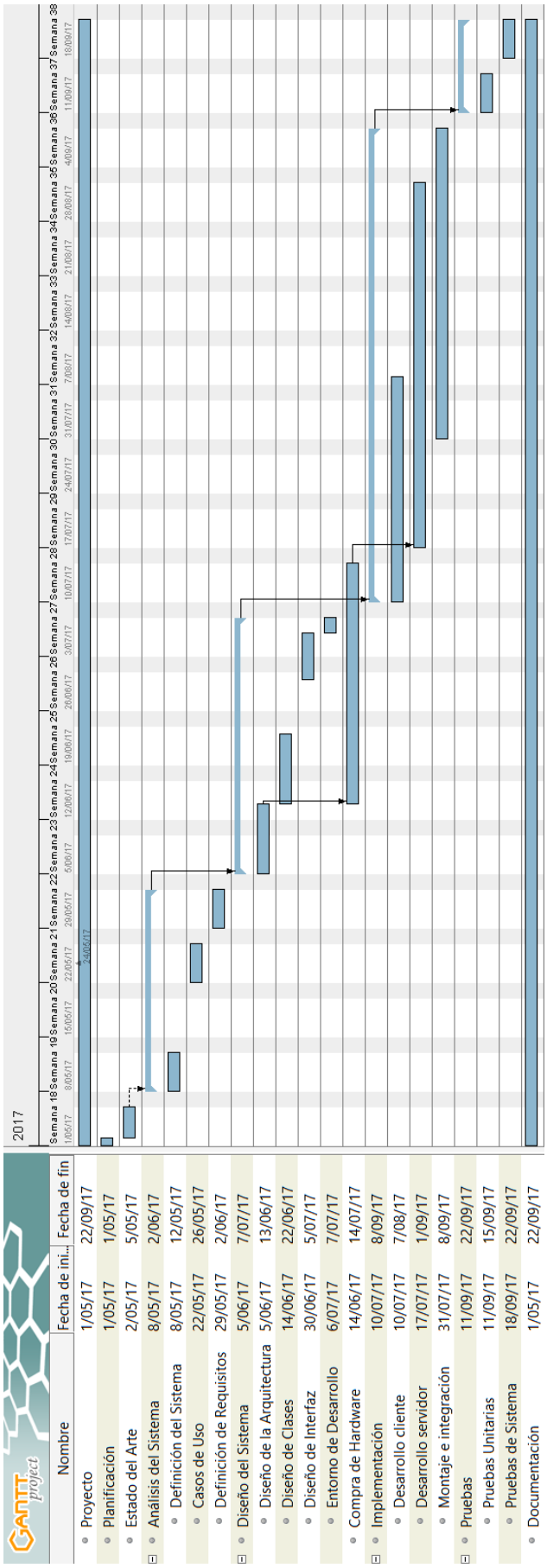

Ilustración 41. Diagrama de Gantt de la planificación real

7.2. Presupuesto

Este apartado expone detalladamente el contexto económico del proyecto a través de la estimación de un presupuesto en función de la planificación real explicada anteriormente. En primer lugar, se presentará el coste total del proyecto y, a continuación, en los siguientes subapartados se desglosará la procedencia de los costes para justificarlo.

7.2.1. Presupuesto total

El proyecto con título: ***Diseño e implementación de un vehículo a escala controlado remotamente***, con una duración de **4 meses y 22 días** (incluidos días no laborables), tiene un presupuesto total para su desarrollo de **22,778.81€ (VEINTIDOS MIL SETECIENTOS SETENTA Y OCHO CON OCHENTA Y UNO EUROS**) (I.V.A incluido).

7.2.2. Desglose del presupuesto

En este apartado se explica la procedencia de cada uno de los gastos que se han generado y se han incluido en el cálculo del coste total especificado en 7.2.1. Presupuesto total.

7.2.2.1. Costes directos

Coste de recursos humanos

Para la realización de las diversas actividades durante la realización del proyecto interviene personal especializado en diferentes ámbitos. Durante el desarrollo de este proyecto en el que solo ha intervenido una persona, dicha persona ha realizado los siguientes roles:

- **Jefe de proyecto**: será el encargado de planificar, dirigir y controlar a los equipos de trabajo que realizan las distintas tareas que comprende el proyecto para alcanzar los objetivos.
- **Analista**: será el encargado de realizar el análisis del sistema sacando los casos de uso y los requisitos de usuario y software.

- **Diseñador**: será el encargado de diseñar el sistema a partir del análisis previo. Realizará la definición de la arquitectura del sistema, la descripción de las clases y la especificación del entorno de desarrollo. Para este proyecto, el diseñador del sistema deberá tener conocimientos de electrónica para el diseño de la parte hardware.
- **Programador**: será el encargado de realizar la implementación del sistema a partir del diseño. Para este proyecto en concreto deberá de tener conocimientos de programación en Android (o Java) y de montaje y programación con Arduino.
- **Gestor de pruebas**: será el encargado de realizar el plan de pruebas teniendo en cuenta el diseño y la implementación para verificar que la funcionalidad del sistema desarrollado cumple con los requerimientos del cliente y se ejecuta correctamente.

