UNIVERSIDAD DEL BÍO-BÍO
FACULTAD DE CIENCIAS EMPRESARIALES
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN Y TECNOLOGÍAS DE LA INFORMACIÓN

"INTEGRACIÓN DE UNA PLATAFORMA SMARTHPHONE A ENTORNOS DE ROBÓTICA MÓVIL"

JORGE ARCADIO CARVAJAL OCARES
GERSON AARON MORA CID

Memoria para optar al título de
Ingeniero Civil en Informática

CHILLÁN, 2011

UNIVERSIDAD DEL BÍO-BÍO
FACULTAD DE CIENCIAS EMPRESARIALES
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN Y TECNOLOGÍAS DE LA INFORMACIÓN

"INTEGRACIÓN DE UNA PLATAFORMA SMARTHPHONE A ENTORNOS DE ROBÓTICA MÓVIL"

JORGE ARCADIO CARVAJAL OCARES
GERSON AARON MORA CID

PROFESOR GUÍA : SR. LUIS GAJARDO DÍAZ
PROFESOR INFORMANTE : SRTA MARÍA SOTO CHICO
NOTA FINAL PROYECTO DE TITULO : _____

Memoria para optar al título de
Ingeniero Civil en Informática

CHILLÁN, 2011

Agradecimientos

Jorge Arcadio Carvajal Ocares

Quiero dar las gracias a mi familia, la cual me ha dado la oportunidad de estudiar y ha sido un apoyo en todo momento en mi vida, sin ellos no sería nada. Gracias Celia, Domingo, Diego, Olga, Cesar y Cecilia por soportar momentos de mal humor, consentirme y estar siempre que lo necesite.

A mi abuela que me cuida y guía desde el cielo, a mi polola María Laudelina por ser mi amiga, confidente, un apoyo fundamental, en resumen el amor de mi vida.

No puedo dejar de agradecer a todas esas personas que nos ayudaron sin dudarlo, en especial a Miriam Abarzua y el señor Luis Merino. A mi compañero por perdonar enojos y apoyarme.

Gerson Aaron Mora Cid

En primer lugar agradecer a Dios por darme la oportunidad de estar aquí y haberme dado las herramientas para conseguir mis objetivos. A mis padres por el apoyo incondicional y la paciencia. A mis compañeros y amigos que de una u otra forma me alentaron en las dificultades, así como también, a las personas que me dieron palabras de apoyo cuando las necesité. Por último agradecer a mi compañero de memoria por soportarme tanto tiempo.

Dedicatoria

Jorge Arcadio Carvajal Ocares

A mi Familia que se esfuerza cada día porque lo tengamos todos y crezcamos como personas, a mi abuela que me quiso ver terminar y lamentablemente no pudo, para ellos es este proyecto.

Gerson Aaron Mora Cid

Este proyecto está dedicado a las personas que me apoyaron de cualquier forma, a todos los que se interesaron en nuestro trabajo, a los que me dieron palabras de apoyo cuando las necesite, en especial a mis amigos y a mi familia que estuvieron junto a mí en todo momento y fueron parte importante tanto en mi proyecto como en lo personal.

Resumen

El presente informe detalla el desarrollo de un sistema orientado a la integración de una plataforma Smartphone en entornos de robótica móvil. Dicho desarrollo consiste en la creación de un sistema de monitoreo y control de dispositivos robóticos, los cuales operan mediante localizaciones geográficas y sensores que permiten la navegación del robot.

La idea principal de esta aplicación es innovar en el ámbito de la robótica utilizando tecnologías modernas y de gran versatilidad, como lo son los Smartphone's, que constituyen una gran opción en el campo del desarrollo dada su gran capacidad de cómputo y sensores. A lo anterior, se añaden la tecnología Android y tecnologías web como Struts 2, Ajax y Google Maps, las cuales son de acceso gratuito, aumentando el abanico de alternativas en la construcción del sistema.

En este proyecto se utiliza una metodología de desarrollo iterativa incremental. Dadas las características de ésta, se utilizó el lenguaje de modelado UML por ser orientación a objetos y amplio uso. Asimismo, se aplicó el patrón de arquitectura Modelo Vista Controlador (MVC) y el patrón de diseño Singleton.

Gracias a lo anteriormente señalado, se ha logrado la construcción de un vehículo robótico capaz de seguir una ruta de coordenadas geográficas y ordenar objetos mediante la identificación de colores por medio de un Smartphone, que coordina las acciones del vehículo mediante comunicación inalámbrica (Bluetooth y WIFI). Para el monitoreo y control del robot se ha creado un sistema web, el cual incorpora el uso de mapas para la visualización de los movimientos del robot, así como también para envío de las posiciones geográficas a visitar mediante la selección de rutas.

En definitiva, podemos mencionar que el proyecto permite sentar un precedente e innovar utilizando Smartphone's en ámbitos para los cuales no fueron específicamente creados, como lo es la Robótica Móvil, lo cual, dicho sea de paso, permitirá aprovechar el potencial de estos dispositivos a otras áreas y utilizar nuevas tecnologías.

Contenido

Índice de tablas

Índice de figuras

Capítulo I

Introducción

Para comprender de qué trata un proyecto es necesario explicar detalladamente algunos aspectos importantes que rodean al problema que se desea solucionar.

Este capítulo interiorizará al lector con los objetivos del proyecto, destacando descripciones de la situación actual de los Smartphone's en Chile y la carencia de desarrollo en esta área.

Adicionalmente hemos incluido los objetivos generales y específicos que tiene el Proyecto a realizar, trabajos realizados con anterioridad por otras personas o instituciones y la metodología seleccionada para el desarrollo del mismo.

Hoy en día nos encontramos en un mundo donde cada vez la tecnología cobra una importancia más relevante, debido a que cada elemento con el cual interactuamos, de una u otra forma está relacionada con ésta. La mayor parte de las actividades que realizamos cotidianamente las realizamos en un ambiente en el cual se encuentran presentes o han interactuado elementos tecnológicos, es por ello que el prescindir de éstos es improbable e incluso, cada vez se potencia el uso de la tecnología hasta en los objetos más inimaginables.

Sin embargo, existen elementos tecnológicos preferidos y que permiten interactuar con nuestros similares, éste es el caso de los teléfonos móviles. Estos dispositivos han evolucionado rápidamente, lo cual, sumado a su capacidad de comunicarse inalámbricamente en cualquier lugar y contexto, han provocado que prácticamente cada persona en el mundo cuente con uno de estos dispositivos (Chile es un claro ejemplo). El uso de estos dispositivos se ha diversificado debido a la amplia gama de actividades que actualmente permiten realizar, tales como: navegación en Internet, reproducción musical, bluetooth, captura de imágenes, etc. Estos elementos y los nuevos avances en esta área han creado una nueva generación de dispositivos móviles denominados SmartPhone o teléfonos inteligentes. Estos dispositivos de tercera generación suman a las capacidades antes mencionadas elementos tales como acelerómetros, compás digital, sensores de iluminación, sistemas operativos especializados y una capacidad de cómputo que asombra.

Diversas áreas de estudio, tal como las personas, han centrado sus miradas en estos dispositivos, como es el caso de la robótica, disciplina compleja y de un alto costo, que ve con buenos ojos el uso de elementos de uso cotidiano, que permiten la resolución de problemas de manera sencilla y con un bajo costo, donde se incorpora en un solo dispositivo una amplia gama de herramientas, las cuales difícilmente se pueden encontrar en otros dispositivos.

Es bajo éste concepto que se desea aprovechar las ventajas antes expuestas, aplicando a la robótica los beneficios y opciones que entregan estos dispositivos.

1.1 Descripción del problema

Según el INE (Instituto Nacional de Estadísticas), hasta mayo del 2009 el número de teléfonos móviles en Chile alcanzó las 15.669.000 unidades y el número de la población estimada a junio de 2009 es de 16.928.873 habitantes, lo que significa que existe casi un teléfono móvil por cada habitante. Más aún, con la reciente llegada de los Smartphone's, es inevitable que cada persona que hasta ahora tiene equipos móviles convencionales migre a equipos con estas tecnologías, abriendo un nuevo mercado a los programadores, ya que estos dispositivos han provocado el aumento de plataformas de desarrollo, con un sin fin de posibilidades para crear nuevas e interesantes aplicaciones. Estas posibilidades son aplicables a diversas áreas como medicina, minería, ramas militares, entre otras, por su gran capacidad de cómputo, confiabilidad, uso y costo. Este último aspecto es sumamente importante porque se cuenta con un dispositivo de un precio abordable, de gran calidad y con una enorme variedad de herramientas (acelerómetros, compás digital, cámara digital, entre otros).

Por otra parte, la robótica ha debido evolucionar como respuesta al mundo dinámico en el cual estamos inmersos. Estos cambios constantes y nuevos retos han propuesto a la robótica un abanico de problemas y necesidades a cubrir. Es por esto que los avances de la tecnología de dispositivos móviles son una herramienta en la cual basarnos para idear innovadoras y factibles soluciones. Sin embargo, en Chile, el desarrollo en el área de la robótica así como en Smartphone's, es escaso. Esto se debe a la reciente llegada y rápida masificación de los Smartphone's en el país y a los costos de desarrollo en el área de la robótica.

En el último tiempo, el costo de los Smartphone's ha disminuido como consecuencia de la irrupción en el mercado de nuevas tecnologías de bajo costo, tales como Android, hecho que ha provocado que los Smartphone's se apliquen a diversas actividades, dado su potencia de cómputo, gran cantidad de sensores, herramientas, y capacidad de ejecutar aplicaciones personalizadas.

Es por esto que el presente proyecto pretende aumentar el desarrollo en el área de los dispositivos móviles de última generación (SmartPhone) e integrarlo a la robótica, de manera de otorgar las bases para posteriores investigaciones y disminuir los costos del desarrollo en la robótica, a modo de fomentar la innovación en estas áreas que tienen un futuro prometedor y que, actualmente, ven truncado su crecimiento por los altos costos que implica y escasa documentación de proyectos similares abordados en estas áreas.

1.2 Objetivos del proyecto

1.2.1 Objetivo general

Desarrollar robots autónomos utilizando Smartphone's, para la resolución de problemas de reconocimiento y localización.

1.2.2 Objetivo específicos

- Construir robots autónomos que cuenten con sensores de aceleración, orientación, visión, localización y proximidad.
- Diseñar un sistema de localización de objetos simple mediante parámetros como color, tamaño, entre otros y control de robots sobre plataforma Smartphone.
- Integrar el software de control entre el Smartphone y el Robot.
- Diseñar un software de monitoreo del status de cada robot en tiempo real, sobre plataforma web.

1.3 Metodología de trabajo utilizada.

El presente proyecto se desarrolla tomando como metodología de trabajo una metodología incremental, específicamente la metodología de desarrollo Iterativa Incremental, la cual permite establecer etapas que son denominadas incrementos, los cuales, a su vez, son de tipo funcional. Por lo tanto, cada incremento es una instancia de análisis, corrección de errores, evolución y avance.

La metodología está orientada a objetos en lo que concierne a procedimientos internos y estructurados en cuanto a las interfaces de usuario.

En este proyecto se utiliza el patrón de arquitectura Modelo Vista Controlador, también conocido como MVC y el patrón de diseño Singleton. Además, se utiliza como lenguaje de desarrollo el lenguaje con orientación a objetos JAVA, dado que es utilizable tanto en dispositivos móviles como en sistemas Web, además de que es un lenguaje robusto, seguro, multiproceso, independiente de la plataforma, y que permite la creación de programas modulares y reutilizables. Se utilizara a UML (Lenguaje Unificado de Modelado) para el modelado mediante diagramas.

Se utilizará un micro-controlador Arduino para que interactúe con los sensores y actuadores del robot; este micro-controlador se programará en lenguaje Processing.

Para la programación se utilizan diversos entornos de desarrollo (IDE), los cuales son de acceso gratuito, que cuentan con gran cantidad de plugin's para la utilización de diferentes tecnologías. Los IDE's utilizados son Netbeans, Eclipse y Arduino Alpha, para el desarrollo del sistema de monitoreo y control Web, aplicación Android y software embebido en micro-controlador Arduino, respectivamente.

1.4 Trabajos relacionados

1.4.1 Trabajos relacionados en la Universidad del Bío-Bío

En la Universidad del Bío-Bío existen proyectos de título creados por alumnos memoristas de diferentes carreras relacionadas al tema del presente informe, dentro de las cuales podemos mencionar las carreras de Ingeniería de Ejecución en Computación e Informática e Ingeniería Civil en Informática, los cuales se encuentran disponibles en las bibliotecas de la dependencia. De los proyectos se presenta una pequeña selección la cual está relacionada de una u otra manera al presente desarrollo.

Nombre de la memoria de título: Desarrollo de agente inteligente móvil, guiado a través de coordenadas geográficas y análisis de su ambiente.
Autor (es): Juan Carlos Figueroa, Sebastián Moraga.
Año: 2006.

Memoria consistente en la creación de un agente móvil capaz de ubicarse mediante coordenadas entregadas por un GPS, éste agente móvil es capaz de escanear el entorno en el cual está inmerso, los datos recogidos de éste análisis son procesados en un software especialmente creado, éste software se ejecuta en un computador, el cual se encuentra a bordo del agente móvil. Para éste proyecto se utilizó la tecnología GPS, además de Micro-controladores, sensores y actuadores de la línea Basic Stamps.

Nombre de la memoria de título: Desarrollo de aplicación móvil sobre plataforma Android en apoyo a visitas médicas
Autor (es): Christopher Arévalo, Verónica Ramírez Norambuena.
Año: 2010.

Memoria consistente en la creación de un sistema sobre la plataforma Android para el apoyo en las visitas médicas, en el cual se puede obtener las fichas de los pacientes, buscar la dirección de un paciente, así como también controlar y coordinar las visitas médicas. Éste sistema ha sido desarrollado sobre Android, por lo cual es utilizable en dispositivos móviles que soporten éste sistema operativo.

1.4.2 Trabajos relacionados fuera del ámbito académico

Nombre de la memoria de título: Estudio de la plataforma de software Android para el desarrollo de una aplicación social

Autor (es): Iván Pérez Rodríguez.

Año: 2009.

El proyecto tiene como finalidad estudiar el funcionamiento y las posibilidades que ofrece el sistema operativo móvil Android comenzando por un enfoque analítico de sus características hasta llegar a la realización de una aplicación que ejemplifique sus funcionalidades básicas.

Nombre de la memoria de título: Diseño y desarrollo de un sistema de posicionamiento en interiores basado en WIFI con tecnología Android.

Autor (es): Iván Pérez Rodríguez.

Año: 2009.

El resultado de este trabajo es un sistema, formado por dos aplicaciones capaces de estimar la posición de un individuo en un entorno cerrado con cobertura WIFI y mostrarla en un dispositivo gestionado por un sistema operativo Android. Además de la estimación de la posición, éste sistema puede crear un mapa de potencias, necesario para conseguir una correcta estimación de la posición. Ambas aplicaciones están preparadas para funcionar en cualquier edificio y en cualquier dispositivo que posea tecnología Android. También se ha buscado que el sistema posea una interfaz sencilla con el propósito de que pueda ser usado por cualquier persona independientemente de su nivel de conocimiento tecnológico.

1.5 Estructura del informe

El presente informe se ha organizado en capítulos, los cuales permitirán ir comprendiendo gradualmente el desarrollo del proyecto. La estructura del informe se puede dividir en 5 partes, las cuales se presentan a continuación.

La primera parte consiste en un grupo de capítulos que tienen como finalidad informar al lector de los detalles que propiciaron este proyecto, así como también contenidos y conceptos necesarios para comprender el desarrollo. Esta primera parte está compuesta por el *Capítulo II: Introducción a la robótica*; *Capítulo III: Tecnologías para dispositivos móviles*; *Capítulo IV: El sistema operativo Android*, y por último *Capítulo V: Tecnologías para el desarrollo Web*.

La segunda parte corresponde al *Capítulo VI: Desarrollo de la aplicación nativa sobre plataforma Android y creación del robot,* este capítulo presenta la solución al problema, el análisis de requerimientos para la construcción del robot, para la creación del software embebido en el micro-controlador Arduino y para la aplicación Android.

La tercera parte se ve representada en el *Capítulo VII: Implementación del robot y aplicación nativa Android,* este capítulo presenta los aspectos relevantes en el desarrollo tanto del robot como de la aplicación Android.

La cuarta parte que compone éste informe corresponde al *Capítulo VIII: Desarrollo e implementación del sistema web de monitoreo y control.* Este capítulo presenta el análisis, diseño e implementación del sistema de monitoreo y control, donde se destaca el uso de nuevas tecnologías.

Finalmente, la quinta parte corresponde a las *Conclusiónes y Trabajos Futuros.*

Capítulo II

Introducción a la Robótica

Hoy en día la robótica es fundamental para algunos procesos, los cuales eran impensables que pudiesen ser realizados por una máquina. Un ejemplo claro de esto son los brazos robóticos que apoyan las líneas de producción o realizan operaciones humanas con una precisión asombrosa.

Es por ello que la robótica hoy en día no es solo asunto de unos pocos; hoy la robótica está presente en todo lugar directa o indirectamente. Sin embargo en la actualidad la robótica es una disciplina costosa, hecho que se desea solucionar con la aplicación de nuevas tecnologías como son los SmartPhone a esta área.

Es así como a continuación se entrega una introducción a la robótica de manera que el lector se interiorice en esta temática y comprenda de mejor forma el contenido del proyecto. En los contenidos del capítulo destaca una introducción breve a la robótica, micro-controlador robótico y su arquitectura, sensores y actuadores.

"Es una disciplina que combina todas aquellas actividades relacionados con el estudio, diseño, construcción, operación y mantención de robots. Es un campo de trabajo que combina diferentes disciplinas como Ingeniería Eléctrica, Ingeniería Electrónica, Ingeniería Mecánica, Ciencias de la computación, Matemáticas, Física, Biología, Neurociencias, etc" [1]. Ya sea que se trate de un robot sencillo que se acerca hacia elementos luminosos o de un robot con un sofisticado sistema de sonares, el diseño e implementación siempre requiere, en mayor o menor medida, de la aplicación de conocimientos interdisciplinarios. Esto hace que los robots constituyan una herramienta ideal para cursos de ingeniería, en los que permiten la puesta en práctica de conceptos y teorías en forma motivadora e integral. Varios han resaltado las posibilidades y el potencial de los robots móviles en éste sentido, entre ellos [2].

Existen diferentes tipos de robot, pero la clasificación más frecuente se basa en su arquitectura, la cual se detalla a continuación [1].

- **Androides:** Este tipo de robot imita tanto la morfología del ser humano como su comportamiento, lo cual no solo se puede observar en su forma corporal, sino también en versiones de busto que representan las emociones humanas.
- **Zoomórficos:** Son aquellos robots que en su desplazamiento buscan imitar a los animales, existen diferentes tipos de robots zoomórficos entre los cuales destacan los siguientes tipos: voladores, acuáticos y terrestres.
- **Móviles:** Son aquellos que su método de desplazamiento es por medio de ruedas o equivalentes, son muy similares al desplazamiento de los vehículos que conocemos (autos o Tanques).
- **Poli-articulados:** Estos robots por lo general consisten en brazos mecánicos que se utilizan para la automatización de procesos en industrias, siendo utilizados para labores como: pintar, desplazar y selección de elementos.

2.1 Micro-controladores

"Son circuitos integrados que contienen gran parte de las cualidades de una computadora de escritorio, tales como: CPU y memoria. Sin embargo, no incluyen ningún dispositivo de comunicación con humanos, como lo son el monitor, teclado o mouse. Los micro-controladores son diseñados para el control de máquinas, más que para interactuar con humanos" [3].

Los micro-controladores se encuentran en innumerables sistemas presentes en la vida diaria, como juguetes, hornos de microondas, frigoríficos, televisores, computadoras, impresoras, automóviles, y otros más especializados y menos comunes, sobre todo de uso industrial como instalaciones eléctricas, controladores de sistemas de naves, etc. Los micro-controladores son el cerebro de la robótica actual, sobre ellos se incorporan sensores y actuadores.

2.1.1 Arquitectura interna del micro-controlador

Existen dos arquitecturas conocidas: la arquitectura clásica de Von Neumann y la arquitectura de Harvard. A continuación se describen ambas arquitecturas.

Arquitectura Von Neumann: Dispone de solo una memoria principal donde se almacenan datos e instrucciones de forma indistinta. A dicha memoria se accede a través de un sistema de buses único (direcciones, datos y control). Para facilitar la comprensión se presenta un esquema de esta arquitectura en la Figura 2.1

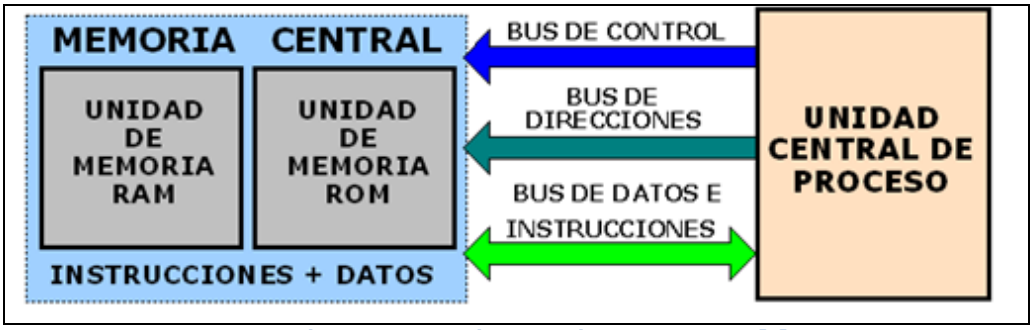


Figura 2.1: Arquitectura de Von Neuman [4]

Arquitectura de Harvard: Dispone de dos memorias independientes, una que contiene sólo instrucciones, y otra que contiene sólo datos. Ambas disponen de sus respectivos sistemas de buses de acceso y es posible realizar operaciones de acceso (lectura o escritura) simultáneamente en ambas memorias. Lo anteriormente descrito se presenta en la Figura 2.2.

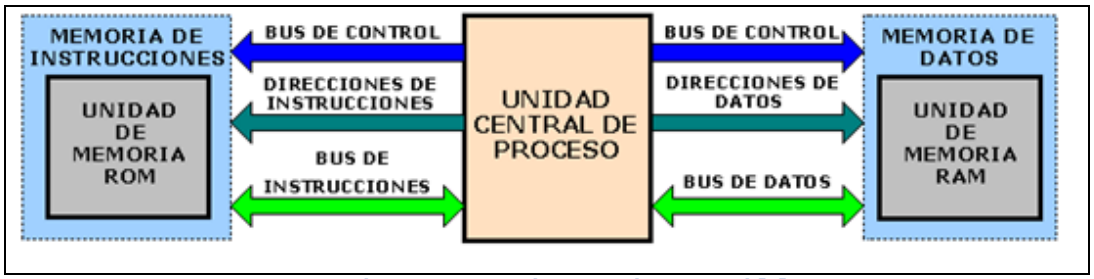


Figura 2.2: Arquitectura de Harvard [4].

2.2 Sensores

"Permiten la medición directa o indirecta del estado de una(s) variable(s) o proceso complejo agregado de N conjuntos de variables y puede convertir una variable de salida en otra que pueda ser más adecuada para un fin específico. Las variables pueden ser físicas (mecánicas, neumáticas, eléctricas, electrónicas) y lógicas (bits, bytes, registros, tablas, archivos, entre otras)" [3].

La variedad de sensores existentes actualmente es inimaginable, siendo los más conocidos los sensores de distancia, presión, luz, de masa de aire, entre otros. Por lo cual no es posible definir la estructura de éstos ni su comportamiento, ya que cada uno se comporta y utiliza de manera diferente [3].

2.3 Actuadores

"Transforma una secuencia de símbolos de un dispositivo determinado como órdenes a otro dispositivo" [3]

Los actuadores son parte importante en el proceso de construcción y ejecución de cualquier elemento que se base en la interacción con el medio. La no utilización de actuadores limita la ejecución de tareas en diversas actividades las cuales incorporan movimientos milimétricamente precisos y continuos, entre los cuales destacan las operaciones médicas de alta precisión y líneas de producción en el campo automotriz, entre otras. Los actuadores se pueden clasificar en: neumáticos, hidráulicos o eléctrico-electrónicos

2.3.1 Mecánica aplicada a la robótica.

"La mecánica es una teoría científica que estudia el movimiento de los cuerpos y sus causas, o bien el equilibrio, es decir, la falta de movimiento" [5].

Teniendo estos conceptos podemos decir que la implicancia de la mecánica en la robótica es amplia en cuanto al apoyo de la teoría del movimiento, potenciando así la robótica móvil como nueva área de interacción. Esto debido a que grandes avances tecnológicos en diferentes áreas se han visto influidos por la robótica móvil, como los robots exploradores, ya sean en campos minados, terrenos desconocidos (de poco acceso para el común de las personas), o lugares donde las condiciones climáticas o atmosféricas no permiten el actuar de las personas (exploración de otros planetas) [5].

Lo antes mencionado es aplicable al proyecto ya que es necesario que el robot cuente con la condición de vehículo móvil para el cumplimiento de su objetivo.

2.3.2 La plataforma Arduino

"Arduino es una plataforma de hardware de código y arquitectura abierta, basada en una sencilla placa con entradas y salidas, analógicas y digitales, en un entorno de desarrollo que está basado en el lenguaje de programación Processing. Esta plataforma que conecta el mundo físico con el mundo virtual, o el mundo analógico con el digital" [6].

En la Figura 2.3 se presenta la distribución que posee el micro-controlador Arduino.

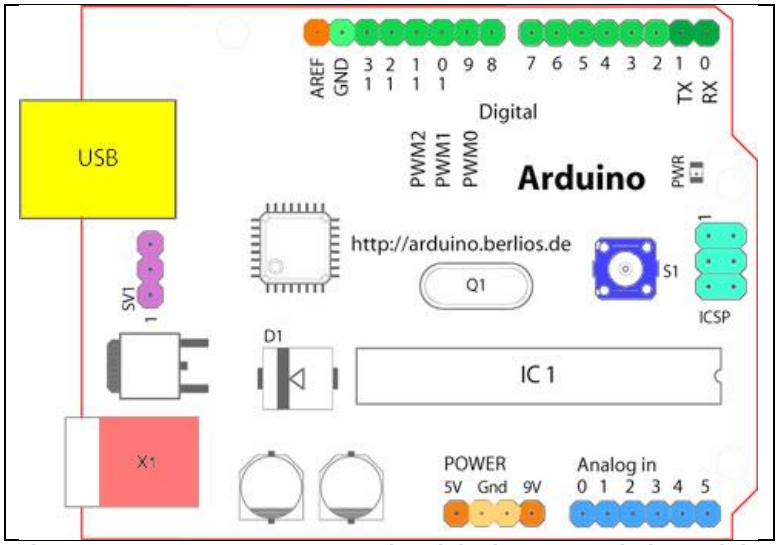


Figura 2.3: Esquema representativo del micro-controlador Arduino [6]

Sus creadores son el zaragozano David Cuartielles, ingeniero electrónico y docente de la Universidad de Mälmo, Suecia y Massimo Banzi, italiano, diseñador y desarrollador Web. El proyecto fue concebido en Italia en el año 2005 [6].

Capítulo III

Tecnologías para dispositivos móviles

La telefonía celular ha crecido a pasos agigantados durante los últimos años. Son miles las personas que cuentan con dispositivos móviles y hacen uso de los servicios ofrecidos por estos.

Para funcionar correctamente estos dispositivos deben encontrarse insertos en un ecosistema tecnológico, el cual se compone de servicios, operadores, redes y plataformas.

En éste capítulo se explica dicho ecosistema y la forma en que se relacionan, de manera de ofrecer un servicio de calidad.

3.1 La telefonía móvil en Chile

Hace 25 años la telefonía en Chile se planteó como iniciativa licitar las frecuencias para abastecer al país de líneas telefónicas. Estas líneas en un comienzo eran exclusivas para empresas que las utilizaban con fines comerciales, siendo inviables para las personas, ya que la capacidad de acceso a dicha tecnología estaba supeditada a la capacidad adquisitiva de las personas debido al alto costo de los equipos y del servicio en sí [7].

En 1988 existían tres empresas que proveían el servicio de telefonía celular, estas empresas eran CTC Celular, Telecom Celular y VTR Celular [8].

"En los noventa la empresa CTC Celular adquiere VTR Celular formando Startel. Esta unión logra posicionar a esta última como la única empresa de telefonía celular chilena que poseía cobertura en todo el territorio nacional" [8] En la década de los 90's el fenómeno de la telefonía celular creció considerablemente aumentando el número de usuarios y disminuyendo el costo del servicio por parte de las compañías proveedoras [8].

Actualmente más de 15.800.000 chilenos poseen un teléfono celular, siendo MoviStar, Entel PCS y Claro los operadores que suscitan la preferencia de los clientes en el territorio nacional [8].

Actualmente la participación de mercado de la telefonía móvil en Chile es del 76% y un 21% de la telefonía fija, lo que indica claramente que la telefonía de red fija está perdiendo cada vez más terreno entre los consumidores, esto según investigaciones del INE (Instituto Nacional de Estadísticas). En cuanto al resto del continente, Chile es considerado uno de los países sudamericanos y a nivel americano con el mayor nivel de penetración de la telefonía móvil [8].

3.2 Operadores de telefonía móvil

Los operadores de telefonía móvil son compañías que ofrecen servicios de telefonía a clientes de teléfonos móviles. En Chile existen tres operadores de telefonía móvil, los cuales son: Movistar, Entel PCS y Claro [8].

Según estadísticas proporcionadas por la Subsecretaría de Telecomunicaciones del Gobierno de Chile, las posiciones de las empresas antes descritas en el mercado de la telefonía móvil es la que se muestra en la Figura 3.1 [8].

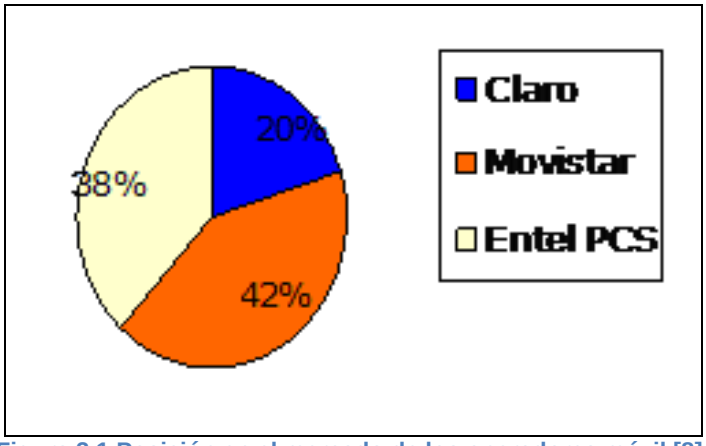


Figura 3.1 Posición en el mercado de los operadores móvil [8].

3.3 Dispositivos móviles

Un dispositivo móvil es cualquier dispositivo que sea lo suficientemente pequeño y cómodo para ser trasladado por una persona, y que cumpla con las siguientes características [8]:

- Tamaño reducido, para facilidad de transporte.
- Ofrecer conexión a Internet, la cual puede ser permanente o intermitente.
- Poseer una memoria limitada.
- Normalmente se asocian al uso individual de una persona, tanto en posesión como en operación, el cual puede ser configurable de acuerdo a las preferencias de cada persona.
- Han sido diseñados específicamente para una función, pero pueden llevar a cabo otras funciones más generales.

Entran en la categoría de dispositivo móvil los siguientes dispositivos [9]:

- **PDA:** Es un computador de mano, el cual originalmente era diseñado como agenda electrónica. Actualmente ofrece servicios similares a los de un PC portátil como por ejemplo: reproducción de películas, creación y modificación de documentos, envío y recepción de correo electrónico, navegar por Internet, etc.
- **Computador portátil:** Es una computadora personal y móvil, son capaces de realizar la mayor parte de las tareas que realizan las computadoras de escritorio, con la ventaja de ser de considerablemente menores en tamaño a estos, más livianas y tienen la capacidad de operar por un período determinado sin estar conectadas a las tomas de electricidad.
- **Teléfono inteligente (SmartPhone):** Es un teléfono con las características de un PC, claro que con capacidades menores. Tienen la característica de que permiten la instalación de programas para incrementar el procesamiento de datos y la conectividad.

3.4 Ecosistema móvil

El ecosistema móvil corresponde al conjunto de operadores, redes, fabricantes de dispositivos y plataformas, las cuales dependen entre ellos para ofrecer servicios de calidad a los usuarios [10].

3.4.1 Operadores de telefonía móvil

Un operador de telefonía Móvil es una compañía telefónica que provee servicios a clientes de teléfonos móviles [10].

Para convertirse en un operador de telefonía móvil en un determinado país, se comienza con la adquisición al gobierno de la licencia de utilización del espectro radioeléctrico. El gobierno correspondiente entregará una parte del espectro según su disponibilidad y el tipo de tecnología móvil que se quiera implementar [10].

Existe otra categoría de operadores de telefonía móvil: los denominados operadores móviles virtuales (OMV), en los cuales para el cliente no existe cambio. Sin embargo, estos operadores no poseen una infraestructura de red, sino que arriendan una parte de esta a otro proveedor [10].

3.5 Tecnologías de telefonía móvil

3.5.1 Primera generación 1G

La primera generación en la telefonía móvil es denominada 1G, la cual hizo su aparición en 1979, a diferencia de los equipos contemporáneos en ese entonces se caracterizaban por ser analógicos y solamente para llamadas por voz. Frecuentemente presentaba problemas de calidad al momento de las llamadas, puesto que la transferencia entre celdas era imprecisa. Los teléfonos móviles tenían baja capacidad y nula seguridad [8].

3.5.2 Segunda generación 2G

La segunda generación hizo su aparición en los años 90, y el avance entre la primera generación y la segunda es el cambio de analógico a digital. En esta generación se utilizan protocolos que permiten mayores velocidades de información para la voz, sin embargo siguen siendo limitados en la comunicación de datos. Se avanzó en servicios como fax y la mensajería de texto [8].

3.5.3 Generación 2.5G

Esta generación como tal no existe, sino que se puede catalogar como perteneciente a esta categoría a teléfonos móviles que incorporen las mejoras de GPRS (General Packet Radio System), HSCSD (High Speed Circuit Switched Data), EDGE (Enhanced Data Rates for Global Evolution), entre otros [8].

3.5.4 Tercera generación 3G

3G es la abreviación o nombre que se le ha asignado a la tercera generación, la cual permite la transmisión de voz y datos a través de telefonía móvil. Sin embargo, el nombre técnico es UTMS (Universal Mobile Telecommunications Service), que tiene como significado "servicio universal de telecomunicaciones móviles" [8].

Los servicios asociados con la tercera generación son: capacidad de transferir voz y datos, es decir, llamada telefónica y video llamadas. A su vez permite el envío y recepción de emails, descarga de archivos y mensajería instantánea. A su vez nace el Internet móvil, con aparatos que incorporan esta capacidad en módems USB (Universal Serial Bus), incluso ha nacido una nueva variedad de computadores portátiles, denominados Netbooks, los cuales incorporan el modem en el equipo requiriendo solo de una tarjeta SIM (Suscriber Identity Module) para su uso [8].

3.5.5 Cuarta generación 4G

Por el momento la cuarta generación se asume como mejoras a la tercera generación, esto dado que no existen definiciones claras de los avances que se incorporarán. Sin embargo se pretende aumentar la velocidad de transmisión en 10 veces [8].

3.6 Red de telefonía móvil pública o PLMN (Public Lan Movile Network)

La red de Telefonía móvil es un sistema de telefonía mediante el cual se permite la combinación de redes de estaciones transmisoras y receptoras de radio, denominadas estaciones base, conectadas a su vez con centrales de conmutación lo cual posibilita la comunicación entre los teléfonos móviles [8].

La Figura 3.2 presenta un esquema de la red de telefonía móvil.

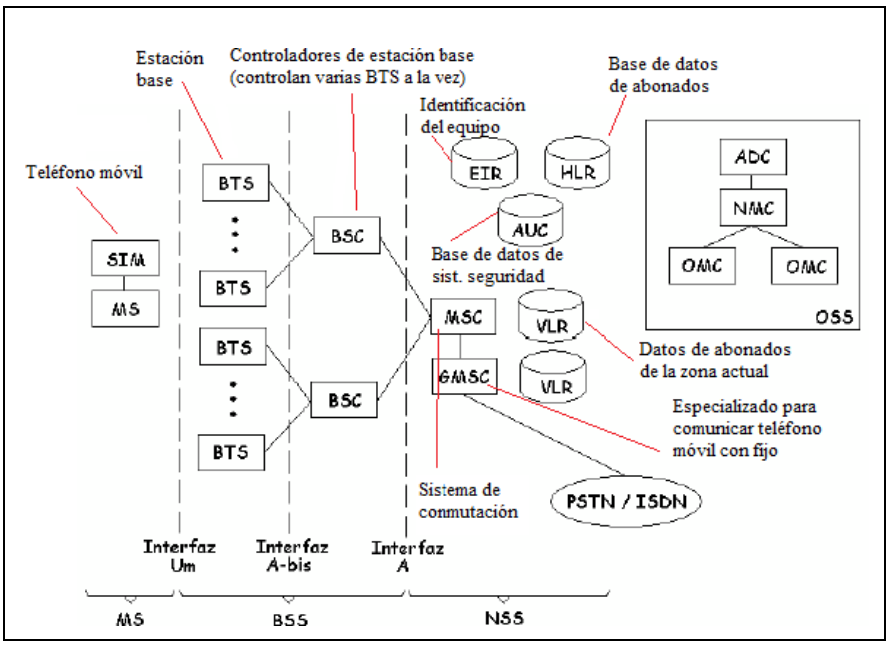


Figura 3.2 Esquema de red de telefonía móvil [8]

Como se puede apreciar en la Figura 3.2 el NSS (Subsistema de red y conmutación) es el centro de procesamiento de la red GSM (Global System for Movile Comunication, se recomienda ver el punto 3.8) y es el responsable de gestionar una comunicación confiable entre la red GSM y las otras redes. Lo más importante es la conexión con la Red Telefónica Pública Conmutada o también llamada PSTN (Public Switched Telephone Network) [8].

Todas las llamadas entre subscriptores, sean éstas originadas hacia el PSTN, originadas en el PSTN y terminadas en un subscriptor móvil, o llamadas entre subscriptores móviles, todas son gestionadas y enrutadas por medio del NSS [8].

3.7 Central de conmutación

Conocida con el acrónimo MSC (Mobile Switching Central), es la encargada de iniciar, terminar y canalizar las llamadas a través del BSC y la BS. Cada MSC está conectado a los BSCs de su área de influencia, pero también a su Visitor Location Register o registro de ubicación de visitante, de ahora en adelante (VLR), y debe tener acceso a los Home Location Register, o registro de ubicación base, de ahora en adelante (HLR) de los distintos operadores e interconexión con las redes de telefonía de otros operadores [11].

3.7.1 Registros de ubicación base (HLR) y de visitante (VLR).

El HLR es una base de datos que almacena la posición del usuario dentro de la red, si está conectado o no y las características de su abono (servicios que puede y no puede usar y tipo de terminal). Es de carácter más bien permanente, ya que cada número de teléfono móvil está adscrito a un HLR determinado y único, que administra su operador móvil [11].

Al recibir una llamada, el MSC pregunta al HLR correspondiente si el número al cual se llama está disponible y dónde está, de estar disponible posibilita y canaliza la llamada o bien, entrega un mensaje de error [11].

El VLR por su parte, es una base de datos volátil que almacena, para el área cubierta por un MSC, los identificativos, permisos, tipos de abono y localizaciones en la red de todos los usuarios activos en ese momento y en ese tramo de la red. El VLR ante la necesidad de realizar una llamada por parte del usuario se contacta con el HLR para determinar si puede o no hacer llamadas según su abono [11]

3.8 Sistema GSM (Global System for Mobile communications)

La red GSM o Sistema global de comunicación móviles, es el estándar más utilizado en el mundo, es denominado estándar de segunda generación (2G) puesto que, a diferencia de la primera generación, las comunicaciones se realizan completamente de manera digital [11].

En 1982 fue estandarizado por primera vez, tomando el nombre de Groupe Spécial Mobile, luego en 1991 adoptó su actual nombre "Sistema global de Comunicaciones Móviles" [11].

El estándar GSM, utiliza bandas de frecuencia las cuales pudiesen ser 850MHz, 900 MHz, 1800 MHz y 1900 MHz. Sin embargo, en ciertos países existen bandas especiales, como es el caso de Estados Unidos que, además de estas bandas, utiliza la banda de 1700 MHz [11].

3.8.1 La red celular

Las redes de telefonía móvil basan su concepto en celdas, es decir, zonas circulares que se superponen para cubrir una determinada área geográfica como se aprecia en la Figura 3.3 [11].

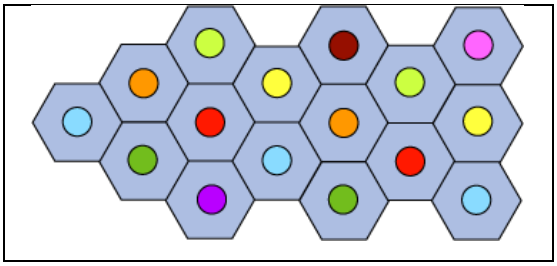


Figura 3.3 Esquema de la red Celular de telefonía móvil [11].

Las redes celulares se basan en el uso de un transmisor-receptor central en cada celda, denominado "estación base" (o Estación base transceptora, BTS) [11].

Cuanto menor sea el radio de una celda, mayor será el ancho de banda. Por lo tanto, en las zonas rurales el radio de las celdas es mayor que las de zonas urbanas por lo cual la cobertura proporcionada es menor [11].