Estos roles recibirán diferentes salarios en función de la actividad realizada y el tiempo dedicado medido en horas, atendiendo al informe sobre el mercado laboral [40].

Rol	Horas empleadas (h)	Coste (€/h)	Coste
Jefe de Proyecto	120	31.76	3,811.20 €
Analista	60	25.17	1,510.20 €
Diseñador	130	25.17	3,272.10 €
Programador	270	20.11	5,429.70 €
Gestor de Pruebas	60	22.88	1,372.80 €
TOTAL	**640**	**-**	**15,396.00 €**

Tabla 88. Costes de recursos humanos

Coste de material

Para la realización del proyecto se ha utilizado material informático tanto hardware como software. Parte del material informático que se ha utilizado para el desarrollo del proyecto era propiedad del autor, por lo que el coste de dichos elementos será la parte de la amortización correspondiente a la duración del proyecto suponiendo que el material se amortizará completamente en un período de 2 años. El coste de otra parte del material informático será incluido íntegramente ya que forma parte del producto final. También se tiene en cuenta los costes relativos al material fungible utilizado.

Unidades	Componente	Coste/ud	Meses de uso	Coste aplicable al proyecto
1	Ordenador portátil MSI	700.00 €	5	145.83 €
1	Móvil Android Meizu M3 Note	180.00 €	5	37.50 €
1	Licencia Windows 10 pro	280.00 €	5	58.33 €
1	Microsoft Office 2016	100.00 €	5	20.83 €
1	Kit Arduino Uno + Complementos	30.00 €	5	6.25 €
2	Módulo wifi esp8266	3.50 €	-	7.00 €
1	Puente H l298n	1.55 €	-	1.55 €
1	Chasis vehículo + 2 motores + Caja 4 pilas AAA	9.00 €	-	9.00 €
TOTAL				**286.29 €**

Tabla 89. Costes de material informático

Unidades	Artículo	Coste unidad	Coste
1	Cuaderno Tauro Plastic	2.00 €	2.00 €
2	Bolígrafos BIC	0.30 €	0.60 €
1	Paquete de 100 Post-it	3.00 €	3.00 €
TOTAL			**5.60 €**

Tabla 90. Costes de material fungible

El montante total de los **costes directos** es **15,687.89 € (QUINCE MIL SEISCIENTOS OCHENTA Y SIETE CON OCHENTA Y NUEVE EUROS)**.

7.2.2.2. Costes indirectos

Los costes indirectos del proyecto asociados a gastos como la luz, la conexión a internet o el teléfono se calculan a partir de una estimación porcentual sobre los costes directos. El porcentaje aplicado para este proyecto será del 5%.

Costes indirectos	
5% sobre costes directos	784.39 €

Tabla 91. Costes indirectos

7.2.2.3. Beneficios

Una vez calculados los costes del proyecto, se realiza el cálculo del beneficio a obtener por el desarrollo del proyecto. Para calcular la ganancia económica que se desea obtener del proyecto se ha decidido aplicar un 15% de beneficio sobre los costes directos del proyecto.

Beneficios	
15% sobre costes directos	2,353.18 €

Tabla 92. Beneficios

7.2.3. Resumen de costes

En este apartado se muestra un resumen del desglose de los costes realizado anteriormente para mostrar el precio final del proyecto sin IVA y finalmente con IVA.

Resumen de Costes	
Costes Directos	
Recursos humanos	15,396.00 €
Material informático	286.29 €
Material fungible	5.60 €
Costes indirectos	784.39 €
Beneficios	2,353.18 €
TOTAL (sin IVA)	**18,825.46 €**
IVA (21%)	3,953.35 €
TOTAL (con IVA)	**22,778.81 €**

Tabla 93. Resumen de costes

8. Marco regulador y ético

En este punto se explica el marco legal en todo lo referente al desarrollo de este proyecto como son los estándares utilizados y las licencias de las tecnologías utilizadas.

8.1. Android Studio

La aplicación cliente ha sido desarrollada para dispositivos móviles con sistema operativo Android. Por ello la herramienta que se ha utilizado para su implementación ha sido Android Studio que utiliza el lenguaje de programación Java. A pesar de que este lenguaje de programación pertenece a la compañía Oracle, este entorno de desarrollo integrado lo incorpora. Por ello solo se atenderá a los aspectos legales de este IDE.

Los términos y condiciones de uso de este entorno de desarrollo para Android se exponen en su página web [41]. Como se indica en el documento, al aceptar las condiciones del Acuerdo de licencia Google otorga para el kit de desarrollo de software Android una licencia limitada, mundial, libre de derechos de autor, no cedible, no exclusiva, no susceptible de someterse a otras licencias y únicamente con el fin de desarrollar aplicaciones para Android.

8.2. IDE Arduino

La parte de servidor formada por el vehículo a escala completo se ha desarrollado utilizado el microcontrolador Arduino UNO y elementos pasivos y activos. Como se indica en su página web [42], Arduino es una compañía de hardware libre de código abierto que diseña y fabrica placas de desarrollo de hardware.