En una red celular, cada celda está rodeada por 6 celdas contiguas (por esto las celdas generalmente se dibujan como un hexágono). Para evitar interferencia, las celdas adyacentes no pueden usar la misma frecuencia. En la práctica, dos celdas que usan el mismo rango de frecuencia deben estar separadas por una distancia equivalente a dos o tres veces el diámetro de la celda [11].

3.8.2 Arquitectura de la red GSM

En una red GSM, el terminal del usuario se llama estación móvil. Cada estación móvil está constituida por una tarjeta SIM (Módulo de identificación al cliente), que permite identificar de manera única al usuario y teléfono móvil [11].

Cada dispositivo se identifica por medio de un número único de identificación, el cual consta 15 dígitos denominado IMEI (International Mobile Equipment Identity), éste dígito es el identificador internacional de equipos móviles. Cada tarjeta SIM a su vez posee un número de identificación único, éste número es secreto y se denomina IMSI (International Mobile Subscriber Identity), lo cual significa identificador internacional de Clientes o abonados móviles, éste código a su vez se protege por una clave de 4 dígitos denominada PIN (Personal Identification Number) [11].

Es por ello que cada tarjeta SIM permite identificar de manera independiente y confiable a cada usuario durante la comunicación con la estación base. El vínculo para la comunicación entre una estación móvil y una estación base se produce a través de un vínculo de radio denominado interfaz de aire [11].

La Figura 3.4 representa la arquitectura de la red GSM.

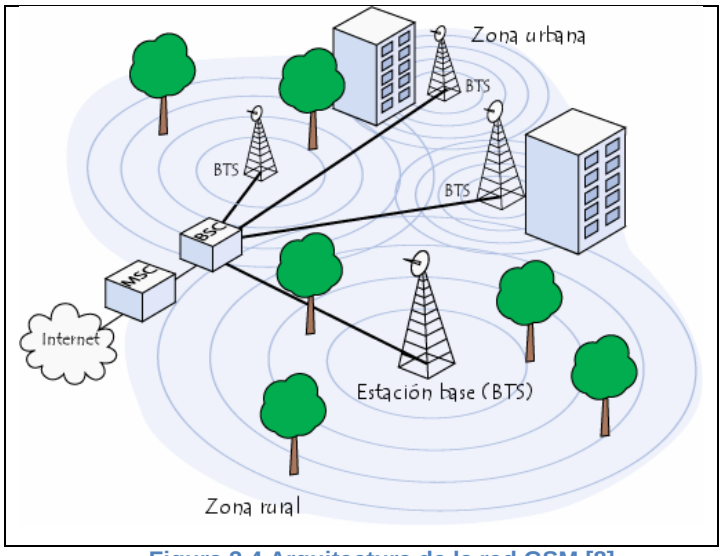


Figura 3.4 Arquitectura de la red GSM [8].

Todas las estaciones base de una red celular están conectadas a un controlador de estaciones base (BSC), el cual administra la distribución de los recursos. El sistema compuesto por el controlador de estaciones base y sus estaciones base conectadas es el Subsistema de estaciones base (BSS) [8].

Por último, los controladores de estaciones base están físicamente conectados al Centro de conmutación móvil (MSC) que los conecta con la red de telefonía pública y con Internet, lo administra el operador de la red telefónica. El MSC pertenece a un Subsistema de conmutación de red (NSS) que gestiona las identidades de los usuarios, su ubicación y el establecimiento de comunicaciones con otros usuarios [8].

La red celular compuesta de esta manera está diseñada para admitir movilidad a través de la gestión de traspasos, los cuales consisten en el traspaso desde una celda a otra. En cuanto al traspaso de una red de un cierto operador a otro se denomina Roaming, es un concepto aceptado y que permite el funcionamiento de equipos en otros países con operadores distintos al que tiene el teléfono móvil por defecto [8].

3.9 Sistemas Operativos móviles

Un sistema operativo móvil, es un sistema operativo que controla a un dispositivo móvil, destacándose sobre los sistemas operativos de computadora por su sencillez, simplicidad y orientación hacia la conectividad inalámbrica. Las mayores empresas dedicadas a éste rubro y las cuales han invertido en el desarrollo de sistemas operativos para estos móviles son: Microsoft, Apple, Nokia, RIM, Palm, y ahora recientemente Google, con el sistema operativo Android [12].

Los sistemas operativos móviles están compuestos por cuatro capas, las cuales se describen a continuación [12]:

- **Kernel:** Kernel o núcleo, proporciona el acceso a los diferentes elementos del hardware. Posee la cualidad de ofrecer servicios a capas de niveles superiores para el manejo de drivers, gestión de procesos, sistema de archivos y la administración de la memoria.
- **Middleware:** El middleware es una capa intermedia, por lo cual para el usuario es totalmente transparente su funcionamiento. Tiene como rol ofrecer servicios de gran importancia como motores de mensajería, comunicación, códec multimedia, interpretes HTML, gestión de dispositivos y seguridad.
- **Entorno de Ejecución de Aplicaciones*:*** Conjunto de interfaces abiertas y programables, las cuales facilitan a los desarrolladores la creación de software.
- **Interfaz de Usuario:** La interfaz de usuario corresponde, a la última capa y con la que el usuario tiene directa relación, ya que esta capa tiene como objetivo facilitar la interacción del usuario con la aplicación. Está compuesta por elementos como Botones, pantallas, listas, etc., además de un marco de interacción.

Por último, los sistemas operativos tienen aplicaciones denominadas nativas, y son las que acompañan al sistema operativo y que permiten el uso básico del dispositivo en su rol de Smartphone, llámese estas aplicaciones: Agenda, Marcación, etc [12].

3.9.1 Características principales del Smartphone

Los Smartphone's son teléfonos Inteligentes, pero su funcionamiento se basa en ciertos elementos, que son de suma importancia y que hacen que destaquen por sobre los demás dispositivos móviles [13].

- **Sistema Operativo:** Los teléfonos inteligentes se basan en un sistema operativo que permita ejecutar aplicaciones de diversos ámbitos. Los sistemas operativos que se ejecutan en un cierto SmartPhone no necesariamente lo harán en otro, depende de la arquitectura que estos tengan, a diferencia de los computadores los Smartphone's solo pueden utilizar el sistema operativo por defecto.
- **Software:** Actualmente la mayor parte de los teléfonos móviles incluyen algún tipo de software, incluso en los teléfonos móviles más básicos encontramos gestores de contactos y listas de llamadas realizadas, sin embargo, un Smartphone tiene la capacidad de ejecutar aplicaciones de producción y con un nivel de elaboración y complejidad mayor. Es común la necesidad de editar documentos, un SmartPhone permite ejecutar aplicaciones tales como Microsoft Office (en el caso del sistema operativo Windows Mobile, de Microsoft). Es así que la gama de aplicaciones y facilidades que entrega un dispositivo inteligente es muy variada, se pueden incluir acciones tales como crear listas de reproducción musical, rutas de manejo mediante GPS, manejar imágenes, geo-localización, enviar email, etc.
- **Acceso a la Web:** Gran parte de los SmartPhone permiten acceder a la web a altas velocidades mediante tecnologías como 3G o WIFI. Sin embargo, los que no permiten acceder a alta velocidad, permiten acceder de una u otra forma.
- **Teclado Qwerty:** El teclado qwerty es una innovación implantada a los Smartphone's. Consiste en un teclado que se despliega en la pantalla táctil (mediante software) o de manera física sobre el dispositivo (hardware). La disposición de las teclas emula a la disposición común de un teclado de computadora.

- **Mensajería:** Todos los teléfonos móviles o celulares permiten el envío y recepción de mensajes de texto, sin embargo los Smartphone's han extendido el servicio, y permiten el envío y recepción de correos electrónicos.

3.10 Otros conceptos y tecnologías afines a los dispositivos móviles.

3.10.1 Sistemas de posicionamiento Global

El sistema de posicionamiento global, o más conocido como GPS (Global positioning System), es un sistema de navegación por satélite que permite determinar en todo el mundo la posición de un objeto, persona, vehículo, etc. La precisión del dispositivo es variable y depende del dispositivo en sí y de la visibilidad entre dispositivo GPS y satélite, sin embargo existen GPS llamados diferenciales que tienen precisión de centímetros. Esta tecnología fue desarrollada por el Departamento de Defensa de Estados Unidos [14].

El GPS funciona mediante una red de 32 satélites, en la cual 28 de esos satélites están constantemente operativos y 4 son de respaldo. Las posiciones son determinadas mediante los satélites ubicados a 20.000 Km de altitud. Las trayectorias de estos satélites están sincronizadas de manera de cubrir completamente la superficie de la tierra. Para al funcionamiento de los GPS estos necesitan como mínimo tres satélites, de estos se reciben señales de identificación y la hora del reloj de cada uno de ellos. Con estos datos el aparato sincroniza el reloj que posee el GPS y calcula el tiempo de demora en llegar estas señales al equipo, mide las distancias al satélite mediante triangulación, método que se basa en determinar la distancia a cada satélite respecto de un punto de medición. Conocidas estas distancias se determina la posición del dispositivo GPS en relación a los satélites, conociendo las coordenadas de los satélites y la distancia a ellos, se asigna la posición absoluta o coordenadas reales al punto de medición [14].

3.10.2 Compás magnético

El compás magnético o brújula es un instrumento que permite determinar la orientación. Tiene como fundamento el uso de agujas imantadas que señalan el norte magnético, el cual es ligeramente diferente en cada zona del planeta y distinto al norte geográfico [14].

3.10.3 Acelerómetro

El acelerómetro es un dispositivo para medir la aceleración y fuerzas inducidas por la gravedad. Se Basa en la segunda ley de Newton [15].

Capítulo IV

El sistema operativo Android

Un sistema operativo móvil (SO móvil) es un conjunto de programas orientados al manejo de procesos que controlan un dispositivo móvil, similar al utilizado en computadoras estos son Windows, Linux, entre otros. Sin embargo, los sistemas operativos móviles son bastantes más simples y están preferentemente orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos.

Android, sistema operativo para dispositivos móviles, se ubica como el sistema operativo de los SmartPhone más vendidos en la actualidad, lo cual no hace más que reconocer sus bondades. Es por ello que en este capítulo se explicarán conceptos básicos y se introducirá al lector a este sistema operativo, dando a conocer su historia, componentes y forma de utilización

4.1 ¿Qué es Android?

Es un sistema operativo orientado a dispositivos móviles basado en una versión modificada del núcleo Linux. Inicialmente fue desarrollado por Android Inc, compañía que luego sería comprada por Google, y en la actualidad lo desarrollan los miembros de la Open Handset Alliance (conjunto de compañías lideradas por Google). La presentación de la plataforma Android se realizó el 5 de noviembre de 2007, esta plataforma permite el desarrollo de aplicaciones por terceros a través del SDK, proporcionado por Google, y mediante el lenguaje de programación Java [16].

Android está constituido por una pila de software pensada especialmente para dispositivos móviles y que incluye tanto un sistema operativo, como middleware y diversas aplicaciones de usuario. El objetivo principal de esta alianza empresarial es el desarrollo de estándares abiertos para la telefonía móvil como medida para incentivar su desarrollo y para mejorar la experiencia del usuario [16].

La Open Handset Allianse incluye a fabricantes de dispositivos y operadores, con firmas tan relevantes como Samsung, LG, Telefónica, Intel o Texas Instruments, entre las más importantes [16].

4.2 Características de Android

A continuación se señalan las principales características de Android.

- No es una plataforma hardware.
- Las aplicaciones Android se construyen en lenguaje Java.
- Android incluye [17]:
    - Kernel basado en Linux, con una rica interfaz de usuario.
    - Funcionalidad telefónica.
    - Aplicaciones para el usuario final.
    - Librerías de código.
    - Framework de aplicación.
    - Soporte multimedia.

4.3 Versiones de SDK para Android.

SDK está disponible para Windows, Mac y Linux. Android libera regularmente nuevas versiones del SDK, dentro de las SDK existentes podemos encontrar las que se enumeran a continuación [16]:

- Android 2.2, lanzada en Mayo 2010
- Android 2.1, lanzada en Enero 2010
- Android 2.0 (Denominado Eclair), lanzada en Octubre 2009
- Android 1.6 (Denominado Donut), lanzada en Septiembre 2009
- Android 1.5 (Denominado Cupcake), lanzada en Abril 2009

Cada SDK incorpora además una versión denominada Google API que implementa funcionalidades de los servicios de Google, destacando el servicio Google Maps [17].

4.4 Características técnicas de Android

4.4.1 Arquitectura de Android

El diseño de Android cuenta, entre otras, con las siguientes características [17]:

- Busca el desarrollo rápido de aplicaciones, que sean reutilizables y verdaderamente portables entre diferentes dispositivos.
- Los componentes básicos de las aplicaciones se pueden sustituir fácilmente por otros.
- Cuenta con su propia máquina virtual, Dalvik, que interpreta y ejecuta código escrito en Java.
- Permite la representación de gráficos 2D y 3D.
- Posibilita el uso de bases de datos, SQL Lite.
- Soporta un elevado número de formatos multimedia.
- Posee servicio de localización GSM.
- Controla los diferentes elementos hardware: Bluetooth, Wi-Fi, cámara fotográfica o de vídeo, GPS, acelerómetro, infrarrojos, etc., siempre y cuando el dispositivo móvil lo contemple.
- Cuenta con un entorno de desarrollo muy cuidado mediante un SDK disponible de forma gratuita: http://developer.android.com/sdk
- Ofrece un plug-in para uno de los entornos de desarrollo más populares, Eclipse, y un emulador integrado para ejecutar las aplicaciones.

La Figura 4.1 presenta las capas que componen la arquitectura de Android, cada una utiliza funcionalidades de la capa anterior:

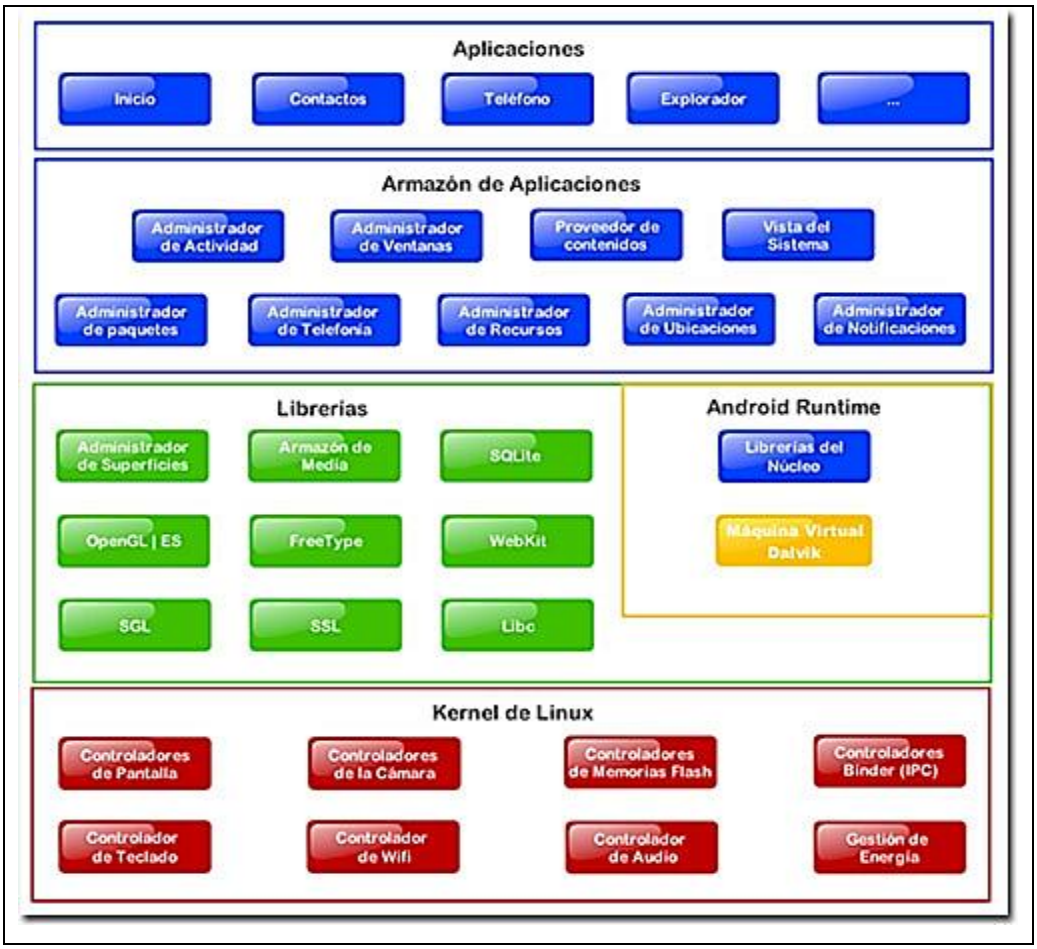


Figura 4.1 Arquitectura de Android [16].

4.5 Núcleo de Android

La capa más inmediata es la que corresponde al núcleo de Android. Android utiliza el núcleo de Linux 2.6 como una capa de abstracción para el hardware disponible en los dispositivos móviles. Esta capa contiene los drivers necesarios para que cualquier componente hardware pueda ser utilizado mediante las llamadas correspondientes [17].

La siguiente capa se corresponde con las librerías utilizadas por Android. Éstas han sido escritas utilizando C/C++ y proporcionan a Android la mayor parte de sus capacidades más características. Junto al núcleo basado en Linux, estas librerías constituyen el corazón de Android [17].

4.6 Componentes de Android

A continuación se detallan los componentes del sistema operativo Android [16]:

- **Framework de aplicación:** permite el uso y reemplazo de componentes.
- **Máquina virtual Dalvik:** máquina virtual para Java, optimizada para dispositivos móviles.
- **Browser integrado:** basado en el motor open source WebKit.
- **Gráficos optimizados:** poderosa librería gráfica 2D; gráficos 3D basados en la especificación OpenGL ES (aceleración hardware opcional).
- **SQLite:** para almacenamiento estructurado de datos.
- **Multimedia:** audio, video e imágenes fijas (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
- **Telefonía GSM:** dependiente del hardware.
- **Bluetooth, EDGE, 3G, y WiFi:** dependiente del hardware.
- **Cámara, GPS, compás y acelerómetro:** dependiente del hardware.
- **Ambiente de desarrollo:** incluye un emulador del dispositivo, herramientas para depuración, configuración de memoria y rendimiento y plugin para el IDE Eclipse.

4.7 Componentes de una aplicación Android

La Figura 4.2 presenta los componentes fundamentales de una aplicación Android.

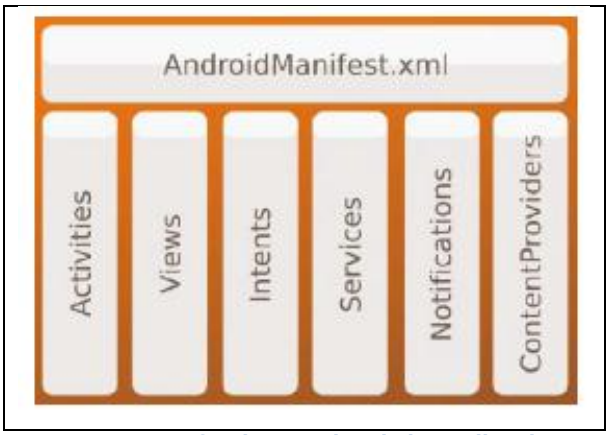


Figura 4.2 Componentes fundamentales de las aplicaciones Android [16].

Uno de los componentes principales de Android son las *Activity*. Cada *Activity* corresponde a una actividad o tarea de una aplicación. Las nuevas actividades van encima de las anteriores, de esta forma en un momento dado, una actividad pasa al primer plano y se coloca por encima de otra formando una pila de actividades [17].

La Figura 4.3 muestra el funcionamiento de las activities en Android, ante la interaccion con el boton *Back* del Smartphone. Éste boton se encuentra disponible en todos los SmartPhone's Android 17.

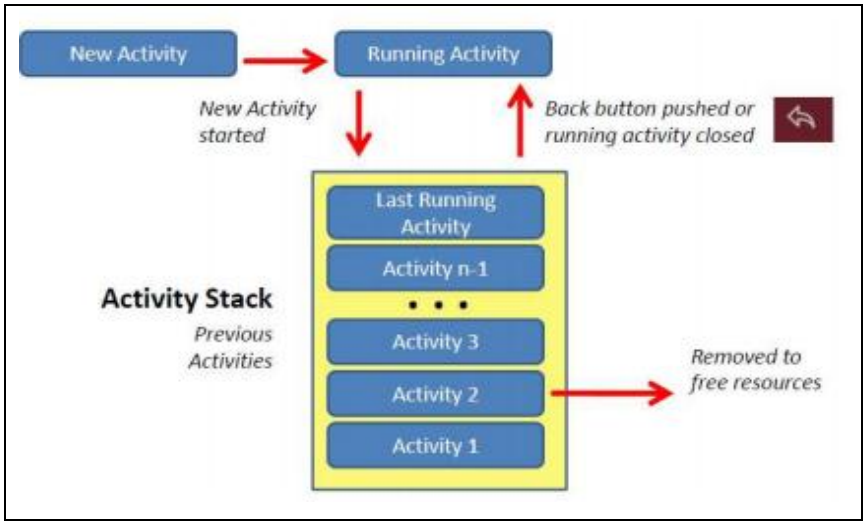


Figura 4.3: Funcionamiento de las *Activities* en Android [17].

Como se puede apreciar en la Figura 4.3, el botón back del Smartphone cierra la actividad y recupera desde la pila la actividad anterior que, por orden de llegada, debe continuar.

La Figura 4.4, presenta un diagrama de actividad que ilustra los estados de las *Activities* en Android y la navegación entre estados de acuerdo a acciones.

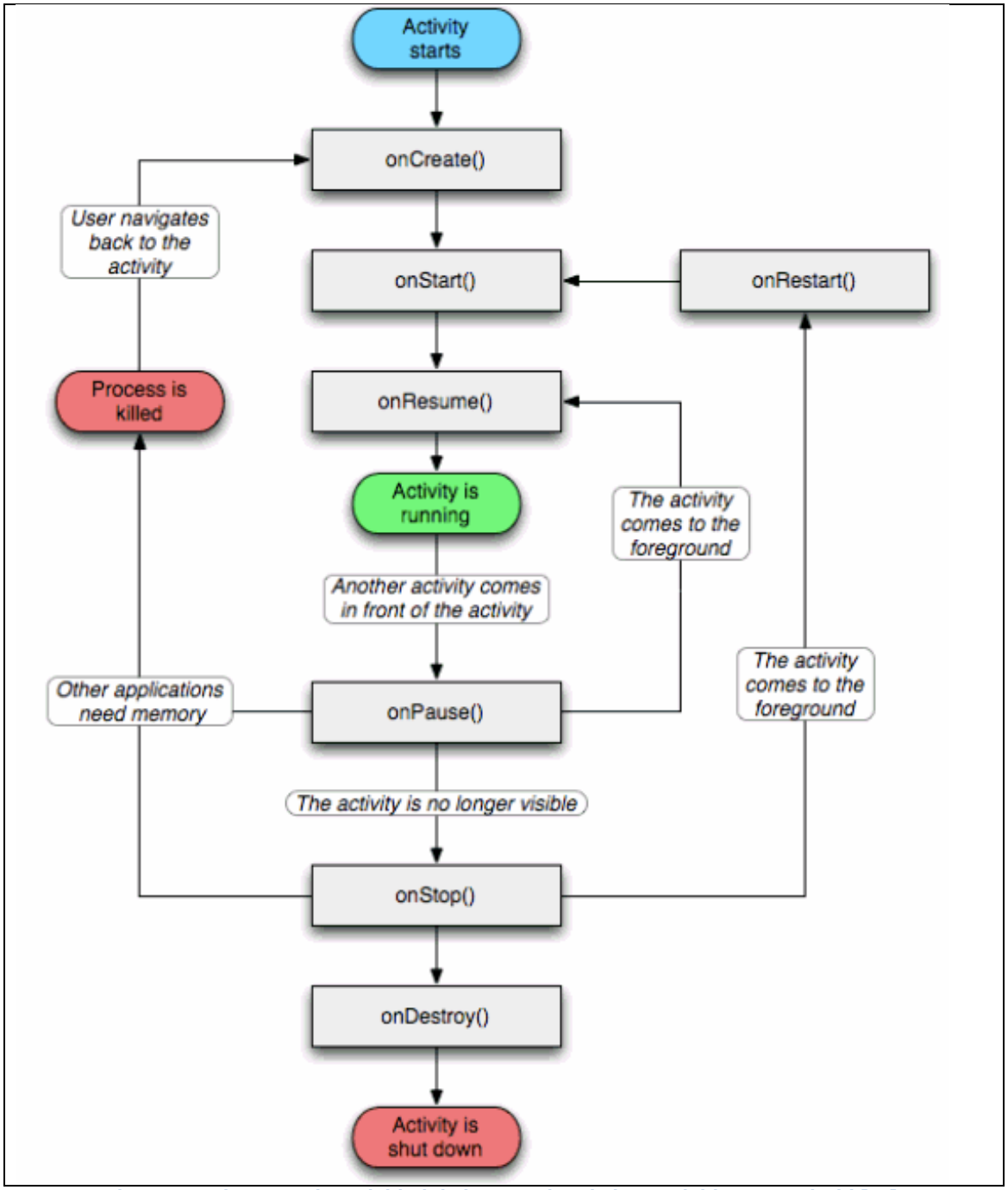


Figura 4.4 Diagrama de actividad de los estados de las *Activities* en Android [17].

Capítulo V

Tecnologías para el desarrollo Web

Actualmente las tecnologías Web centran las miradas de los desarrolladores de software, esto impulsado por la masificación de Internet y gran versatilidad de funcionamiento.

Es por ello que las nuevas tecnologías recientemente lanzadas son el foco de atención. Algunas de estas tecnologías cambian formas de funcionamiento o promueven marcos de trabajo para el desarrollo de sistemas en entorno Web.

Dado lo anterior, en este capítulo se presentarán nuevas tecnologías web y sus características. Estas tecnologías han sido creadas de manera independiente pero se complementan para crear nuevos y novedosos sistemas para la web.

5.1 Struts 2

Struts 2 es un Framework que permite el desarrollo de aplicaciones Web de manera sencilla; estas aplicaciones se basan en Java EE, bajo la arquitectura Modelo Vista Controlador (MVC) [18].

Dado que se basa en el lenguaje de programación Java, es posible la aplicación de patrones, estándares y un diseño orientado a objetos [18].

5.2 Google Maps

El API de Google Maps permite utilizar mapas con gran nivel de detalle en páginas Web, mediante la utilización de JavaScript. El API proporciona diversas utilidades para manipular mapas y añadir contenido a los mismos mediante diversos servicios, permitiendo crear sólidas aplicaciones de mapas en sitios web [19].

Este servicio se encuentra disponible en diferentes lenguajes y frameworks, a continuación se presentan los servicios disponibles para la utilización de Google Maps [19]:

- Maps API for Flash
- Maps JavaScript API (Versión 2 y 3).
- Static Maps API

5.2.1 Características de Google Maps.

Google Maps fue presentado el 8 de Enero de 2005. Originalmente soportaría sólo a los usuarios de Internet Explorer y Mozilla Firefox, sin embargo el soporte para Opera y Safari fue agregado el 25 de febrero de 2005. El software estuvo en su fase beta por 6 meses antes de convertirse en parte de Google Local, el 6 de octubre de 2005 [19].

Google Maps fue una de las primeras aplicaciones basadas en AJAX de uso masivo por parte de los usuarios. Su gran éxito ha provocado que compañías imiten y realicen nuevas propuestas, sin embargo, Google Maps continúa acaparando la preferencia de los usuarios [19].

Google ofrece de forma gratuita un API para el desarrollo de aplicaciones a medida basadas en su servicio de mapas, con esto se permite integrar los mapas en otras aplicaciones e incluso hacer *mash-up* o mezclas de Google Maps y otras aplicaciones web que también disponen de una API pública [19].

Sin embargo, Google impone limitaciones en el uso de su API. A continuación se presentan tales limitaciones [19].

- No existe limitación en el número de visitas diarias a la página que se puede generar mediante Google Maps API.
- El número de solicitudes de codificación geográfica que se pueden enviar diariamente es limitado.
- El API de Google Maps no incluye publicidad.
- Los usuarios finales deberán acceder de forma gratuita al servicio.
- No se pueden modificar ni ocultar los logotipos ni la atribución del mapa.
- Se puede utilizar el API (salvo en el caso de Static Maps API) en sitios Web o en aplicaciones de software. En el caso de sitios Web, se debe registrar la URL del servidor que soportará la aplicación que contenga los mapas.

5.2.2 Google Maps JavaScript API

La versión de JavaScript permite la incrustación de mapas dinámicos en páginas Web, así como la utilización de geo-localización, es un API de gran estabilidad y eficiencia, que necesita Browser compatibles con JavaScript y acceso a Internet para su normal funcionamiento [19].

5.3 Ajax

El término AJAX se presentó por primera vez en el artículo "Ajax: A New Approach to Web Applications, publicado por Jesse James Garrett el 18 de Febrero de 2005. Hasta ese momento, no existía un término normalizado que hiciera referencia al nuevo tipo de aplicaciones Web que estaba apareciendo" [20].

"En realidad, el término AJAX es un acrónimo de Asynchronous JavaScript + XML, que se puede traducir como: JavaScript asíncrono + XML" [20].

"Ajax no es una tecnología en sí mismo, en realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes" [20].

Las tecnologías que forman AJAX son [20]:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML, XSLT y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.

La Figura 5.1 presenta las tecnologías que componen AJAX, así como la interacción entre cada una de ellas.

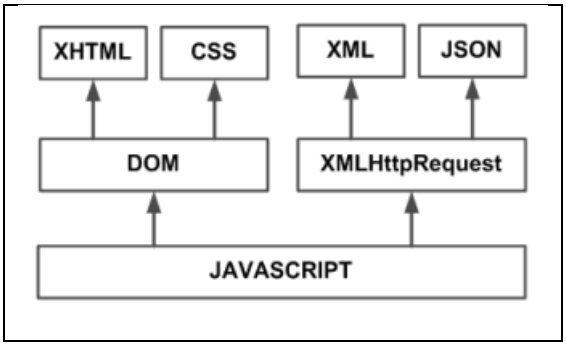


Figura 5.1 Diagrama de tecnologías integrantes de AJAX [20].

En las aplicaciones Web tradicionales, las acciones del usuario en la página desencadenan llamadas al servidor, éste las procesa, construye la respuesta y generará el código HTML de la página que será devuelta para que finalmente se visualice a través del navegador. En la Figura 5.2 se presenta la estructura de las aplicaciones Web creadas de manera clásica y usando AJAX [20].

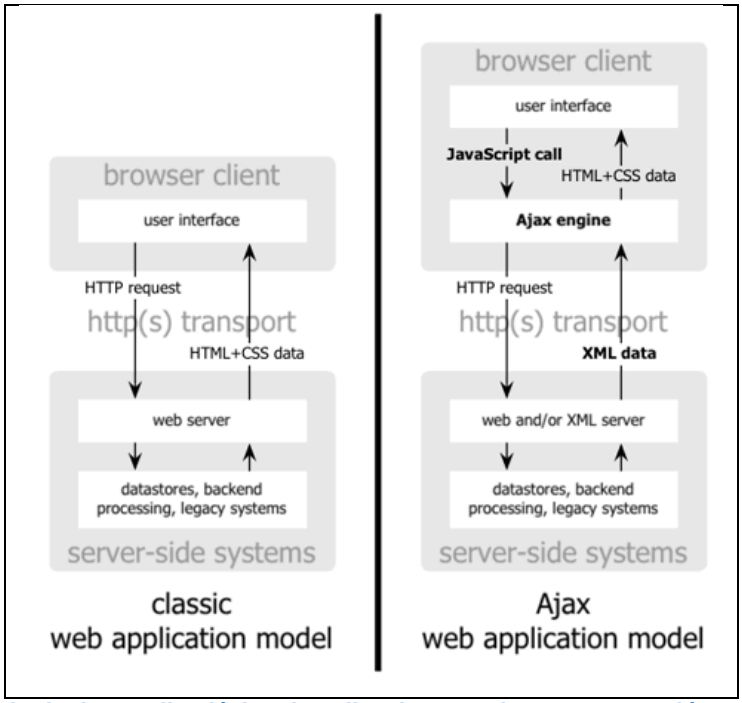


Figura 5.2 Modelo de desarrollo clásico de aplicaciones web, en comparación con el modelo de desarrollo Ajax [20].

La técnica tradicional para crear aplicaciones Web funciona correctamente, pero no es adecuada a las necesidades de los usuarios. Al realizar una petición al servidor, el usuario debe esperar a que se recargue la página con los cambios solicitados. Si la aplicación debe realizar peticiones, su uso se convierte en algo molesto. AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano [20].

Las aplicaciones construidas con AJAX eliminan la recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor. La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se

encuentra con una ventana del navegador vacía esperando la respuesta desde el servidor [20].

Las peticiones HTTP al servidor se sustituyen por peticiones JavaScript que se realizan al elemento encargado de AJAX. Las peticiones más simples no requieren intervención del servidor, por lo que la respuesta es inmediata. Si la interacción requiere una respuesta del servidor, la petición se realiza de forma asíncrona mediante AJAX. En éste caso, la interacción del usuario tampoco se ve interrumpida por recargas de página o largas esperas por la respuesta del servidor [20].

Capítulo VI

Desarrollo de la aplicación nativa Android y creación del robot

En el presente capítulo se presentará la solución al problema planteado en el punto 1.1, así como también se abordará el desarrollo del robot y la aplicación embebida en el Smartphone Android.

Para el desarrollo de la aplicación Android, se realiza un análisis de requerimientos funcionales y no funcionales, así como también la definición de los actores del sistema, realización de diagrama de casos de uso, secuencia de sistema y de comunicación. Esto para explicar de mejor forma al lector la aplicación a implementar.

6.1 Descripción de la solución

Dada la descripción del problema, vista en el punto 1.1, se creará un robot móvil el cual será capaz de realizar una ruta en base a sus sensores. Una vez que haya realizado la ruta, el robot tendrá la capacidad de distinguir entre ciertos objetos y ordenarlos de acuerdo a su color.

Para dar cumplimiento a lo señalado en el párrafo anterior, se construye un robot móvil el que tiene en su parte superior un Smartphone Android, para el cual se desarrolla una aplicación nativa. Esta aplicación será la encargada de ejecutar la lógica necesaria para ir de una localización geográfica a otro, y recorrer la ruta asignada, la que se compone de un grupo de localizaciones geográficas que el usuario seleccionará desde el sistema de monitoreo y control, cuya selección se realizará mediante un mapa. Luego de realizar la ruta asignada el robot será capaz de identificar objetos de un determinado color, objetos que deberá ser capaz de tomar y trasladar a un lugar representado por un objeto de un color totalmente diferente.

En cuanto al robot, éste contiene un micro-controlador Arduino, el cual se comunica vía bluetooth con la aplicación Android, de manera que el robot ejecute las decisiones que tome la aplicación. El robot además cuenta con un sensor ultrasónico el cual permitirá detectar la cercanía de los objetos a ordenar.

La selección de rutas, como ya se mencionó, se realizará mediante la utilización de un mapa, el cual ante cada selección sobre él asignará un punto a la localización geográfica seleccionada y se unirá mediante una línea con la localización seleccionada anteriormente. Las rutas se componen de, a lo menos dos localizaciones geográficas: la primera localización geográfica de cada ruta es en la cual se encuentra posicionado el robot. El sistema de monitoreo y control, tiene la capacidad de monitorear y controlar a dos robots, sin embargo en esta ocasión sólo se construye un robot.

En este capítulo se presenta la creación del robot y aplicación Android. Para facilitar el entendimiento del desarrollo e implementación del proyecto, se recomienda ver el punto 6.1.1, el cual consiste en la definición del nombre del sistema y sus módulos.

6.1.1 Definición del nombre del sistema

Ante el objetivo de crear una aplicación embebida en el SmartPhone Android, software Arduino para el control del robot, y un sistema Web que oficie de servidor y despliegue el sistema de monitoreo y control. De acuerdo a lo anterior se ha decidido nombrar al proyecto como *SkyNet*, ante lo cual se denomina *SkyNet módulo Android* al módulo del proyecto que se ejecutara sobre en el Smartphone, al sistema Web se llamará como *SkyNet módulo Web*. De manera de que en adelante se entienda de esta manera a cada uno de los elementos que conforman éste proyecto.

6.1.2 Diagrama de despliegue

Los diagramas de despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación.

La Figura 6.1 presenta el diagrama de despliegue que muestra la distribución de procesos y componentes en los nodos procesadores. Diagrama que tiene como finalidad explicar al lector el funcionamiento de los módulos que conforman a SkyNet.

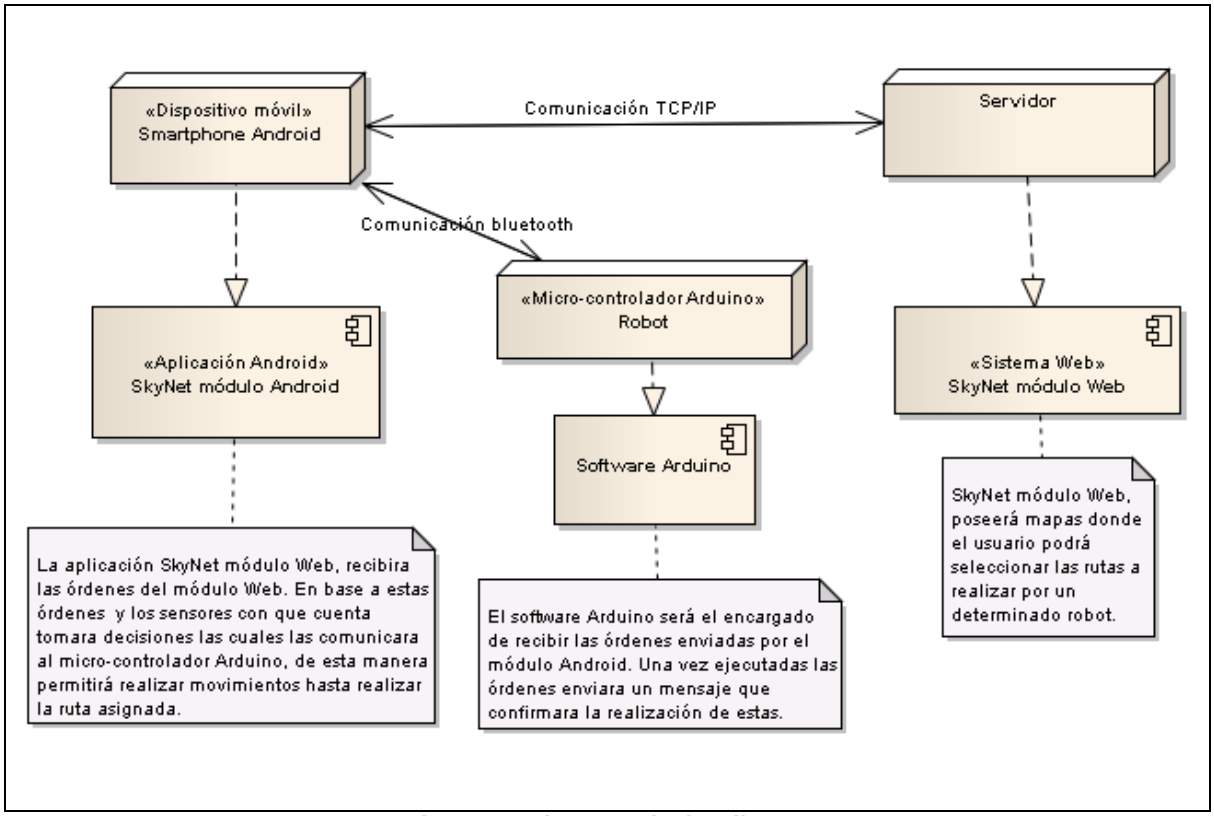


Figura 6.1 Diagrama de despliegue.

Como se puede apreciar en la Figura 6.1, el sistema SkyNet, nombre del sistema desarrollado para el proyecto "Integración de una plataforma SmartPhone en entornos de robótica móvil", consta de tres subsistemas: El módulo SkyNet módulo Web, SkyNet módulo Android y la aplicación embebida en el micro-controlador Arduino. La interacción entre los tres subsistemas mencionados anteriormente permite el correcto funcionamiento del sistema.

6.2 Análisis de requerimientos del robot

6.2.1 Requerimientos físicos del robot

Para el establecimiento de requerimientos físicos del robot se ha tomado en cuenta la variabilidad del terreno, adherencia del terreno, pendientes e inclinaciones en el terreno, además de aspectos electrónicos y mecánicos del robot. A continuación se presentan los requerimientos físicos del robot:

- Autonomía y capacidad electrónica para la utilización de sensores como ultrasonido, aceleración, orientación, visión y localización.
- El robot debe tener la capacidad de detectar colores, por lo cual ante la utilización del SmartPhone para esta función, debe contar con un espacio para el montaje del celular de manera de garantizar su seguridad.
- El robot debe ser capaz de tomar objetos de un color especifico y trasladar a un lugar representado por un objeto de un color diferente al que traslada.
- Para que el robot tome el objeto mencionado en el ítem anterior, debe contar con un brazo mecánico o mecanismo que lo emule, y que permita tomar y trasladar los objetos
- El robot debe contar con pilas y baterías de manera de permitir su funcionamiento de manera autónoma. El voltaje entregado por estos elementos debe ser mayor o igual a 9 Volts y menor que 12 Volts.
- El robot debe ser construido con elementos sólidos de manera que se garantice su integridad ante los movimientos que realice.
- El robot debe poseer elementos que permitan la comunicación con un computador y el SmartPhone, esto para la carga de códigos y/o comunicación con el software del SmartPhone.
- El robot debe poseer elementos que le permitan desplazarse en ambientes abiertos.
- El robot no debe superar las dimensiones de 40x40x40 cm, a fin de permitir su transporte y facilitar su uso.