El software de Arduino también es de código abierto y los términos de uso expuestos en su página web [38] se aceptan al descargarlo. El código fuente para el entono desarrollado en Java se publica bajo la Licencia Pública General de GNU, que garantiza a los usuarios finales la libertad de usar, estudiar, compartir/copiar y modificar el software [43]. Las bibliotecas C/C++ de los microcontroladores utilizadas para la programación están bajo la Licencia Pública General Reducida de GNU, que garantizar la libertad de compartir y modificar el software asegurando que el software es libre para todos sus usuarios [44].

8.3. Git

Git es software de gestión de versiones que ha permitido gestionar el repositorio en el que se ha almacenado este proyecto, así como controlar las nuevas contribuciones realizadas al mismo. Como se indica en su página web [45], Git se publica con licencia de código abierto bajo la Licencia Pública General de GNU para garantizar la libertad de compartir y cambiar software libre asegurándose de que es gratuito para todos sus usuarios.

8.4. Bitbucket

Bitbucket, perteneciente a la compañía Atlassian, es un servicio de alojamiento web para proyectos que utilizan el sistema de control de revisiones Mercurial y Git, como en este proyecto. Bitbucket propone una solución gratuita y otra solución comercial en función del número de repositorios/proyectos se deseen almacenar. Para la realización de este proyecto se ha utilizado la versión gratuita incluye la creación de hasta 5 repositorios privados. Al crear la cuenta y aceptar las condiciones se acepta el acuerdo definido en la página [46]. La ventaja de este tipo de repositorios es que en un futuro se podría compartir con la comunidad convirtiendo el repositorio en público.

9. Trabajos futuros y conclusiones

En este apartado se exponen las conclusiones obtenidas del trabajo realizado para el desarrollo de este proyecto y del producto final desarrollado. También se presentan las conclusiones personales sobre la realización del Trabajo de Fin de Grado y los trabajos futuros y las líneas de desarrollo que podría seguir este proyecto.

9.1. Trabajos futuros

El ámbito del control remoto en la actualidad está adquiriendo mucha importancia ya que facilita tareas habituales. Por eso no se espera que este proyecto termine aquí como un producto final, sino que se pretende que sirva de base para el desarrollo de proyectos mayores con la introducción de nuevas funcionalidades. Algunas funcionalidades que se podrían introducir son:

1. La primera mejora que se podría realizar es sustituir los sensores de obstáculos de ultrasonidos por sensores infrarrojos que tienen una mayor fiabilidad.
2. Se podría añadir una cámara que retransmitiera en directo el camino que está siguiendo el vehículo. De esta forma a través de la tecnología wifi se podría controlar el vehículo desde cualquier dispositivo que se encuentre en cualquier punto con conexión a internet. Esta nueva funcionalidad es posible que requiriese de un cambio de controlador y de chasis para poder introducir la cámara sobre el vehículo e incluso permitir que girase.
3. En función de la finalidad del sistema se podrían añadir otros instrumentos como por ejemplo un brazo mecánico que permitiese realizar acciones como coger, sujetar, cortar o taladrar. Esta funcionalidad podría interesar en campos como la defensa para desarrollar sistemas de desactivación de explosivos.
4. También en función del terreno se podría pensar en modificar el chasis del vehículo para mejorar su manejo y se podría pensar en implementar el sistema para más variedad de vehículos como por ejemplo drones.
5. El punto máximo de desarrollo seria llegar a mejorar el proyecto que Telefónica y Ericsson presentaron en em Mobile World Congress de 2017 [14].

9.2. Conclusiones

La elección de este proyecto como trabajo de fin de grado se realizó especialmente por el interés de conocer el montaje y funcionamiento de un sistema distribuido para controlar remotamente un vehículo a escala. Investigando en el campo en que se enmarca el proyecto, existen multitud de ejemplos muy similares que ya han explorado el control remoto de vehículos a escala implementados en Arduino y manejados desde Android.

A diferencia de estos ejemplos, en este proyecto se ha intentado aumentar la complejidad y utilizar una red wifi en lugar de Bluetooth. La utilización de la tecnología Bluetooth es la más extendida en este ámbito debido al mayor conocimiento de utilización de los módulos Bluetooth de Arduino con respecto a los modulo wifi y a que es más fácil de integrar con Android. Pero el rango del Bluetooth es muy pequeño y no permite la conexión desde otras redes externas como podría permitir una red wifi bien configurada para que se pudiese controlar el vehículo desde cualquier parte del mundo con Internet.

Otra complejidad que se decidió introducir con respecto a los ejemplos existentes es el control del giro a través de la rotación del dispositivo móvil. Esta capacidad ha tenido un resultado que no llega a las expectativas esperadas en comparación con el incremento de dificultad de diseño que supuso.

En cuanto al prototipo generado, ahora mismo es un juguete que ha servido de base para comprender el desarrollo de un sistema de estas características y servir de base para desarrollos futuros como se indica anteriormente. Por lo que el producto final del proyecto actualmente tiene como única función el ocio y no realiza un gran aporte a la sociedad. En cuanto al aspecto económico su valor puede ser un poco excesivo para un único vehículo, pero ahora que existe un diseño y un prototipo se podría pensar en pulir algunos detalles o incluir funcionalidad para realizar una producción en serie y obtener mayores beneficios económicos.

9.3. Conclusiones personales

La conclusión personal obtenida de la realización de este proyecto de fin grado es la ilusión de realizar un proyecto en el que tú decides como se realiza cada tarea y el diseño. También me ha sorprendido la capacidad que he adquirido de aplicar los conocimientos y experiencia adquiridos durante estos años.

La selección del proyecto de un vehículo a escala controlado remotamente se debió a que cuando lo vi me llamo mucho la atención de cómo se realizaría un proyecto de este tipo, ya que cuando era pequeño me preguntaba cómo podía mover un coche de juguete desde un mando. Ahora que he aprendido como se puede construir sistemas completos y he pasado por esta experiencia de desarrollar un sistema real y funcional puedo comprobar que cuando el trabajo da sus frutos es muy gratificante.

Aunque durante los años del grado y el desarrollo de este proyecto he encontrado problemas que me han costado resolver, el trabajo duro me ha permitido llegar a este punto: las conclusiones. Ahora veo que con la entrega de este trabajo me encuentro más cerca de terminar esta etapa tan buena que comenzó hace 4 años.

10. Referencias

[1] 5 ELEMENTOS CLAVES DE UN ROBOT, 2017. ANDORobots [En línea]. Disponible en: http://www.andorobots.com/blog/5-elementos-claves-de-un-robot?language=es [Último acceso: 25 septiembre 2017]

[2] OMARH, MONOGRAFIAS.COM, 2017, Sistemas distribuidos - Monografias.com. *Monografias.com* [En línea]. 2017. Disponible en: http://www.monografias.com/trabajos16/sistemas-distribuidos/sistemas-distribuidos.shtml [Último acceso: 25 septiembre 2017]

[3] VILLADA ROMERO, JOSÉ LUIS, 2015, *Instalación y configuración del software de servidor web (UF1271)*. Málaga: IC Editorial.

[4] Definición de Robot, 2017. *Dle.rae.es* [En línea]. Disponible en: http://dle.rae.es/?id=WYRlhzm [Último acceso: 25 septiembre 2017]

[5] Robots móviles (I), 2017. *Xatakaciencia.com* [En línea]. Disponible en: https://www.xatakaciencia.com/robotica/robots-moviles-i [Último acceso: 25 septiembre 2017]

[6] *Informe Mobile en España y en el Mundo 2017*, 2017. [En línea]. Disponible en: www.amic.media/media/files/file_352_1289.pdf [Último acceso: 25 septiembre 2017]

[7] EL CONFIDENCIAL, 2017, Adiós definitivo a Windows Phone: el 99% de los móviles en el mundo son iOS o Android. [En línea]. Disponible en: https://www.elconfidencial.com/tecnologia/2017-02-16/windows-phone-android-ios-moviles-smartphones_1333287/ [Último acceso: 25 septiembre 2017]

[8] PAe - Métrica v.3, 2017. *Administracionelectronica.gob.es* [En línea]. Disponible en: https://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Metrica_v3.html#.Wb0zw8hJZPY [Último acceso: 25 septiembre 2017]

[9] Definición de remoto — Definicion.de, 2017. *Definición.de* [En línea]. Disponible en: https://definicion.de/remoto/ [Último acceso: 25 septiembre 2017]

[10] Control remoto - EcuRed, 2017. *Ecured.cu* [En línea]. Disponible en: https://www.ecured.cu/Control_remoto#Definici.C3.B3n_del_t.C3.A9rmino [Último acceso: 25 septiembre 2017]

[11] Simple RC Car for Beginners (Android Control Over Bluetooth), 2017. *Instructables.com* [En línea]. Disponible en: http://www.instructables.com/id/Simple-RC-car-for-beginners-Android-control-over-/ [Último acceso: 25 septiembre 2017]

[12] Multifunction Bluetooth Controlled Robot Smart Car Kits For Arduino, 2017. www.banggood.com [En línea]. Disponible en: https://www.banggood.com/Multifunction-Bluetooth-Controlled-Robot-Smart-Car-Kits-For-Arduino-p-906628.html [Último acceso: 25 septiembre 2017]

[13] BB-8 Sphero Droide Star Wars + force band, 2017. *Juguetrónica* [En línea]. Disponible en: https://www.juguetronica.com/bb-8-sphero-droide-star-wars-force-band?gclid=CjwKCAjwjJjOBRBVEiwAfvnvBO_f3EsvJQrUvwSLTWLowXbKPXGvD_qgbtLrxHRSR3LluXc5CSlhHBoC6WIQAvD_BwE [Último acceso: 25 septiembre 2017]