- El robot no debe superar los 2 kilogramos de peso.

El robot que se construirá contará con un micro-controlador Arduino el que deberá programarse a fin de que le permita desplazarse adecuadamente considerando lo indicado en la lista anterior. Los requisitos que deberá cumplir el software embebido en este micro-controlador se indica en el siguiente párrafo.

6.2.2 Requerimientos del software embebido en el micro-controlador Arduino

El software alojado en el micro-controlador Arduino incluido en el robot, debe proveer funciones que permitan realizar los movimientos determinados por el SmartPhone. Por lo tanto, esta aplicación embebida deberá cumplir con los siguientes requerimientos:

- Permitir al robot realizar movimientos básicos, los cuales serían adelante, atrás, izquierda, derecha, detenerse.
- Permitir el movimiento del brazo mecánico o mecanismo implementado, de manera de que el robot sea capaz de tomar y trasladar los objetos detectados.
- Permitir la utilización de sensores de distancia de manera de detectar obstáculos en el camino.
- Permitir la comunicación con el SmartPhone, más específicamente el robot debe comunicarse de manera serial con el dispositivo bluetooth del SmartPhone.

6.3 Diseño físico del robot

El diseño físico del robot considera aspectos tales como: diversidad de terrenos, elementos que obstaculicen el libre movimiento del robot, terrenos con mayor o menor adherencia y un brazo mecánico o mecanismo similar que permita tomar los objetos a trasladar. Es por ello que se ha optado por utilizar un chasis que posea orugas que permitan realizar movimientos sobre su eje y además de otorgar fuerza a su desplazamiento, el diseño tiene como base las orugas de los tanques de guerra, el cual considera orugas de metal las cuales abarcan el largo total del tanque, y presentan una inclinación en los extremos del tanque para permitir sobrepasar obstáculos.

La Figura 6.2 muestra un tanque de guerra del cual se tomó su chasis como la base para el diseño del chasis del robot.

Figura 6.2 Modelo de tanque guerra, base para el diseño del chasis del robot.

Para el diseño del mecanismo que se encargara de tomar los objetos, y que se ubicara en la parte superior del robot. Se decide replicar el funcionamiento de una pinza (ver Figura 6.3), sin embargo se agregaran modificaciones.

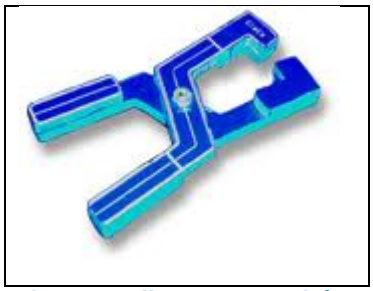


Figura 6.3 Modelo de pinza a replicar, que servirá para tomar los objetos.

La primera modificación a incorporar será que solo uno de los “dientes” de la pinza será móvil, el diente que se encontrara fijo se ubica sobre un costado de la parte delantera del robot. El otro “diente” será móvil y se ubicara en el costado contrario de la parte delantera del robot, este diente se alargara de manera que realice un barrido más amplio de los elementos que se encuentren en el frente del robot, de esta forma moverá el objeto de su ubicación hacia el “diente” que se encuentra fijo en el costado contrario. Esta modificación se incorpora con el ubicar en un lugar que deje libre la visual para que se detecte un nuevo objeto.

El resultado es un tanque de chasis de madera, orugas de goma. Una base de madera donde se encuentra la electrónica del robot, y sobre esta, una plataforma acrílica transparente que permite el anclaje de mecanismos para tomar y dejar objetos, y además de un soporte para el SmartPhone.

Las dimensiones del tanque son: 22 cm de ancho, 26 cm de largo y 26 cm de alto, con un peso de 1,5 kg. El tanque ha sido pintado de color negro, de manera que parte de él no interfiera en la detección de objetos.

Las orugas contemplan una inclinación de 45 grados en la parte frontal y posterior del robot, para superar obstáculos que se presenten en el desplazamiento del robot. Las orugas se mueven mediante reducciones, la cuales consisten en engranajes que permiten reducir la velocidad de los motores para otorgar mayor fuerza al movimiento del robot, estas reducciones poseen dos motores los cuales finalmente permitirán el movimiento independiente de cada oruga, esto permitirá la particularidad de girar sobre su eje.

6.4 Software embebido en micro-controlador Arduino.

El software construido para ser ejecutado en el micro controlador Arduino consiste en una serie de funciones las cuales dependerán de la información que se envíe desde el software que se ejecutará en el SmartPhone. La comunicación entre estas dos aplicaciones se ha implementado mediante bluetooth, siendo el software del robot de tipo reactivo ya que actúa de acuerdo a órdenes enviadas por parte del SmartPhone Android.

Las funciones básicas que se han implementado son: avanzar, retroceder, girar hacia la izquierda, girar hacia la derecha, detener, capturar objeto y liberar objeto. Además, se ha considerado el uso de sensores de ultrasonido, mediante la captura del rebote del sonido en elementos que se encuentran frente a él, esto permite determinar la distancia a la cual se encuentra el robot, replicando el funcionamiento de un sonar. Es importante destacar que el robot deberá responder cada vez que haya realizado una acción, de manera de confirmar las acciones realizadas.

La programación se realizó mediante el entorno de desarrollo proporcionado por Arduino, llamado Arduino Alpha, software gratuito y disponible en su sitio Web *http://arduino.cc/en/Main/Software.*

6.5 Desarrollo de la aplicación nativa sobre plataforma Android

6.5.1 Análisis de requisitos

El Análisis de requisitos corresponde a la primera fase del desarrollo de un software, y permite conocer mediante ciertas tareas las condiciones o funcionalidades que debe cumplir el software a construir.

Siempre se debe tener en cuenta la visión del usuario, pues será quien finalmente haga uso de la aplicación [22].

6.5.1.1 Requisitos no funcionales

Los requisitos no funcionales son aquellos con características que pueden afectar diversos aspectos de una aplicación, como por ejemplo, el rendimiento, mantenimiento, seguridad, portabilidad y estándares.

La Tabla 6.1 describe los requisitos no funcionales para la aplicación a construir en el SmartPhone Android.

Atributo	Detalle
Plataforma	Android SDK 1.5 en adelante
Tipo de aplicación	Aplicación móvil
Lenguaje de programación	Java y XML
Gestor de base de datos	SQLite
Herramientas de desarrollo	Eclipse
Tiempos de respuesta	No mayor a 7 segundos para cualquier operación.
Disponibilidad	Para el manejo mediante el sistema de monitoreo y control se necesita de WIFI en todo momento, para asegurar el normal funcionamiento de la aplicación.

Tabla 6.1: Requisitos no funcionales de SkyNet módulo Android.

6.5.1.2 Requisitos Funcionales

"Los requisitos son una descripción de las necesidades o deseos de un producto" [21]. Los requisitos funcionales reflejan las acciones del sistema frente a determinados escenarios.

En la Tabla 6.2 se presentan las categorías de los requisitos, posteriormente estas se utilizan para identificar los requisitos que pudiesen pasar inadvertidos [21].

Categoría de la Función	Significado
Evidente	Debe realizarse, y el usuario debería saber que se ha realizado.
Oculta	Debe realizarse, aunque no es visible para los usuarios. Esto se aplica a los servicios técnicos subyacentes, como guardar información en un mecanismo persistente de almacenamiento. Las funciones ocultas a menudo se omiten (erróneamente) durante el proceso de obtención de los requerimientos.
Superflua	Opcionales; su inclusión no repercute significativamente en el costo ni en otras funciones.

Tabla 6.2: Categoría de los requisitos [21].

6.5.1.3 *Funciones básicas del sistema*

En la Tabla 6.3 se presentan las funciones básicas del SkyNet módulo Android, que luego serán ampliadas, dado que contienen otras funciones relevantes para el sistema.

Ref. #	Función	Categoría
R.1	Control mediante SkyNet	Evidente
R.2	Autenticar Usuario	Evidente

Tabla 6.3: Funciones Básicas de SkyNet módulo Android.

La Tabla 6.4, se presenta el detalle para la funcionalidad *Control mediante SkyNet*.

Ref. #	Función	Categoría
R.1.1	Iniciar proceso de monitoreo y control.	Evidente
R.1.2	Conectar a un dispositivo bluetooth.	Evidente
R.1.3	Conectar manualmente a un dispositivo bluetooth.	Evidente
R.1.4	Controlar manualmente el robot.	Superflua

Tabla 6.4: Detalle Funcionalidad *Controlar mediante SkyNet.*

La Tabla 6.5 contiene el detalle de la funcionalidad *Autenticar usuario.*

Ref. #	Función	Categoría
R.2.1	Ingresar a aplicación SkyNet módulo Android.	Evidente
R.2.2	Modificar el password de ingreso a la aplicación.	Superflua.

Tabla 6.5: Detalle Funcionalidad *Autenticar Usuario.*

6.5.2 Identificación de Actores de SkyNet módulo Android

Para el software desarrollado en el SmartPhone se ha identificado dos actores, los cuales se describen en la Tabla 6.6 y 6.7.

Actor	Usuario SkyNet
Descripción	El usuario SkyNet es una persona que cuenta con acceso a SkyNet módulo Web. Además esta persona debe tener acceso a un SmartPhone, con el módulo Android instalado y un equipo robótico. El equipo robótico deberá permitir la comunicación bluetooth y el desplazamiento del robot mediante urugas, así como también, un brazo en su parte superior.
Características	Podrá hacer uso limitado del sistema SkyNet módulo Web y uso de la totalidad de las funcionalidades del módulo Android. Las funcionalidades que podrá utilizar del módulo web corresponden a la asignación de rutas y al monitoreo de estas.

Tabla 6.6: Actor Usuario SkyNet.

Actor	SkyBot
Descripción	El actor SkyBot es un robot, el cual tendrá en su parte superior un Smartphone Android, el que ejecutara a SkyNet módulo Android.
Características	Es un robot móvil, el cual posee orugas de goma y reducciones para proporcionar fuerza al movimiento. Tiene un micro-controlador Arduino que es el encargado de ejecutar las órdenes enviadas por SkyNet módulo Android. Además, SkyBot posee un sensor ultrasónico, y un brazo. Estos elementos en conjunto permiten tomar los objetos que SkyNet módulo Android Indique.

Tabla 6.7: Actor SkyBot.

6.5.3 Modelo de casos de uso

En la Figura 6.4 se presentan los casos de usos identificados para el actor *Usuario SkyNet y* actor *SkyBot*, es importante destacar que este diagrama de casos de uso corresponde solo a SkyNet módulo Android.

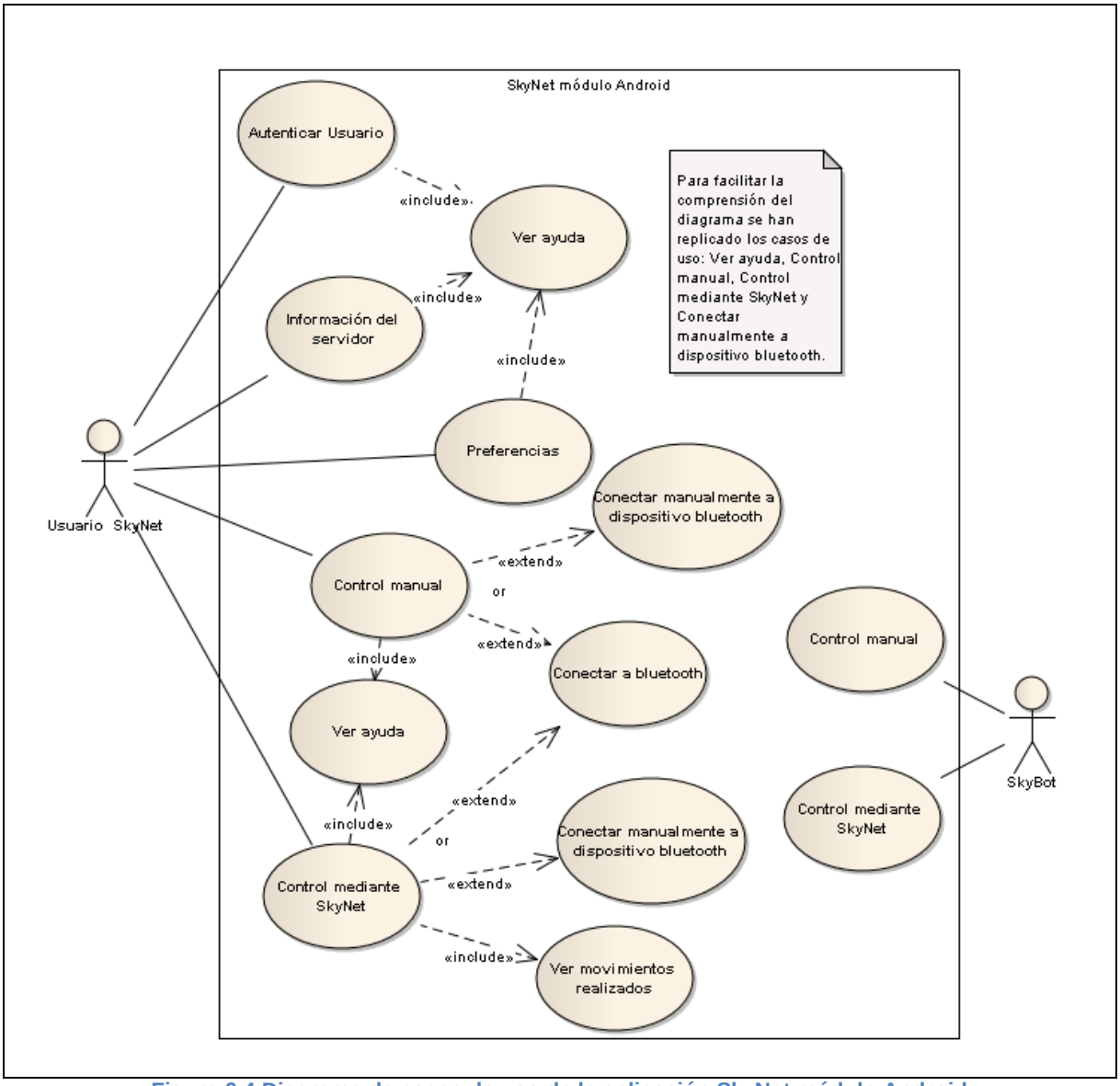


Figura 6.4 Diagrama de casos de uso de la aplicación SkyNet módulo Android.

6.5.4 Aspectos importantes para la usabilidad del sistema embebido en SmartPhone Android

Para el desarrollo de la interfaz de usuario de la aplicación Android denominada SkyNet módulo Android, se aplicaron conceptos de usabilidad de manera de facilitar al usuario la utilización del sistema.

Ante el hecho de que la aplicación se ejecuta en un dispositivo móvil se deben considerar aspectos que puedan jugar en contra de la usabilidad, como pueden ser los diferentes tamaños de las pantallas de los Smartphone's y capacidad de computo reducida, entre otros aspectos [22].

Es así como se ha optado por un diseño sencillo y funcional que no abuse de elementos gráficos, dado que la interacción entre el usuario y la aplicación es limitada. Es por ello que se ha considerado lo siguiente:

- Utilizar iconos simples, fáciles de reconocer y seleccionar, de manera que el usuario pueda intuir mediante el icono la funcionalidad que éste representa.
- Utilizar colores adecuados, con un nivel de contraste que permita observar los iconos en condiciones de luz natural (luz directa del sol en ocasiones provoca que se dificulte la visión de lo que está contenido en la pantalla del SmartPhone).
- Nombres representativos y claros, dando prioridad a la utilización de nombres y no verbos.
- Realización de acciones comunes asociadas a iconos comúnmente reconocidos (ejemplo signo de interrogación para la ayuda).
- Uso de menús para facilitar uso de funcionalidades del sistema y el dispositivo.
- Incluir opciones de navegación, cancelar y/o parar.

6.5.5 Modelo entidad relación.

El modelo entidad relación es una herramienta utilizada para modelar datos de una aplicación, con éste modelo se consigue representar de manera gráfica la estructura lógica de una base de datos [23].

Nace a mediados de los años setenta, fue propuesto por Chen con el fin de permitir la representación conceptual de los problemas. Adopta la forma de un grafo, donde sus elementos fundamentales son las entidades y las relaciones. Una entidad caracteriza a un tipo de objeto ya sea real o abstracto, teniendo nombre y atributos definidos. Además se considera entidad a todo aquello respecto del cual se desee almacenar información. Las entidades se suelen representar mediante rectángulos. La relación es una asociación entre varias entidades, estas poseen nombres, y se representan mediante flechas y rombos [23].

La Figura 6.5 presenta el diagrama de entidad relación de SkyNet módulo Android.

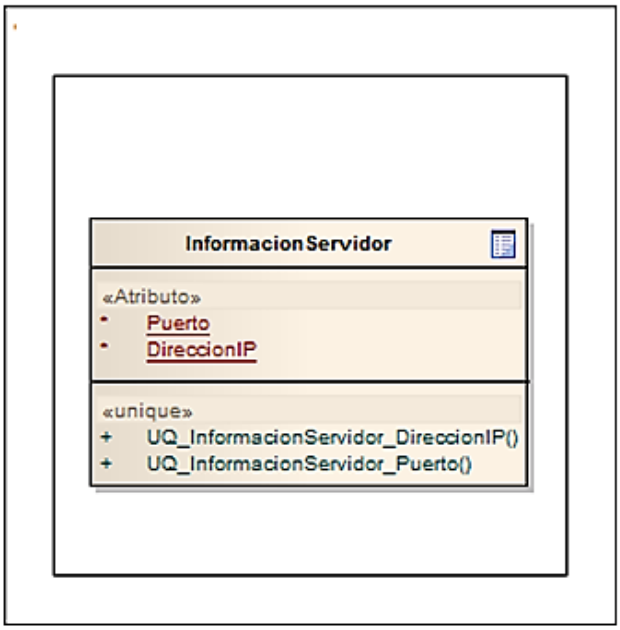


Figura 6.5 Diagrama entidad relación de SkyNet módulo Android.

6.5.6 Diagrama de clases del diseño

La Figura 6.6 presenta el diagrama de clases correspondiente al paquete *logica*, de la aplicación SkyNet módulo Android.

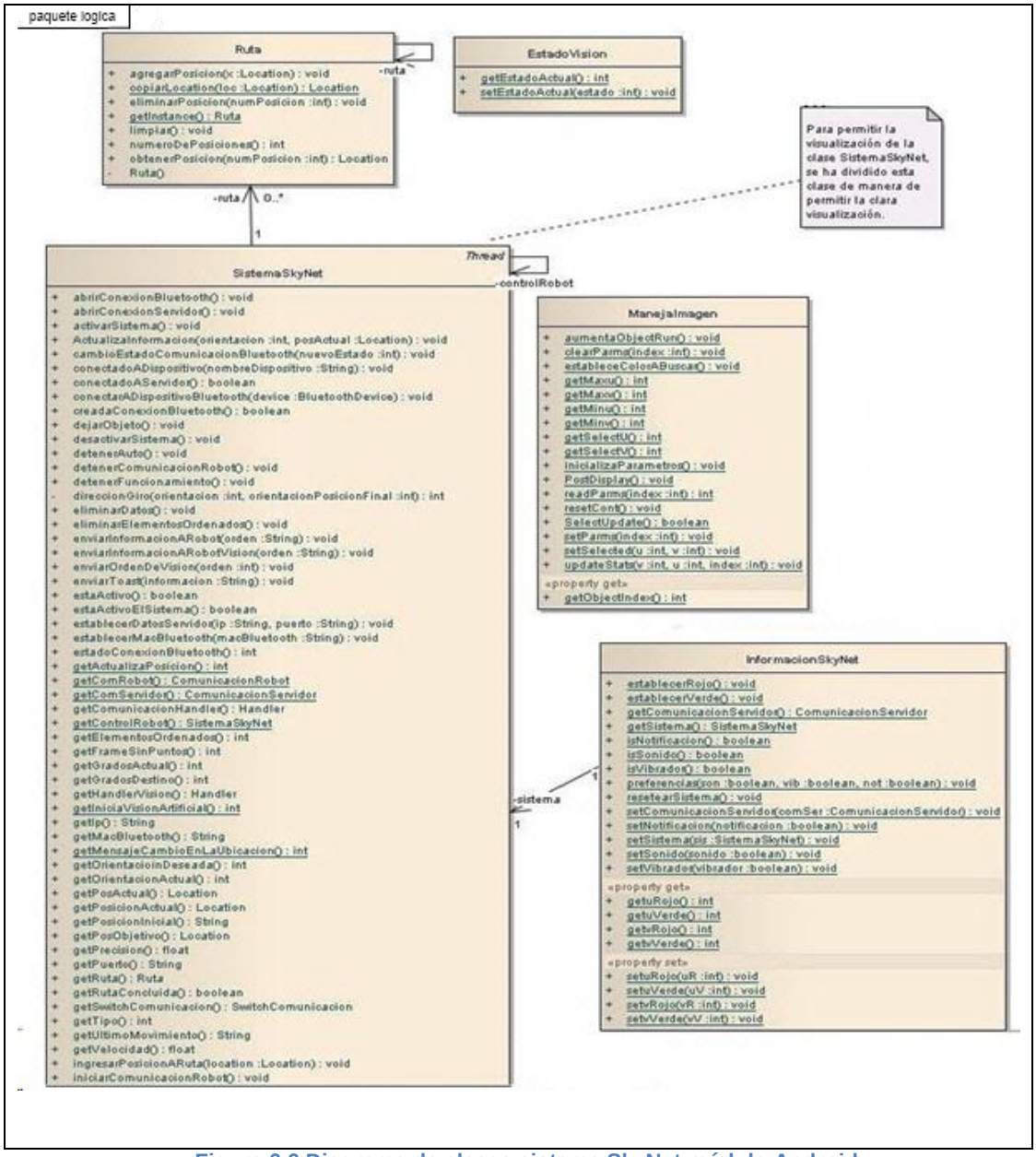


Figura 6.6 Diagrama de clases sistema SkyNet módulo Android.

Dada la cantidad de métodos existentes para la clase *SistemaSkyNet* (correspondiente al módulo Android de SkyNet), se ha dividido esta clase. A continuación en la Figura 6.7 se presentan los restantes métodos de la clase *SistemaSkyNet*.

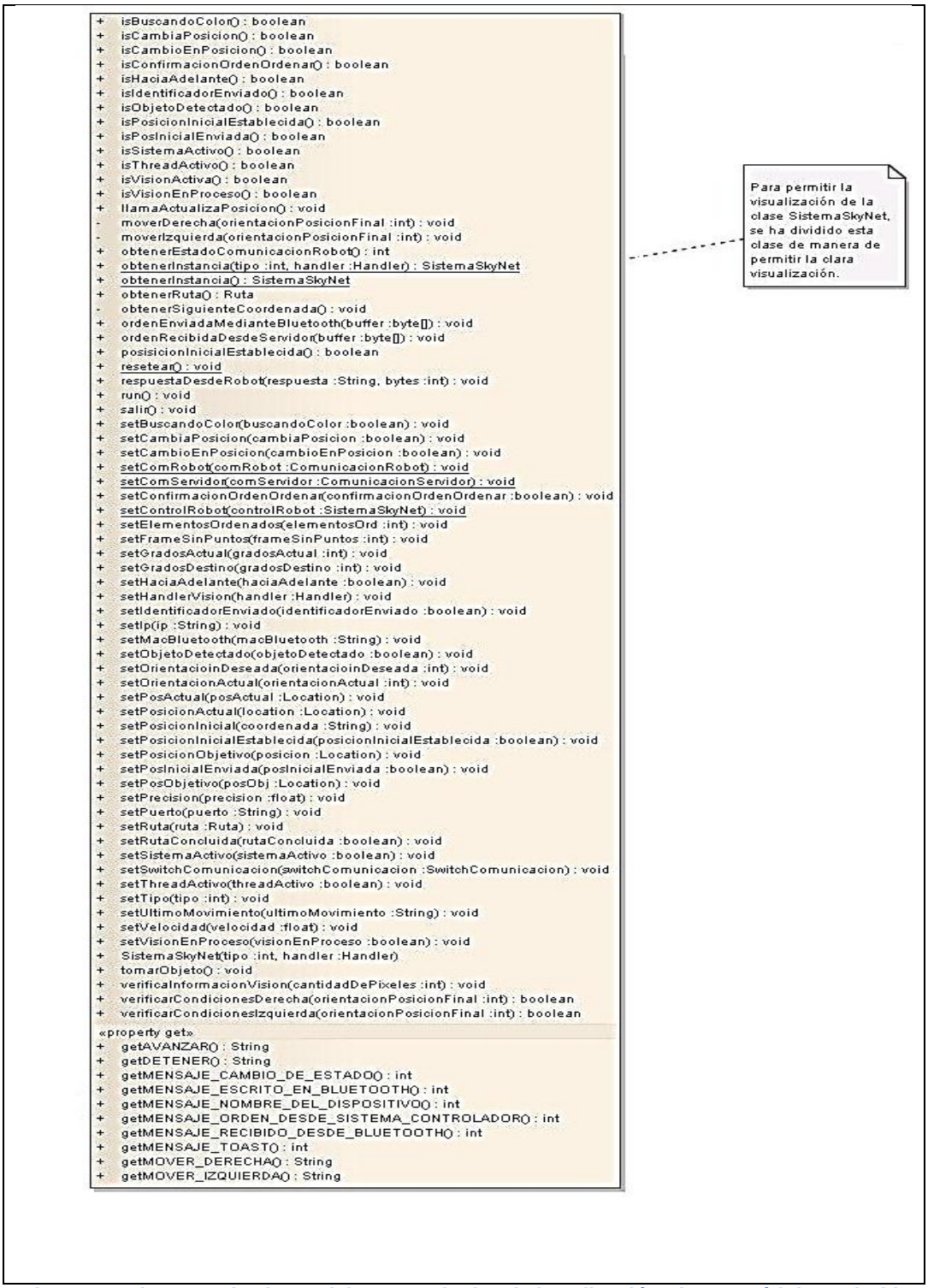


Figura 6.7 Diagrama de clases del paquete logica de la aplicación SkyNet módulo Android.

La Figura 6.8 muestra el diagrama de clases del paquete *comunicacion*.

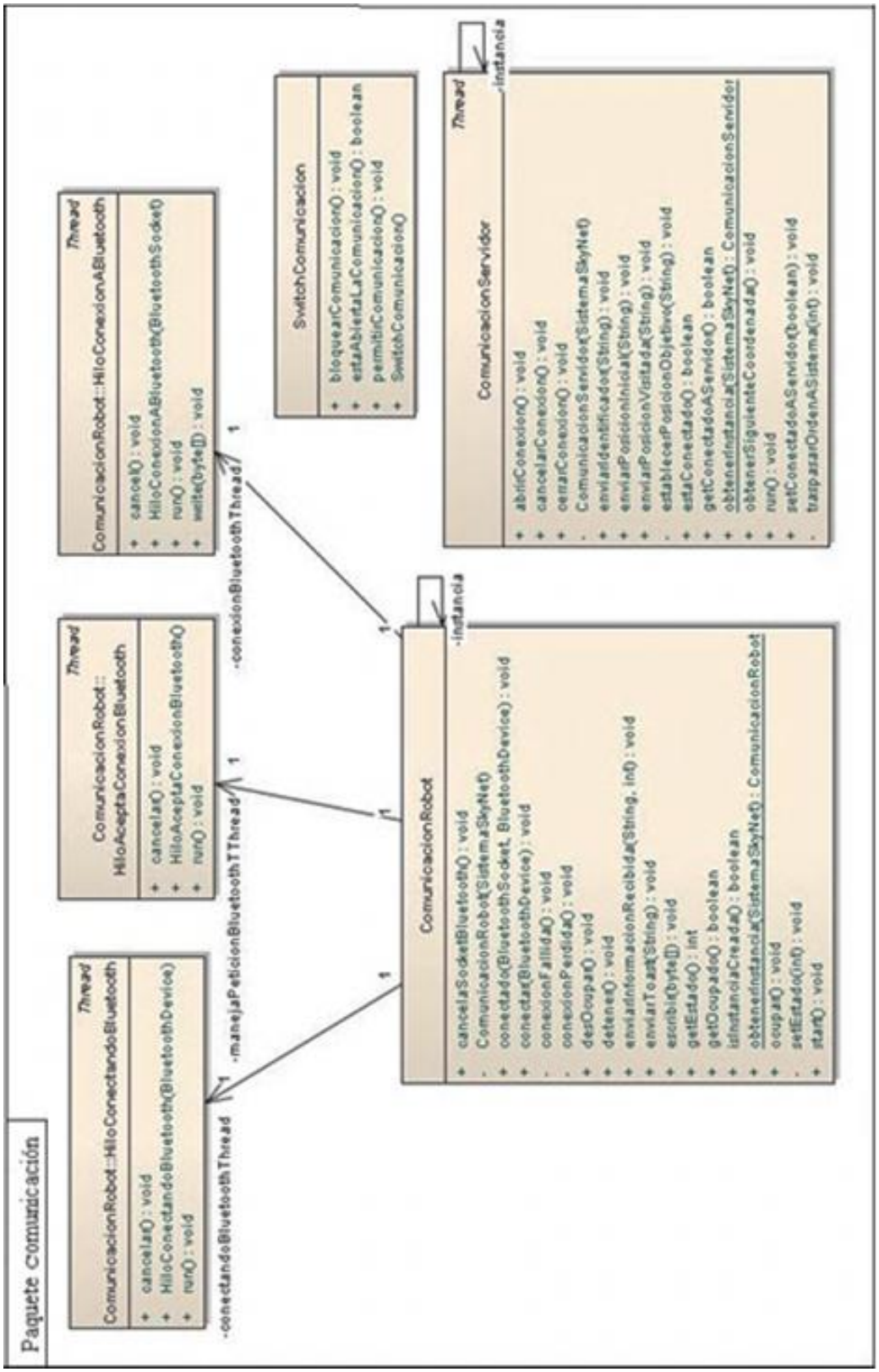


Figura 6.8 Diagrama de clases del paquete comunicación.

La Figura 6.9 presenta el diagrama de clases del paquete *vistas* de SkyNet módulo Android.

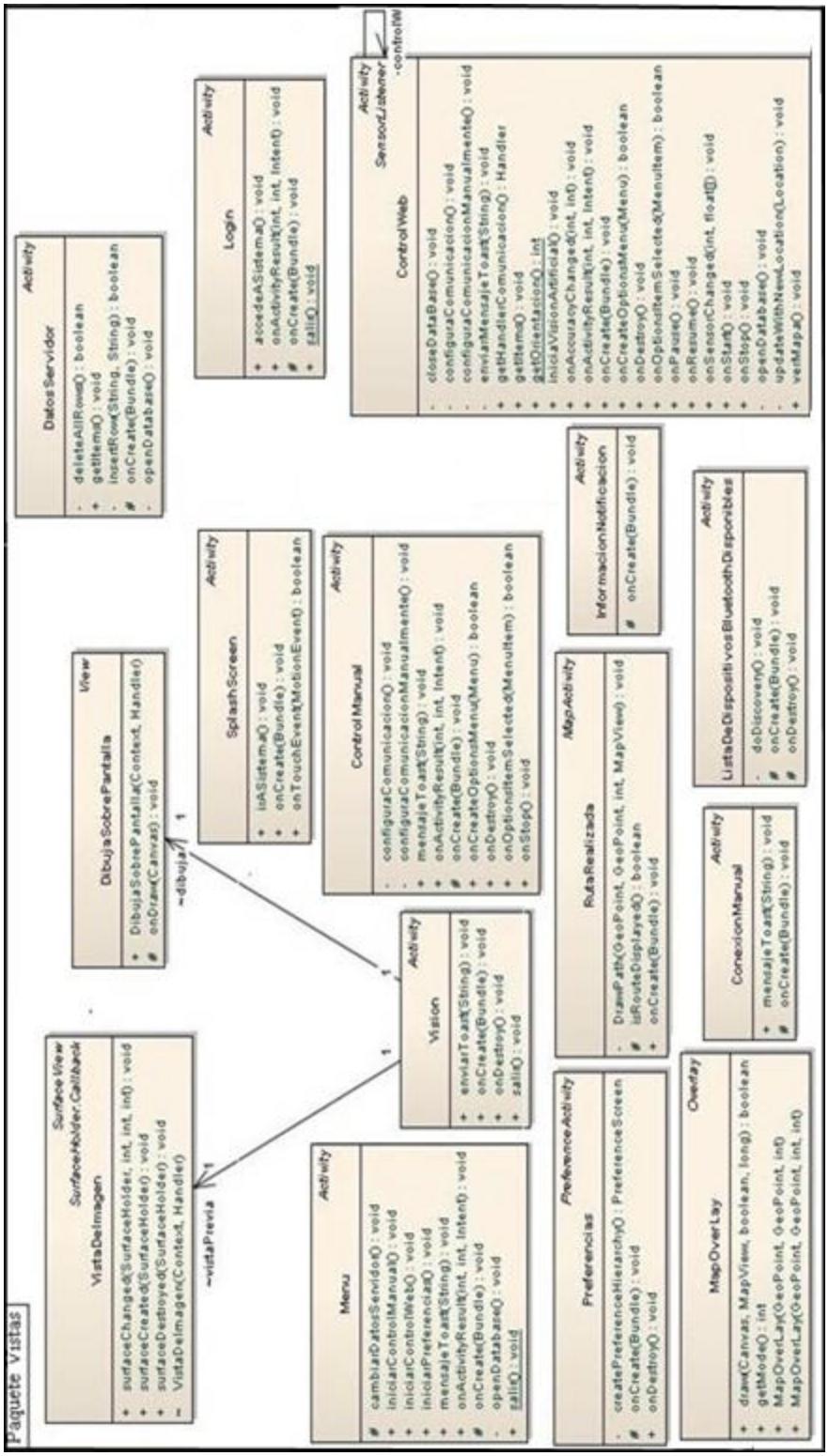


Figura 6.9 Diagrama de clases del paquete vistas.

La Figura 6.10 ilustra un diagrama aclaratorio de la interacción entre clases de diferentes paquetes. Es importante destacar que estas clases tienen interacción con otras clases, interacción que se puede apreciar en los diagramas de cada paquete. Los elementos del paquete *vistas* corresponden en su mayoría a *Activities* de Android que no tienen relación directa con la clase controladora *SistemaSkyNet*, sin embargo, estas operan la Base de datos y las preferencias de la aplicación, elementos que son utilizados por las actividades *ControWeb* y *ControlManual*.

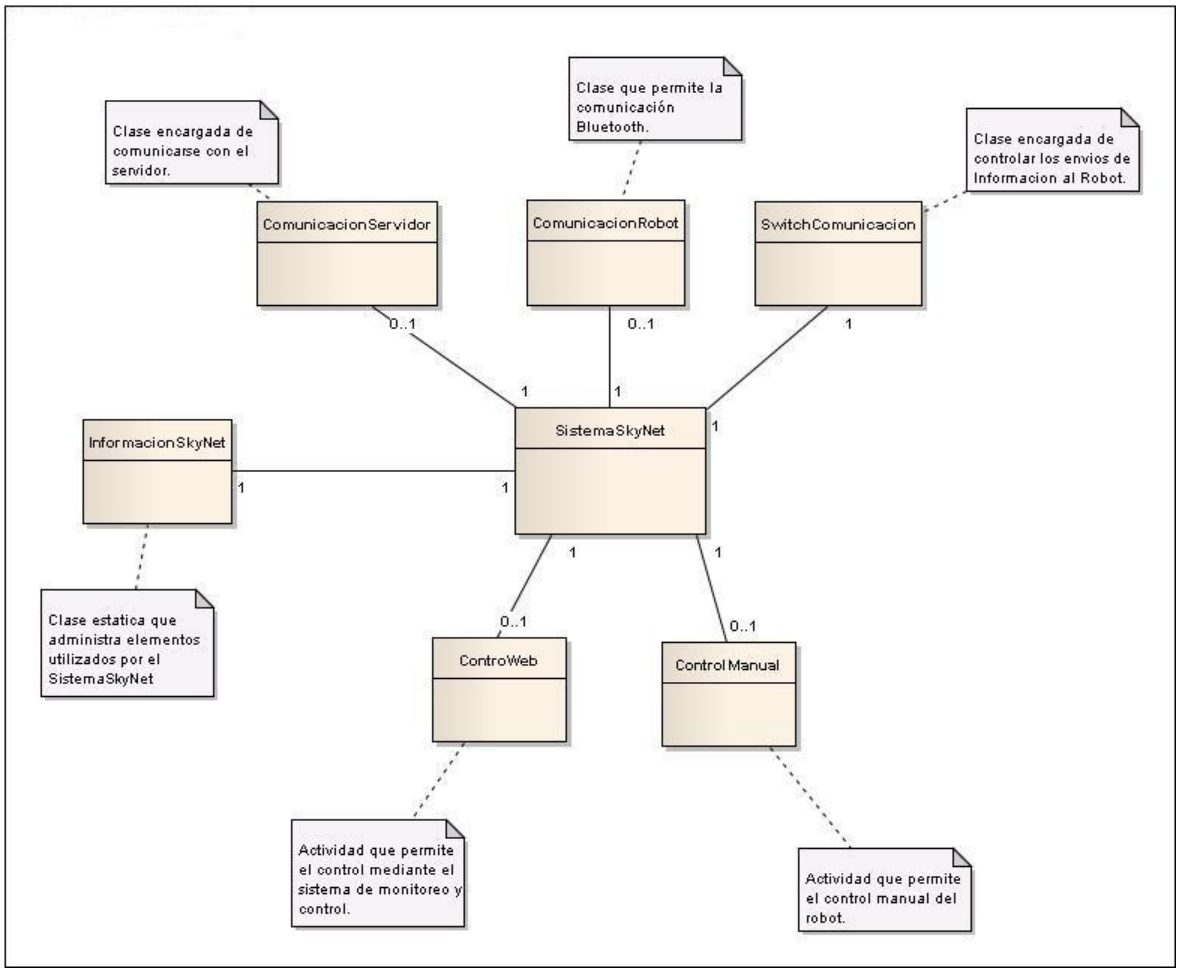


Figura 6.10 Diagrama de interacción entre paquetes.

6.5.7 Diagrama de Secuencia de sistema

El diagrama de secuencia es una representación que muestra, en determinado escenario de caso de uso, los eventos generados por actores externos, y su orden y los eventos internos del sistema.

Los actores involucrados dependiendo del caso de uso son: Usuario SkyNet y SkyBot. Para mayor detalle de los actores del módulo Android vea el Punto 6.5.2.

6.5.7.1 Diagrama de secuencia de sistema del método conectar

En la Figura 6.11 se presenta el diagrama de secuencia del método *conectar (dispositivoBluetooth),* el cual se encarga de iniciar la conexión a un dispositivo bluetooth.

Tal como se aprecia en la Figura 6.11 existen tres recuadros que representan alternativas, los dos primeros recuadros indican que ante la eventualidad de que exista un intento de conexión en curso, ésta se cancela (llamada 1 de la Figura 6.11) y anula el thread encargado de realizar la conexión. El último recuadro *alt* indica que si existe una conexión establecida, al igual que en los dos recuadros anteriores, se cancela la conexión (llamada 2 de la Figura 6.11) y se anula el thread encargado de manejar esta conexión.

Luego para realizar la conexión a un dispositivo bluetooth, se crea una nueva instancia del thread *HiloConectandoBluetooth*, este thread recibe como parámetro en su constructor el dispositivo con el cual se desea establecer comunicación. Finalmente, se invoca al método *start* (llamada 4 de la Figura 6.10) para que arranque el nuevo thread y se establece como estado de la comunicación *ESTADO_CONECTANDO* (llamada 5 de la Figura 6.11).

Tal como se ha podido apreciar existen tres threads encargados de aceptar las peticiones de conexión, realizar conexiones, y manejar la conexión. Pero no todos permanecen constantemente en ejecución, solo se utilizan en determinados momentos y funcionan como una línea de producción donde uno da paso al otro, hasta alcanzar la comunicación con el robot. Para mejorar la comprensión se recomienda la lectura de la especificación de casos de uso correspondientes al módulo Android de la aplicación SkyNet, para ello ver Anexo I.