[14] EUROPA PRESS, 2017, Telefónica y Ericsson realizan la primera prueba de conducción remota con tecnología 5G. [En línea]. Disponible en: http://www.europapress.es/economia/noticia-telefonica-ericsson-realizan-primera-prueba-conduccion-remota-tecnologia-5g-20170227175030.html [Último acceso: 25 septiembre 2017]

[15] ¿Qué significa Android e iOS? ¿Cuál es mejor?, 2017. *ValorTop* [En línea]. Disponible en: http://www.valortop.com/blog/android-ios [Último acceso: 25 septiembre 2017]

[16] PICmicro® MCU Estudio - ¿qué es un microcontrolador?, 2017. *Electronicaestudio.com* [En línea]. Disponible en: http://www.electronicaestudio.com/microcontrolador.htm [Último acceso: 25 septiembre 2017]

[17] RASPBERRY PI – Historia de la Informática, 2017. *Histinf.blogs.upv.es* [En línea]. Disponible en: http://histinf.blogs.upv.es/2013/12/18/raspberry-pi/ [Último acceso: 25 septiembre 2017]

[18] Arduino - Introduction, 2017. *Arduino.cc* [En línea]. Disponible en: https://www.arduino.cc/en/Guide/Introduction [Último acceso: 25 septiembre 2017]

[19] Red inalámbrica, 2017. *Es.wikipedia.org* [En línea]. Disponible en: https://es.wikipedia.org/wiki/Red_inalámbrica [Último acceso: 25 septiembre 2017]

[20] *Sistemas distribuidos ** Panorama **,* 2017. [En línea]. Disponible en: http://www.tamps.cinvestav.mx/~vjsosa/clases/sd/sistemas_distribuidos_panorama.pdf [Último acceso: 25 septiembre 2017]

[21] Requisito no funcional, 2017. *Es.wikipedia.org* [En línea]. Disponible en: https://es.wikipedia.org/wiki/Requisito_no_funcional [Último acceso: 25 septiembre 2017]

[22] JOSEVALLEP1, MONOGRAFIAS.COM, 2017, Definición arquitectura cliente servidor - Monografias.com. *Monografias.com* [En línea]. Disponible en: http://www.monografias.com/trabajos24/arquitectura-cliente-servidor/arquitectura-cliente-servidor.shtml#resum [Último acceso: 25 septiembre 2017]

[23] Arquitectura cliente servidor, 2017. *Oposicionestic.blogspot.com.es* [En línea]. Disponible en: https://oposicionestic.blogspot.com.es/2011/06/arquitectura-cliente-servidor.html [Último acceso: 25 septiembre 2017]

[24] PROTOCOLOS, MODELO, 2017, Modelo Cliente / Servidor - Protocolos. *Informaticaintroduccion20152.blogspot.com.es* [En línea] Disponible en: http://informaticaintroduccion20152.blogspot.com.es/2015/11/modelo-cliente-servidor-protocolos_2.html [Último acceso: 25 septiembre 2017]

[25] *COMUNICACIÓN ENTRE PROCESOS SOCKETS*, 2017. [En línea]. Disponible en: http://sopa.dis.ulpgc.es/ii-dso/leclinux/ipc/sockets/sockets.pdf [Último acceso: 25 septiembre 2017]

[26] Arduino y WIFI ESP8266 | Tutoriales Arduino, 2017. *Prometec.net* [En línea]. Disponible en: https://www.prometec.net/arduino-wifi/ [Último acceso: 25 septiembre 2017]

[27] *ESP8266 AT Instruction Set*, 2017. [En línea]. Disponible en: https://www.espressif.com/sites/default/files/documentation/4a-esp8266_at_instruction_set_en.pdf [Último acceso: 25 septiembre 2017]

[28] Controlar motores de corriente continua con Arduino y L298N, 2017. *Luis Llamas* [En línea]. Disponible en: https://www.luisllamas.es/arduino-motor-corriente-continua-l298n/ [Último acceso: 25 septiembre 2017]

[29] Arduino - PWM, 2017. *Arduino.cc* [En línea]. Disponible en: https://www.arduino.cc/en/Tutorial/PWM [Último acceso: 25 septiembre 2017]

[30] Medir distancia con Arduino y sensor de ultrasonidos HC-SR04, 2017. *Luis Llamas* [En línea]. Disponible en: https://www.luisllamas.es/medir-distancia-con-arduino-y-sensor-de-ultrasonidos-hc-sr04/ [Último acceso: 25 septiembre 2017]

[31] Buzzers o zumbadores | Tutoriales Arduino, 2017. *Prometec.net* [En línea]. Disponible en: https://www.prometec.net/buzzers/ [Último acceso: 25 septiembre 2017]

[32] Medir nivel de luz con Arduino y fotoresistencia LDR (GL55), 2017. *Luis Llamas* [En línea]. Disponible en: https://www.luisllamas.es/medir-nivel-luz-con-arduino-y-fotoresistencia-ldr/ [Último acceso: 25 septiembre 2017]

[33] ¿Qué es Java y para qué es necesario?, 2017. *Java.com* [En línea]. Disponible en: https://www.java.com/es/download/faq/whatis_java.xml [Último acceso: 25 septiembre 2017]