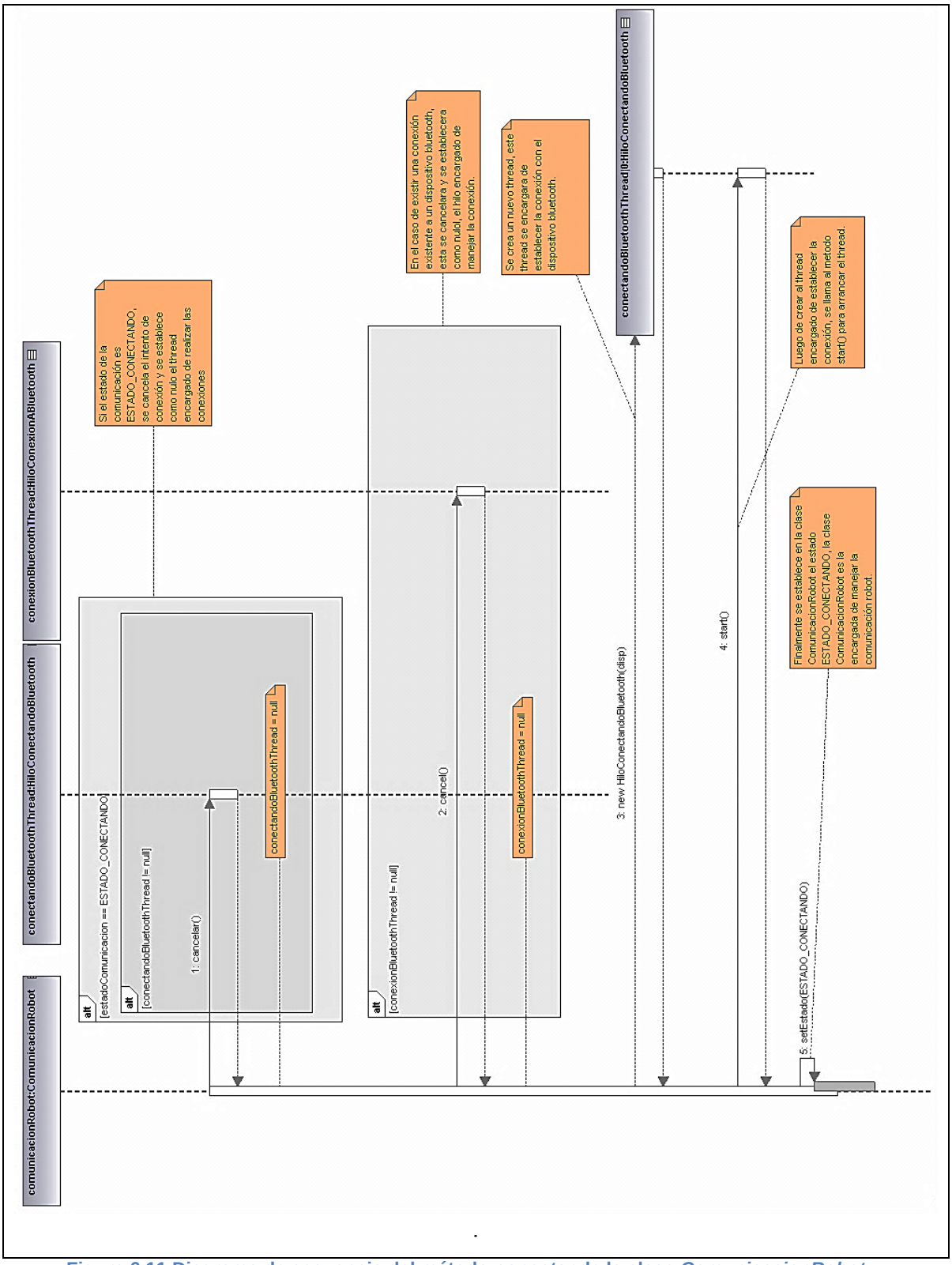


Figura 6.11 Diagrama de secuencia del método *conectar* de la clase *ComunicacionRobot*.

6.5.7.2 *Diagrama de secuencia de sistema del método escribir*

La Figura 6.12 muestra el diagrama de secuencia del método *escribir*, método que pertenece a la clase *HiloConexionABluetooth*, *thread* encargado de enviar información hacia el receptor bluetooth, en este caso el robot. El envío y recepción de información mediante bluetooth se realiza de manera serial.

Como se puede apreciar en el recuadro *alt,* el cual representa la alternativa que indica la inexistencia de una conexión a un dispositivo bluetooth, si esta alternativa se cumple, el envío de información se cancela. En caso contrario se asigna a una variable el hilo encargado de la conexión y se invoca al método *write* (llamada 1, de la Figura 6.12*)* , método que recibe como parámetro la información a enviar mediante el bluetooth, una vez recibida la información la transforma a bytes y la envía de manera serial por el *Stream* de salida.

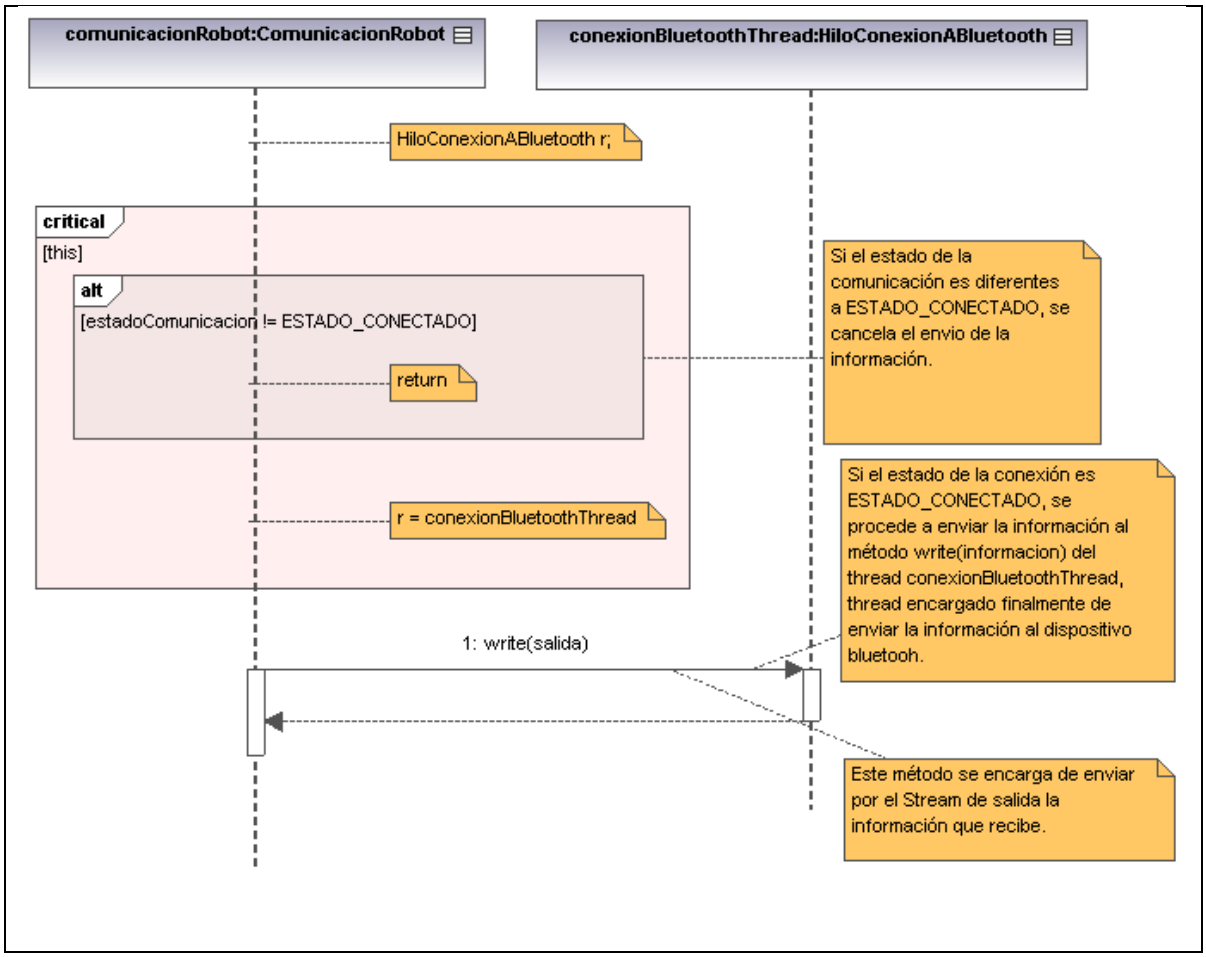


Figura 6.12 Diagrama de secuencia método *escribir*.

6.5.7.3 Diagrama de secuencia del método moverDerecha

Tal como se puede apreciar en la Figura 6.13, el método *moverDerecha* contiene un ciclo (recuadro *loop* de la Figura 6.13) el cual posee como condición de término la respuesta del método *verificarCondicionesDerecha* (llamada 1 de la Figura 6.13), éste método verifica si el robot posee la orientación deseada. De no estar en la orientación deseada envía la orden al robot para que se mueva a la derecha (llamada 2 de la Figura 6.13).

SkyNet módulo Android no solo considera los movimientos a la derecha para alcanzar la orientación deseada, dado que puede ser movimientos a la izquierda o derecha dependiendo de cuál represente un menor número de movimientos considerando la orientación actual y la deseada.

Una vez alcanzada la orientación deseada se muestra en la pantalla un mensaje *Toast*, mensaje que tiene la cualidad de aparecer por un determinado tiempo y luego desaparecer (llamada 3 de la Figura 6.13). Luego se libera la comunicación, se envía la orden para que el robot avance y se establece cómo falso la existencia de cambios en la posición, esto se puede apreciar en la llamada 4, 5 y 6 respectivamente, de la Figura 6.13.

De esta forma se avanza a la espera de una actualización de la localización geográfica, este movimiento se realiza en base al compass del dispositivo móvil de manera de mantener la orientación deseada el robot.

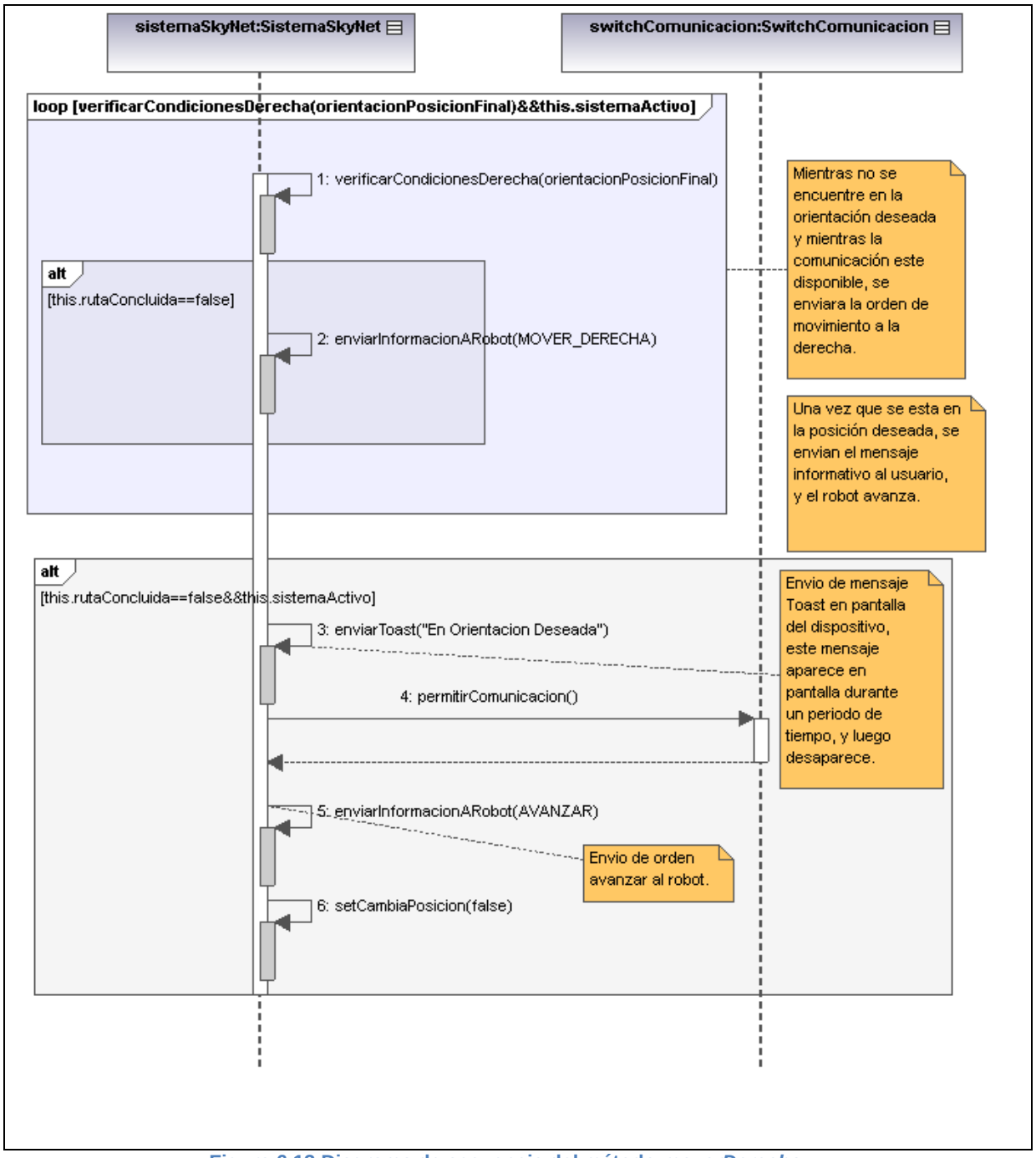


Figura 6.13 Diagrama de secuencia del método *moverDerecha*.

6.5.8 Diagramas de comunicación

En el Lenguaje Unificado de Modelado (UML) 2.0, un diagrama de comunicación es una versión simplificada del diagrama de colaboración de la versión de UML 1.x. [24].

Un diagrama de comunicación modela las interacciones entre objetos o partes en términos de mensajes en secuencia. Los diagramas de comunicación representan una combinación de información tomada desde el diagrama de clases, secuencia, y diagrama de casos de uso describiendo tanto la estructura estática como el comportamiento dinámico de un sistema [24].

Los diagramas de comunicación y de secuencia describen información similar, y con ciertas transformaciones, pueden ser transformados unos en otros sin dificultad [24].

Para mantener el orden de los mensajes en un diagrama de comunicación, los mensajes son etiquetados con un número y ubicados cerca del enlace por el cual se desplaza el mensaje. Leer un diagrama de comunicación conlleva comenzar en el mensaje 1.0, y seguir los mensajes desde un objeto hasta el siguiente [24].

6.5.8.1 *Diagrama de comunicación del método conectar*

La Figura 6.14 presenta el diagrama de comunicación del método *conectar*, el cual se ha detallado en el punto 6.5.7.1.

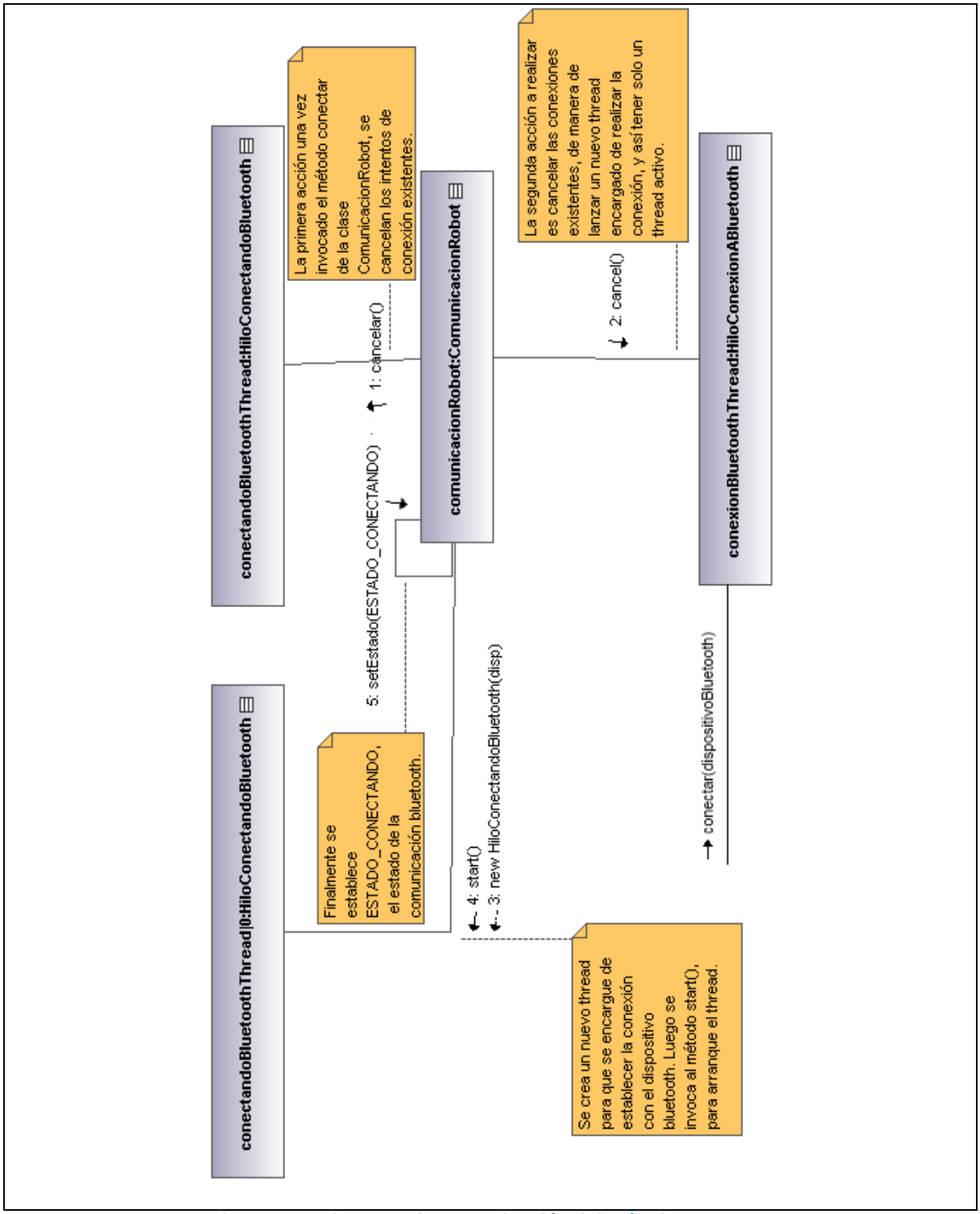


Figura 6.14 Diagrama de comunicación del método *conectar*.

6.5.8.2 Diagrama de comunicación del método escribir

La Figura 6.15 presenta el diagrama de comunicación del método *escribir*, método encargado de realizar un primer apronte para el envió de la información, mediante la comunicación bluetooth. Tal como se ha descrito en el punto 6.5.7.2, éste método luego de asegurar las condiciones para el envió de la información, invoca al método *write* el cual finalmente envía la información en bytes, utilizando el *Stream* de salida.

Este *Stream* se crea una vez establecida la comunicación con un dispositivo bluetooth. Existen dos *Stream:* uno dedicado a la recepción de información por medio del bluetooth y otro encargado del envió de información.

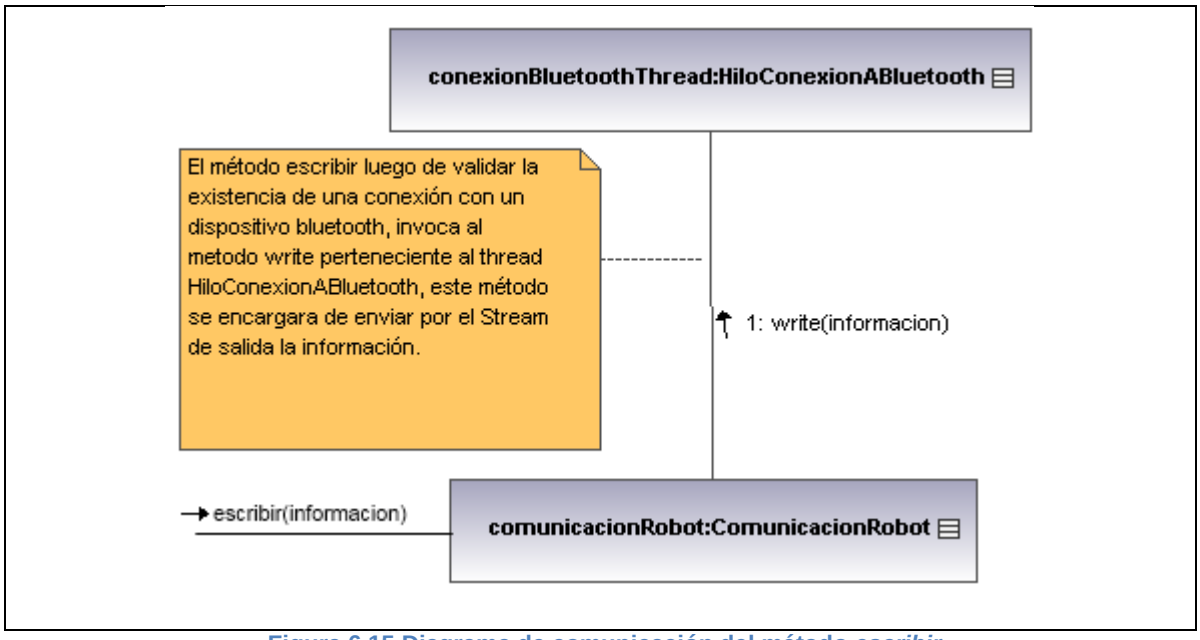


Figura 6.15 Diagrama de comunicación del método *escribir*.

6.5.8.3 *Diagrama de comunicación del método moverDerecha*

En la Figura 6.16 se presenta el diagrama de comunicación del método *moverDerecha*, este método se ha descrito anteriormente en el punto 6.5.7.3.

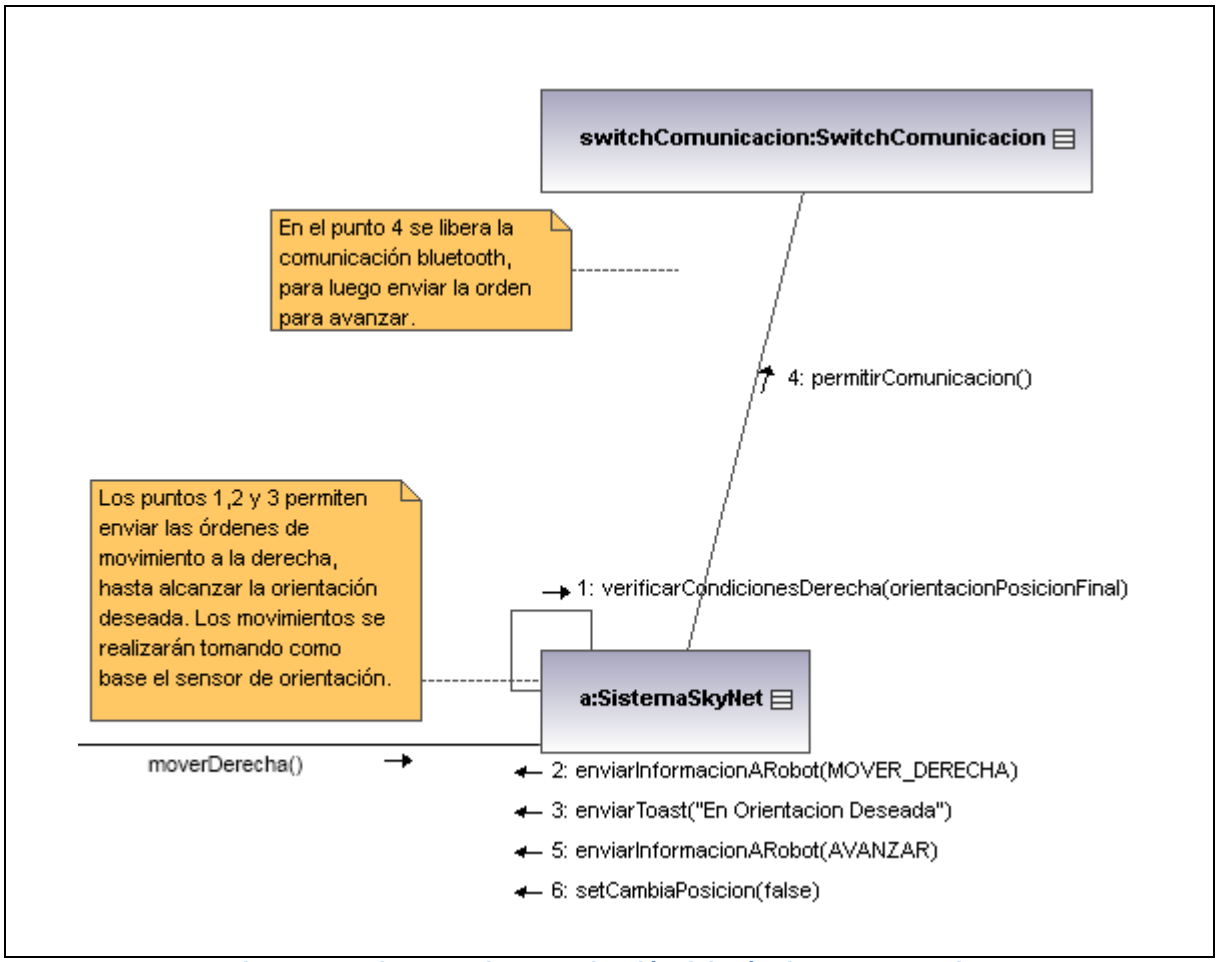


Figura 6.16 Diagrama de comunicación del método *moverDerecha*.

6.5.9 Patrones de Diseño

Los patrones de diseño entregan solución a problemas que se han repetido constantemente en el tiempo. Para que una solución sea considerada patrón se debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Y además debe ser re-utilizable lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias [25].

El desarrollo de la aplicación Android considera el uso de 2 patrones, los cuales se detallan a continuación.

6.5.9.1 Patrón modelo vista controlador (MVC)

El modelo vista controlador, también conocido como MVC es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica en tres componentes distintos [25].

Este patrón para la arquitectura de aplicaciones de software fue descrito por primera vez en el año 1979 por Trygve Reenskaug, cuando trabajaba en los laboratorios de Investigación de Xerox. La Figura 6.17, presenta una imagen representativa del patrón Modelo Vista Controlador [25].

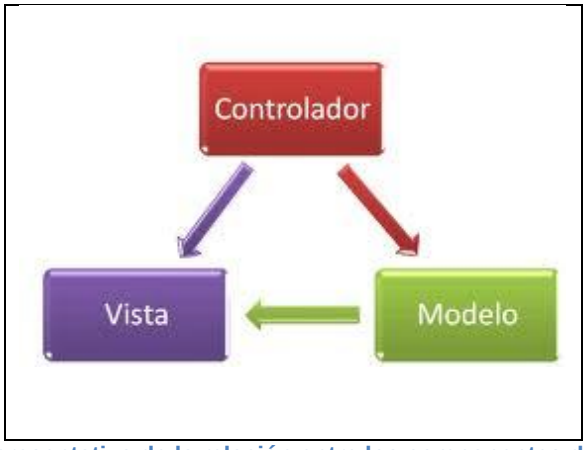


Figura 6.17 Imagen representativa de la relación entre los componentes del patrón Modelo Vista Controlador [25].

6.5.9.2 Singleton

El patrón de diseño Singleton o de instancia única, está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención es garantizar que una clase tenga solo una instancia y proporcionar un punto de acceso global a ella [25].

La Figura 6.18, presenta una imagen representativa del patrón Singleton.

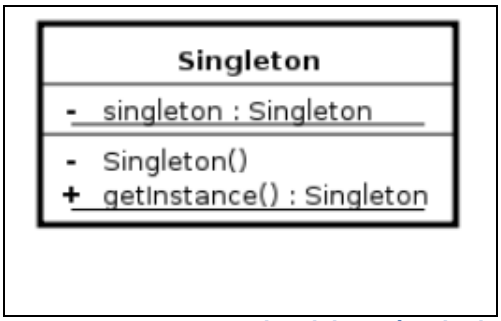


Figura 6.18 Imagen representativa del patrón Singleton [25].

6.5.10 Mapa de Navegación

Los mapas de navegación permiten explicar la disposición de las actividades o vistas de la aplicación, de manera que el usuario visualice el lugar donde se encuentra y verifique las actividades a realizar para dar cumplimiento a su deseo.

La Figura 6.19 presenta el mapa de navegación de la aplicación SkyNet módulo Android. En esta figura cada rectángulo representa una vista, vista que en Android se le da el nombre de *Actividad o Activity*.

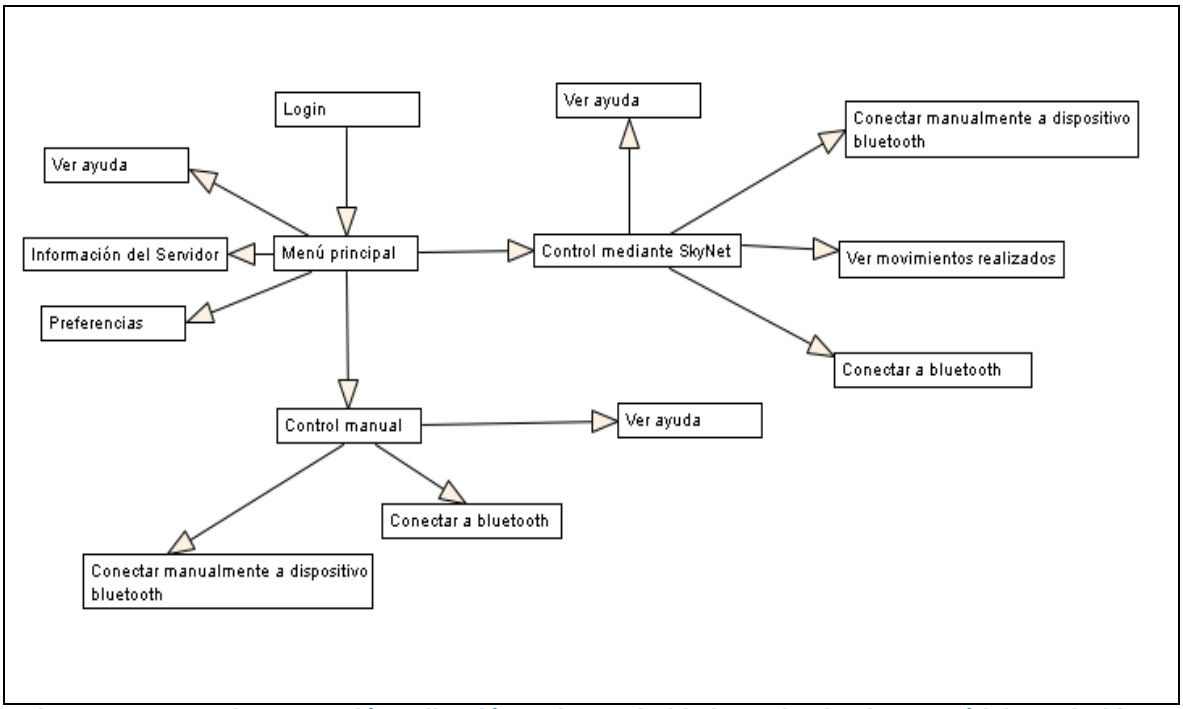


Figura 6.19 Mapa de navegación aplicación nativa Android, denominada SkyNet módulo Android.

6.5.10.1 Detalle Mapa de Navegación

A continuación se describe las actividades pertenecientes a la aplicación SkyNet módulo Android.

- **Login:** Es la primera vista que se presenta al usuario al momento de seleccionar la aplicación, en esta vista el usuario deberá ingresar la clave de acceso a la aplicación (su valor por defecto es 123). De ser válida la clave ingresada tendrá acceso al resto de opciones disponibles que se detallan en los puntos siguientes.
- **Menú principal:** Es la vista que presenta todas las opciones disponibles para el usuario.
- **Información del servidor:** Vista en la cual se presenta la información del servidor (Dirección IP y puerto), además se puede cambiar esta información si fuese necesario.
- **Ver ayuda:** Se presenta información de ayuda, y que tiene directa relación con la actividad en curso.
- **Control manual:** Vista que permite el control del robot de manera manual, mediante la selección de botones direccionales. Para el funcionamiento de esta opción se debe conectar anteriormente a un dispositivo bluetooth.
- **Conectar a bluetooth:** Esta opción permite realizar la conexión a un dispositivo bluetooth mediante la selección de un dispositivo de una de dos listas, las cuales pueden ser: lista de dispositivos antes conectados y la lista de nuevos dispositivos.
- **Conectar manualmente a dispositivo bluetooth:** Opción en la cual se ingresa la dirección MAC del dispositivo bluetooth con el cual se desea conectar, ante la invisibilidad del dispositivo en la opción "Conectar a Bluetooth" (esto dado que algunos dispositivos es posible configurarlos de manera que no sean visibles por otros dispositivos bluetooth).
- **Control mediante SkyNet:** Opción que permite recibir las órdenes del módulo Web de SkyNet, es decir, recibir una ruta que contiene coordenadas a visitar. Una vez que se ha concluido la ruta seleccionada por el usuario, el robot automáticamente comienza a buscar objetos de un determinado color y los

traslada a un lugar demarcado con otro color. Los movimientos realizados por el robot se ven representados en SkyNet módulo web en un mapa.

6.5.11 Conexión mediante socket TCP/IP, para la comunicación SmartPhone Servidor

La comunicación entre el SmartPhone y servidor se realiza mediante sockets TCP/IP, estos sockets son los encargados de permitir la comunicación entre el servidor (Sistema de monitoreo denominado SkyNet modulo Web) y el cliente, que en éste caso serán los robots existentes los cuales tienen como nombre *Robot negro* y *Robot amarillo*.

Los sockets son una interfaz normalizada de comunicación entre procesos, utilizado inicialmente en la arquitectura TCP/IP. Cada extremo se individualiza mediante una dirección IP o nombre DNS de la máquina y el número de puerto por el cual se recibe la información, que puede adquirir un valor entre 1 y 65535. El socket tiene como objetivo permitir que un proceso pueda enviar o recibir datos a través de la red [26].

Los sockets orientados a la conexión (TCP) son sockets donde, antes de la transferencia de datos, existe una fase de establecimiento de conexión, se establece un canal dedicado para la comunicación entre el cliente y servidor. Los datos en esta comunicación se reciben uno tras otro. Otra característica importante es la utilización de un canal de salida y uno de entrada, representados por objetos *InputStream* y *OutputStream* [27].

6.5.12 Procesamiento de imágenes digitales

Para la detección de colores y masas mediante la utilización del SmartPhone Android, se ha utilizado el procesamiento de imágenes digitales sencillo. La utilización de esta visión artificial sencilla se realiza sin el apoyo de librerías que optimicen y apoyen el trabajo, dado que las librerías existentes para Android se encuentran disponibles desde la versión de SDK 2.1 en adelante, y el común de los SmartPhone hasta ahora existentes en el país cuenta con Android 1.5.

A continuación se presenta el formato de imagen utilizada por los SmartPhone Android.

6.5.12.1 El estándar YUV

El estándar YUV (también conocido como CCIR 601), es un modelo de representación del color dedicado al video análogo. Se basa en un modo de transmisión de video con componentes separados, que utiliza tres cables (o variables) diferentes para llevar información con respecto a los componentes de luminosidad y los dos componentes de crominancia (color). Es el formato utilizado en las normas PAL (Phase Alternation Line) y SECAM (Séquentiel Couleur À Mémoire). El parámetro “Y” representa la luminiscencia, esto quiere decir que entrega la información correspondiente a negro o blanco, mientras que “U” y “V” representan la crominancia, es decir la información con respecto al color. Éste modelo se desarrolló para permitir la transmisión de información a color en televisores a color y, a la vez, garantizar que los televisores blanco y negro existentes continuaran mostrando una imagen en tonos grises. [28]

El modelo YUV está más próximo al modelo humano de percepción que el estándar RGB usado en el hardware de gráficos por ordenador, pero no tan cerca como el espacio de color HSL y espacio de color HSV.

Capítulo VII

Implementación del robot y aplicación nativa Android

Para profundizar el entendimiento de la aplicación SkyNet módulo Android se ha dedicado este apartado a la implementación de éste módulo. Se explica cómo fueron desarrollados algunos de los aspectos más importantes del módulo Android, tales como la utilización de los sensores, diseño de interfaz, y detección de colores y seguimiento de estos.

7.1 Construcción del robot

La construcción del robot se ha realizado acorde a lo señalado en la Figura 7.1, respetando las medidas indicadas.

Ante los requisitos capturados se planteó un primer diseño, el cual por razones de estabilidad, tamaño, problemas con las orugas (cadenas que emulan a las ruedas) y su tracción, se decide dar paso a un nuevo diseño. Las imágenes del primer diseño se presentan en la Figura 7.1.

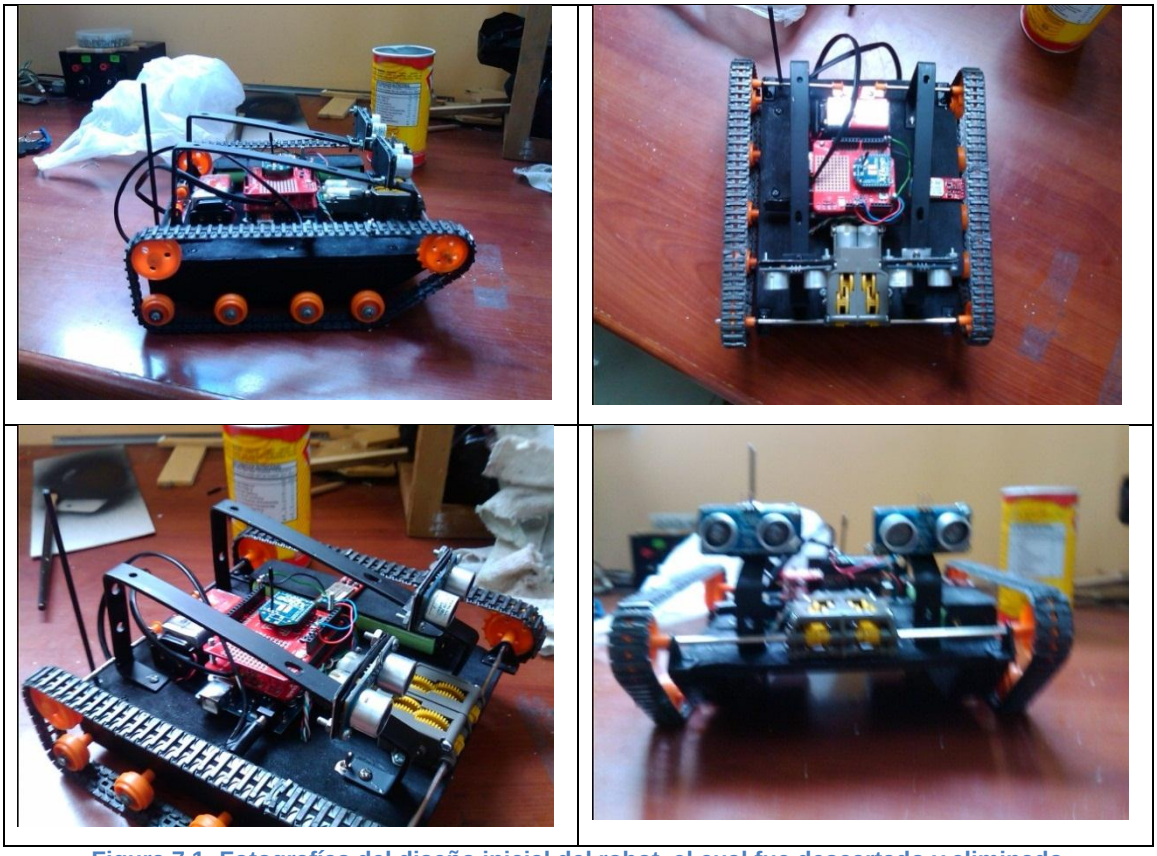


Figura 7.1: Fotografías del diseño inicial del robot, el cual fue descartado y eliminado

Para dar solución a los problemas mencionados anteriormente, se realiza un modelo con orugas y una base acrílica sobre la cual se encuentra la pinza o brazo que permite tomar los objetos. Además se incorporan nuevas modificaciones físicas que se describen a continuación:

- Modificación de la caja de reducciones de manera de otorgar el triple de fuerza que la versión anterior
- Rieles que protegen las orugas y evitan descarrilamientos de estas
- Soporte adaptable para SmartPhone (Regulable según el SmartPhone)
- Plataforma acrílica sobre los circuitos y micro-controladores para su protección
- Inclusión de una nueva línea de orugas de manera de duplicar el ancho de estas y mejorar la adherencia.
- Aumento en altura del robot.
- Incorporación de otra caja reductora, la cual se suma al aumento de fuerza de cada caja reductora.

Estas modificaciones dan lugar al actual modelo de robot que es capaz de operar en ambientes diversos y adversos (de acuerdo a su tamaño).

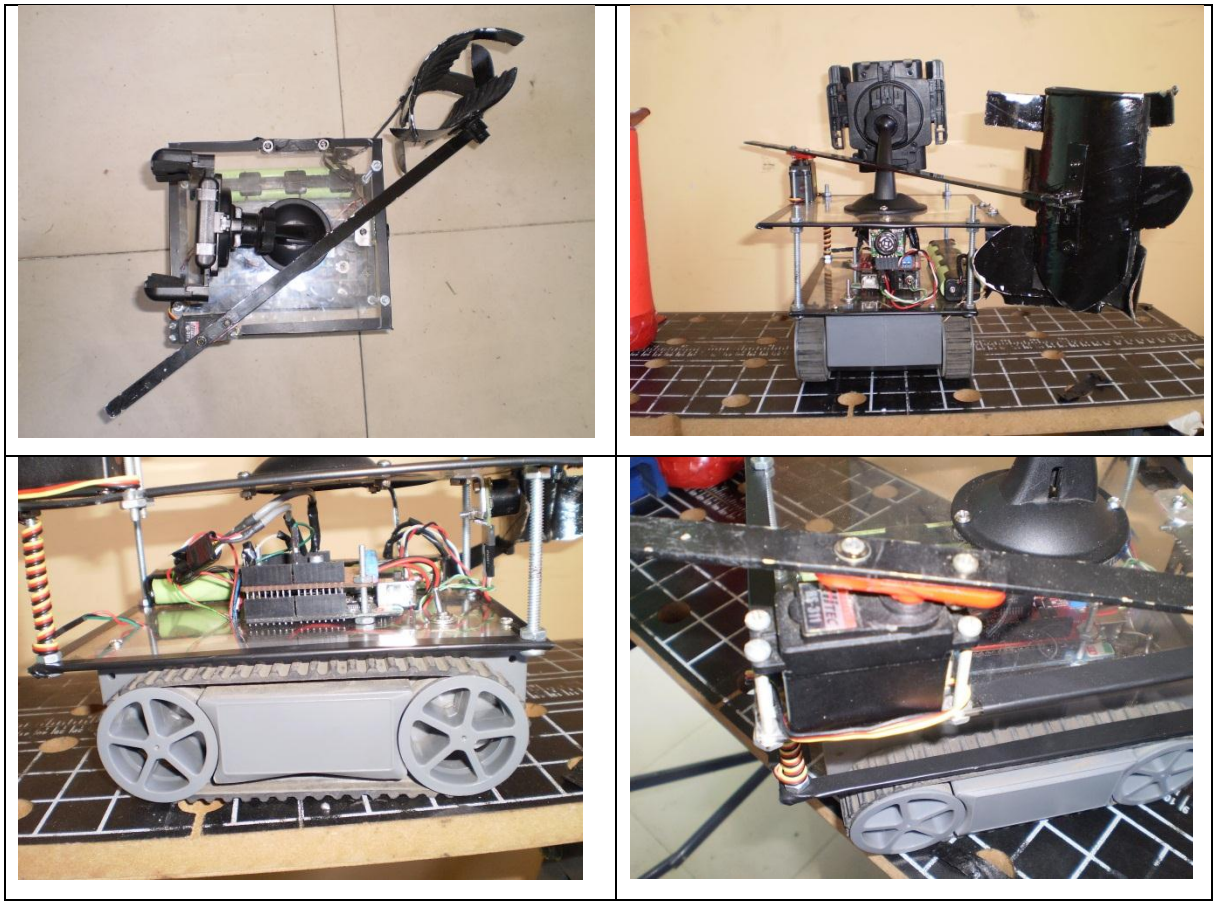
La Figura 7.2 presenta el segundo diseño del robot, así como su construcción. Este diseño mejoraba el funcionamiento de la primera versión pero continuo con problemas en sus orugas, las cuales se salían de sus ejes cuando se operaba en terrenos de gran adherencia.

Figura 7.2: Segundo diseño del robot, el cual fue desechado.

Finalmente se adquiere un chasis marca Pololu (modelo RP5-CH02), este chasis posee reducciones de gran potencia las cuales permiten operar en diferentes terrenos sin notar una diferencia en el desplazamiento. A su vez se adquiere un sensor ultrasónico Maxbotix (modelo LV-EZ3) que es compatible con Arduino, sensor que mejoro ostensiblemente la medición de distancias y funcionamiento durante largos periodos de tiempo. Este nuevo sensor permite medir con exactitud distancias mayores a 12 centímetros y menores a 3 metros, hecho que supera al sensor anterior (modelo Ping de la marca Parallax) que solo tenía precisión hasta los 40 centímetros.

Se reutilizan los micro-controladores, pilas, porta pilas, bases acrílicas y pinza. De esta forma se construye un robot similar en apariencia a los anteriores, pero con mejores componentes los cuales permiten un buen desempeño de la tarea asignada.

En la Figura 7.3 se presenta el robot final, el cual incorpora los elementos mencionados anteriormente.



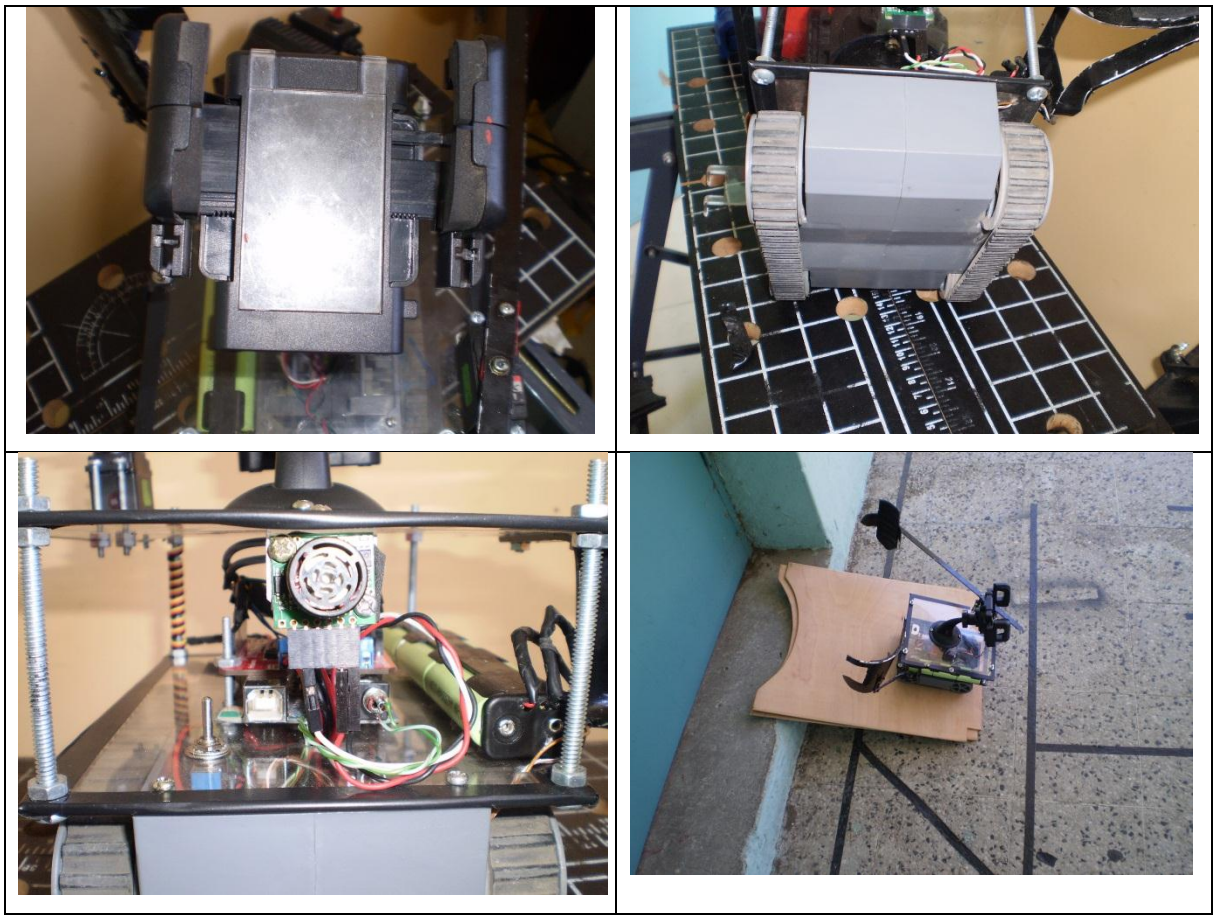


Figura 7.3 Diseño final del robot.

7.2 Desarrollo de la aplicación embebida sobre micro-controlador Arduino

El software Arduino ha sido desarrollado tomando en cuenta las consideraciones mencionadas en el Punto 6.3. Es por ello que el resultado considera movimientos básicos hacia adelante, atrás, giros a la izquierda, derecha y detenerse. A esto se suma la utilización de un sensor de distancia ultrasónico. La comunicación entre SmartPhone y robot es mediante bluetooth, basado en el envío y recepción de información de manera serial.

La Figura 7.4 presenta el código Arduino correspondiente a la función *adelante* del robot.

```
1 void adelante(){
2   analogWrite(pwm_a, 255);
3   analogWrite(pwm_b, 255);
4   digitalWrite(dir_a, HIGH);
5   digitalWrite(dir_b, HIGH);
6   Serial.print(1);
7 }
```

Figura 7.4 Función adelante (código Arduino).

Como se puede apreciar en la Figura 7.4, el método adelante establece la potencia en 255 (línea 2 y 3) que es el máximo permitido por el micro-controlador Arduino, y activa los motores estableciendo como *HIGH* las líneas de salida *dir_a* y *dir_b* (línea 4 y 5), la combinación de ambas líneas de salida en *HIGH* permite el movimiento de ambos motores hacia adelante. Finalmente la función adelante envía el código de confirmación de la operación por el puerto serial, el cual corresponde a "*1*" (línea 6).

7.3 Implementación de la aplicación nativa Android

7.3.1 Implementación de la interfaz

La implementación de la interfaz se ha realizado utilizando el plugin de Eclipse para Android, éste plugin permite realizar el diseño de la interfaz ya sea programando el código XML o mediante la tecnología *drag and drop*.

A continuación en la Figura 7.5 se presenta un fragmento del código XML correspondiente al menú de opciones de la aplicación, esta vista considera dimensiones, imágenes y elementos a utilizar en la aplicación.

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3    android:orientation="vertical" android:layout_width="fill_parent"
4    android:layout_height="fill_parent"
5    android:background="@drawable/fondo_de_pantalla">
6      <TextView android:layout_width="wrap_content"
7      android:layout_height="wrap_content"
8       android:layout_gravity="center"
9       android:id="@+id/titulo_menu_principal"
10      android:layout_marginTop="60px" android:text="Menú">
11     </TextView>
12     <ImageButton android:layout_width="wrap_content"
13      android:layout_height="wrap_content"
14      android:layout_gravity="right"
15      android:background="@drawable/pregunta"
16      android:id="@+id/ayuda_menu_principal"
17      android:layout_marginRight="15px">
18     </ImageButton>
19     …
20  </LinearLayout>
```

Figura 7.5 Código ejemplo, correspondiente al menú de la aplicación.

Como se puede apreciar en la Figura 7.5 y línea 2, el contenedor de la vista elegido es de tipo *LinearLayout* el cual por defecto posee una orientación horizontal. Al incluir *android:orientation=”vertical”* se indica que cada elemento se ubicará de forma horizontal y a su vez uno debajo de otro.

Los demás elementos existentes en el código son botones (*ImageButton* específicamente, consiste en un botón representado por una imagen), y *TextView* los cuales son conocidos en otros lenguajes. A estos elementos se aplican las propiedades de *android:loyout_width* y *android:layout_height* para indicar el ancho y alto de los componentes, también pudiese ser *wrap_content*, que indica que utilice el menor espacio posible o de tipo *fill_parent*, que es el mayor espacio posible [29].

El código XML anterior genera la siguiente interfaz, la cual se presenta en la Figura 7.6.

Figura 7.6 Vista del menú de la aplicación Android.

7.3.2 Manifest XML y permisos de la aplicación

Éste archivo XML se genera automáticamente al crear un proyecto Android y en él se declaran todas las especificaciones de la aplicación. Cuando hablamos de especificaciones hacemos mención a las *Activities* utilizadas, los *Intents*, bibliotecas, el nombre de la aplicación, el hardware que se necesitará, los permisos de la aplicación, el SDK mínimo para la aplicación y el requerido.

Los componentes del Manifest son:

- **<manifest>**: Tag raíz, dentro de él se pueden contienen todos los atributos de la aplicación, los cuales se describen en los puntos posteriores.
- **<application>**: Tag que engloba a las *Activities*, *Services*, *Providers*, Receivers y las bibliotecas que se usan en la aplicación.
- **<activity>** Se tendrá una *Activity* por cada vista de la aplicación y dentro de él van sus atributos y los distintos *Intents* para comunicarse.
- **<supports-screens>**: Éste tag es utilizado para describir el tipo de pantallas soportadas por la aplicación.
- **<uses-permissions>**: Mediante éste tag se especifican los permisos otorgados a la aplicación para que esta se ejecute, además son los que deberá aceptar el usuario antes de instalarla. Por ejemplo, si se desea utilizar funcionalidades con Internet o el vibrador del teléfono, hay que indicar que la aplicación requiere esos permisos.
- **<uses-sdk>**: En éste tag se determinan las distintas versiones Android que va a utilizar nuestra aplicación, tanto sobre qué versiones va a correr como qué versión fue utilizada para realizar nuestras pruebas. Mediante el atributo *android:minSdkVersion* establecemos a partir de qué versión de Android la aplicación podrá correr.

7.3.3 Permisos de SkyNet módulo Android

SkyNet módulo Android necesita permisos para poder utilizar librerías y servicios del sistema operativo, es por ello que estos permisos se declaran en el *Manifest*. Para que el sistema operativo Android facilite a la aplicación los recursos ahí descritos.

Dentro de los recursos solicitados en los permisos del *Manifest* nos encontramos con permisos de Internet, Bluetooth, GPS mediante Satélites, Cámara, Led, Vibrador.

A continuación en la Figura 7.7 se presentan el fragmento de código del *Manifest* donde se otorgan los permisos a la aplicación, permisos que el usuario debe aceptar al momento de instalar la aplicación en el Smartphone.

```
1   <uses-permission android:name="android.permission.INTERNET"></uses-permission>
2   <uses-permission android:name="android.permission.BLUETOOTH"></uses-permission>
3   <uses-permission android:name="android.permission.BLUETOOTH_ADMIN">
4   </uses-permission>
5   <uses-permission android:name="com.example.bluetooth.BACKPORT_BLUETOOTH">
6   </uses-permission>
7   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
8   <uses-permission android:name="android.permission.CONTROL_LOCATION_UPDATES">
9   </uses-permission>
10  <uses-permission android:name="android.permission.CAMERA"></uses-permission>
11  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE">
12  </uses-permission>
13  <uses-permission android:name="android.permission.FLASHLIGHT"></uses-permission>
14  <uses-permission android:name="android.permission.VIBRATE"></uses-permission>
15  <uses-feature android:name="android.hardware.camera"></uses-feature>
16  <uses-sdk android:minSdkVersion="3"/>
```

Figura 7.7 Permisos solicitados en el Manifest para SkyNet módulo Android.

7.3.4 Uso de los sensores

SkyNet módulo Android basa su funcionamiento en la utilización de los sensores de orientación y el GPS para determinar la posición donde se encuentra el robot y las acciones a realizar para lograr ir de un lugar a otro.

Android provee elementos para la utilización de estos sensores, como es el caso de *LocationManager*, y la interfaz *SensorListener*. Es importante destacar que en los SmartPhone Android, generalmente, se encuentran presentes los sensores localización (GPS), sensor de orientación y el acelerómetro.

7.3.5 Utilización del GPS

El uso del GPS en Android se realiza gracias a la utilización de *LocationManager*. Esta clase proporciona acceso a los servicios del sistema de localización. Estos servicios permiten a las aplicaciones obtener actualizaciones periódicas de la situación geográfica del dispositivo, o para arrancar una aplicación específica (*Intent*) cuando el dispositivo entra en la proximidad geográfica de un lugar determinado.

La localización es determinada mediante la triangulación de satélites, de antenas celulares, o mediante conexiones WIFI. Sin embargo dado que el robot operará en ambientes abiertos, se ha decidido utilizar solo la triangulación mediante satélites, hecho que se ve reflejado en que solo se ha dado el permiso para ese tipo de determinación de localización en el archivo *Manifest*. La Figura 7.8 muestra el permiso para esta triangulación, que es la más precisa de todas.

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

Figura 7.8 Permiso de utilización de la triangulación por satélites.

Por otro lado, es necesario configurar el funcionamiento de éste servicio, esto se obtiene de la manera señalada en la Figura 7.9.

```
LocationManager locationManager = (LocationManager) getSystemService(context);
```

Figura 7.9 Obtención del servicio de localización.

El funcionamiento del servicio de localización se debe configurar, para ello se utiliza un objeto de la clase *Criteria*, clase que indica los criterios a considerar para la selección de un proveedor de localización. Los proveedores consideran la precisión, uso de energía, capacidad de informar altitud, velocidad, y el costo monetario de la determinación de la posición en cada caso (si es que el proveedor es de pago).

La Figura 7.10 presenta la configuración realizada para la obtención del proveedor de localización.

```
1    Criteria criteria = new Criteria();
2    criteria.setSpeedRequired(true);
3    criteria.setAccuracy(Criteria.ACCURACY_FINE);
4    criteria.setAltitudeRequired(true);
5    criteria.setBearingRequired(true);
6    criteria.setPowerRequirement(Criteria.POWER_LOW);
```

Figura 7.0.10 Configuración del proveedor de localización.

Una vez establecida la configuración se debe obtener el proveedor de localización deseado para la configuración establecida. Es por ello que se utiliza la clase *LocationManager* la cual proporciona servicios para la localización. Además se utiliza la configuración antes mencionada en la Figura 7.10, para solicitar el mejor proveedor disponible que satisfaga dicha configuración.

La Figura 7.11 presenta el fragmento de código representativo de lo descrito anteriormente.

```
String provider = locationManager.getBestProvider(criteria, true);
```

Figura 7.11 Selección del proveedor de localización.

Para establecer la posición actual, una buena base es obtener la última localización detectada por el proveedor. Para ello se utiliza el objeto *locationManager* que es una instancia de *LocationManager*, y se realiza la llamada al método *getLastKnownLocation(proveedor)*, éste método retorna un objeto de tipo *Location*, éste objeto contiene la latitud, longitud, altitud, velocidad, entre otros parámetros.

Lo antes descrito se ilustra en la Figura 7.12.

```
Location posicionInicial = locationManager.getLastKnownLocation(provider);
```

Figura 7.0.12: Obtención de la última localización geográfica visitada

Por último, para mantener un ente encargado de monitorear los cambios en la localización geográfica se crea un objeto *Listener* denominado *cambiosEnLocalizacion*, y se configura el servicio de localización para que ante actualizaciones en la localización se envie la localización al *Listener cambiosEnLocalizacion*. Las actualizaciones se programan para que se detecten cada 1 milisegundo o cada 1 metro de recorrido (teóricamente, dado que en la práctica no es así). Para facilitar el entendimiento de lo anteriormente señalado se presenta al lector la Figura 7.13, la cual contiene la línea de código correspondiente.

```
locationManager.requestLocationUpdates(provider, 1,1,cambiosEnLocalizacion);
```

Figura 7.13: Fragmento de código que establece un *Listener* para la captura de actualizaciones de la posición.

7.3.6 Cálculo de distancia y bearing entre posiciones geográficas

El cálculo del bearing y distancia, permite a la aplicación determinar qué movimientos realizar para trasladar el robot de una posición geográfica a otra. Ambos cálculos se encuentran disponibles en el API de Android, más específicamente en la clase *Location*, clase que representa localizaciones geográficas en Android.

El bearing entre dos localización geográficas, corresponde a cuántos grados con respecto al norte se encuentra la localización de destino. En Android se calcula mediante la función *bearingTo*, la cual se aplica sobre dos objetos *Location* (representan localizaciones geográficas). En la Figura 7.14 se presenta la forma de utilización, mediante un fragmento de código de SkyNet módulo Android.

```
float grados = posicionActual.bearingTo(posicionObjetivo);
```

Figura 7.0.14: Cálculo de *bearing* entre dos localizaciones geográficas.

Para el cálculo de distancia la situación es similar al caso del cálculo de Bearing. El cálculo de distancias entre localizaciones está implementado en la clase *Location* de Android. Y se utiliza de la manera descrita por la Figura 7.14, correspondiente a un fragmento de código de SkyNet módulo Android.

```
float distancia = posicionActual.distanceTo(posicionObjetivo);
```

Figura 7.15: Cálculo de distancia entre dos localizaciones geográficas

En estos dos cálculos se basa la lógica de movimiento del robot, complementándose con el sensor de orientación, el cual permite fijar el rumbo, evitando desviarse de éste. El desplazamiento se produce hasta la distancia indicada por el método *distanceTo*, o hasta que se produzca un cambio en la localización GPS del robot.

7.3.7 Utilización del sensor de orientación

El sensor de orientación proporcionado por Android provee una interfaz llamada *SensorListener*, la cual proporciona los métodos *onSensorChanged(int sensor, float[] values)* y *onAccuracyChanged(int sensor, int accuracy)*. Estos métodos se invocan al momento de existir un evento que provoque el cambio en los valores correspondientes a ese sensor.

Una vez en ejecución el método se debe obtener el sensor del cual se ha generado el evento, para ello se pregunta por el tipo de sensor que generó el evento, utilizando la clase *SensorManager*, los sensores más utilizados y disponibles en Android son (presentación de acuerdo al nombre con el cual se identifican en Android):

- SENSOR_ACCELEROMETER.
- SENSOR_LIGHT.
- SENSOR_MAGNETIC_FIELD.
- SENSOR_ORIENTATION.
- SENSOR_PROXIMITY.

Los dispositivos Android disponen de una serie de sensores y dependerá de cada equipo los sensores con que cuente (no todos los SmartPhone disponen de los mismos sensores, varían de acuerdo al fabricante y versión), sin embargo, los sensores de orientación y acelerómetro se incorporan en todos los SmartPhone Android.

Lo anteriormente descrito se presenta en la Figura 7.16, figura que consiste en un extracto del código encargado de capturar los eventos producidos por los sensores del SmartPhone.

```
1  public void onSensorChanged(int sensor, float[] values) {
2            if (sensor == SensorManager.SENSOR_ORIENTATION) {
3                   orientacion = (int) values[3];
4            }
5  …
```

Figura 7.16: Identificación del sensor y obtención de sus valores, fragmento del código de SkyNet módulo Android.

7.3.8 Implementación de la comunicación bluetooth

La comunicación entre SmartPhone y robot se realizó mediante comunicación bluetooth, dado que Android no proporciona elementos para la utilización del bluetooth se ha usado la librería *Backport-Bluetooth*. Esta librería construida en lenguaje Java permite el manejo de dispositivos bluetooth, así como también el envío y recepción de información mediante estos.

Mediante la librería antes señalada se construyeron tres threads para el manejo de la petición de conexión, conexión y la aceptación de una petición, a estos threads se les ha denominado como *HiloAceptaConexionBluetooth*, *HiloConexionABluetooth* y *HiloConectandoBluetooth,* respectivamente.

Para realizar la conexión a un dispositivo bluetooth, en primer lugar se debe obtener el dispositivo bluetooth, el cual se selecciona desde una lista de dispositivos, en esta lista se despliegan aquellos con los cuales ya se ha establecido conexión, nuevos dispositivos detectados, y se entrega la posibilidad de ingresar manualmente la MAC de un dispositivo no visible. Una vez seleccionado el dispositivo al cual se desea conectar, se envía al método conectar de la clase *ComunicacionRobot*, éste método recibe como parámetro un objeto de tipo *BluetoothDevice* (línea 1, Figura 7.17). El método *conectar* verifica si existe comunicación en curso con un dispositivo bluetooth, si es así se cancela dicha conexión (línea 4 y 5, Figura 7.17), crea el hilo encargado de realizar la conexión a un dispositivo bluetooth y establece como estado *ESTADO_CONECTANDO* (línea 14, Figura 7.17).

A continuación se presenta la Figura 7.17, correspondiente a un segmento de código correspondiente al método *conectar* de la clase *ComunicacionRobot*.

```
1  public synchronized void conectar(BluetoothDevice disp) {
2        if (estadoComunicacion == ESTADO_CONECTANDO) {
3            if (connectThread != null) {
4            connectThread.cancelar();
5            connectThread = null;
6            }
7        }
8        if (connectedThread != null) {
9            connectedThread.cancel();
10           connectedThread = null;
11       }
12        connectThread = new HiloConectandoBluetooth(disp);
13        connectThread.start();
14        setEstado(ESTADO_CONECTANDO);
15 }
```

Figura 7.17: Método *conectar*, perteneciente a la clase *ComunicacionRobot*.

Finalmente, se presenta el fragmento de código que realiza el envío de la información al dispositivo bluetooth con el cual se ha establecido comunicación, éste método corresponde a la clase thread *HiloConexionABluetooth*, clase interna de la clase *ComunicacionRobot*. Éste método envía de manera serial los bytes de información, siempre y cuando el estado de la comunicación sea conectado. A continuación se presenta la Figura 7.18, correspondiente al fragmento de código del método *escribir*.

```
1  public void escribir(byte[] salida) {
2        HiloConexionABluetooth r;
3        synchronized (this) {
4            if (estadoComunicacion != ESTADO_CONECTADO) return;
5            r = connectedThread;
6        }
7        r.write(salida);
8    }
```

Figura 7.18: Fragmento de código correspondiente al método *escribir.*

7.3.9 Comunicación SmartPhone-Servidor vía socket TCP/IP

7.3.9.1 Implementación de la comunicación socket

Para la implementación de los sockets es necesario habilitar un puerto en común en la máquina utilizada como servidor, éste puerto se utiliza para la comunicación entre SkyNet módulo Web y SkyNet módulo Android. Este sistema posee un servidor que atenderá las peticiones provenientes de SkyNet módulo Android y será el encargado de enviar las rutas a realizar por cada robot. A la Figura 7.19 se detalla la creación de un objeto de la clase *InetAddress*, este objeto contendrá la dirección IP donde está alojado el servidor (línea 1). Luego con este objeto y con el puerto por el cual se comunicara el socket cliente con el servidor, se creará un nuevo objeto denominado *remoteAddr* de la clase *SocketAddress* (línea 2), este objeto contendrá la información para realizar la conexión del socket.

Luego en la línea 3 podemos observar la creación de un objeto *socket* el cual es una instancia de la clase *Socket,* objeto que posteriormente nos permitirá conectarnos a el servidor. En la línea 4 se ordena que el objeto *socket* creado recientemente establezca conexión, a una cierta dirección y puerto (representado por el objeto *remoteAddr)* y con un tiempo máximo de 2000 milisegundos, es decir 2 segundos. Una vez establecida la conexión se obtendrán las líneas de entrada y salida de información las cuales se representa en las líneas 5 y 6 respectivamente.

Las líneas de entrada y salida son objetos de tipo *DataInputStream* y *DataOutputStream* respectivamente, son líneas exclusivas para el envío y recepción de información, lo que permitirá que el envío no interfiera con la recepción de información.

```
1   InetAddress addr = InetAddress.getByName(direccionIPServidor);
2   SocketAddress remoteAddr = new InetSocketAddress(addr, puerto);
3   socket=new Socket();
4   socket.connect(remoteAddr, 2000);
5   entrada = new DataInputStream(socket.getInputStream());
6   salida   = new DataOutputStream(socket.getOutputStream());
```

Figura 7.19: Establecimiento de la conexión por parte de la aplicación Android, método perteneciente a la clase Thread *ComunicacionServidor*.

El fragmento de código de la Figura 7.20 corresponde el método *run()* de la clase *ComunicacionServidor*, éste método es invocado mediante la llamada *start()* y es el encargado de iniciar el funcionamiento del thread. Tiene como finalidad esperar continuamente solicitudes de parte del servidor, ante la recepción de una solicitud reacciona de acuerdo a la petición y realiza la acción correspondiente.

El citado fragmento se presenta en la Figura 7.20, correspondiente al fragmento de código del método *run()* de la clase *ComunicacionServidor*.

```
1  public void run() {
2      int opcion=ESPERAR;
3      try{
4          do{
5            if(socket.isConnected()){
6                 if(opcion==ENVIAR_ORDEN_A_PLACA){
7                    int orden = Integer.parseInt(entrada.readUTF());
8                    traspasarOrdenASistema(orden);
9                }if(opcion==PETICION_IDENTIFICADOR_AUTO){
10                 salida.writeUTF(sistema.getMacBluetooth());
11               }if(opcion==PETICION_POSICION_INICIAL){
12                      String posicion=sistema.getPosicionInicial();
13                      salida.writeUTF(posicion);
14               }if(opcion==TRANSFERENCIA_SIGUIENTE_COORDENADA){
15                      String posicionObjetivo=entrada.readUTF();
16                             sistema.enviarToast("posición Objetivo
17                       establecida");
18                      establecerPosicionObjetivo(posicionObjetivo);
19              }
20           opcion=ESPERAR;
21           do{
22                    opcion = Integer.parseInt(entrada.readUTF());
23           }while(opcion==ESPERAR);
24         }
25      }while(opcion!=CERRAR);
26      cerrarConexion();
27  }catch(Exception e){
28     cerrarConexion();
29 }}
```

Figura 7.20: Método *run* de la clase *ComunicacionServidor*, éste método se encarga de escuchar las peticiones enviadas desde el servidor.

7.3.10 Reconocimiento de colores mediante la utilización de la cámara del SmartPhone

Para el reconocimiento de objetos y colores se ha utilizado una lógica de visión sencilla dado que las librerías de visión existentes para Android se encuentran disponibles de la versión 2.0 en adelante. Es por ello que se han utilizado los elementos provistos por Android para visualizar la cámara del SmartPhone, esto permite utilizar SkyNet módulo Android por cualquier SmartPhone con sistema operativo Android.

El procesamiento de la imagen consiste en la utilización de los métodos de la interfaz *SurfaceHolder* y de la clase padre *SurfaceView*, estas clases permiten monitorear el ancho de la visualización de la cámara, así como también los pixeles y cambios producidos en la visualización. En primer lugar se debe obtener la cámara, para ello se debe efectuar la llamada descrita en la Figura 7.21.

```
camara = Camera.open();
```

Figura 7.21: Obtención de la cámara.

Una vez obtenida la cámara, se establece que cuando se inicie la visualización se procesará uno de cada ocho frames, esto dado que como no existen librerías que optimicen el procesamiento de una imagen se debe restringir al máximo los procesos sobre la imagen.

Al momento de procesar un frame se calculan los valores para U y V (Para mayor comprensión se recomienda ver el Punto 7.3.10.1, correspondiente al estándar YUV), estos valores se calculan a partir del arreglo de datos que se obtiene desde la cámara, este arreglo de datos contiene la información de la imagen en formato YUV.

Una vez obtenidos los valores de U y V, se envían en conjunto con la posición del arreglo accesada a una clase estática llamada *ManejaImagen* (línea 11, Figura 7.22). Esta clase estática compara los valores de U y V con los valores mínimos y máximos de U y V previamente configurados para un cierto color. Si el valor obtenido para U y V se encuentra en el rango de valores aceptados, se establece como *1* la posición del arreglo que contiene la información de la imagen, el *1* significa presencia del color en la imagen, de lo contrario se establece como *0*.

Al mismo tiempo existen contadores que verifican la cantidad de unos existentes y la cantidad máxima de unos en el arreglo, que se encuentran de manera correlativa en el mismo, esto para determinar de manera sencilla la mayor masa con presencia del color.

Finalmente, ante el procesamiento de un frame se ordena al sistema controlador que verifique la información obtenida y actúe según se decida (ver línea 15, Figura 7.22). Para evitar sobrecargar el Smartphone se vuelve a establecer en 8 el contador de frames que se descartarán (ver línea 17, Figura 7.22), y de esta forma procesar solamente uno de cada ocho frames.

La Figura 7.22, presenta un fragmento de código correspondiente a la obtención de los valores de U y V, cruciales para la detección del color en una imagen.