[34] *Lenguaje C*, 2017. [En línea]. Disponible en: http://informatica.uv.es/estguia/ATD/apuntes/laboratorio/Lenguaje-C.pdf [Último acceso: 25 septiembre 2017]

[35] Lenguaje de programación C++, 2017. *Aprendiendo Arduino* [En línea]. Disponible en: https://aprendiendoarduino.wordpress.com/2015/03/26/lenguaje-de-programacion-c/ [Último acceso: 25 septiembre 2017]

[36] Guía Breve de Tecnologías XML, 2017. *W3c.es* [En línea]. Disponible en: http://www.w3c.es/Divulgacion/GuiasBreves/TecnologiasXML [Último acceso: 25 septiembre 2017]

[37] Conoce Android Studio | Android Studio, 2017. *Developer.android.com* [En línea]. Disponible en: https://developer.android.com/studio/intro/index.html?hl=es-419 [Último acceso: 25 septiembre 2017]

[38] Arduino - Software, 2017. *Arduino.cc* [En línea]. Disponible en: https://www.arduino.cc/en/main/software [Último acceso: 25 septiembre 2017]

[39] ¿Qué es un diagrama de Gantt y para qué sirve?, 2017. *Obs-edu.com* [En línea]. Disponible en: http://www.obs-edu.com/es/blog-project-management/diagramas-de-gantt/que-es-un-diagrama-de-gantt-y-para-que-sirve [Último acceso: 25 septiembre 2017]

[40] Sector IT & Telecom - Informe de tendencias salariales, 2017. *Research.randstad.es* [En línea]. Disponible en: https://research.randstad.es/tendencias/sector-it-telecom-informe-de-tendencias-salariales [Último acceso: 25 septiembre 2017]

[41] Terms and Conditions | Android Studio, 2017. *Developer.android.com* [En línea]. Disponible en: https://developer.android.com/studio/terms.html [Último acceso: 25 septiembre 2017]

[42] Arduino - FAQ, 2017. *Arduino.cc* [En línea]. Disponible en: https://www.arduino.cc/en/Main/FAQ [Último acceso: 25 septiembre 2017]

[43] GNU General Public License, 2017. *Es.wikipedia.org* [En línea]. Disponible en: https://es.wikipedia.org/wiki/GNU_General_Public_License [Último acceso: 25 septiembre 2017]

[44] GNU Lesser General Public License, 2017. *Es.wikipedia.org* [En línea]. Disponible en: https://es.wikipedia.org/wiki/GNU_Lesser_General_Public_License [Último acceso: 25 septiembre 2017]

[45] About - Git, 2017. *Git-scm.com* [En línea]. Disponible en: https://git-scm.com/about/free-and-open-source [Último acceso: 25 septiembre 2017]

[46] Customer Agreement | Atlassian, 2017. *Atlassian* [En línea]. Disponible en: https://www.atlassian.com/legal/customer-agreement [Último acceso: 25 septiembre 2017]

Anexo A: Summary

Abstract

The goal of this end-degree project is the design and implementation of a remotely controlled scale vehicle using an application developed for mobile phones with Android operating system. The scale vehicle will be designed and implemented on an Arduino UNO microcontroller and its available hardware elements. The communication between the Android application and the physical device will be done through a Wi-Fi network. Thus, the structure of the project will be defined by the Client-Server model.

Android app on the mobile device will be the Client responsible for making requests to the server with the actions that the user wants to execute. For this, with the mobile connected to the vehicle Wi-Fi network, the user will be able to control the forward and backward moves, the lights and the sound through buttons and the rotation of the mobile. This way the mobile will be used as a command that is simple and intuitive for the user.

The server will be the physical device that receives and interprets the requests made by the application and activates the actuators necessary to execute the actions considering the environment using sensors. Therefore, the physical device is considered a robot [1]. The communication between Client-Server through the Wi-Fi technology will allow to exploit its benefits as the range or the connection from other networks.

The final objective of the project is the design and implementation of a scale vehicle prototype that will make available a Wi-Fi network to which a user can connect from his Android device and control it in real time through the application.

Introduction

At present, the use of distributed systems has been extended due to the development and use of cheap and reduced size microprocessors and to the impulse of networks and wireless communications [2]. Distributed systems could be defined as "systems whose hardware and software components are networked and communicate and coordinate for a specific purpose, offering a single system vision" [3].

The distributed systems are subdivided according to the centralized processing model, set of servers and Client-Server, being the Client-Server model that prevails today. This way, the objective system of this project could be explained from three points: the server that would be the robot that represents the scale vehicle, the client that would be the Android application and the wireless Wi-Fi network that will communicate client and server.

As previously mentioned, the reduction of costs and size of microcontrollers has also boosted the use of robots. A robot is a programmable electronic system capable of performing operations reserved to people [4] in an autonomous, programmable and environmentally friendly manner [1]. The vehicle parts are typical of robots: control systems and planners for a correct execution of actions, actuators that generate movements, internal sensors that measure and guide the actuators, external sensors that perceive the environment and a power supply that allows its operation without physical connections [5].