```
1  camara.setPreviewDisplay(holder);
2     camara.setPreviewCallback(new PreviewCallback() {
3     public void onPreviewFrame(byte[] data, Camera _camera) {
4           if( --rangoDeProcesamiento == 0 ){
5                 ManejaImagen.resetCont();
6                 int offset = 320*240;
7                 int v, u;
8                 for( int index=0; index<(320*240/4); index++ ){
9                       v = (0x000000ff & data[offset]);
10                      u = (0x000000ff & data[offset+1]);
11                      ManejaImagen.updateStats( v, u, index );
12                      offset += 2;
13                }
15                skynet.verificaInformacionVision(ManejaImagen.cont);
16                postInvalidate();
17                rangoDeProcesamiento = 8;
18           }
19    });
```

Figura 7.22: Obtención de valores U y V.

7.3.11 Seguimiento de objetos

Para el seguimiento de un objeto, se establece márgenes que dividen la pantalla en izquierda, centro y derecha. Tal como se mencionaba en el punto anterior, para la detección de masas se verifican la mayor cantidad de elementos vecinos en un arreglo, si bien es cierto es una técnica bastante simple es muy efectiva, y no provoca una sobrecarga para el procesador del SmartPhone.

Como se obtiene la mayor presencia o "masa" de color, y esta consta en posiciones en un arreglo se obtiene el centro de esta "masa". Y mediante la transformación del arreglo a una matriz de las dimensiones de la pantalla, se obtiene la posición en X e Y que tendría el centro de la "masa" de color. De esta forma utilizando la posición en X de la masa, se obtiene si el centro de esta masa se encuentra en la izquierda, centro o derecha.

De esta manera se ha implementado una forma sencilla de detección de color y masas, así como también su seguimiento mediante el robot. El seguimiento ha sido complementado con un sensor ultrasonido el cual actúa como sonar y permite detectar la distancia a objetos, ante lo cual se puede seguir un color y detenerse a una cierta distancia, para tomar un objeto y trasladarlo hasta un objeto de otro color.

Para apoyar el traslado del objeto se ha dotado al robot con una pinza que sostiene contra un costado el objeto de manera de dejar a la cámara la visual disponible para seguir un nuevo color.

Capítulo VIII

Desarrollo e implementación del sistema Web de monitoreo y control

En este capítulo se abordan temas pertinentes al desarrollo e implementación del Sistemas de monitoreo y control denominado SkyNet módulo Web. El desarrollo de este módulo contempla la etapa de análisis de requisitos así como algunos diagramas que ayudaron en la creación del sistema.

La fase de implementación que se presenta en este capítulo considera también la explicación de aspectos relevantes de las tecnologías y herramientas utilizadas en SkyNet módulo Web como el uso de mapas Google, así como la incorporación de Ajax para él envío de la información a la lógica del sistema.

8.1 SkyNet módulo Web

Tal como se describió en el punto 6.2, SkyNet se compone de dos módulos, el módulo Android y el módulo Web. En los capítulos VI y VII se ha descrito el desarrollo e implementación del robot y aplicación nativa Android, respectivamente. Sin embargo, para dar cumplimiento a la solución del problema (ver punto 6.1), resta describir el desarrollo e implementación del módulo web de SkyNet.

Es importante destacar que SkyNet módulo web permite el ingreso simultáneo de usuarios, puesto que es un sistema Web. Sin embargo, no ha sido creada para que se use de esta forma, porque los usuarios que ingresen de manera simultánea accederán a la misma vista y sus acciones podrían entorpecer las órdenes enviadas por los demás usuarios.

En consideración a lo anteriormente señalado, se invita al lector a interiorizarse con el presente capítulo, el cual detalla el desarrollo e implementación del módulo Web de SkyNet.

8.2 Análisis de requisitos de SkyNet módulo Web

Como se especificó anteriormente en el Punto 6.4.2**,** el análisis de requisitos es fundamental para la correcta formulación de las bases que orienten el desarrollo de cualquier sistema, razón por lo cual se presentan los resultados del análisis de requisitos para el sistema SkyNet módulo Web.

8.2.1 Requisitos no funcionales

Los requisitos no funcionales son aquellos con características que pueden afectar diversos aspectos de una aplicación, como por ejemplo, rendimiento, mantenimiento, seguridad, portabilidad y ajuste a estándares.

En la Tabla 8.1 se describen los requisitos no funcionales para la aplicación Web a construir.

Atributo	Detalle
Plataforma	J2EE
Tipo de aplicación	Aplicación Web
Lenguaje de programación	Java y XML
Tiempos de respuesta	No Mayor a 3 segundos para cualquier operación.
Disponibilidad	Al ser una aplicación web tiene una disponibilidad continua.

Tabla 8.1: Requisitos no funcionales de SkyNet módulo Web.

8.2.2 Requisitos funcionales

En la Tabla 8.2 se reiteran las categorías de los requisitos a analizar para el módulo Web de SkyNet, las cuales posteriormente se utilizan para identificar los requisitos que podrían pasar inadvertidos (21). Se recomienda al lector ver el Punto 6.5.1.2, mencionado anteriormente y que detalla a los requerimientos funcionales y sus categorias.

Categoría de la Función	Significado
Evidente	Debe realizarse, y el usuario debería saber que se ha realizado.
Oculta	Debe realizarse, aunque no es visible para los usuarios. Esto se aplica a los servicios técnicos subyacentes, como guardar información en un mecanismo persistente de almacenamiento. Las funciones ocultas a menudo se omiten (erróneamente) durante el proceso de obtención de los requerimientos.
Superflua	Opcionales; su inclusión no repercute significativamente en el costo ni en otras funciones.

Tabla 8.2: Categoría de los requisitos.

8.3 Funciones básicas del sistema

En la Tabla 8.3 se presentan las funciones básicas del sistema, que luego serán ampliadas, dado que contienen otras funciones relevantes para el sistema.

Ref. #	Función	Categoría
R.1	Asignar rutas	Evidente
R.2	Gestionar usuarios	Evidente
R.3	Monitorear robots	Evidente
R.4	Modificar robots a controlar	Superflua
R.5	Autenticar usuario	Evidente

Tabla 8.3: Funciones Básicas del sistema.

En la Tabla 8.4, se presenta el detalle para la funcionalidad *Asignar rutas*.

Ref. #	Función	Categoría
R.1.1	Ingresar una o más localizaciones geográficas, conformando una ruta, para que el o los robots la realicen. Una vez realizada la ruta se ordenara objetos de un determinado color.	Evidente
R.1.2	Eliminar la ruta asignada a uno o ambos robots.	Evidente
R.1.3	Enviar la o las rutas ingresadas a él o los robots deseados.	Evidente

Tabla 8.4: Detalle función *Asignar rutas.*

La Tabla 8.5, contiene el detalle de la funcionalidad *Gestionar usuarios*.

Ref. #	Función	Categoría
R.2.1	Registrar usuarios. Sólo el administrador podrá agregar o eliminar usuarios.	Evidente
R.2.2	El usuario administrador podrá eliminar a un usuario registrado	Evidente
R.2.3	Deberá permitir el acceso a usuarios registrados (autenticar)	Evidente

Tabla 8.5: Detalle función *Gestionar usuarios.*

La Tabla 8.6 contiene el detalle de la funcionalidad *Monitorear robots*.

Ref. #	Función	Categoría
R.3.1	Permitir a un usuario registrado, ver los movimientos realizados por uno o más robots, luego de que se les haya asignado una ruta.	Evidente
R.3.2	Detener y cancelar el movimiento de uno o más robots.	Evidente

Tabla 8.6: Detalle función *Monitorear robots*

La Tabla 8.7 presenta la funcionalidad *Modificar robots a controlar*.

Ref. #	Función	Categoría
R.4.1	Verificar la MAC del Smartphone que controlará un determinado robot.	Superflua
R.4.2	Cambiar la MAC de uno o más Smartphone, de manera de no enviar información a estos dispositivos.	Superflua

Tabla 8.7: Detalle función *Modificar robots a controlar.*

La Tabla 8.8 presenta la funcionalidad *Autenticar usuario*.

Ref. #	Función	Categoría
R.5.1	Ingresar al módulo web, mediante el nombre de usuario y password.	Evidente

Tabla 8.8 Detalle función *Autenticar usuario*.

Para las funcionalidades *Gestionar rutas, Gestionar ruta,* y *Gestionar Monitoreo*, se ha detectado un requerimiento de categoría Oculta, el cual es la selección y visualización de las rutas en un mapa. El hecho de seleccionar en un mapa la ruta a visitar, resulta útil para el usuario, dado que ingresar manualmente localizaciones geográficas con su latitud y longitud es compleja y se incurriría en errores. Por otro lado, la visualización de los movimientos del robot en un mapa provoca que el usuario tenga una idea clara de los movimientos realizados y la ubicación del robot.

8.4 Identificación de actores del sistema

Para SkyNet módulo Web se han definido dos usuarios, estos se presentan en la Tabla 8.9 y 8.10.

Actor	Administrador SkyNet
Descripción	El administrador es el encargado de registrar a los posibles usuarios, ya que no existe otro modo de ingresar usuarios, esto tiene el objetivo de mantener un control en el ingreso al sistema.
Características	Podrá tener acceso a todas las funciones y solo él será el encargado de ingresar a los usuarios.

Tabla 8.9 Descripción actor *Administrador SkyNet* del sistema SkyNet módulo Web

El usuario SkyNet, tal como ya se mencionó en el punto 6.5.2, tiene relación tanto con el módulo Android como con el módulo Web. En la Figura 8.10 se presenta el detalle del actor *Usuario SkyNet*.

Actor	Usuario SkyNet
Descripción	El usuario SkyNet es una persona que cuenta con acceso a SkyNet módulo Web. Además esta persona debe tener acceso a un SmartPhone, con el módulo Android instalado y un equipo robótico. El equipo robótico deberá permitir la comunicación bluetooth y el desplazamiento del robot mediante urugas, así como también, un brazo en su parte superior.
Características	Podrá hacer uso limitado del sistema SkyNet módulo Web y uso de la totalidad de las funcionalidades del módulo Android. Las funcionalidades que podrá utilizar del módulo web corresponden a la asignación de rutas y al monitoreo de estas.

Tabla 8.10 Descripción actor *Usuario SkyNet*, el cual tiene acceso al módulo Android y Web de SkyNet.

8.5 Modelo de casos de uso

En la Figura 8.1 se presentan los casos de usos identificados en SkyNet módulo Web. Para ver el detalle de los casos de uso dirigirse al Anexo III.

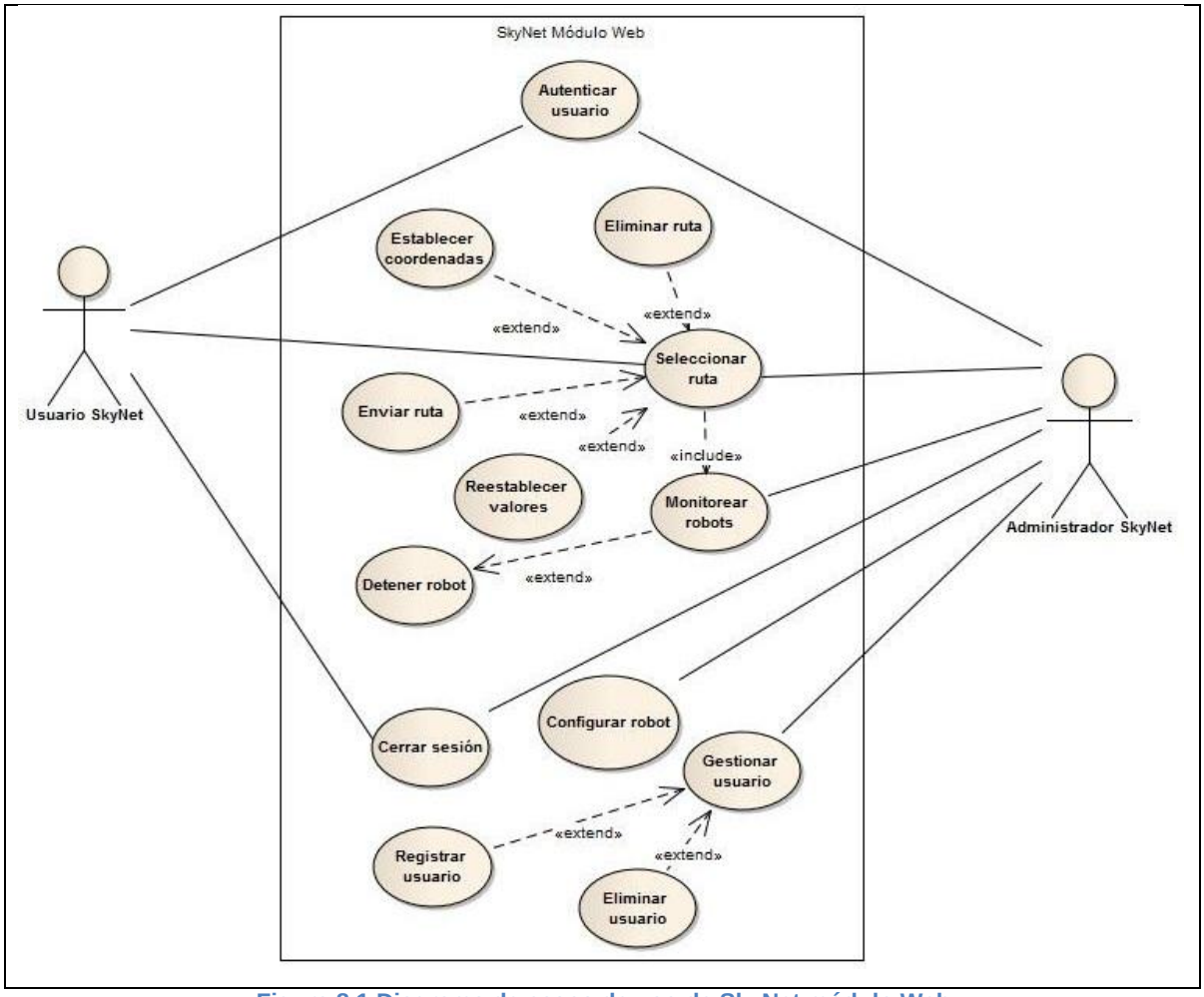


Figura 8.1 Diagrama de casos de uso de SkyNet módulo Web.

8.6 Modelo entidad relación

Como se explicó anteriormente, el modelo entidad relación se encarga de representar gráficamente el mundo relacional que modelan las bases de datos. La Figura 8.2 presenta las entidades existentes en la base de datos que maneja SkyNet módulo Web. El modelo es sencillo ya que sólo es utilizado para llevar un registro de los usuarios y teléfonos con los cuales el sistema se comunica.

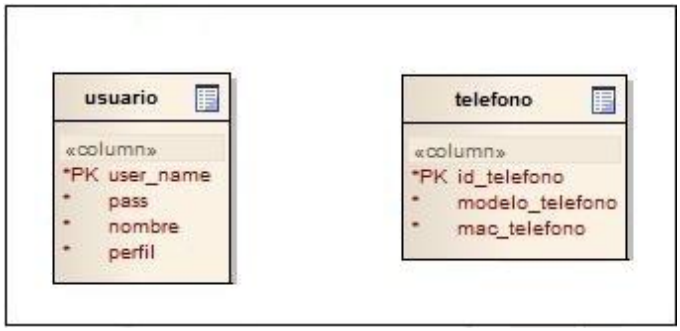


Figura 8.2 Diagrama entidad relación SkyNet módulo Web.

8.7 Diagrama de clases

El diagrama presentado en la Figura 8.3 muestra las clases del sistema SkyNet módulo Web y sus relaciones. Estas clases tienen como objetivo implementar la lógica del módulo Web.

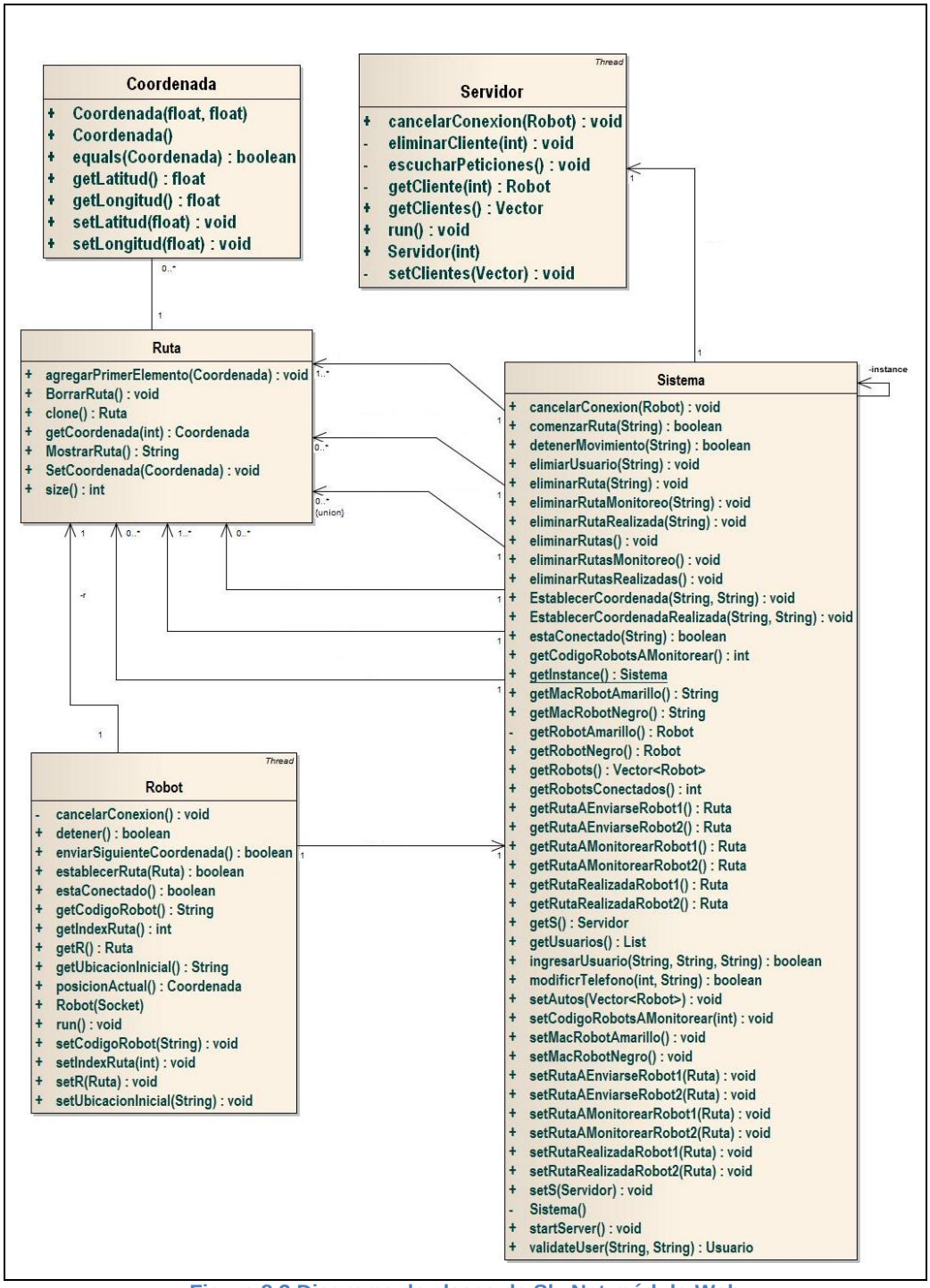


Figura 8.3 Diagrama de clases de SkyNet módulo Web

En la Figura 8.4 se presenta el diagrama de clases de análisis, este diagrama de clases no incorpora métodos ni atributos de manera de mostrar las relaciones y las clases que intervienen en la lógica del sistema. Con este diagrama se presente interiorizar al lector de las clases que intervienen en la lógica del módulo web de SkyNet.

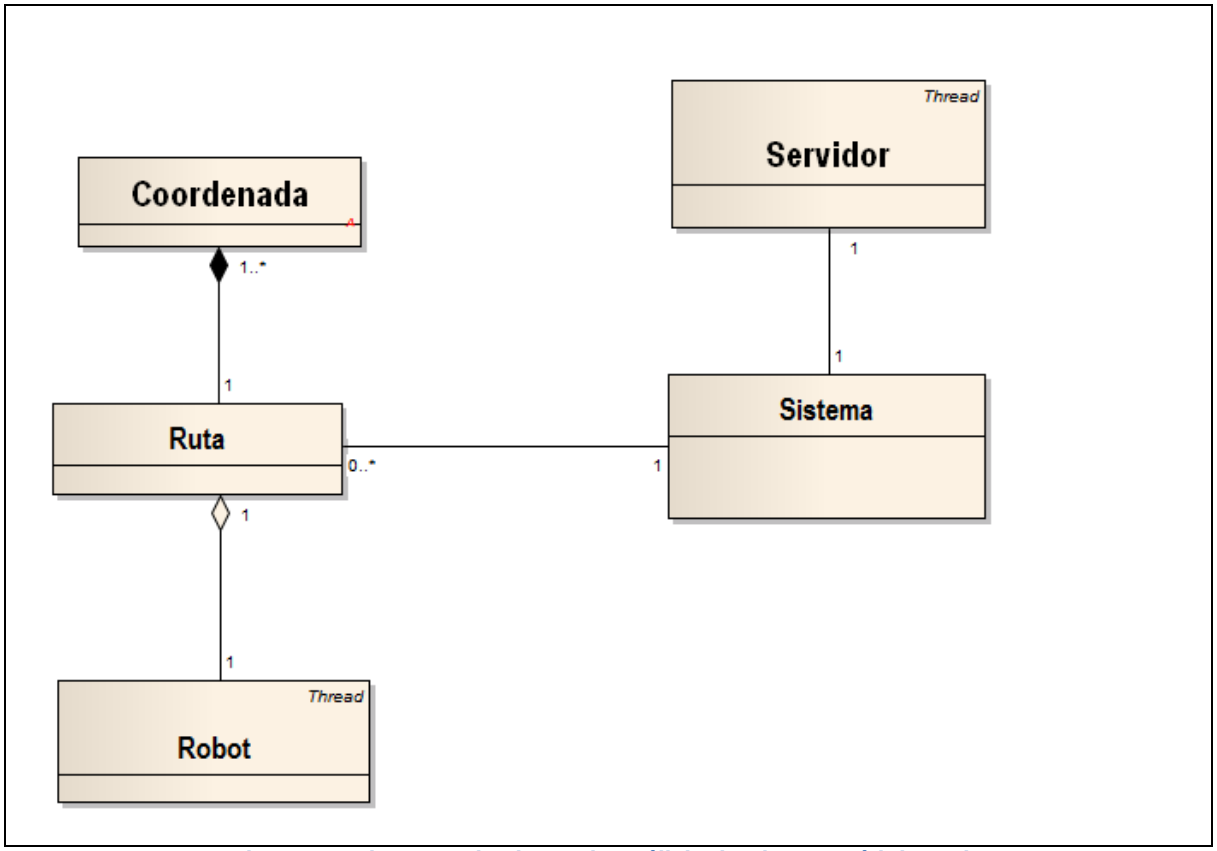


Figura 8.4 Diagrama de clases de análisis de SkyNet módulo Web.

8.8 Diagrama de secuencia de sistema

Según la definición de diagramas de secuencia entregada, se definen a continuación los diagramas más representativos de SkyNet módulo Web.

8.8.1 Diagrama de secuencia de sistema del caso de uso *Autenticar usuario.*

El diagrama presentado a continuación en la Figura 8.5, describe el flujo extendido del diagrama correspondiente al caso de uso *Autenticar usuario*. Muestra la interacción que realiza el sistema de manera de autenticar a un determinado usuario ante la solicitud de ingreso al sistema.

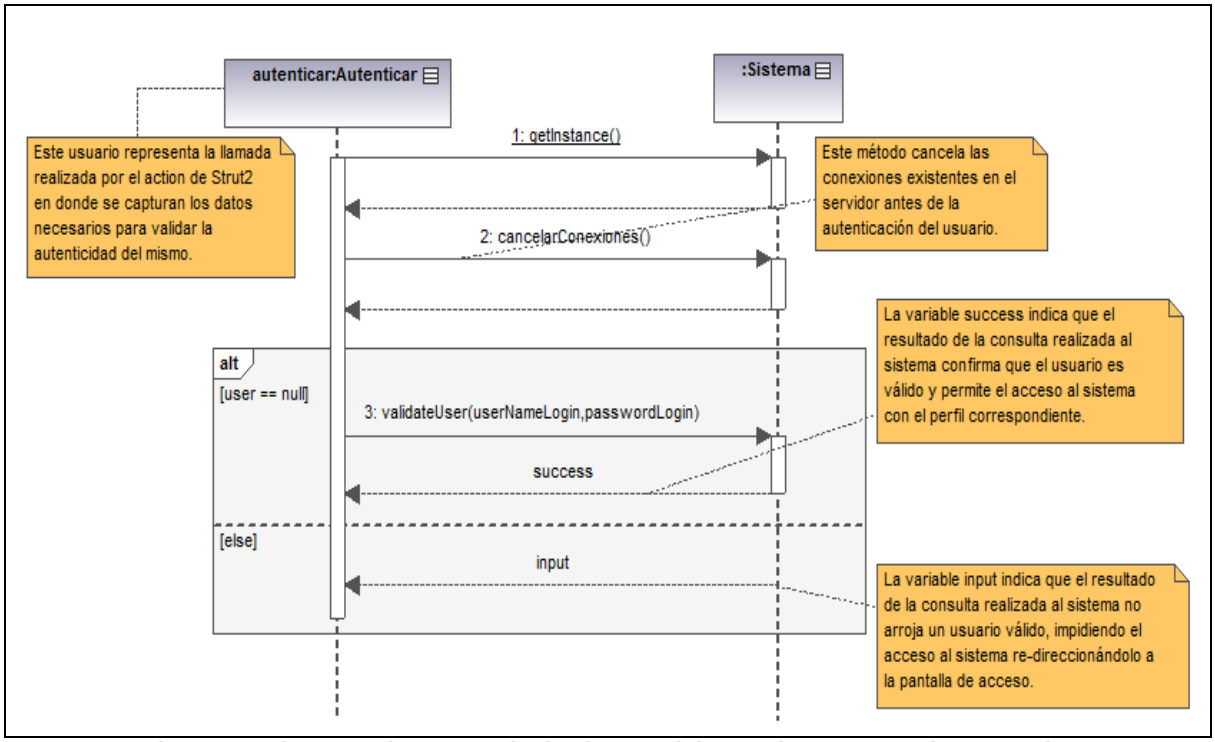


Figura 8.5 Diagrama de secuencia de sistema del caso de uso Autenticar usuario.

8.8.2 Diagrama de secuencia de sistema del caso de uso *Establecer coordenada*

En la Figura 8.6, se presenta el diagrama de secuencia correspondiente al Caso de Uso *Establecer coordenadas*. Éste caso de uso muestra la interacción que realiza el sistema al momento de seleccionar una ruta para un determinado robot (Anexo IV, Tabla IV.11).

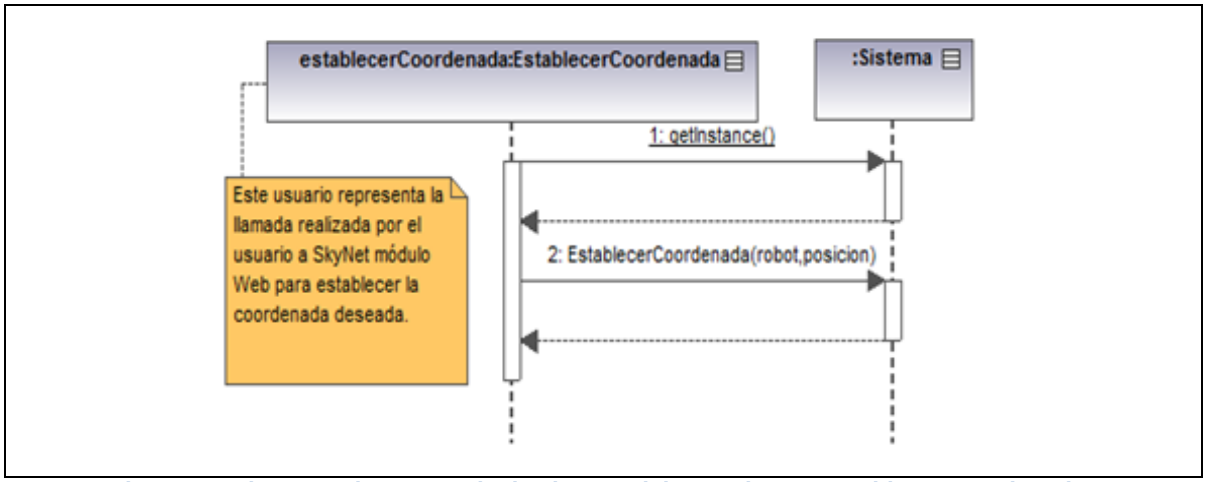


Figura 8.6 Diagrama de secuencia de sistema del caso de uso Establecer coordenadas.

8.8.3 Diagrama de secuencia de sistema del caso de uso *Enviar ruta*

El diagrama presentado a continuación, en la Figura 8.7, muestra el flujo extendido del diagrama de secuencia de sistema correspondiente al caso de uso *Enviar ruta*. Caso de uso que envía la ruta seleccionada en el mapa al robot para su realización (el sistema se ha diseñado para el manejo de dos robots como máximo, aunque, para este proyecto solamente se construyó un robot).

El método *comenzarRuta("robotNegro")* en primer lugar obtiene las coordenadas ingresadas a través del mapa para el robot negro, para posteriormente enviarlas mediante el la comunicación socket al módulo Android. Una vez enviadas se da comienzo a la ruta generada por las coordenadas, así también, el método *getRutaAEnviarseRobotNegro()* y el método *setRutaAMonitorearRobotNegro (s.getRutaAEnviarseRobotNegro())* se encargan de establecer la ruta de monitoreo en base a la ruta enviada. Luego de esto se eliminan las coordenadas de la ruta a enviarse al *robot negro* mediante el método *"eliminarRutas()"* para poder almacenar posteriores rutas y se eliminan las rutas de monitoreo del *robot amarillo* para mantener libre esta ruta, evitando que se mezclen en el módulo de monitoreo. En este caso de uso en particular, si la opción de comenzar ruta no corresponde al *robot negro* se eliminan tanto las rutas a enviarse como las rutas de monitoreo de ambos autos o robots.

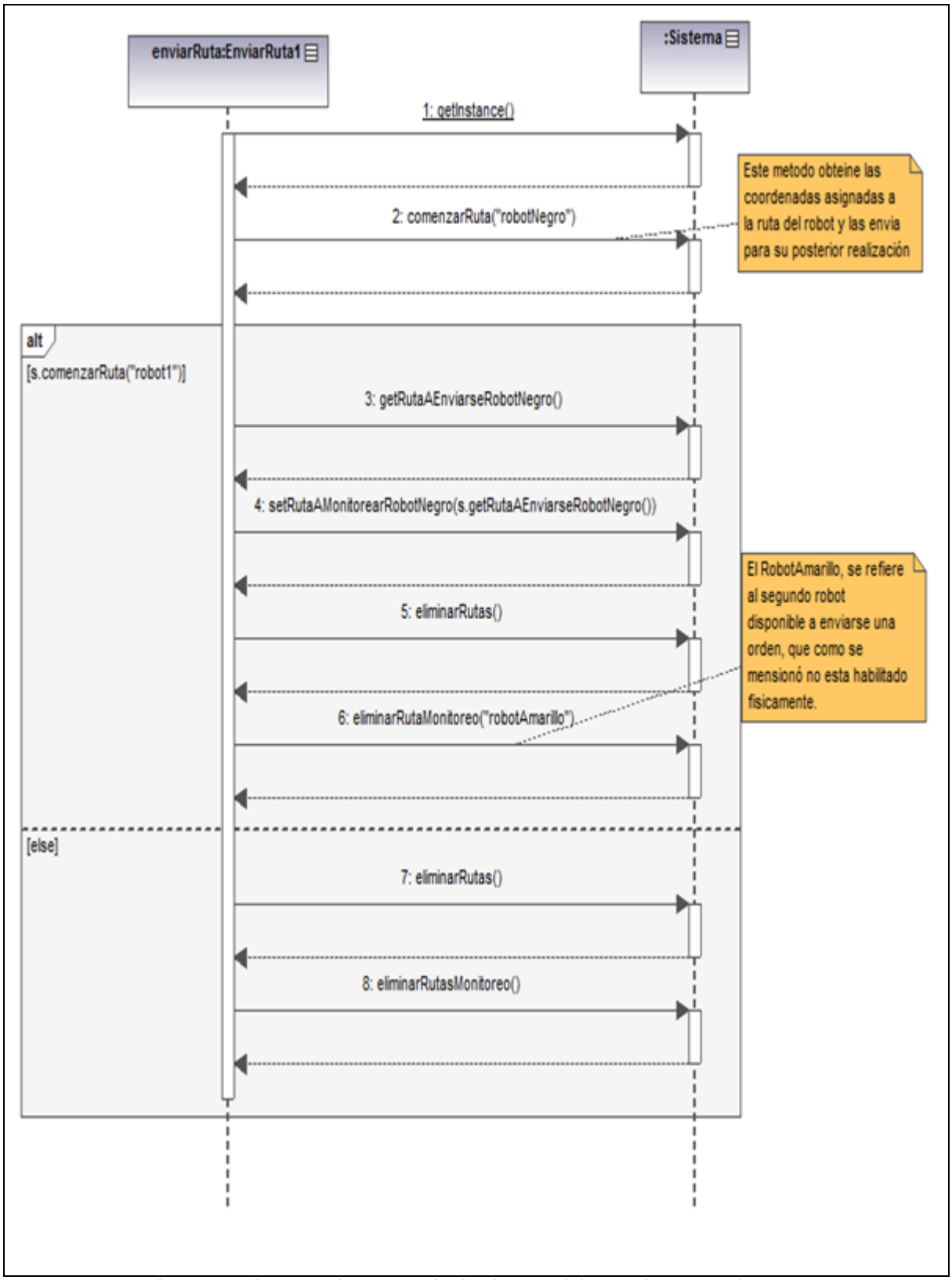


Figura 8.7 Diagrama de secuencia de sistema del caso de uso Enviar ruta.

8.9 Diagramas de comunicación de SkyNet módulo Web

Teniendo como base lo anteriormente señalado en el Punto 6.5.8 (Definición detallada sobre los diagramas de comunicación), se presentan los diagramas de comunicación de procesos relevantes dentro de SkyNet módulo Web.

8.9.1 Diagrama de comunicación del caso de uso *Autenticar usuario*

El diagrama de comunicación del caso de uso *Autenticar usuario* presentado en la Figura 8.8, corresponde al flujo del proceso de autenticación y los retornos que el mismo pueda entregar, siguiendo el orden en los cuales se deben ejecutar los procesos para un correcto funcionamiento.

A continuación se presenta, en la Figura 8.8 el diagrama de comunicación del caso de uso *Autenticar usuario,* para conocer el detalle de este caso de uso se recomienda la lectura del Anexo IV, Tabla IV.7.

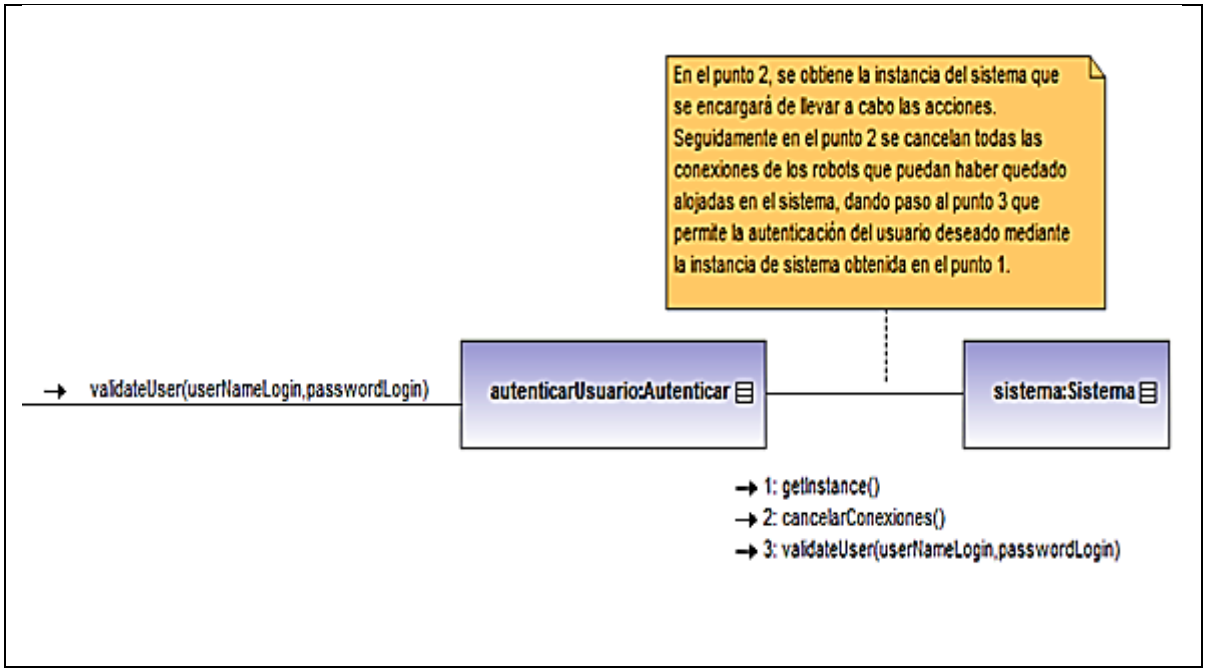


Figura 8.8 Diagrama de comunicación del caso de uso Autenticar usuario.

La Figura 8.9 presenta el flujo que realiza SkyNet módulo Web, al enviar una ruta a un robot. En este caso al existir la posibilidad de seleccionar más de un robot, el módulo deberá discriminar entre los Smartphone's conectados, dependiendo de la orden enviada por el usuario.

A continuación se presenta la Figura 8.9, correspondiente al diagrama de comunicación del caso de uso *Enviar ruta.*

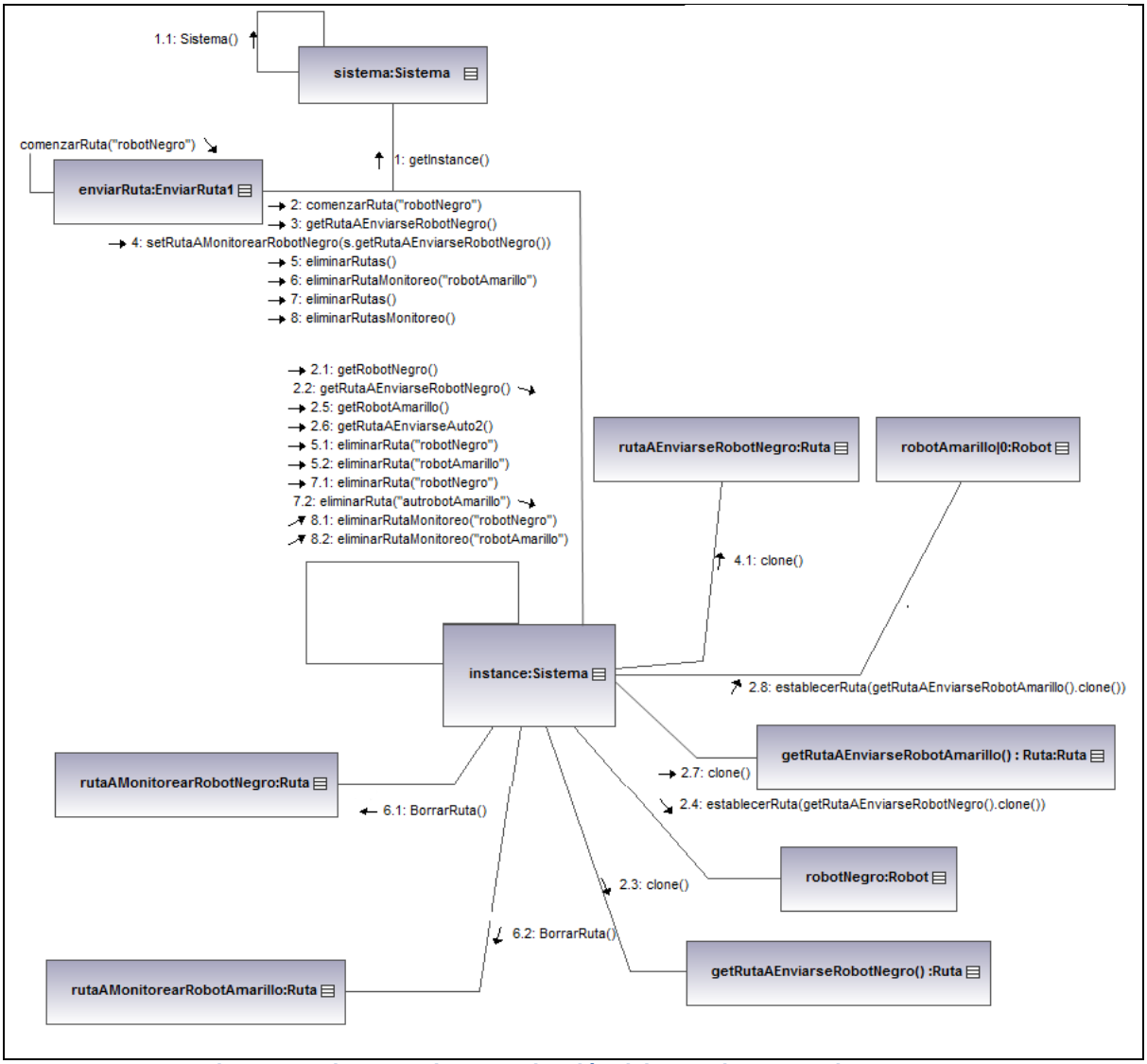


Figura 8.9 Diagrama de comunicación del caso de uso Enviar ruta.

8.9.2 Diagrama de comunicación del caso de uso *Obtener datos a monitorear*

El diagrama presentado a continuación, en la Figura 8.10, muestra las acciones realizadas para obtener la información correspondiente a rutas y posiciones, las cuales se presentan al usuario mediante un mapa proporcionado por la API de Google Maps.

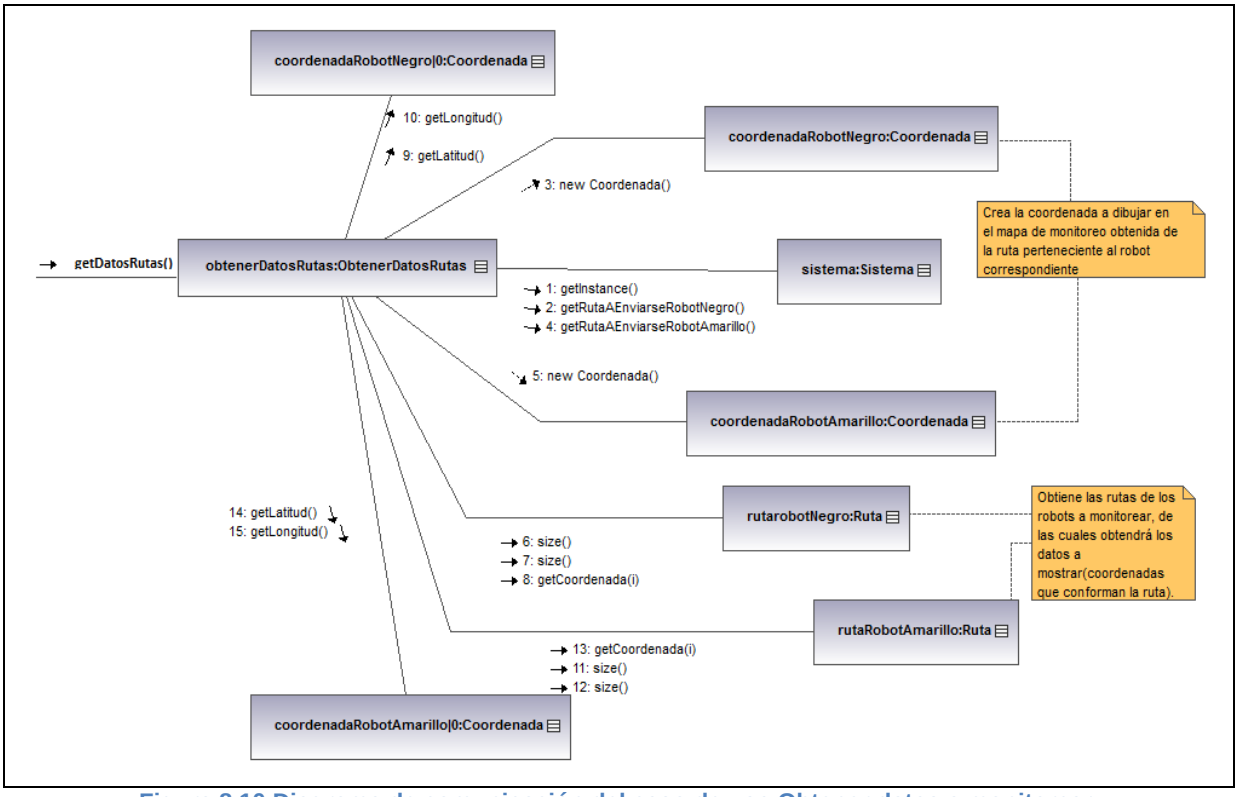


Figura 8.10 Diagrama de comunicación del caso de uso Obtener datos a monitorear.

8.10 Patrones

En el desarrollo de SkyNet módulo Web, al igual que en SkyNet módulo Android, se han incorporado buenas prácticas, como lo es la utilización de patrones. Debido a que ya han sido explicados anteriormente solo se mencionarán los patrones implementados en el desarrollo de este módulo.

Los patrones implementados en SkyNet módulo Web son:

- Patrón de arquitectura Modelo Vista Controlador (MVC).
- Patrón de diseño Singleton.

8.11 Implementación de SkyNet módulo Web

El sistema web ha sido desarrollado en lenguaje JAVA y utilizando el Framework de desarrollo Struts 2 sobre el entorno de desarrollo Netbeans 6.8, y el gestor de base de datos relacional utilizado ha sido Postgres. La elección de Netbeans 6.8 se debe en primer lugar a la existencia de un plugin de Struts 2, además es una herramienta gratuita, con la cual se tiene experiencia. Se ejecuta en computadores de 800MHz de procesador, con al menos 512 MB de RAM (Read Access Memory), hecho que le permite un buen desempeño en los computadores actuales.

Para la base de datos se selecciona Postgres dado que es un gestor de código abierto, con el cual se ha trabajado anteriormente, es estable, seguro, posee gran documentación, y que permite alta concurrencia, la utilización de gran variedad de tipos nativos, funciones, vistas, integridad transaccional.

SkyNet módulo Web tiene como finalidad informar visualmente al usuario del grado de avance del robot con respecto a la ruta que le fue asignada previamente en un mapa, de tal manera que el usuario tome decisiones con esta información, estas pueden ser: detener los movimientos de él o los robots en movimiento, seleccionar nuevas rutas o simplemente finalizar la ejecución de la ruta en curso. Toda la información que se presente al usuario, así como el ingreso de información que éste realice se hará mediante mapas Google.

8.11.1 Integración de Google Maps

En el desarrollo de SkyNet módulo Web, es necesaria la presencia de mapas que permitan establecer la ruta a cumplir por un determinado robot. Para lo cual se evaluaron opciones, esto arrojó como mejor opción, la incorporación de mapas mediante JavaScripts. Para esto, se utilizó Google Maps, puesto que provee una API especializada para trabajar con mapas, de manera dinámica y eficiente.

A continuacion el código ilustrado en la Figura 8.11, representa la declaración del uso de mapas google en una página web. Esta declaración se realiza mediante la incrustación de un script que declara la *key* con la cual se accesará a los servidores de Google. Esta operación es característica y necesaria en los sistemas que incorporan Google Maps.

```
1 <script
2      src="http://maps.google.com/maps?file=api&amp;v=2&amp;
3      sensor=false&amp;
4      key=ABQIAAAAzr2EBOXUKnm_jVnk0OJI7xSosDVG8KKPE1-m51RBrvYughuyMxQ-
5      i1QfUnH94QxWIa6N4U6MouMmBA"
6      type="text/javascript">
7 </script>
```

Figura 8.11 Representación del código de establecimiento de *Key* de acceso.

En el caso de éste módulo (SkyNet módulo Web), la implementación del mapa incorpora la capacidad de identificar en qué localización geográfica se ha realizado una selección, esto dado que es necesario que el usuario verifique visualmente los lugares que desea establecer como rutas para él o los robots. El hecho que la interación entre el usuario y el sistema sea mediante un mapa aumenta la usabilidad y proporciona al usuario seguridad en cuanto a las localizaciones ingresadas, dado que las verifica visualmente.

A continuación se presenta la Figura 8.12, la cual contiene el fragmento de código JavaScript que permite, ante la selección de una posicion el mapa, capturar la coordenada, establecer el icono, trazar la ruta y enviar la información a los metodos encargados de traspasar la selección al sistema.

```
1   var myEventListener = GEvent.bind(this.map, "click", this, function(overlay, latlng) {
2      if(latlng) {
3         if(robotEnProceso==1){
4            getResult(latlng)
5         var marcador=new GMarker(latlng, markerOptions);
6            vectorMarcadoresNegros[numeroDeMarcadoresNegros]=marcador;
7            numeroDeMarcadoresNegros++;
8            this.map.addOverlay(marcador);
9            var polyline = new GPolyline([posicionAnteriorNegro,latlng], lineColor, 3);
10           this.map.addOverlay(polyline);
11           vectorOverlaysNegros[numeroDeOverlaysNegros]=polyline;
12           numeroDeOverlaysNegros++;
13           posicionAnteriorNegro=latlng;
14        }else if(robotEnProceso==2){
15           getResult(latlng);
16           var marcador=new GMarker(latlng, markerOptions);
17           vectorMarcadoresAmarillos[numeroDeMarcadoresAmarillos]=marcador;
18           numeroDeMarcadoresAmarillos++;
19           this.map.addOverlay(marcador);
20           var polyline = new GPolyline([posicionAnteriorAmarillo,latlng],lineColor,3);
21           this.map.addOverlay(polyline);
22           vectorOverlaysAmarillos[numeroDeOverlaysAmarillos]=polyline;
23           numeroDeOverlaysAmarillos++;
24           posicionAnteriorAmarillo=latlng;
25        }else if(robotEnProceso==3){
26           alert("Debes seleccionar un robot!!!");
27        }
28     }});
```

Figura 8.12 Código para agregar marcadores al mapa.

Mediante el fragmento de código expuesto en la Figura 8.12, cada vez que el usuario realiza una selección sobre el mapa, se captura el evento ingresando una nueva posición al canvas, es decir, dibuja la posición seleccionada en el mapa. En esta acción intervienen marcadores de tipo *GMarker*, los cuales establecen un icono configurable en una determinada posición geográfica, tal como se puede apreciar en la línea 5 de la Figura 8.12. Otro elemento importante en la implementación del mapa son los Overlays, los cuales no son más que capas que se agregan sobre un objeto *Map*, los overlay utilizados en el sistema SkyNet son de tipo *GMarker* y *PolyLine*, estos elementos se agregan a un mapa mediante la llamada *addOverlay(overlay) (*Para ejemplificar ver línea 10 y 19, de la Figura 8.12).

La implementación de lo anteriormente descrito se ve reflejada en la Figura 8.13.

Figura 8.13 Vista de ejemplo del mapa de selección de rutas.

Para apoyar la funcionalidad de generar rutas, se incorpora la capacidad de dibujar los trazos entre cada selección del usuario, para esto se utiliza la API de Google Maps que permite crear objetos de tipo *GPolyLine*. Estos objetos dibujarán, entre dos localizaciones geográficas, una línea que tendrá una configuración de color y grosor, dependiendo del vehículo al cual se le esté asignando una ruta.

La representación de lo anteriormente señalado se presenta en la Figura 8.14.

```
1       var polyline = new GPolyline([posicionAnteriorNegro,latlng], lineColor, 3);

2       this.map.addOverlay(polyline);
```

Figura 8.14 Dibujo de líneas para representar rutas en el mapa.

La puesta en práctica del código presentado en la Figura 8.14, entrega la cualidad del dibujo de líneas las que finalmente representarán la ruta.

Figura 8.15 Vista de ejemplo de la representación de una ruta.

8.11.2 Incorporación de AJAX para el manejo de mapas.

Uno de los mayores problemas en el desarrollo de éste proyecto surge en la mantención de los datos ingresados al mapa, puesto que los mapas están construidos en lenguaje JavaScript. Al ser éste un lenguaje interpretado, surge el problema que ante la actualización de la página, la información se pierde totalmente, hecho que provocaría la confusión del usuario.

Es por ello que se ha incorporado Ajax al funcionamiento del sistema, mediante esta técnica, se ha logrado interactuar con la lógica del sistema enviando cada acción realizada sobre el mapa a la lógica, de manera de mantener la información ante eventuales recargas de página.

Se utilizan llamadas asíncronas para enviar a la lógica de SkyNet módulo Web, las selecciones ocurridas sobre el mapa, de manera de mantenerlas y utilizarlas para él envió de ellas al robot, o recuperar las selecciones ante la recarga del mapa. El hecho de que se recargue la página provoca una llamada, que obtiene la información anteriormente ingresada y la despliega nuevamente en el mapa, esta acción, al realizarse de manera asíncrona, provoca que sea totalmente transparente para el usuario.

Asimismo se ha sido utilizado en la presentación de la posición actual del robot en el mapa, una vez que se ingresa a la selección de rutas, de manera de conocer el inicio de la ruta a ingresar.

A continuación se presenta, en la Figura 8.16, una llamada asíncrona que obtiene la información de los datos ingresados a la ruta.