Currently, mobile is the device with the most popularity and extension: 66% of the world's population has one and sales of smartphones continue to grow [6]. As it is such a broad market, the development of mobile applications has become very important. Applications vary depending on the operating system used by mobile phones. Therefore, the client application will be for Android devices because its implementation is higher (81.7% in 2016) than its main competitor iOS (17.9% in 2016) and others (0.4%) [7].

Interconnected devices and the Internet of Things (IoT) are growing exponentially in recent years, both in the industrial area and in daily use [6]. Connections between different devices can be made through different technologies such as satellite, infrared, GSM, Bluetooth or Wi-Fi. Currently the most used connections, due to the expansion of smartphones discussed above, are GSM, Bluetooth and Wi-Fi.

Based on the above information, this project proposes the design and construction of a scale terrestrial robot in which a server will be mounted. The Android client application will communicate with the server through a wireless network so that the user can control the vehicle remotely.

Motivation

Considering the introduction, the evolution of hardware and communications technologies, as well as the high availability of mobile devices, it has facilitated the remote control of any action that one wishes to perform on another device located at a distance.

The technological advances allow to design and implement the remote control of the scale vehicle reducing the workload and the costs derived from the realization of the project. The different forms of hardware implementation and the different possibilities of communication between existing devices provide a real-time, simple and effective development environment for the remote control.

Remotely controlled systems are systems that are expanding especially in sectors such as home automation and control of vehicles such as UAVs or automobiles. The remote control will expand to other sectors and in the future, many of the usual tasks will be performed remotely from our mobile device, so understanding their operation is very useful.

In this way, the main motivation for the completion of this end degree project is the realization of a remotely controlled scale vehicle from an Android device in a simple, economical, efficient and real-time way to understand how this type of distributed systems are designed and composed and to generate an open work to future extensions.

Objectives

The objective of this project carried out as an end grade project is the design and implementation of a remotely controlled scale vehicle from an Android application as briefly explained in the previous points.

The project intends to design and implement functionalities like a car so that the actions can be controlled from the application in a simple and intuitive way for the users. In addition, it is intended to introduce safety conditions with the aim of ensuring the integrity and protection of the scale vehicle independently of the user's decisions, such as distance sensors to detect obstacles.

Another aspect that is intended to be introduced is the control of some aspects through the sensors of the user's mobile device in order that the controls of the vehicle are assimilated to a steering wheel or the controls of any vehicle. Another objective related to this topic would be to facilitate its future integration in virtual reality controls, a field that is currently expanding.

Through this document it is intended to illustrate the phases developed in the realization of the project to reach the previously presented objectives and to explain the development of the prototype of scale vehicle.

Document

This section indicates how this document has been structured to explain the implementation of the project. The structure followed is like that indicated in the methodology Metric Version 3 [8], proposed by the Ministry of Finance and Public Administration of the Government of Spain for the systematization of the activities that support the software life cycle.

State of the art

This section performs a first approach and a field analysis of the remote control of hardware elements, context in which this project is framed. Subsequently, some of the existing products and the tools and technologies available and used will be analysed.

To better understand of the project development, it is necessary to carry out a preliminary analysis of the remote-control technology. Traditionally the remote control has referred to the devices that through the transmission of infrared waves allow to send control information to the machine or main system located to a certain distance with the objective of facilitating the comfort of the users. The most popular traditional remote-control elements are television controls, DVD or music equipment [9].

This way users can dictate commands through actions such as pressing buttons or moving the controls to operate the devices completely remotely without the control device and the remote device being connected by cables. This capability makes these systems comfortable and accessible [10].

But today, remote control within systems has acquired great importance in fields such as home automation or automotive and in devices of daily living. This is due to the advancement of telecommunications and technologies used to link remote devices together with the constant increase in the use of mobile devices, as discussed in the introduction. The rise of remote control is also closely related to the importance of distributed systems.

Distributed systems whose hardware and software components are in separate components connected in network and communicate/coordinate the actions through a pre-established protocol. These types of systems are very important because of Internet, mobile networks and local area networks that allow remote access to other devices to control it from any geographical point with an Internet connection. The current popularity of distributed systems is due to the development of more powerful, small and cheap microprocessors linked to the development of communications and wireless networks.

System analysis

In this section the remotely controlled scale vehicle system analysis will be performed. The analysis will be carried out following the structure proposed in Metric Version 3 [8], which consists in obtaining a specification of the system to be built by capturing the needs and problems to solve them and to model the problem. This point will be the basis of the system design to be carried out later.

The system to be developed consists of a robotically vehicle controlled remotely through an Android application in a device connected to the same Wi-Fi network. Control over the vehicle will mainly be to execute movements safely and, secondarily, to perform other actions available to vehicles, such as lights or horn.

The complete functionality of the system will be defined throughout this document, which is directed first to the project manager (Carlos González Hernández) and the project supervisor (Javier Fernández Muñoz). Secondly, it is directed to all those interested in the development of the prototype resulting from the realization of this project.