```
1      function obtieneInformacion() {
2             createXmlHttp();
3             var url;
4             url = "<%=basePath%>" + "secure/ObtenerDatosRutas.action";
5             xmlHttp.open("GET", url, true);
6             xmlHttp.onreadystatechange = recibeInformacion;
7             xmlHttp.send(null);
8      }
```

Figura 8.16 Llamada asíncrona al sistema que obtiene los datos de las rutas seleccionadas.

La Figura 8.16 es un ejemplo de una llamada asíncrona al sistema, la cual se realiza mediante el metodo GET y establece un *Listener* (línea 6 de la Figura 8.16, objeto listener llamado *recibeInformacion*) que verifica la recepción de la información. Éste proceso se realiza mediante el parámetro *onreadystatechange* al cual se le asigna una función que en éste caso es *recibeInformacion*, de esta forma ante el error de la petición u obtención de la información se llamará a *recibeInformacion*, y esta se accionará de acuerdo al suceso. En caso de obtener la información, se divide el texto recibido mediante parsing y lo despliega en el mapa y, en caso de errores en la petición, comunicará en pantalla el mensaje informativo correspondiente.

Es así como Ajax se ha incorporado para el envío y obtención de coordenadas desde y hacia el sistema, de manera de permitir una mayor libertad al usuario al momento de manipular el mapa. Además de evitar que ante actualizaciones no deseadas, o sin conocimiento de consecuencias, se pierda la información ingresada al mapa.

8.11.3 Diferencias entre localizaciones obtenidas por el SmartPhone y las localizaciones obtenidas mediante la selección en el mapa Google.

Un aspecto importante, descubierto durante el desarrollo de las pruebas, es que la selección de localizaciones en los mapas Google actualmente difiere levemente de las obtenidas en el SmartPhone Android. Esto dado que el SmartPhone obtiene la localización solicitando la posición a los satélites, petición que se realiza constantemente y de manera precisa. Sin embargo, los mapas Google son fotografías digitales de alta resolución de todo el planeta, las cuales se alinean una al lado de otra, existiendo un "mapeo" de las coordenadas.

Actualmente se puede apreciar que existe un desfase entre lo que Google traza como calles y lo que realmente aparece como calles en las imágenes, esto indica la existencia de pequeños desfases, de los cuales no existe documentación alguna.

La diferencia entre las coordenadas obtenidas mediante el SmartPhone y comparadas con las obtenidas en los mapas Google, se ha estimado en alrededor de 6 metros. Éste hecho atenta directamente a la precisión de los movimientos realizados por el robot, dado que la posición de término del robot será diferente a la indicada en SkyNet módulo Web. Es por ello que se ha establecido un margen de precisión de 9 metros, para lo que es el desplazamiento del robot hacia la posición objetivo.

La Figura 8.17, muestra una imagen representativa del corrimiento existente en los mapas Google.

Figura 8.17 : Imagen representativa del corrimiento detectado en los mapas Google[1].

[1] De amarillo la calle indicada por Google y de plomo oscuro la ubicación real de la calle.

Conclusiones

En el desarrollo del proyecto documentado en este informe se han logrado obtener valiosas conclusiones acerca de la situación actual de la ciudad de Chillán, y Chile en general, respecto de la tecnología móvil y la robótica. Se comprobó que la existencia de proyectos que mezclen las tecnologías móviles con algún otro tipo de tecnologías es limitado, en especial en el área de desarrollo de software, esto se ve reflejado en que la documentación Web existente proviene de páginas extranjeras y no académicas, esto puede atribuirse a que la mayoría de las tecnologías aquí tratadas son nuevas.

Si bien la barrera que presentan los Smartphone's en cuanto a adquisición se está desvaneciendo, aún existe un problema gravitante, el cual es poco incentivo en el desarrollo de éste tipo de proyectos y la poca difusión de estas tecnologías.

En cuanto al desarrollo del proyecto en sí, éste se abordó mediante una metodología iterativa incremental, dada las características del mismo. La metodología escogida cumplió con las expectativas, otorgando las suficientes herramientas de control para poder llevarlo a cabo dentro de los plazos establecidos.

Otro factor relevante en el desarrollo del proyecto fueron las tecnologías de apoyo aplicadas al sistema. Una de estas tecnologías es Strut2, que fue preponderante en el desarrollo de la aplicación Web provocando un ahorro en horas de trabajo y permitiendo así desarrollar una aplicación que cumpliese con los requerimientos deseados. Además, la incorporación de mapas proporcionados por el API de Google, proporcionó la base para la aplicación que en éste informe se detalla, dado que sin la inclusión de mapas, no hubiese sido posible implementar, de manera amigable el ingreso de coordenadas y el despliegue de los movimientos realizados, truncando la rapidez de ingreso y dificultando la comprensión de los datos contenidos en la página.

Así también, el uso de AJAX solucionó el problema del manejo de la información

que contenían los mapas, permitiendo la interacción entre el usuario y el sistema sin provocar pérdidas de información en estos producto de recargas de la página.

Es necesario también mencionar los problemas que surgieron durante la ejecución del proyecto, como por ejemplo, la construcción de los vehículos a manejar. Este ha sido un gran problema ya que no es un ámbito en el cual se tenga dominio, lo que provocó más de algún contratiempo. Otro problema relevante ha sido la conectividad, dado que no existe red disponible a la cual conectarse, para realizar pruebas dentro de la universidad.

En conclusión, este proyecto constituye un aporte en cuanto estimula y fortalece el desarrollo de software en dispositivos móviles y sistemas Web, integrando estas tecnologías a la robótica móvil. Este tipo de proyectos abre las puertas a la interacción, con profesionales de distintas especialidades, como automatización, biología, mecánica, electrónica, entre otras. Además, permite reducir costos en el desarrollo de robótica, dado que se han utilizado nuevos componentes y tecnologías, de gran potencial, bajo costo, emergentes y de un futuro prometedor, como lo son los Smartphone Android y componentes Arduino.

Trabajos futuros

Como trabajo futuro se plantea el reconocimiento de formas, ya que por el momento sólo es posible el reconocimiento de colores. Para ello es necesaria la implementación de una librería de visión, ya que no existe para Android 1.5 una librería que permita trabajar en éste ámbito, sólo está disponible para versiones superiores.

También se propone como trabajo futuro la implementación del sistema de monitoreo multi-usuario, ya que por ahora es posible sólo el manejo de los vehículos de manera mono usuario, esto debido a que, a pesar de que el sistema admite conexiones masivas, el funcionamiento de la lógica no controla la interferencia en las rutas entre dos o más usuarios.

Por último, se propone como trabajo futuro, la integración de trabajo colaborativo entre los robots, permitiendo el desarrollo de tareas que impliquen una lógica más avanzada.

Bibliografía

1. **Ruiz del Solar, Javier y Salazar, Roberto.** *Introducción a la Robótica.* [En Linea] Santiago, Chile : s.n.

2. **Garrido Z., Ignacio y Bassi A., Danilo.** *Gbot: Un Robot Móvil Educacional.* [En Linea] Santiago, Chile : s.n., 2000.

3. **Coronado, Emiro y Correa, Manuel.** *Sensors and actuators Objects in Heterogeneous System.* [En Linea] Trujillo, Venezuela : s.n., 25 de 11 de 2009.

4. perso.wanadoo.es. [En línea] [Citado el: 24 de Agosto de 2010.] http://perso.wanadoo.es/pictob/microprg.htm#arquitectura_von_neumann_y_arquitectura_harvard.

5. **Teoría, Depto. de Mecánica de Medios Continuos y.** Grupo de Mecánica Computacional. [En línea] [Citado el: 15 de 11 de 2020.] http://w3.mecanica.upm.es.

6. Arduino. *http://arduino.cc/es/.* [En línea] [Citado el: 3 de Octubre de 2010.] http://arduino.cc/es/.

7. El mercurio. [En línea] [Citado el: 20 de Octubre de 2010.] http://www.elmercuriodelosestudiantes.cl/B3B2BC648E1D49F29E9B979377B5B9A8/71BAC0D1CA4F4DB4A7BFC7B4B338E65A/573BCB6A47524A99A18C4412614C0960/816751505353425A8B2BBBDBFF8EDC02/articulo/5423.asp.

8. **Arévalo, Christopher y Ramirez, Verónica.** Memoria de Titulo. *Desarrollo de Aplicacion Movil Sobre Plataforma Android en Apoyo a Visitas Medica.* Chillán, Chile : s.n., 2010.

9. **Baz Alonso, Arturo y Ferreira Artime, Irene.** *Dispositivos móviles* . [En Linea] España : s.n., 2009.

10. **Gajardo Díaz, Luis.** Dispositivos Móviles. Chillan, Chile : s.n., 2010.

11. **Fain, Pablo Alejandro.** http://www.pabloalejandrofain.com.ar. [En línea] [Citado el: 20 de Octubre de 2010.] http://www.pabloalejandrofain.com.ar/como-funciona-la-red-gsm..

12. **Gajardo Díaz, Luis.** *OS Móvil.* Chillan, Chile : s.n., 2010.

13. http://www.techpluto.com. [En línea] [Citado el: 24 de Octubre de 2010.] http://www.techpluto.com/smartphone-characteristics/.

14. **Kaplan, Elliott.** *Understanding GPS: Principles and Applications.* 2008.

15. http://www.worldlingo.com. [En línea] [Citado el: 15 de Octubre de 2010.] http://www.worldlingo.com/ma/enwiki/es/Accelerometer..

16. http://developer.android.com. [En línea] [Citado el: 20 de Octubre de 2010.] http://developer.android.com/guide/basics/what-is-android.html.

17. **Gajardo Díaz, Luis.** *Gajardo, Luis. Introducción a la Plataforma Android.* [Electronico] Chillan, Ñuble, Chile, Región del Bio Bio. : s.n., Abril-Junio de 2010.

18. Apache Strut 2. [En línea] [Citado el: 24 de Octubre de 2010.] http://struts.apache.org/2.x/docs/home.html.

19. Google Maps. [En línea] [Citado el: 24 de Octubre de 2010.] http://maps.google.com/support/bin/static.py?hl=es&page=guide.cs&guide=21670&from=21670&rd=1.

20. Ajax. [En línea] [Citado el: 25 de Octubre de 2010.] http://www.uberbin.net/archivos/internet/ajax-un-nuevo-acercamiento-a-aplicaciones-web.php.

21. **Larman, Craig.** *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Procss.* s.l. : Prentice Hall., 2001.

22. **Somerville, Ian.** *Ingenieria del Software.* s.l. : Prentice Hall, 2005. ISBN: 8478290745.

23. **Saeidian, Hossein.** *Una evaluación del del modelo entidad relación extendido.* [En Linea] 1996.

24. **W. Ambler, Scott.** Agilemodeling. [En línea] Ambysoft, 2010. [Citado el: 15 de 12 de 2010.] http://www.agilemodeling.com/artifacts/communicationDiagram.htm.

25. **Pressman, Roger S.** *Ingeniería de Software un enfoque práctico.* s.l. : Mc Graw Hill, 2008.

26. Chuidiang. [En línea] [Citado el: 10 de Noviembre de 2010.] http://www.chuidiang.com/java/sockets/socket.php.

27. **España, Ministerio de Educación y Ciencia de.** hera.cnice.mec.es. [En línea] [Citado el: 4 de Noviembre de 2010.] hera.cnice.mec.es/redes2/contenido/Pdf/mod1_4.pdf.

28. Wikipedia. [En línea] 12 de Septiembre de 2010. [Citado el: 17 de Noviembre de 2010.] http://es.wikipedia.org/wiki/YUV.

29. **Burnette, Ed.** *Hello, Android.* s.l. : O'Reilly, 2009.

30. [En línea] [Citado el: 22 de agosto de 2010.] http://www.mailxmail.com/curso-gsm/sub-sistema-conmutacion-red-nss.

31. http://www.worldlingo.com. [En línea] [Citado el: 15 de Octubre de 2010.] http://www.worldlingo.com/ma/enwiki/es/Accelerometer.

Anexos

Proyecto
Integración de una plataforma
Smartphone a entornos de robótica móvil.

Anexo I

Especificación de casos de uso SkyNet módulo Android

Para el correcto funcionamiento de algunas de las funcionalidades de esta aplicación se asume que se cuenta con conexión a una red WIFI en todo momento, y que el SmartPhone Android utilizado cuente con dispositivo Bluetooth y GPS, de no ser así la aplicación está sujeta a errores de conexión o no se permitirá el acceso a otras funcionalidades.

A continuación se presentan los casos de uso del sistema. En primer lugar la Tabla I.1 detalla al caso de uso *Autenticar usuario*.

Caso de Uso	**Autenticar usuario**
Id	1
Descripción	Caso de uso encargado de verificar si el usuario que desea ingresar a la aplicación tiene permitido el acceso.
Objetivo	Permitir el acceso solo a personas que previamente autorizadas mediante la entrega de una clave.
Actores del Sistema	Usuario SkyNet
Actores Secundarios	No Existen
Precondiciones	No Existen
Flujos Principales	1.- Este caso de uso comienza cuando el usuario SkyNet desea ingresar a la aplicación, por lo cual se dirige al menú del smartphone, y selecciona la aplicación llamada SkyNet módulo Android. 2.- Como respuesta a la acción del ítem 1, arranca la aplicación SkyNet módulo Android y presenta al usuario la pantalla de autentificación, solicitando el password de la aplicación, este password se ingresa a la opción *Preferencias* de la aplicación. El password por defecto de la aplicación es "*123*". 3.- El usuario ingresa el password, ya sea el por defecto o el ingresado en reemplazo de este. 4.- El sistema verifica si password ingresado, es el registrado para la aplicación. Si lo es permite el ingreso al menú de la aplicación. En el caso de que el password sea el por defecto se comunica que se debe cambiar. 5.- El caso de uso finaliza.
	En caso de ser válido el password ingresado por el usuario, el usuario tiene acceso a las funcionalidades de la aplicación.
Flujos Excepcionales	4a.: El sistema verifica el password, el que no es encontrado en los registros, por lo cual comunica al usuario SkyNet que password ingresado no es válido y no permite el ingreso a la aplicación.

Tabla I.1: Caso de uso *Autenticar usuario*.

La Tabla 1.2 detalla el caso de uso *Información del servidor*.

Caso de Uso	**Información del servidor**
Id	2
Descripción	Caso de uso que permite verificar la dirección IP y puerto del servidor, y modificar los datos mencionados anteriormente si es que existe un cambio en la IP donde es alojado el servidor y el puerto por el cual se comunica.
Objetivo	Establecer la dirección IP y Puerto mediante el cual se realizara la comunicación con el sistema de monitoreo y control, SkyNet módulo Web.
Actores del Sistema	Usuario SkyNet.
Actores Secundarios	No Existen
Precondiciones	El Usuario SkyNet se ha autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet desea establecer los parámetros para la comunicación con el servidor, es decir dirección IP y puerto, por lo cual ingresa al menú *Información del servidor*. 2.- El sistema despliega la pantalla de ingreso de los parámetros de comunicación, mostrando los datos actuales y solicitando la nueva dirección IP y puerto del servidor. 3.- El Usuario SkyNet verifica los parámetros de comunicación, ingresando los nuevos parámetros y presionando el botón guardar. 4.- El sistema válido la existencia de ambos datos, es decir la dirección IP y el puerto de comunicación. Y Almacena los datos ingresados. 5.- El caso de uso finaliza.
Post-Condiciones	La dirección IP y puerto han sido actualizados, registrando los datos ingresados por el usuario SkyNet.
Flujos Excepcionales	3a.: El usuario SkyNet verifica los parámetros actualmente existentes, ante lo cual desea cancelar el ingreso seleccionando el botón cancelar y volver al menú principal de la aplicación. 4a.: El sistema verifica la existencia de todos los datos, de no existir todos los datos solicitados la operación de guardado no se realizara. Y se comunicará el problema sucedido al usuario.

Tabla I.2: Caso de uso *Información del servidor*.

La Tabla I.3 detalla el caso de uso *Control mediante SkyNet*.

Caso de Uso	**Control mediante SkyNet**
Id	3
Descripción	Caso de uso que permite establecer comunicación con un dispositivo bluetooth (perteneciente a SkyBot), y además conectarse al servidor de manera de recibir las localizaciones geográficas a visitar. Una vez enviada la primera coordenada desde el servidor, la aplicación Android inicia sus actividades.
Objetivo	El objetivo del caso de uso consiste en permitir conectar la aplicación a un dispositivo bluetooth, así como también iniciar, pausar o cancelar la aplicación. Las coordenadas son enviadas mediante un sistema web. El dispositivo bluetooth con el cual se comunicara pertenece a SkyBot, el cual es un robot y este responderá ante las órdenes de SkyNet módulo Android.
Actores del Sistema	Usuario SkyNet
Actores Secundarios	SkyBot
Precondiciones	El usuario SkyNet debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet desea hacer uso de la aplicación por completo, es decir el modulo web y el módulo Android (Monitoreo y Control web, del robot denominado SkyBot). Por lo cual se dirige al menú *Control mediante SkyNet*. 2.- Se despliega la pantalla de presentación de la información, esta pantalla entregara la información de la comunicación entre servidor-SmartPhone y bluetooth (SkyBot)-SmartPhone, además se presentaran los botones necesarios para realizar la conexión a un dispositivo bluetooth y para iniciar, detener o cancelar la aplicación. 3.- El Usuario SkyNet verificara la información y decidirá la acción a realizar, ingresando al menú o seleccionando el botón deseado. 4.- El sistema capturara la selección de una acción y reaccionará de la siguiente forma. 4a.: Si la selección corresponde al menú conectar a dispositivo realiza el caso de uso *conectar a bluetooth*. 4b.: Si la selección corresponde al botón conectar manualmente realiza el caso de uso *conectar manualmente a dispositivo bluetooth*. 5.- El caso de uso finaliza.
Post-Condiciones	Se ha realizado el monitoreo y control, mediante SkyNet.
Flujos Excepcionales	No Existen.

Tabla I.3: Caso de uso *Control mediante SkyNet*.

La Tabla 1.4 describe el caso de uso *Conectar a bluetooth*.

Caso de Uso	**Conectar a bluetooth**
Id	4
Descripción	Caso de uso que permite seleccionar de una lista los dispositivos bluetooth con los cuales anteriormente se ha establecido conexión, y establecer comunicación con él. O buscar nuevos dispositivos dentro de alcance del SmartPhone para posteriormente conectarse a él seleccionándolo desde una lista. El dispositivo bluetooth al cual conectarse pertenece a SkyBot.
Objetivo	Conectar el SmartPhone mediante bluetooth a SkyBot.
Actores del Sistema	Usuario SkyNet.
Actores Secundarios	SkyBot
Precondiciones	El usuario SkyNet debe haberse autenticado inicialmente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet desea conectar la aplicación a un dispositivo bluetooth, por lo cual ingresa al menú *Conectar a bluetooth*. 2.- Se despliega la pantalla con los dispositivos sincronizados anteriormente, y a se entrega el botón utilizar la función buscar nuevos dispositivos. 3.- El usuario SkyNet selecciona el dispositivo de la lista de dispositivos antes sincronizados, o en su defecto selecciona el botón buscar nuevos dispositivos. 4.- El sistema conecta la aplicación al dispositivo bluetooth seleccionado. 5.- El caso de uso finaliza.
Post-Condiciones	La aplicación se ha conectado a un dispositivo bluetooth, en el caso que se seleccionase un dispositivo.
Flujos Excepcionales	4a.: El sistema ante la selección del botón *Buscar nuevos dispositivos*, agrega a una lista los dispositivos encontrados. 3a.: El Usuario SkyNet verifica la lista de dispositivos antes sincronizados y la de nuevos dispositivos detectados, y selecciona el dispositivo que desee.

Tabla I.4: Caso de uso *Conectar a bluetooth*.

La Tabla I.5 detalla el caso de uso *Conectar manualmente a dispositivo Bluetooth*.

Caso de Uso	**Conectar manualmente a dispositivo bluetooth**
Id	5
Descripción	Caso de uso que permite establecer la comunicación con un dispositivo bluetooth que no sea visible o no sea detectado por el SmartPhone, es por ello que permite el ingreso de una dirección MAC, para de esta manera conectar a dicho dispositivo bluetooth. El dispositivo bluetooth al cual se conectara pertenece a SkyBot, el cual es un robot que realizara las ordenes enviadas por SkyNet módulo Android, y responderá una vez realizada estas acciones.
Objetivo	Conectarse a un dispositivo bluetooth que no aparezca en la búsqueda realizada por el caso de uso con ID 4.
Actores del Sistema	Usuario SkyNet
Actores Secundarios	No Existen
Precondiciones	El usuario SkyNet debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet desea conectar la aplicación a un dispositivo bluetooth, de manera manual, por lo cual ingresa al menú *conectar manualmente a dispositivo bluetooth*. 2.- Se despliega la pantalla de ingreso de dispositivo bluetooth, solicitando la dirección MAC del dispositivo a conectar. 3.- El usuario SkyNet ingresa la dirección MAC del dispositivo al cual se desea conectar, presiona el botón conectar. 4.- El sistema verifica si la dirección MAC, si es válida se conecta la aplicación a este dispositivo. 5.- El caso de uso finaliza.
Post-Condiciones	La aplicación se ha conectado a un dispositivo bluetooth.
Flujos Excepcionales	4a.: El sistema ante la validación, y detección de errores en el formato o que no se ha ingresado alguna dirección, comunica el error al usuario SkyNet mediante un mensaje informativo, y posteriormente realiza la acción 2.

Tabla I.5: Caso de uso *Conectar manualmente a dispositivo bluetooth*.

La Tabla I.6 contiene el detalle del caso de uso *Control manual*.

Caso de Uso	**Control manual**
Id	6
Descripción	Caso de uso que permite el control del robot mediante comunicación bluetooth y la selección de movimientos desde botones direccionales.
Objetivo	Este caso de uso comienza cuando el usuario decide controlar los movimientos del robot mediante botones direccionales.
Actores del Sistema	Usuario SkyNet
Actores Secundarios	SkyBot
Precondiciones	El usuario SkyNet debe haberse autenticado inicialmente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet desea controlar de forma manual los movimientos del Robot, ante esto el usuario SkyNet ingresa al menú *Control manual*. 2.- Se despliega la pantalla de control manual, con sus respectivos botones direccionales. Además el sistema despliega menús de conexión manual y automática a dispositivos Bluetooth. 3.- El usuario SkyNet visualiza la pantalla de control manual, ante lo cual presiona el botón menú y se conecta a un dispositivo bluetooth mediante los métodos de conexión automática y manual a dispositivo Bluetooth (Caso de uso *Conectar a bluetooth* y *Conectar manualmente a dispositivo bluetooth* respectivamente). 4.- Una vez establecida la comunicación a un dispositivo Bluetooth, el sistema permite la selección de botones y envió de información a robot, para realizar el movimiento seleccionado. 5.- El caso de uso finaliza.
Post-Condiciones	La aplicación se ha conectado a un dispositivo bluetooth. El robot ha realizado los movimientos seleccionados por el usuario.
Flujos Excepcionales:	3a.: El sistema ante la selección del menú *Control manual*, despliega caso de uso *Conectar a bluetooth*. Una vez realizado este caso de uso de manera satisfactoria es decir, el Usuario SkyNet se ha conectado a un dispositivo bluetooth. Vuelve al curso del caso de uso actual. 3b.: El sistema ante la selección de menú *Conectar a Bluetooth*, despliega caso de uso *Conectar a manualmente dispositivo bluetooth*. Una vez realizado este caso de uso de manera satisfactoria es decir, el usuario SkyNet se ha conectado a un dispositivo bluetooth. Vuelve al curso del caso de uso actual.

Tabla I.6: Caso de uso *Control manual.*

La Tabla I.7 contiene el detalle del caso de uso *Ver ayuda*.

Caso de Uso	**Ver ayuda**
Id	7
Descripción	Caso de uso que permite visualizar información de ayuda, de manera de orientar al usuario.
Objetivo	Informar al usuario la forma de utilización de las opciones de SkyNet módulo Android.
Actores del Sistema	Usuario SkyNet
Actores Secundarios	
Precondiciones	El usuario SkyNet debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet desea visualizar la información de ayuda, por lo cual selecciona el botón *Ver ayuda* (circulo azul con un signo de interrogación de color blanco en su interior). 2.- Se despliega la pantalla información que indica como operar en el menú que se encuentra.
Post-Condiciones	El Usuario SkyNet ha visualizado la información de ayuda.
Flujos Excepcionales:	No existen.

Tabla I.7: caso de uso *Ver ayuda*.

Anexo II

Plan de pruebas de SkyNet módulo Android

II Plan de pruebas de SkyNet módulo Android

II.1 Propósito

Describir el plan de pruebas para el módulo Android del sistema "SkyNet". En concreto se define los siguientes objetivos específicos:

- Identifica los elementos que a probar.
- Describe la estrategia de pruebas que se va a seguir en el proceso de prueba.
- Identifica los recursos necesarios para llevar a cabo el proceso de prueba y estima los esfuerzos que conlleva.
- Lista los resultados que se obtienen de las actividades de prueba.

II.2 Ámbito

Este Plan de Pruebas describe las pruebas de unidad, integración y del sistema que se aplicarán a la aplicación Android desarrollada. Teniendo como objetivo probar todos los requisitos definidos en la Especificación de requisitos y en el Modelo de casos de uso.

II.3 Requerimientos de las pruebas

La lista que proporcionamos en esta sección identifica los elementos (casos de uso, requisitos funcionales y requisitos no funcionales) que son objetivos de las pruebas. Es decir, los elementos qué se van a probar.

II.3.1 Pruebas de integridad de los datos:

Verificar la recuperación correcta de las modificaciones realizadas en la base de datos.

II.3.2 Pruebas de funcionalidad:

Para realizar las pruebas de funcionalidad se verificara el funcionamiento de los siguientes casos de uso:

- Verificar funcionamiento caso de uso Autenticar usuario.
- Verificar funcionamiento caso de uso Ver ayuda.
- Verificar funcionamiento caso de uso Información del servidor.
- Verificar funcionamiento caso de uso Control manual.
- Verificar funcionamiento caso de uso Control mediante SkyNet.
- Verificar funcionamiento caso de uso Conectar a bluetooth.
- Verificar funcionamiento caso de uso Conectar manualmente a dispositivo Bluetooth.
- Verificar funcionamiento caso de uso Ver movimientos realizados
- Verificar funcionamiento caso de uso Preferencias.

II.3.3 Pruebas de interfaz de usuario:

- Verificar que la navegación a través de las vistas se realiza de manera sencilla.
- Navegar a través de todos los casos de uso, verificando que cada interfaz de usuario se comprende fácilmente.

II.3.4 Estrategia de Prueba

En esta sección presentamos la estrategia a utilizar, para probar la aplicación Android denominada *Skynet módulo Android*. En la sección anterior hemos descrito qué elementos del sistema software se probarán, y en esta sección se define cómo se realizarán las pruebas.

II.4 Tipos de pruebas y técnicas

II.4.1 Pruebas de integridad de datos.

A continuación en la Tabla II.1, se presentan las pruebas de integridad a realizar en la aplicación SkyNet módulo Android.

Objetivos de la prueba	**Comprobar que los procedimientos y métodos de acceso a los datos funcionan correctamente.**
Técnicas	Invocar cada procedimiento o método de acceso a los datos con entradas válidas e inválidas. Inspeccionar los datos para asegurar que estos son los previstos
Criterios de finalización	Todos los procedimientos y métodos de acceso funcionan como se diseñaron y sin ningún error en los datos.

Tabla II.1: Pruebas de integridad de los datos

II.4.2Pruebas de funcionalidad.

Las pruebas de funcionalidad se centraran en los requisitos que puedan ser trazados directamente de los casos de uso y reglas de negocio. El objetivo de estas pruebas es verificar la aceptación, procesamiento y recuperación de datos, y la adecuada implementación de las reglas de negocio. Este tipo de pruebas están basadas en técnicas de caja negra, es decir, verificar la aplicación interaccionando a través de las interfaces de usuario y analizando los resultados.

En la Tabla II.2, se presenta el detalle de las pruebas de funcionalidad a aplicarse en SkyNet modulo web.

Objetivos de la prueba	**Asegurar la navegación correcta de la aplicación, la entrada de datos, su procesamiento y recuperación.**
Técnicas	Ejecutar cada caso de uso y flujo del caso de uso con datos válidos e inválidos para verificar lo siguiente: • Cuando se utilizan datos correctos se obtienen los resultados esperados. • Cuando se utilizan datos incorrectos se obtienen los mensajes de error o advertencias adecuadas. • Cada regla de negocio se ha aplicado correctamente.
Criterios de finalización	• Todas las pruebas planificadas se han ejecutado. • Todos los defectos identificados se han considerado.
Consideraciones	No Existen

Tabla II.2: Pruebas de funcionalidad.

II.4.3 Pruebas de interfaz de usuario.

Las pruebas de interfaz de usuario verifican la interacción del usuario con el sistema software. El objetivo de esta prueba es asegurar que la interfaz de usuario permite al usuario acceder y navegar a través de toda la funcionalidad de la aplicación.

La Tabla II.3, correspondiente al detalle de las pruebas de Interfaz de usuario.

Objetivos de la prueba	**Verificar los siguientes objetivos:** • La navegación a través de la aplicación refleja adecuadamente la lógica de funcionamiento deseada. • Las ventanas y sus características, como menús, tamaño, posición y estado cumplen los estándares.
Técnicas	Crear o modificar pruebas para cada vista con el objetivo de verificar la correcta navegación y su estado.
Criterios de finalización	Cada ventana se ha verificado con éxito y es consistente con la versión de referencia o con los estándares utilizados.
Consideraciones	No existen

Tabla II.3: Pruebas de interfaz de usuario.

II.4.4 Herramientas a utilizar para el desarrollo de las pruebas

La Tabla II.4, correspondiente a las herramientas que se utilizaran para llevar a cabo el proceso de prueba:

Tipo de Prueba	Herramienta
Gestión del proyecto	Microsoft Project
Herramienta DBMS	SQLite

Tabla II.4: Herramientas

II.5 Recursos a utilizar en las pruebas

A continuación se describen los recursos necesarios para realizar el proceso de prueba, sus principales responsabilidades y características.

II.5.1 Recursos hardware

En la Tabla II.5, se presentan los recursos Hardware a utilizaran para la aplicación de las pruebas.

Recurso	Cantidad	Nombre y tipo
PC	2	Diseño de las pruebas
PC	2	Ejecución de las pruebas

Tabla II.5: Recursos de hardware

II.6 Configuración del entorno de prueba

No existe configuración del entorno de prueba.

II.6.1 Recursos humanos

En la Tabla II.6, se presentan los diferentes roles necesarios así como también la cantidad necesitada en cada rol, de manera de realizar las pruebas.

Recursos humanos		
Rol	**Mínimos recursos recomendados**	**Responsabilidades específicas o comentarios**
Gestor de prueba	1	Proporcionar una gestión adecuada. Responsabilidades: • Proporcionar una dirección técnica. • Adquirir los recursos apropiados. • Informar de la gestión.
Diseñador de prueba	1	Identificar, priorizare implementar los casos de prueba. Responsabilidades: • Generar el Plan de pruebas. • Diseñar los Casos de prueba. • Evaluar el esfuerzo de prueba.
Probador (Tester)	2	Ejecutar las pruebas. Responsabilidades: • Ejecutar pruebas. • Recuperar los errores. • Documentar los defectos.

Tabla II.6: Recursos humanos

II.7 Tareas de la etapa de pruebas

A continuación se presentan las tareas a realizar en cada una de las actividades.

II.7.1 Planificación de las pruebas

- Identificar los requisitos para las pruebas.
- Desarrollar la estrategia de pruebas.
- Identificar los recursos necesarios para realizar las pruebas.
- Generar el Plan de pruebas.

II.7.2 Diseño de las pruebas

- Desarrollo de las pruebas.
- Identificar y describir los casos de prueba.

II.7.3 Implementación de las pruebas

- Establecer el entorno de prueba.
- Desarrollar las clases de prueba, los componentes de prueba y los datos de prueba.

II.7.4 Ejecución de las pruebas

- Ejecutar los casos de prueba.
- Evaluar la ejecución del proceso de prueba.
- Verificar los resultados.
- Investigar los resultados no esperados.
- Registrar los defectos.

II.7.5 Evaluación de las pruebas

- Evaluar la cobertura de los casos de prueba.
- Evaluar la cobertura del código.
- Analizar los defectos.
- Determinar si se han alcanzado los criterios de las pruebas.

II.8 Casos de Prueba

A continuación se presentan los casos de prueba de la aplicación Android, denominada SkyNet módulo Android. En primer lugar la Tabla II.7, presenta el caso de prueba denominado *Verificar caso de uso Autenticar usuario*.

De aquí en adelante se entenderá que existe solo un actor que interactúa con seleccionando las opciones de la aplicación SkyNet módulo Android, este actor recibe el nombre de Usuario SkyNet, sin embargo por simplicidad de aquí en adelante se le llamara simplemente usuario. Para facilitar el entendimiento de lo señalado anteriormente se recomienda la lectura de los puntos 6.1.1 y 6.5.2.

Caso de prueba de aceptación
Código Caso de Prueba: 1.0 Verificar el caso de uso Autenticar usuario.
Descripción de la Prueba: Esta prueba intentará autenticar a un usuario válido, el cual posee un password registrado en la aplicación.
Condiciones de Ejecución: No se requieren condiciones especiales.
Entrada / Pasos de Ejecución: **1.** Se selecciona la aplicación SkyNet módulo Android. **2.** Se ingresa un password válido, y se presiona el botón *Ingresar*.
Resultado Esperado: **1.** A presionar el botón ingresar la aplicación permite el ingreso al menú de SkyNet módulo Android.
Evaluación de la Prueba: Evaluación Satisfactoria

Tabla II.7: Caso de prueba *Verificar el caso de uso Autenticar usuario*.

En la Tabla II.8, se presenta el caso de prueba denominado *Verificar caso de uso Información del servidor*.

Caso de prueba de aceptación
Código Caso de Prueba: 2.0 Verificar caso de uso Información del Servidor
Descripción de la Prueba: Esta Prueba pretende comprobar la correcta modificación de los datos del servidor, los cuales consisten en dirección IP y puerto de comunicación.
Condiciones de Ejecución: El usuario debe haberse autenticado exitosamente.
Entrada / Pasos de Ejecución: **1.** Se selecciona el menú *Información del servidor*, correspondiente al caso de uso que posee el mismo nombre. **2.** Se despliega una vista que contempla la información actual del servidor y los campos para el ingreso de la nueva información. **3.** El usuario completa los campos solicitados y envía la información. **4.** El usuario realiza nuevamente el caso de uso *Información del servidor*. **5.** El usuario comprueba que la información que ingreso se registra en la aplicación.
Resultado Esperado: A presionar el botón *Información del Servidor*, se desplegara una vista en la cual se presentara la información actualmente existente para el servidor, a su vez se presentara los campos para ingresar los nuevos datos. Al registrar una nuevos datos para el servidor y volver a realizar el caso de uso *Información del Servidor*, se visualiza la información ingresada por el usuario a modo de prueba.
Evaluación de la Prueba: Evaluación satisfactoria

Tabla II.8: Caso de prueba denominado *verificar caso de uso Información del servidor.*

En la Tabla II.9, se presenta el caso de prueba *Verificar funcionamiento de caso de uso Ver ayuda*.

Caso de Prueba de Aceptación
Código Caso de Prueba: 3.0 Verificar funcionamiento caso de uso Ver ayuda
Descripción de la Prueba: Esta Prueba pretende comprobar la aparición de la ayuda en pantalla.
Condiciones de Ejecución: El usuario debe haberse autenticado exitosamente.
Entrada / Pasos de Ejecución: **1.** Se selecciona el botón *Ver ayuda*, correspondiente al caso de uso *Ver ayuda*. Este botón está representado por un círculo azul con un signo de interrogación blanco en su interior.
Resultado Esperado: A presionar el botón *Ver Ayuda*, se desplegara una ventana informativa. La cual contendrá la información de ayuda para la aplicación.
Evaluación de la Prueba: Evaluación satisfactoria

Tabla II.9: Caso de prueba denominado *Verificar funcionamiento de caso de uso Ver ayuda*.

La Tabla II.10, presenta el caso de prueba *Verificar funcionamiento de caso de uso Control manual*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 4.0	Verificar funcionamiento de caso de uso Control manual
Descripción de la Prueba: Esta Prueba intentará realizar el caso de uso control manual, el cual permite conectarse a un dispositivo Bluetooth el cual es parte de SkyBot, de manera de controlar mediante botones direccionales las acciones que realice el SkyBot(robot).	
Condiciones de Ejecución: El usuario debe haberse autenticado exitosamente.	
Entrada / Pasos de Ejecución: **1.** Se selecciona el caso de uso *Control manual*, seleccionando el icono *Control manual*, el cual es representado por una imagen de un joystick. **2.** Luego el usuario presiona el botón menú, e ingresa al caso de uso *Conectar manualmente a dispositivo Bluetooth*. **3.** El usuario presiona el botón menú, y selecciona el caso de uso *Conectar a bluetooth*. **4.** El usuario ante la realización satisfactoria del paso 2 y 3, se encuentra conectado a un dispositivo bluetooth. **5.** El usuario selecciona de acuerdo a sus preferencias los botones direccionales y el robot reacción ante cada selección.	
Resultado Esperado: A presionar el botón *Control manual*, y selecciona luego de conectarse a un bluetooth un botón direccional, ante lo cual el robot reacciona.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla II.10: Se presenta el caso de prueba *Verificar funcionamiento de caso de uso Control manual*.

La Tabla II.11, presenta el caso de prueba *Verificar funcionamiento de caso de uso Conectar a bluetooth*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 5.0	Verificar funcionamiento caso de uso Conectar a bluetooth
Descripción de la Prueba: Esta Prueba pretende comprobar que el caso de uso en cuestión realmente permite establecer conexión con un dispositivo bluetooth.	
Condiciones de Ejecución: El usuario debe haberse autenticado exitosamente.	
Entrada / Pasos de Ejecución: **1.** Se selecciona el botón *Conectar a bluetooth*, correspondiente al caso de uso *Conectar a bluetooth*. **2.** Seleccionar un dispositivo de la lista. **3.** Verificar el estado de la comunicación, debería ser conectado al dispositivo elegido.	
Resultado Esperado: Al seleccionar el dispositivo la aplicación comienza a establecer la comunicación, la cual se logra en casi automáticamente.	
Evaluación de la Prueba: Evaluación Satisfactoria	

Tabla II.11: Se presenta el caso de prueba *Verificar funcionamiento de caso de uso Conectar a bluetooth*.

La Tabla II.12, se presenta el caso de prueba *Verificar funcionamiento de caso de uso Preferencias*.

Caso de Prueba de Aceptación
Código Caso de Prueba: 6.0 Verificar funcionamiento caso de uso Preferencias
Descripción de la Prueba: Esta Prueba pretende comprobar que el caso de uso preferencias.
Condiciones de Ejecución: El usuario debe haberse autenticado exitosamente.
Entrada / Pasos de Ejecución: **1.** Se selecciona el botón *Preferencias*, ubicado en el menú principal, correspondiente al caso de uso *Preferencias*. **2.** Seleccionar se ingresa a las preferencias modificando los datos y selecciones ahí descritas. **3.** Se selecciona guardar **4.** Se realiza el paso 1 nuevamente **5.** Se verifica que las selecciones realizadas en el paso 2 hayan sido registradas.
Resultado Esperado: Las selecciones de preferencias realizadas han sido guardadas.
Evaluación de la Prueba: Evaluación satisfactoria

Tabla II.12: Se presenta el caso de prueba *Verificar funcionamiento de caso de uso Conectar a bluetooth*.

La Tabla II.13, se presenta el caso de prueba *Verificar funcionamiento de caso de uso Conectar manualmente a dispositivo bluetooth*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 7.0	Verificar funcionamiento caso de uso Conectar manualmente a dispositivo bluetooth
Descripción de la Prueba: Esta prueba pretende comprobar que el caso de uso en cuestión realmente permite establecer conexión con un dispositivo bluetooth mediante el ingreso de su dirección MAC.	
Condiciones de Ejecución: El usuario debe haberse autenticado exitosamente.	
Entrada / Pasos de Ejecución: 1. Se selecciona el botón *Conectar manualmente a dispositivo bluetooth*, correspondiente al caso de uso *Conectar manualmente a dispositivo a Bluetooth*. 2. Se Ingresa una dirección MAC. Y se presiona conectar. 3. Verificar el estado de la comunicación, debería ser conectado al dispositivo elegido, en caso de que la MAC sea de un dispositivo disponible.	
Resultado Esperado: Al seleccionar el dispositivo la aplicación comienza a establecer la comunicación, la cual se logra en casi automáticamente.	
Evaluación de la Prueba: Evaluación Satisfactoria, aunque dependiente de la veracidad de la información de la dirección MAC. En caso de no estar disponible dicha dirección no conecta y no ocurre ningún problema como resultado a esta opción.	

Tabla II.13: Se presenta el caso de prueba *Verificar funcionamiento de caso de uso Conectar manualmente a dispositivo bluetooth*.

La Tabla II.14, presenta el caso de prueba *Verificar funcionamiento del caso de uso Ver movimientos realizados*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 8.0	Verificar funcionamiento del caso de uso Ver movimientos realizados
Descripción de la Prueba: Esta prueba pretende comprobar que ante movimientos realizados, en el caso de ser controlado mediante el sistema de monitoreo y control, estos se registraran y graficarán en un mapa.	
Condiciones de Ejecución: El usuario debe haberse autenticado exitosamente.	
Entrada / Pasos de Ejecución: **1.** Se selecciona el botón *Ver movimientos realizados*, el cual corresponde al caso de uso *Ver Movimientos realizados*. **2.** En caso de estar en funcionamiento el monitoreo y control mediante SkyNet, se registraran las localizaciones geográficas visitadas, las cuales se desplegaran en un mapa.	
Resultado Esperado: Si existen localizaciones visitadas, estas son graficadas en el mapa.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla II.14: Se presenta el caso de prueba *Verificar funcionamiento de caso de uso Ver movimientos realizados*.

La Tabla II.15, presenta el caso de prueba *Verificar funcionamiento de caso de uso Control mediante SkyNet*.

Caso de Prueba de Aceptación
Código Caso de Prueba: 9.0 Verificar funcionamiento caso de uso Control mediante Skynet.
Descripción de la Prueba: Esta Prueba pretende comprobar que el caso de uso Control mediante SkyNet, se realice correctamente.
Condiciones de Ejecución: El usuario debe haberse autenticado exitosamente.
Entrada / Pasos de Ejecución: **1.** Se selecciona el botón *Control mediante SkyNet*, correspondiente al caso de uso *Control mediante SkyNet.* **2.** Se establece conexión con el servidor. **3.** Se establece conexión con un bluetooth. **4.** Se reciben las órdenes desde el servidor. **5.** SkyNet módulo Android recibe las ordenes **6.** SkyNet módulo Android envía las ordenes a SkyBot. **7.** SkyBot ejecuta las acciones
Resultado Esperado: El resultado esperado es que se establece conexión con el servidor y un dispositivo bluetooth, se recibe las órdenes y se ejecutan de manera que el robot realiza la ruta que se nos indica y llega a un objetivo.
Evaluación de la Prueba: Evaluación Satisfactoria, existe margen de error con respecto al objetivo, el cual es de 9 metros.

Tabla II.15: Se presenta el caso de prueba *Verificar funcionamiento de caso de uso Control mediante SkyNet.*

Anexo III

Especificación de casos de uso SkyNet módulo Web

Especificación de casos de Uso Sistema SkyNet módulo Web.

Dado los requerimientos obtenidos y detallados en el Punto 8.2, luego de su análisis se han obtenido los siguientes casos de uso.

A continuación se presentan los casos de uso del Sistema SkyNet Módulo web. En primer lugar la Tabla III.1 correspondiente al caso de uso *Autenticar usuario*.

Caso de Uso	**Autenticar usuario**
Id	1
Descripción	Caso de uso encargado de autenticar al Usuario SkyNet o Administrador SkyNet, de manera de que tengan acceso al sistema sólo los usuarios registrados.
Objetivo	Permitir el ingreso de usuarios que estén registrados en el sistema.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No Existen
Precondiciones	El usuario (Administrador SkyNet o Usuario Skynet) debe estar registrado en el sistema para poder ingresar.
Flujos Principales	1.- Este caso de uso comienza cuando el usuario (ya sea Administrador SkyNet o Usuario SkyNet) desea ingresar al sistema, por lo cual ingresa la URL correspondiente al sistema. 2.- Ante la presencia de la pantalla de inicio de sesión, el usuario ingresa la información correspondiente a los campos *Usuario* y *Password* y luego presionando el botón *Ingresar*. 3.- El sistema comprueba que la información ingresada corresponda a un usuario registrado en el sistema. La información corresponde a un usuario registrado en el sistema por lo tanto permite el ingreso del usuario al sistema. 4.- Este caso de uso finaliza.
Post-Condiciones	El usuario debe tener acceso a las opciones del sistema si se autentico correctamente.
Flujos Excepcionales	3a.- Si el usuario o password son incorrectos se recarga la página indicando que hubo un error al pie de las casillas de *Usuario* y *Password*.

Tabla III.1: Caso de uso *Autenticar usuario*.

La Tabla III.2 detalla el caso de uso *Seleccionar ruta*.

Caso de Uso	**Seleccionar ruta**
Id	2
Descripción	Este caso de uso permite acceder al mapa en donde se realiza la selección de la ruta a seguir.
Objetivo	Este caso de uso permite al Usuario SkyNet o Administrador SkyNet acceder a la vista donde se puede seleccionar una ruta para un determinado robot.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No Existen.
Precondiciones	El usuario debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet o Administrador SkyNet selecciona la opción *Seleccionar Ruta*. 2.- El sistema verifica que por lo menos exista un robot conectado de los dos existentes en el sistema, de haber robots conectados, se obtiene el identificador de los robots conectados indicándole al sistema las opciones a habilitar para establecer la ruta de cada robot. 3.- El sistema permite el acceso al usuario a la página de selección de ruta. 4.- El caso de uso finaliza.
Post-Condiciones	El usuario accede al mapa de selección de rutas así como a las opciones que permiten establecer una ruta.
Flujos Excepcionales	3a.- Al no existir un robot conectado el sistema envía al usuario a una página, la cual le indica al usuario que no hay robots conectados al sistema.

Tabla III.2: Caso de uso *Seleccionar ruta*.

La Tabla III.3 detalla el caso de uso *Establecer coordenadas*.

Caso de Uso	**Establecer coordenadas**
Id	2.1
Descripción	En este caso de uso se permite agregar coordenadas a una ruta que será seguida por el robot.
Objetivo	Seleccionar una o varias posiciones al robot, con el fin de que éste visite cada una de éstas y realice una determinada acción, esta acción está determinada por la preferencia del usuario que la solicita.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No Existen
Precondiciones	El Usuario SkyNet o Administrador SkyNet debe haber sido autenticado. Debe existir al menos un robot (SkyBot) conectado que reciba la orden.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet o Administrador SkyNet selecciona el robot que debe visitar las coordenadas seleccionadas. 2.- El usuario presiona en el mapa disponible para establecer un punto que es parte de la ruta a enviar al o los robots seleccionados. 3.- El sistema obtiene las componentes de la coordenada solicitada por el usuario las transforma en objetos coordenadas para posteriormente agregarla a una ruta perteneciente al auto que debe seguirla. 4.- El sistema queda a la espera de la decisión a tomar del usuario con respecto a la ruta generada por la selección de los puntos. 5.- El caso de uso Finaliza.
Post-Condiciones	Se deben haber establecido las coordenadas que el usuario ingreso en la ruta correspondiente a visitar por el robot.
Flujos Excepcionales	4a.- El usuario selecciona otro robot para ser enviadas las coordenadas, lo cual provoca que se elimine la ruta trazada hasta el momento. 4b.- El usuario decide volver a seleccionar la ruta por lo cual borra la colección de coordenadas establecidas y comienza la selección nuevamente retornando al paso 1.

Tabla III.3: Caso de uso *Establecer coordenadas*.

La Tabla III.4, detalla el caso de uso *Enviar ruta*.

Caso de Uso	**Enviar ruta**
Id	2.2
Descripción	Este caso de uso permite enviar rutas a los robots conectados, las rutas a enviar previamente se seleccionaron del mapa.
Objetivo	Enviar una ruta al robot, con el fin de que éste visite cada una de las coordenadas que esta ruta posee y realice una determinada acción, esta acción está determinada por la preferencia del usuario que la solicita.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No Existen
Precondiciones	El usuario debe estar autenticado.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNeto o Administrador SkyNet selecciona la opción *Enviar ruta robot negro*. 2.- El sistema obtiene la ruta establecida para el robot seleccionado. 3.- El sistema verifica que el robot al cual será enviada la ruta se encuentre conectado al sistema. 4.- El sistema establece la ruta a enviarse para cada robot. 5.- Envía la primera coordenada iniciando la ruta del robot. 6.- El sistema inicia el caso de uso *Monitorear robot(s).*
Post-Condiciones	Se deben haber enviado las coordenadas que el usuario desea que visite el robot.
Flujos Excepcionales	1a.-El usuario selecciona la opción Enviar ruta robot amarillo, continuando el flujo normalmente. 1b.- El usuario selecciona la opción Enviar ruta ambos robots, continuando el flujo normalmente. 3a.- El robot no está conectado, por lo que el sistema envía al usuario a una página que le indica dicho problema.

Tabla III.4: Caso de uso *Enviar ruta*.

La Tabla III.5 detalla el caso de uso *Eliminar ruta*.

Caso de Uso	**Eliminar ruta**
Id	2.3
Descripción	Este caso de uso se encarga de borrar la ruta del robot seleccionado, en el caso de que el usuario lo estime conveniente, ya sea para iniciar una ruta nueva o sólo evitar que se envíe dicha ruta.
Objetivo	Eliminar la ruta a ser enviada a un cierto robot.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No Existen
Precondiciones	El Usuario SkyNet o Administrador SkyNet debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet o Administrador SkyNet selecciona la opción *Eliminar ruta auto negro*. 2.- El elimina las coordenadas pertenecientes a la ruta del mismo vehículo. 3.- El sistema elimina del mapa las coordenadas del auto seleccionado
Post-Condiciones	Se debe haber eliminado la ruta del vehículo que el usuario seleccionó.
Flujos Excepcionales	1a.- El usuario selecciona la opción *Eliminar ruta robot amarillo* y sigue el flujo normal. 1b.- El usuario selecciona la opción *Eliminar ruta ambos robots* y sigue el flujo normal para cada auto conectado.

Tabla III.5: Caso de uso *Eliminar ruta*.

La Tabla III.6, detalla el caso de uso *Restablecer valores*.

Caso de Uso	**Restablecer valores**
Id	2.4
Descripción	Este caso de uso se encarga de eliminar las rutas y las preferencias seleccionadas.
Objetivo	Elimina los datos ingresados por el usuario.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No existen
Precondiciones	El Usuario SkyNet o Administrador SkyNet debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNet o Administrador SkyNet presiona el botón *Restablecer valores*. 2.- El sistema elimina las rutas de los robots insertadas por el usuario y los datos relacionados a dichas rutas.
Post-Condiciones	Se deben haber restablecido los valores por defecto del sistema.
Flujos Excepcionales	No Existen

Tabla III.6: Caso de uso *Restablecer valores*.

La Tabla III.7 detalla el caso de uso *Monitorear robot(s)*.

Caso de Uso	**Monitorear robots**
Id	3
Descripción	Este caso de uso tiene como objeto llevar un control de la ruta realizada por él o los robots seleccionados en comparación a la ruta enviada por el Usuario SkyNet o Administrador SkyNet.
Objetivo	Tener un registro visual de las etapas en las cuales se encuentra el robot, de manera que el usuario tenga conocimiento de la posición aproximada donde se encuentra el vehículo y la ruta que ha seguido para llegar a dicha posición.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No Existen
Precondiciones	El usuario debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando finaliza el envió de la ruta, dando inicio al seguimiento del o los robots si por lo menos uno de éstos se encuentra conectado. 2.- El sistema muestra en el mapa la o las rutas a seguir, diferenciando la ruta de cada robot por color. 3.- Periódicamente las coordenadas que el robot visita se dibujarán en el mapa para representar la ruta real que el robot realiza.
Post-Condiciones	El robot realiza la ruta realizada.
Flujos Excepcionales	**3a.:** El sistema no puede conectarse con el robot por un error de conexión, comunica al usuario el error.

Tabla III.7: Caso de uso *Monitorear robot(s).*

La Tabla III.8, detalla el caso de uso *Detener* robot.

Caso de Uso	**Detener robot**
Id	3.1
Descripción	Este caso de uso se encarga de detener la acción en curso de uno o más robots.
Objetivo	Dar la orden para que el o los robots detengan su movimiento.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No existen
Precondiciones	El Usuario SkyNeto o Administrador SkyNet debe estar registrado en el sistema para poder ingresar.
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNeto o Administrador SkyNet presiona el botón *Detener robot negro*. 2.- El sistema envía la orden al robot de que detenga la acción en curso. 3.- El caso de uso finaliza.
Post-Condiciones	El robot recibe la orden de detener las acciones.
Flujos Excepcionales	1a.- El usuario presiona el botón *Detener robot amarillo*. 1b.- El usuario presiona el botón *Cancelar monitoreo*. 2a.: El sistema no puede conectarse con el robot por un error de conexión, enviando al usuario a una página de error.

Tabla III.8: Caso de uso *Detener robot.*

La Tabla III.9, detalla el caso de uso *Cerrar sesión*.

Caso de Uso	**Cerrar sesión**
Id	4
Descripción	Este caso de uso invalida la sesión actual, impidiendo que se pueda acceder a las opciones de SkyNet módulo Web.
Objetivo	Invalidar la sesión en curso, prohibiendo la selección de funcionalidades del sistema.
Actores del Sistema	Administrador SkyNet, Usuario SkyNet
Actores Secundarios	No Existen
Precondiciones	El Usuario SkyNeto o Administrador SkyNet debe haberse autenticado correctamente
Flujos Principales	1.- Este caso de uso comienza cuando el Usuario SkyNeto o Administrador SkyNet presiona la opción *Cerrar sesión*. 2.- El sistema obtiene la actual sesión. 3.- El sistema invalida la sesión. 4.- El sistema envía al usuario a la página de inicio, indicándole que ha cerrado su sesión. 5.- El caso de uso finaliza.
Post-Condiciones	Se cierra la sesión luego de detener cualquier acción de él o los robot(s).
Flujos Excepcionales	3a.**:** El sistema recibe un error, por lo cual invalida la sesión y continúa el flujo normal.

Tabla III.9: Caso de uso *Cerrar sesión*.

La Tabla III.10, detalla el caso de uso *Ingresar usuario*.

Caso de Uso	**Gestionar usuario**
Id	5
Descripción	Este caso de uso se encarga de gestionar a los usuarios que interactúan con el sistema, es decir, se encarga de registrar nuevos usuarios o eliminar usuarios registrados según sea el caso.
Objetivo	Ingresar nuevos usuarios al sistema.
Actores del Sistema	Administrador SkyNet.
Actores Secundarios	No Existen.
Precondiciones	El usuario debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Administrador SkyNet presiona el menú *Gestionar Usuario.* 2.- El sistema envía al Administrador SkyNet a una página que contiene el formulario para el registro de nuevos usuarios que se explica en el caso de uso 5.1 y la lista de usuarios en el sistema, pudiendo eliminarlos mediante el caso de uso 5.2 presente en el anexo III. 3.- El administrador escoge la opción de registrar usuario o eliminar usuario, prosiguiendo el flujo detallado en el caso de uso correspondiente.
Post-Condiciones	El administrador ingresa a la pantalla que le permitirá gestionar al usuario deseado.
Flujos Excepcionales	4a.- El sistema no presenta flujos excepcionales.

Tabla III.10: Caso de uso *Gestionar usuario*.

La Tabla III.11, detalla el caso de uso *Registrar usuario*.

Caso de Uso	**Registrar usuario**
Id	5.1
Descripción	Este caso de uso se encarga de agregar usuarios al sistema, para que tengan acceso a las funcionalidades que este contiene.
Objetivo	Ingresar nuevos usuarios al sistema.
Actores del Sistema	Administrador SkyNet.
Actores Secundarios	No Existen.
Precondiciones	El usuario debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Administrador SkyNet presiona el *Gestionar usuario.* 2.- El sistema envía al Administrador SkyNet a una página que contiene el formulario con los datos del nuevo usuario a ingresar, los cuales son el nombre de usuario, su password, el nombre y el perfil que utilizará. 3.- El administrador completa el formulario y lo envía para ser ingresado. 4.- El sistema valida la existencia de los datos, e ingresa los datos al sistema. 5.- El caso de uso finaliza.
Post-Condiciones	El nuevo usuario es ingresado al sistema, y puede acceder al sistema.
Flujos Excepcionales	4a.- El sistema detecta un error en el formulario, por lo cual indica al Administrador SkyNet el error para su posterior corrección y vuelve al paso 3.

Tabla III.11: Caso de uso *Registrar usuario*.

La Tabla III.12, detalla el caso de uso *Eliminar usuario*.

Caso de Uso	**Eliminar usuario**
Id	5.2
Descripción	Caso de uso que permite eliminar a un usuario del sistema, para de esta manera prohibir su ingreso.
Objetivo	Eliminar usuarios existentes en el sistema.
Actores del Sistema	Administrador SkyNet
Actores Secundarios	No Existe
Precondiciones	El usuario debe haberse autenticado exitosamente.
Flujos Principales	1.- Este caso de uso comienza cuando el Administrador SkyNet presiona el menú *Gestionar usuario*. 2.- El sistema envía al Administrador a una página que contiene una lista con los usuarios existentes en el sistema. 3.- El administrador presiona el botón *Eliminar usuario*. 4.- El sistema envía un mensaje de alerta, indicando que eliminará al usuario y preguntando si desea continuar. 5.- El usuario selecciona la opción aceptar. 6.- El sistema identifica al usuario y lo elimina del sistema. 7.- El caso de uso finaliza.
Post-Condiciones	El usuario seleccionado fue eliminado.
Flujos Excepcionales	5a.- El usuario selecciona la opción cancelar, por lo cual el caso de uso termina, sin eliminar un usuario.

Tabla III.12: Caso de uso *Eliminar usuario*.

Anexo IV

Pruebas de SkyNet Módulo Web

Plan de pruebas SkyNet módulo Web

IV.1 Propósito

Este documento describe el Plan de pruebas para el sistema de monitoreo “SkyNet”. En concreto define los siguientes objetivos específicos:

- Identifica los elementos que se van a probar.
- Describe la estrategia de pruebas que se va a seguir en el proceso de prueba.
- Identifica los recursos necesarios para llevar a cabo el proceso de prueba y estima los esfuerzos que conlleva.
- Lista los resultados que se obtienen de las actividades de prueba.

IV.2 Ámbito

Este Plan de Pruebas describe las pruebas de unidad, integración y del sistema que se aplicarán al sistema software desarrollado. El objetivo es probar todos los requisitos definidos en la Especificación de requisitos y en el Modelo de casos de uso.

IV.3 Requerimientos de las pruebas

La lista que proporcionamos en esta sección identifica los elementos (casos de uso, requisitos funcionales y requisitos no funcionales) que son objetivos de las pruebas. Es decir, los elementos qué vamos a probar.

IV.3.1 Pruebas de integridad de los datos

- Verificar la recuperación correcta de las modificaciones realizadas en la base de datos.
- Verificar accesos simultáneos de lectura de datos.

IV.3.2 Pruebas de funcionalidad:

- Verificar el caso de uso 1.0 Autenticar
- Verificar el caso de uso 2.0 Seleccionar ruta.
- Verificar el caso de uso 2.1 Establecer coordenadas
- Verificar el caso de uso 2.2 Enviar ruta
- Verificar el caso de uso 2.3 Eliminar ruta
- Verificar el caso de uso 2.4 Restablecer valores.
- Verificar el caso de uso 3.0.Monitorear robot(s).
- Verificar el caso de uso 3.1 Detener robot(s).
- Verificar el caso de uso 5.1 Ingresar usuario.
- Verificar el caso de uso 5.2 Eliminar usuario.

IV.3.3 Pruebas de interfaz de usuario:

- Verificar que la navegación a través de un conjunto de pantallas es fácil.
- Navegar a través de todos los casos de uso, verificando que cada interfaz de usuario se comprende fácilmente.

IV.3.4 Estrategia de Prueba

En esta sección presentamos el enfoque que vamos a utilizar para probar el sistema software. En la sección anterior hemos descrito qué elementos del sistema software vamos a probar, y en esta sección se define cómo se realizaran las pruebas.

IV.4 Tipos de pruebas y técnicas

IV.4.1 Pruebas de integridad de los datos.

En la Tabla IV.1, se presenta las pruebas de integridad de los datos

Objetivos de la prueba	Comprobar que los procedimientos y métodos de acceso a los datos funcionan correctamente.
Técnicas	Invocar cada procedimiento o método de acceso a los datos con entradas válidas e inválidas. Inspeccionar los datos para asegurar que estos son los previstos
Criterios de finalización	Todos los procedimientos y métodos de acceso funcionan como se diseñaron y sin ningún error en los datos.

Tabla IV.1: Pruebas de integridad de los datos.

IV.4.2 Pruebas de funcionalidad.

Las pruebas de funcionalidad se deberían centrar en cualquier requisito que pueda ser trazado directamente de los casos de uso y reglas de negocio. El objetivo de estas pruebas es verificar la aceptación, procesamiento y recuperación de datos y la adecuada implementación de las reglas de negocio. Este tipo de pruebas están basadas en técnicas de caja negra, es decir, verificar la aplicación interaccionando a través de las interfaces de usuario y analizando los resultados. Esto se ve reflejado en la Tabla IV.2

Objetivos de la prueba	Asegurar la navegación correcta de la aplicación, la entrada de datos, su procesamiento y recuperación.
Técnicas	Ejecutar cada caso de uso y flujo del caso de uso con datos válidos e inválidos para verificar lo siguiente: • Cuando se utilizan datos correctos se obtienen los resultados esperados. • Cuando se utilizan datos incorrectos se obtienen los mensajes de error o advertencias adecuadas. • Cada regla de negocio se ha aplicado correctamente.
Criterios de finalización	Todas las pruebas planificadas se han ejecutado. Todos los defectos identificados se han considerado.
Consideraciones	Ninguna.

Tabla IV.2: Pruebas de funcionalidad.

IV.4.3 Pruebas de interfaz de usuario.

Las pruebas de interfaz de usuario verifican la interacción del usuario con el sistema software. El objetivo de esta prueba es asegurar que la interfaz de usuario permite al usuario acceder y navegar a través de toda la funcionalidad de la aplicación. Esto se refleja en la Tabla IV.3.

Objetivos de la prueba	Verificar los siguientes objetivos
	• La navegación a través de la aplicación refleja adecuadamente las reglas de negocio y los requisitos incluyendo ventana a ventana, campo a campo y métodos de acceso (tabulador, movimientos del ratón y teclas de función). • Las ventanas y sus características, como menús, tamaño, posición y estado cumplen los estándares.
Técnicas	Crear o modificar pruebas para cada ventana con el objetivo de verificar la correcta navegación y su estado.
Criterios de finalización	Cada ventana se ha verificado con éxito y es consistente con la versión de referencia o con los estándares utilizados.
Consideraciones	Ninguna.

Tabla IV.3: Pruebas de interfaz de usuario.

IV.4.4 Herramientas a utilizar en la realización de las pruebas

Las siguientes herramientas se usarán para llevar a cabo el proceso de prueba: y se muestran en la Tabla IV.4.

Tipo de Prueba	Herramienta
Gestión del proyecto	Microsoft Project
Herramienta DBMS	pgAdmin
Interfaz de usuario	Dreamweaber

Tabla IV.4: Herramientas.

IV.5 Recursos a utilizar en la ejecución de las pruebas

En esta sección describimos los recursos necesarios para realizar el proceso de prueba, sus principales responsabilidades y características.

IV.5.1 Recursos hardware

En la Tabla IV.5, se presentan los recursos Hardware necesarios para la realización de las pruebas.

Recurso	Cantidad	Nombre y Tipo
PC	2	Diseño de las pruebas
PC	2	Ejecución de las pruebas

Tabla IV.5: Recursos hardware

IV.5.2 Recursos humanos

La Tabla 1.6, muestra el recurso humano necesario para la realización de las pruebas.

RECURSOS HUMANOS		
Rol	**Mínimos recursos recomendados**	**Responsabilidades específicas o comentarios**
Gestor de prueba	1	Proporcionar una gestión adecuada. Responsabilidades: • Proporcionar una dirección técnica. • Adquirir los recursos apropiados. • Informar de la gestión.
Diseñador de prueba	2	Identificar, priorizare implementar los casos de prueba. Responsabilidades: • Generar el Plan de pruebas. • Diseñar los Casos de prueba. • Evaluar el esfuerzo de prueba.
Probador (Tester)	2	Ejecutar las pruebas. Responsabilidades: • Ejecutar pruebas. • Recuperar los errores. • Documentar los defectos.

Tabla IV.6: Recursos humanos.

IV.6 Tareas de la etapa de pruebas

Las tareas que se realizan en cada una de las actividades son las que se presentan a continuación.

IV.6.1 Planificación de las pruebas

- Identificar los requisitos para las pruebas.
- Desarrollar la estrategia de pruebas.
- Identificar los recursos necesarios para realizar las pruebas.
- Generar el plan de pruebas.

IV.6.2 Diseño de las pruebas

- Desarrollo de las pruebas.
- Identificar y describir los casos de prueba.

IV.6.3 Implementación de las pruebas

- Establecer el entorno de prueba.
- Desarrollar las clases de prueba, los componentes de prueba y los datos de prueba.

IV.6.4 Ejecución de las pruebas

- Ejecutar los casos de prueba.
- Evaluar la ejecución del proceso de prueba.
- Verificar los resultados.
- Investigar los resultados no esperados.
- Registrar los defectos.

IV.6.5 Evaluación de las pruebas:

- Evaluar la cobertura de los casos de prueba.
- Evaluar la cobertura del código.
- Analizar los defectos.
- Determinar si se han alcanzado los criterios de las pruebas.

IV.7 Casos de Prueba

La Tabla IV.7, presenta la descripción del caso de prueba 1.01 correspondiente al caso de uso *Autenticar usuario*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1.0.1	**Código Caso de Uso:** 1.0 Autenticar Usuario.
Descripción de la Prueba: Esta prueba intentará lograr a un usuario válido con un usuario y un password correctos y un perfil Administrador SkyNet.	
Condiciones de Ejecución: No se requieren condiciones especiales.	
Entrada / Pasos de Ejecución: 1. Se ingresa a la página correspondiente al sistema. 2. Se ingresa un usuario válido con perfil Administración SkyNet. 3. Se ingresa la contraseña correspondiente al usuario	
Resultado Esperado: Al presionar el botón ingresar el sistema permite al usuario ingresar a las opciones correspondientes al perfil Administrador SkyNet.	
Evaluación de la Prueba: Evaluación satisfactoria	

Tabla IV.7: Caso de prueba *Autenticar usuario*

En la Tabla IV.8, se presenta la descripción del caso de prueba 1.0.2 correspondiente al caso de uso *Autenticar usuario*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1.0.2	**Código Caso de Uso:** 1.0 Autenticar usuario.
Descripción de la Prueba Esta prueba intentará autentificar a un usuario válido con un usuario y un password correctos, y perfil Usuario SkyNet.	
Condiciones de Ejecución: No se requieren condiciones especiales.	
Entrada / Pasos de Ejecución: 1. Se ingresa a la página correspondiente al sistema. 2. Se ingresa un usuario válido con perfil usuario SkyNet. 3. Se ingresa la contraseña correspondiente al usuario SkyNet.	
Resultado Esperado: Al presionar el botón ingresar el sistema permite al usuario ingresar a las opciones correspondientes al perfil Usuario SkyNet.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.8: Caso de prueba *Autenticar usuario.*

La Tabla IV.9, presenta la descripción del caso de prueba 1.03 correspondiente al caso de uso *Autenticar usuario*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1.0.3	**Código Caso de Uso:** 1.0 Autenticar usuario.
Descripción de la Prueba Esta Prueba intentará autenticar a un usuario válido con un usuario correcto, un password incorrecto y un perfil Usuario SkyNet o Administrador SkyNet.	
Condiciones de Ejecución: No se requieren condiciones especiales.	
Entrada / Pasos de Ejecución: 1. Se ingresa a la página correspondiente al sistema. 2. Se ingresa un usuario válido con perfil Usuario SkyNet o Administrador SkyNet. 3. Se ingresa una password inválido para el usuario que desea ingresar.	
Resultado Esperado: Al presionar el botón ingresar el sistema devuelve al usuario a la pantalla de inicio y le indica que el usuario o el password ingresados son incorrectos	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.9: Caso de prueba para *Autenticar usuario*.

En la Tabla IV.10, se presenta la descripción del caso de prueba 2.0.1 correspondiente al caso de uso *Seleccionar ruta*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.0.1	Código Caso de Uso: 2.0 Seleccionar ruta.
Descripción de la Prueba Esta prueba intentará acceder a la pantalla de selección de rutas de manera correcta.	
Condiciones de Ejecución: El usuario debe estar autenticado ya sea con perfil Usuario SkyNet o Administrador SkyNet.	
Entrada / Pasos de Ejecución: 1. Presionar el menú *Seleccionar ruta.*	
Resultado Esperado: Si se encuentra conectado por lo menos un robot al sistema, se permitirá el acceso a la pantalla de selección de rutas, habilitando él o los autos conectados para poder establecer las coordenadas, de lo contrario se enviará al usuario a una página la cual indica que no hay robots conectados donde el usuario podrá seleccionar la opción menú y volver a la pantalla de selección de ruta inicialmente dada.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.10: Caso de prueba para el caso de uso *Seleccionar ruta.*

La Tabla IV.11 presenta la descripción del caso de prueba 2.1.1 correspondiente al caso de uso *Establecer coordenada*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.1.1	**Código Caso de Uso:** 2.1 Establecer Coordenada
Descripción de la Prueba Esta prueba permitirá verificar si es posible establecer una coordenada a la ruta perteneciente al robot negro, robot amarillo o ambos.	
Condiciones de Ejecución: 1. El usuario debe haberse autenticado ya sea con perfil Usuario SkyNet o Administrador SkyNet. 2. Debe existir al menos un robot conectado al sistema. 3. Debe encontrarse en la pantalla de selección de ruta.	
Entrada / Pasos de Ejecución: 1. Seleccionar el robot deseado al que se le asignara una ruta. 2. Presionar en el mapa el punto a agregar a la ruta	
Resultado Esperado: Se debe dibujar el punto en el mapa, incluyendo una línea del color correspondiente al robot seleccionado (negro o amarillo).	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.11: Caso de prueba para el caso de uso *Establecer coordenada.*

La Tabla IV.12, presenta la descripción del caso de prueba 2.2.1 correspondiente al caso de uso *Enviar ruta*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.2.1	**Código Caso de Uso**: 2.2 Enviar ruta
Descripción de la Prueba Esta prueba permitirá verificar el no envío de una ruta sin localizaciones a visitar.	
Condiciones de Ejecución: 4. El usuario debe estar autenticado ya sea con perfil Usuario SkyNet o Administrador SkyNet. 5. Debe existir al menos un robot conectado al sistema. 6. Debe acceder a la opción *Seleccionar Ruta*	
Entrada / Pasos de Ejecución: 1. Presionar una de las opciones: *Enviar ruta robot negro*, *Enviar ruta auto amarillo* o *Enviar ruta a ambos robot*.	
Resultado Esperado: Se debe mostrar una alerta al usuario indicando que la o las rutas deben contener al menos una localización por visitar.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.12: Caso de prueba para el caso de uso *Enviar ruta.*

La Tabla IV.13, presenta la descripción del caso de prueba 2.2.2 correspondiente al caso de uso *Enviar ruta*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.2.2	**Código Caso de Uso:** 2.2 Enviar ruta
Descripción de la Prueba Esta prueba permitirá verificar el envío de una o ambas rutas a los robots correspondientes.	
Condiciones de Ejecución: 7. El usuario debe estar autenticado ya sea con perfil Usuario SkyNet o Administrador SkyNet. 8. Debe existir al menos un robot conectado al sistema. 9. Debe encontrarse en la selección de ruta. 10. Debe haber por lo menos seleccionado una localización geográfica, para cada uno de los robots ingresados	
Entrada / Pasos de Ejecución: 1. Presionar una de las opciones: *Enviar ruta robot negro*, *Enviar ruta auto amarillo* o *Enviar ruta a ambos robots*.	
Resultado Esperado: Se enviará la ruta al o los robots seleccionados, el usuario se redirige a la pantalla de monitoreo de rutas, en la cual deberá estar dibujada la ruta del o los robot(s) a los cuales se les envió la ruta, y paulatinamente se dibujara los movimientos que realiza el o los robots.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.13: Caso de prueba para el caso de uso *Enviar ruta*.

La Tabla IV.14, presenta la descripción del caso de prueba 2.3.1 correspondiente al caso de uso *Eliminar ruta*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.3.1	**Código Caso de Uso:** 2.3 Eliminar ruta
Descripción de la Prueba Se intentara eliminar la ruta del robot negro, siendo que la ruta no contiene localizaciones geográficas.	
Condiciones de Ejecución: 1. El usuario debe haberse autenticado exitosamente. 2. El robot negro debe estar conectado. 3. Se debe acceder a la pantalla de selección de rutas. 4. No se ingresan localizaciones geográficas a la ruta.	
Entrada / Pasos de Ejecución: 1. Presionar la opción *Borrar ruta robot negro*.	
Resultado Esperado: Se le informará al usuario que no es posible eliminar la ruta, dado que no contiene localizaciones geográficas.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.14: Caso de prueba para el caso de uso *Eliminar ruta.*

La Tabla IV.15, presenta la descripción del caso de prueba 2.3.2 correspondiente al caso de uso *Eliminar ruta*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.3.2	**Código Caso de Uso:** 2.3 Eliminar ruta
Descripción de la Prueba Esta prueba permitirá verificar la eliminación del robot amarillo, cuando esta no contiene localizaciones geográficas asignadas.	
Condiciones de Ejecución: 1. El usuario debe haberse autenticado exitosamente. 2. El robot amarillo debe estar conectado. 3. Se debe acceder a la pantalla de selección de ruta. 4. No se debe agregar localizaciones geográficas a la ruta del robot amarillo.	
Entrada / Pasos de Ejecución: 1. Presionar la opción *Borrar ruta robot amarillo.*	
Resultado Esperado: Se le informará al usuario que no es posible eliminar la ruta, puesto que no se han asignado localizaciones geográficas a la ruta.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.15: Caso de prueba para el caso de uso *Eliminar ruta.*

La Tabla IV.16, presenta la descripción del caso de prueba 2.3.3 correspondiente al caso de uso *Eliminar ruta.*

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.3.3	**Código Caso de Uso:** 2.3 Eliminar ruta
Descripción de la Prueba Esta prueba permitirá verificar la correcta eliminación de las rutas cuando contienen localizaciones geográficas a visitar.	
Condiciones de Ejecución: 5. El usuario debe haberse autenticado exitosamente. 6. Debe existir al menos un robot conectado al sistema. 7. Se debe acceder a la pantalla de selección de ruta. 8. Debe haberse agregado al menos una localización geográfica a alguna de las rutas.	
Entrada / Pasos de Ejecución: 1. Presionar la opción *Borrar rutas*	
Resultado Esperado: Eliminará la ruta trazada en el mapa y las asignadas cada robot.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.16: Caso de prueba para el caso de uso *Eliminar ruta*.

La tabla IV.17 presenta la descripción del caso de prueba 2.3.4 correspondiente al caso de uso *Eliminar ruta*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.3.4	**Código Caso de Uso:** 2.3 Eliminar ruta
Descripción de la Prueba Esta prueba permitirá verificar la correcta eliminación de la ruta asignada al robot negro cuando esta contiene al menos una localización geográfica.	
Condiciones de Ejecución: 1. El usuario debe haberse autenticado exitosamente. 2. El robot negro debe estar conectado. 3. Debe encontrarse en la pantalla de selección de ruta. 4. Debe haber al menos una localización geográfica asignada a la ruta del robot negro.	
Entrada / Pasos de Ejecución: 1. Presionar la opción *Borrar ruta robot negro.*	
Resultado Esperado: Se eliminará del mapa la ruta trazada para el robot negro al igual que las localizaciones geográficas asignadas en la lógica del sistema, y se mantendrán la ruta y las coordenadas del auto amarillo (En caso de que esté conectado y se haya trazado una ruta).	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.17: Caso de prueba para el caso de uso *Eliminar ruta.*

La Tabla IV.18, presenta la descripción del caso de prueba 2.3.5 correspondiente al caso de uso *Eliminar ruta*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.3.5	**Código Caso de Uso**: 2.3 Eliminar ruta
Descripción de la Prueba Esta prueba permitirá verificar la eliminación de la ruta del robot amarillo, a ambos robots se les han asignado localizaciones geográficas a la ruta.	
Condiciones de Ejecución: • El usuario debe haberse autenticado exitosamente. • Debe existir al menos un robot conectado al sistema. • Se debe acceder a la pantalla de selección de ruta. • Debe haber al menos una localización asignada a la ruta de cada robot.	
Entrada / Pasos de Ejecución: Presionar la opción *Borrar ruta robot amarillo.*	
Resultado Esperado: Se eliminará del mapa la ruta trazada para el robot amarillo al igual que las localizaciones geográficas en dicha ruta, además se mantendrán la ruta y las coordenadas del robot negro.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.18: Caso de prueba para el caso de uso *Eliminar ruta*.

La Tabla 1.19, presenta la descripción del caso de prueba 2.4.1 correspondiente al caso de uso *Restablecer valores*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 2.4.1	**Código Caso de Uso**: 2.4 Restablecer valores
Descripción de la Prueba Esta prueba permitirá verificar la correcta eliminación de las rutas correspondientes a los robots disponibles al igual que quedará sin selección el robot al cual se le ingresará la ruta debiendo seleccionar el robot deseado.	
Condiciones de Ejecución: • El usuario debe estar autenticado ya sea con perfil Usuario SkyNet o Administrador SkyNet.	
Entrada / Pasos de Ejecución: 1. El usuario presiona el botón *Restablecer valores*.	
Resultado Esperado: Se eliminarán del mapa la rutas trazadas y quedarán sin selección los *radio buttons* que indican el robot seleccionado a que se le ingresará una ruta.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.19: Caso de prueba para el caso de uso *Restablecer valores*

La Tabla IV.20, presenta la descripción del caso de prueba 3.0.1 correspondiente al caso de uso *Monitorear robot(s)*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 3.0.1	**Código Caso de Uso**: 3.0 Monitorear robot(s)
Descripción de la Prueba Esta prueba permitirá verificar el envió de la ruta a los robots seleccionados, y el monitoreo de los movimientos realizados por estos.	
Condiciones de Ejecución: 1. El usuario debe haberse autenticado exitosamente. 2. Debe existir al menos un vehículo conectado al sistema. 3. Debe haber al menos una ruta con coordenadas que se halla enviado.	
Entrada / Pasos de Ejecución: 1. Presionar la opción *Enviar ruta a robot negro*, *Enviar ruta a robot amarillo* o *Enviar ruta a ambos robots*.	
Resultado Esperado: Se enviará la ruta a él o los robots seleccionados, se redirigirá a la pantalla de monitoreo, donde se representarán las rutas asignadas a él o los robots a los que se les envió una ruta así como también, se dibujará la ruta real que realice el robot.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.20: Caso de prueba para el caso de uso *Monitorear robot(s)*

La Tabla IV.21, presenta la descripción del caso de prueba 3.0.2 correspondiente al caso de uso Detener robot(s).

Caso de Prueba de Aceptación	
Código Caso de Prueba: 3.0.2	**Código Caso de Uso:** 3.0 Detener robot(s)
Descripción de la Prueba Esta prueba permite verificar el correcto funcionamiento de la opción *Detener auto negro*, *Detener auto amarillo* y *Cancelar monitoreo*	
Condiciones de Ejecución: • El usuario debe haberse autenticado exitosamente, ya sea con perfil Usuario SkyNet o Administrador SkyNet y debe existir al menos un vehículo en monitoreo.	
Entrada / Pasos de Ejecución: 1. Presionar la opción *Detener robot negro*, *Detener robot amarillo* o *Cancelar monitoreo*.	
Resultado Esperado: Al presionar la opción *Detener robot negro* o *Detener robot amarillo*, debe enviarse la orden al vehículo, deteniéndolo. En el caso de presionar *Detener monitoreo*, se enviará la orden a ambos autos deteniendo el monitoreo y enviando al usuario al menú del sistema.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.21: Caso de prueba para el caso de uso Detener robot(s).

La Tabla IV.22, presenta la descripción del caso de prueba 5.1.1 correspondiente al caso de uso *Registrar usuario*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 5.1.1	**Código Caso de Uso:** 5.1 Registrar usuario
Descripción de la Prueba: Esta Prueba intentará registrar un nuevo usuario al sistema.	
Condiciones de Ejecución: El Administrador SkyNet debe haberse autenticado exitosamente.	
Entrada / Pasos de Ejecución: 1. Se ingresa a la opción *Gestionar usuario*. 2. Se completan todos los campos. 3. La contraseña y la confirmación de esta deben ser iguales. 4. Se envía formulario en la opción *Ingresar usuario*.	
Resultado Esperado: 1. Se ingresa el usuario al sistema	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.22: Caso de prueba para el caso de uso Registrar usuario.

La Tabla IV.23, presenta la descripción del caso de prueba 5.2.1 correspondiente al caso de uso *Eliminar usuario*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 5.2.1	**Código Caso de Uso:** 5.2 Eliminar usuario
Descripción de la Prueba: Esta prueba eliminará un usuario del sistema, estando este registrado en el perfil de Administrador SkyNet.	
Condiciones de Ejecución: • El Administrador SkyNet a eliminar debe haberse registrado anteriormente.	
Entrada / Pasos de Ejecución: 1. Presiona el botón *Eliminar* correspondiente al usuario que se desea eliminar. **2.** Acepta el mensaje: *¿Está seguro de eliminar el usuario?*	
Resultado Esperado: Se elimina el usuario.	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.23: Caso de prueba para el caso de uso *Eliminar usuario.*

La Tabla IV.24, presenta la descripción del caso de prueba 5.0.2 correspondiente al caso de uso *Eliminar usuario*.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 5.0.2.	**Código Caso de Uso:** 5.0 Eliminar usuario
Descripción de la Prueba: Esta prueba intentara eliminar un usuario del sistema, estando este registrado en el perfil de Administrador SkyNet. Una vez que se pida la confirmación de eliminación se cancelara el mensaje de eliminación.	
Condiciones de Ejecución: • El a usuario debe estar registrado en el sistema.	
Entrada / Pasos de Ejecución: 1. Presiona el botón *Eliminar* correspondiente al usuario deseado. 2. No acepta el mensaje: *¿Está seguro de eliminar el usuario?*	
Resultado Esperado: **1.** No se elimina el usuario y se mantiene en la página	
Evaluación de la Prueba: Evaluación satisfactoria.	

Tabla IV.24: Caso de prueba para el caso de uso *Eliminar usuario*.