System design

The architecture selected to achieve the objectives of the system is the Client-Server model because it is the most appropriate. It defines an organization where the processes work on separate machines, i.e. the process Client (the Android application of the system) consumes the Server (the vehicle) services and they interact by a pre-established message passing mechanism called Protocol [22].

The Client is the subsystem that has the process in charge of claiming the services of the Server through requests and the Server is the subsystem that contains the process that provides the services to the Client based on the requests made and responds by returning to the Clients the obtained result. Message passing [22] is the mechanism necessary for the interaction support that allows a Client to obtain a service from a Server. The use of a message passing protocol allows connecting Clients and Servers implemented in different platforms and with different programming language.

Implementation

In this section, the implementation of the System is explained with very little depth, since the design carried out in section 4 has been followed in detail. The main objective of this section is to present the technologies used in the implementation of the system and the result of the system.

Test

This section aims to describe the process of verification that after the implementation of the system have been included and developed all the functionalities defined in section 3. To perform this test battery the hardware defined in 4.4.1 has been used with the help of the simulated Android device that offers Android Studio.

Temporary Planning

In this section the planning of the activities and tasks performed for system development is done. The technique used for planning is the representation of Gantt diagrams, a tool to plan and schedule tasks over a given period that allows visualizing the activities to be carried out, the interdependence between them and their planning in the time of the project [39].

Budget

The project with title***: Design and implementation of a remotely controlled scale vehicle***, with a duration of **4 months and 22 days** (including non-working days), has a total budget for its development of **22,778.81 € (TWENTY TWO THOUSAND SEVEN HUNDRED SEVENTY EIGHT AND EIGHTY ONE EUROS** (IVA included).

Regulatory and ethical framework

This point explains the legal framework in everything related to the development of this project as are the standards used and the licenses of the technologies used: Android Studio, IDE Arduino, Git and BitBucket.

Future lines of research

The scope of remote control is now becoming very important as it facilitates normal tasks. That is why this project is not expected to end here as a final product, but is intended to serve as a basis for the development of larger projects with the introduction of new functionalities. Some functionalities that could be introduced are:

1. The first improvement that could be made is to replace the ultrasonic obstacle sensors with infrared sensors that have a higher reliability.
2. You could add a camera that would retransmit live the road that the vehicle is following. This way, through Wi-Fi technology, you can control the vehicle from any device that is at any point with an internet connection. This new functionality may require a change of controller and chassis to be able to introduce the camera on the vehicle and even allow it to rotate.
3. Depending on the purpose of the system, other instruments could be added, such as a mechanical arm that would allow actions such as picking, holding, cutting or drilling. This functionality could interest in fields like the defence to develop systems of deactivation of explosives.
4. Also depending on the terrain, you could think about modifying the vehicle chassis to improve its handling and you could think about implementing the system for more variety of vehicles such as UAV.
5. The maximum development point would be to improve the project that Telefónica and Ericsson presented at the Mobile World Congress in 2017 [14].

Conclusions

The choice of this project as end degree work was made especially for the interest of knowing the assembly and operation of a distributed system to remotely control a scale vehicle. Investigating the field in which the project is framed, there are many similar examples that have already explored the remote control of scale vehicles implemented in Arduino and managed from Android.

Unlike these examples, this project has tried to increase the complexity and use a Wi-Fi network instead of Bluetooth. The use of Bluetooth technology is the most widespread in this area due to the increased knowledge of the use of Arduino Bluetooth modules with respect to the Wi-Fi module and Bluetooth is easier to integrate with Android. But the range of the Bluetooth is very small and does not allow the connection from other external networks as it could allow a Wi-Fi network well configured so that the vehicle could be controlled from anywhere in the world with Internet.

Another complexity that was decided to introduce with respect to the existing examples is the control of the rotation through the rotation of the mobile device. This capacity has had a result that does not reach the expectations expected in comparison with the difficulty of design that supposed.

As for the prototype generated, it is now a toy that has served as a basis for understanding the development of a system of these characteristics and serve as a basis for future developments as indicated above. So, the final product of the project currently has the sole function of leisure and does not make a great contribution to society. As for the economic aspect, its value may be a bit excessive for a single vehicle, but now that there is a design and a prototype you could think of polishing some details or include functionality to perform a series production and obtain greater economic benefits.

Personal conclusions

The personal conclusion obtained from the completion of this final project is the illusion of carrying out a project in which you decide how each task and design is done. I have also been amazed at the ability I have acquired to apply the knowledge and experience gained over the years.

The selection of the project of a remotely controlled scale vehicle was due to the fact that when I saw it I was very attracted to how a project of this type would be carried out, since when I was little I wondered how I could move a toy car from a remote. Now that I have learned how to build complete systems and have gone through this experience of developing a real and functional system I can see that when the work is fruitful it is very rewarding.

Although during the years of the degree and development of this project I have encountered problems that have cost me to solve, hard work has allowed me to reach this point: conclusions. Now I see that with the delivery of this work I find myself closer to finishing this stage so good that started 4 years ago.